MCR-75-17
Contract NAS5-11996

Volume II | Final Report | April 1975

PROGRAM USERS' GUIDE

COMPUTER PROGRAM SYSTEM FOR DYNAMIC SIMULATION AND STABILITY ANALYSIS OF PASSIVE AND ACTIVELY CONTROLLED SPACECRAFT

Authors

Carl S. Bodley
A. Darrell Devers
A. Colton Park

Approved

George Morosow
Program Manager

MARTIN MARIETTA CORPORATION
P. O. Box 179
Denver, Colorado 80201

FOREWORD

This report, prepared by the Dynamics and Loads Section, Martin Marietta Corporation, Denver Division, under Contract NAS5-11996, presents the results of a study whose purpose was to develop a computer program system for dynamic simulation and stability analysis of passive and actively controlled spacecraft. The study was performed from May 1973 to April 1975 and was administered by the National Aeronautics and Space Administration, Goddard Space Flight Center, Greenbelt, Maryland, under the direction of Mr. Joseph P. Young.

The report is published in four volumes:

Volume I - Theory
Volume II - Program Users' Guide
Volume III - Demonstration Problems
Volume IV - Program Listing

ACKNOWLEDGEMENTS

The authors wish to acknowledge the assistance provided by Goddard Space Flight Center personnel. Mr. Harold Frisch and Mr. James Donohue contributed many valuable technical comments and suggestions throughout the program. In particular, they layed out the basic approach which was to be used for data input, defined exactly what should be included in the transfer function and stability analysis portion of the program and defined 8 of the 11 demonstration problems. They provided the associated data which was used to verify both the nonlinear time response and linear transfer function and stability analysis portions of the program. Mr. Raymond Welch provided the subroutine used in the generation of root locus plots. Dr. William Case provided valuable advice on the interfacing with NASTRAN output and also he generated the demonstration problem utilized to validate the interface subroutine (NASFOR). During the very early program development stage, Dr. James Mason offered significant advice on the need to compute internal forces at the interconnect points. Mr. Reginald Mitchell contributed invaluable advice on requirements for making the program compatible with the GSFC IBM 360/95 computer system. In addition, he supplied the contractor with a 360 system compatible plot package, furnished the contractor a self authored subroutine to read NASTRAN output, and was responsible for running all of the eleven demonstration problems on the 360/95 computer. Finally, the authors wish to acknowledge the encouragement and efforts of Mr. Joseph P. Young, Technical Monitor, who made numerous valuable comments and suggestions throughout the study.

CONTENTS

		Page
I.	PROGRAM SYSTEM OVERVIEW	I-1
A.	Introduction	I-1
B.	Simulation Overview and Nomenclature	I-2
C.	General Usage Information	I-6 and I-7
II.	PROGRAM SEGMENTATION	II-1 thru II-3
III.	DELINEATION OF USER-SUPPLIED MODULES	III-1
A.	Logic Flow	III-1
B.	Specifics Related to User-Supplied Modules	III-3
C.	Discussion of Selected Common Block Information	III-9 thru III-12
IV.	PROGRAM INPUTS	IV-1
A.	Discussion of Subroutines READ and READIM	IV-1
B.	Input Data Stream	IV-2 thru IV-49
V.	PROGRAM OUTPUTS	V-1 thru V-25
VI.	AUXILIARY PROGRAMS	VI-1
A.	REDIM - The Redimension Program	VI-1
B.	NASFOR - The NASTRAN Interface Program	VI-4 thru VI-21
	APPENDIX INPUT/OUTPUT SUBROUTINE EXPLANATIONS	A-1 thru A-22
	BASELINE USER-PAK DEFINITION	B-1 thru B-10

Figure

I.B-1 Simulation Nomenclature I-3
II.A-1 DISCOS Program Segmentation II-1
III.A-1 Chronology of Addressing User-Pak III-2
III.B-1 Subroutine TORQUE Flow Diagram III-4
IV.B-1 Program System DISCOS Data Stream Flow IV-3
IV.B-1 Logic Flow, Program NASFOR VI-11

Table

II.A-1 DISCOS Overlay Description II-2
VI.B-1 Description of NASFOR Subroutines VI-9

ABSTRACT

A theoretical development and associated digital computer program system for the dynamic simulation and stability analysis of passive and actively controlled spacecraft is presented. The dynamic system (spacecraft) is modeled as an assembly of rigid and/or flexible bodies not necessarily in a topological tree configuration. The computer program system may be used to investigate total system dynamic characteristics including interaction effects between rigid and/or flexible bodies, control systems, and a wide range of environmental loadings. Additionally, the program system may be used for design of attitude control systems and for evaluation of total dynamic system performance including time domain response and frequency domain stability analyses.

Volume I presents the theoretical developments including a description of the physical system, the equations of dynamic equilibrium, discussion of kinematics and system topology, a complete treatment of momentum wheel coupling, and a discussion of gravity gradient and environmental effects.

The development of synthesis and analysis techniques for the linearized system includes a discussion of the numerical linearization technique, procedures for definition of system transfer functions, and linear time domain response.

Volume II is a program users' guide and includes a description of the overall digital program code, individual subroutines and a description of required program input and generated program output.

Volume III presents the results of selected demonstration problems that illustrate all program system capabilities.

Volume IV contains a listing of the digital code.

I. PROGRAM SYSTEM OVERVIEW

This volume is intended to provide the reader with sufficient understanding of program system DISCOS* and its capabilities so as to permit a user to employ the program as a basic tool to analyze the behavior of a wide range of dynamical problems. Specific emphasis will be on a simulation for multiply-interconnected spinning elastic bodies responding under the combined influences of external environments and either active or passive control.

A. INTRODUCTION

The simulation employs a state-space approach that was developed in detail in Volume I. The state-space formulation provides an attractive basis for simulation of nonlinear dynamical problems in a general sense as well as permitting linearization of the governing equations to provide an additional foundation with which to evaluate frequency domain and linearized time domain characteristics.

An attempt has been made to relieve the user from the requirement of having to communicate with the digital program via large amounts of bulk data input. Although the program has many options available, the program data stream has been organized to require only a minimal amount of basic input data for a particular simulation. The data requirements have been further consolidated in a manner that is quite definitive for the physical system being simulated. In summary, the user can quite easily relate to the particular elements of the program requirements and thus minimize setup time required to prepare data input for a given problem. In addition, a set of self-checking features has been included in an attempt to identify and check certain compatabilities that are necessary for a proper simulation of a physically realizable system.

In an overall sense, the digital program can be employed by the user to obtain

1. nonlinear time response,
2. interaction constraint forces,

---

* Dynamic Interaction Simulation of Controls and Structure

3. total system resonance properties,
4. frequency domain response and stability information,
5. linearized time response.

The program outputs consist of printed and plotted results depicting

1. dynamic model construction,
2. time domain response,
3. frequency domain characteristics.

The printed outputs are of a fixed form while the user controls the plotted information through the input data stream.

B. SIMULATION OVERVIEW AND NOMENCLATURE

This discussion identifies the basic nomenclature used to synthesize a typical assembly of interconnected bodies. The theoretical development, program users' manual, and demonstration problems make extensive reference to various terminologies that are clarified here. Figure I.B-1 provides a visual display that illustrates many of the items being discussed and will be repeatedly referred to in the ensuing discussions.

The overall system "topology" is identified by the user via the input integer array ITOPOL that contains the necessary information describing which "hinges" interface which bodies. Each body contains a body reference point that is the origin of an orthogonal cartesian body axis system. This point need not coincide with the body center of mass.

Contiguous "bodies" are interfaced through a "hinge". We say interfaced in lieu of connected to emphasize the fact that the common "hinge" point between contiguous bodies may actually permit relative translational motion of the two bodies at the hinge. The degree of fixity at the hinge is identified by the user via the input integer array IHDATA. A typical body may contain "sensor" points that identify particular points where additional information is required to complete the desired simulation. A sensor point might sense on position or rate for the control system inputs, but could also represent a point on a body where certain other information is desired, such as a momentum wheel location or a point of force/torque application.

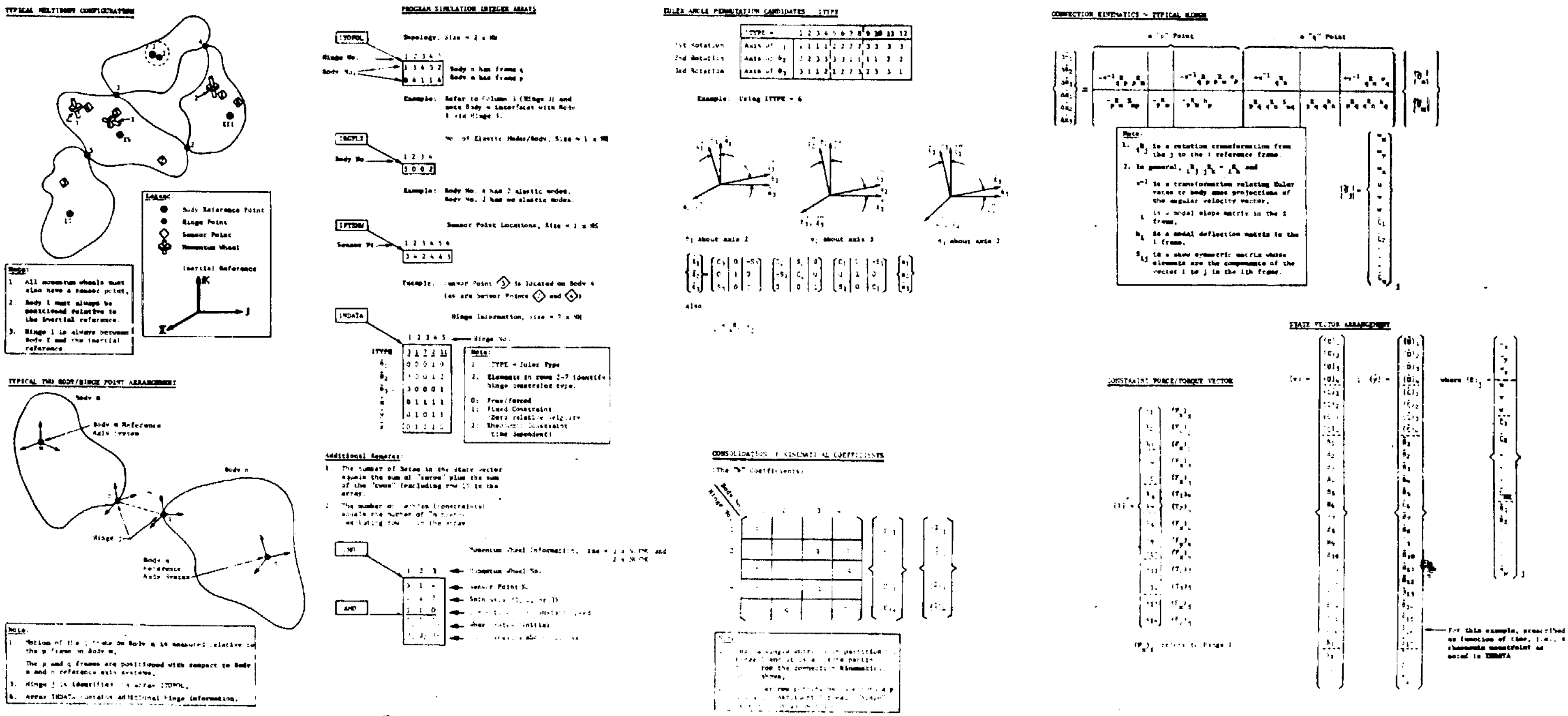


Figure I.B-1 Simulation Nomenclature

ORIGINAL PAGE IS OF POOR QUALITY

The integer input array IFTSMW identifies the body, where a particular sensor point is located.

A body may also contain "momentum wheels". This special consideration accommodates a disk or rotating mass with a single relative rotational degree of freedom into the simulation without introducing another body. The momentum wheel capability is more efficient for the simulation of a single degree of freedom rotating mass than is constraining 5 of 6 rigid body degrees of freedom via constraint equations. All momentum wheels must have an associated sensor point. A wheel may either be active or constant speed; an active wheel has a variable spin rate and receives an input torque (generally via some sensor output relationship) and a shaft torque is applied to the wheel inducing a wheel angular acceleration. The array IMO identifies whether or not the wheel is active, and which axis is the spin axis. The reference axis for the wheel is the same as the sensor point axis system where the wheel is located. The array AMO identifies the wheel spin rates (initial rate only for the active wheel) and the wheel spin inertia about the spin axis.

The system state vector is arranged in a specific manner within the program and it is necessary for the user to be very familiar with this arrangement for a number of reasons. First, the user must know where certain variables are located so that he can couple the control law into the simulation, and secondly, the user must know the order of the state variables in order to interpret results. Figure I.B-1 presents the state variable order consistent with the illustrative problem and other related information. The state variables shown do indeed represent a typical arrangement in that all of the various types of variables resulting from the multiple options available within the simulation are present. The order of the constraints (λ) is also noted. Note that the user introduces the control variables into the state vector but these variables (δ) will always appear after the betas (β). Furthermore, the user may also introduce auxiliary variables (plant sensor signals and control system outputs) for use in the linearized studies. These auxiliary variables should be placed (by the user) after the control variables (δ) and in the order: plant sensor signals (X_{SS}) followed by the control system outputs (B).

C. GENERAL USAGE INFORMATION

There are a number of guidelines that must be adhered to in setting up a particular simulation. Some were detailed previously and are concisely summarized here:

1. there must be at least two bodies; a single body problem is simulated by including a dummy body that is not connected to the body to be analyzed;

2. body no. 1 is always positioned relative to the inertial reference;

3. bodies are numbered from 1 to NB in an arbitrary order;

4. every body (except body 1) must have at least <u>one</u> hinge; body 1 must have at least two hinges;

5. hinges are numbered from 1 to NH in an arbitrary order but hinge no. 1 is, by definition, the hinge on body 1 between body 1 and the inertial origin; hence, hinge no. 1 can only appear on body 1;

6. there must be at least one sensor point for a given simulation;

7. sensor points are numbered from 1 to NS in an arbitrary manner;

8. a typical flexible body requires mass and modal data that reflects a coordinate system that is consistent with the body axis reference system for that body, e.g., a modal coupling approach establishing modal properties for a given body would have to use the same reference body axis system;

9. for frequency domain studies, there can only be as many control output variables identified to introduce into the state equations as there are control system variables to begin with. Similarly, there can be no more sensor signal variables identified than plant variables which appear in the original independent state equations.

10. the user must make certain that the user supplied package has dimensions consistent with NHMAX for the arrays

SK(NSK, NHMAX), DK(NDK, NHMAX), and HNGT(NHT, NHMAX)

where NHMAX = dimensioned maximum number of hinges,

NSK = 3 or 6 depending upon nature of hinge freedom,

NDK = 3 or 6 depending upon nature of hinge freedom,

NHT = 3 or 6 depending upon nature of hinge freedom,

and if rotation only, then NSK = NDK = NHT = 3, if rotation and translation, then NSK = NDK = NHT = 6;

11. the inertial properties of all the momentum wheels in a particular body must be included in that body's inertia description (whether rigid or flexible) since inertial coupling is used.

II. PROGRAM SEGMENTATION

The digital code has been segmented into an executive overlay which governs the succeeding program flow and four supporting primary overlays, each with a separate and dedicated purpose. The basic program flow is depicted in Figure II.A-1.

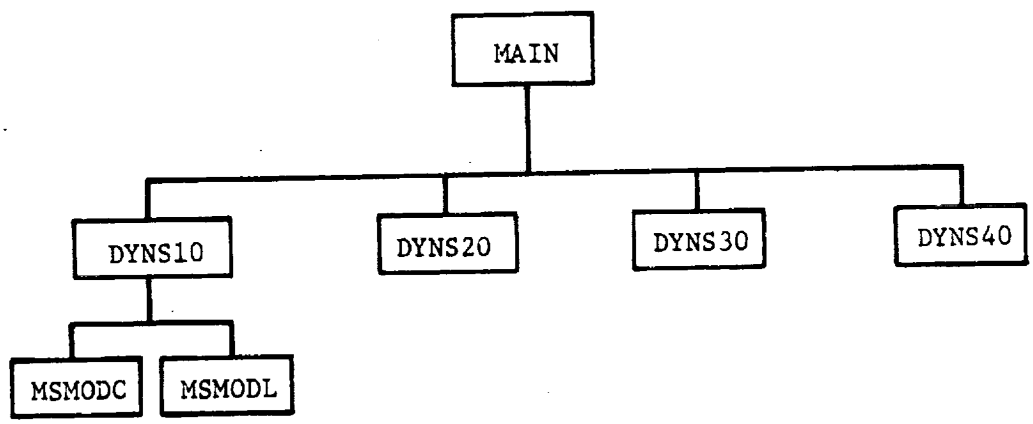


Figure II.A-1. DISCOS Program Segmentation

Table II.A-1 summarizes the intended purpose of the fundamental components in the program structure.

Table II.A-1 DISCOS Overlay Description

Primary Overlays	
Segment Name	Purpose
MAIN	Executive program control
DYNS10	Data input
DYNS20	Simulation of problem, linearization of state eq's, nonlinear time response
DYNS30	Plot results from nonlinear or linearized time response
DYNS40	Frequency domain analysis, linearized time response, frequency domain displays, (Bode, Nichols, Nyquist, Root Locus)
Secondary Overlays (called from DYNS10)	
MSMODL	Flexible body data inputs for lumped mass representation
MSMODC	Flexible body data inputs for consistent mass representation

The executive overlay (MAIN) initiates the simulation by reading job identification information and then passes control to the first primary overlay (DNYS10) which represents the basic data input segment. This overlay may be viewed as the program segment which builds the model from the input data. A series of topology checks are made as the data is loaded within this overlay to better assure proper modeling of the physical system. This overlay utilizes two additional secondary overlays for processing certain types of inertial and modal data.

After overlay DYNS10 has structured the basic data for simulation, control is returned to the executive overlay which in turn passes control on to the second overlay DYNS20. This overlay performs the actual problem mechanization and develops the nonlinear formulation which is the foundation for the entire dynamic simulation program.

During a given simulation, the executive overlay always calls both the first and second primary overlays (DYNS10 and DYNS20) but, depending upon certain input control parameters, may or may not call the time history plot overlay DYNS30 or the linearized system analysis overlay DYNS40.

Simulation of a particular problem has its basis within the algorithms contained in the program subroutine YDØT which establishes the canonical first-order differential equations that govern the dynamical motion. This routine in turn addresses another subprogram TØRQUE which in turn activates the user supplied modules that relate to the particular simulation being considered. These modules are discussed in further detail in a following section.

III. DELINEATION OF USER-SUPPLIED MODULES

The program has been written under the assumption that certain user-supplied modules are available to complete a given problem. In this manner, the user has considerable latitude with regard to how certain particulars related to a given simulation are to be handled. Control law specification, external torque inputs, and identification of plant sensor signals and control system outputs are examples of items handled by the user. With this concept in mind, several subprograms have been placed under user control but with certain restrictions and guidelines to which the user must adher. Later comments will identify certain requirements associated with these user supplied modules.

A. LOGIC FLOW

It is worthwhile to consider a flow chart segment of the program (Figure III.A-1) and its chronology within the solution process. The order which the user supplied modules are called is indicated by the integers 1 through 7 (for subroutines) and 8 and 9 for functions.

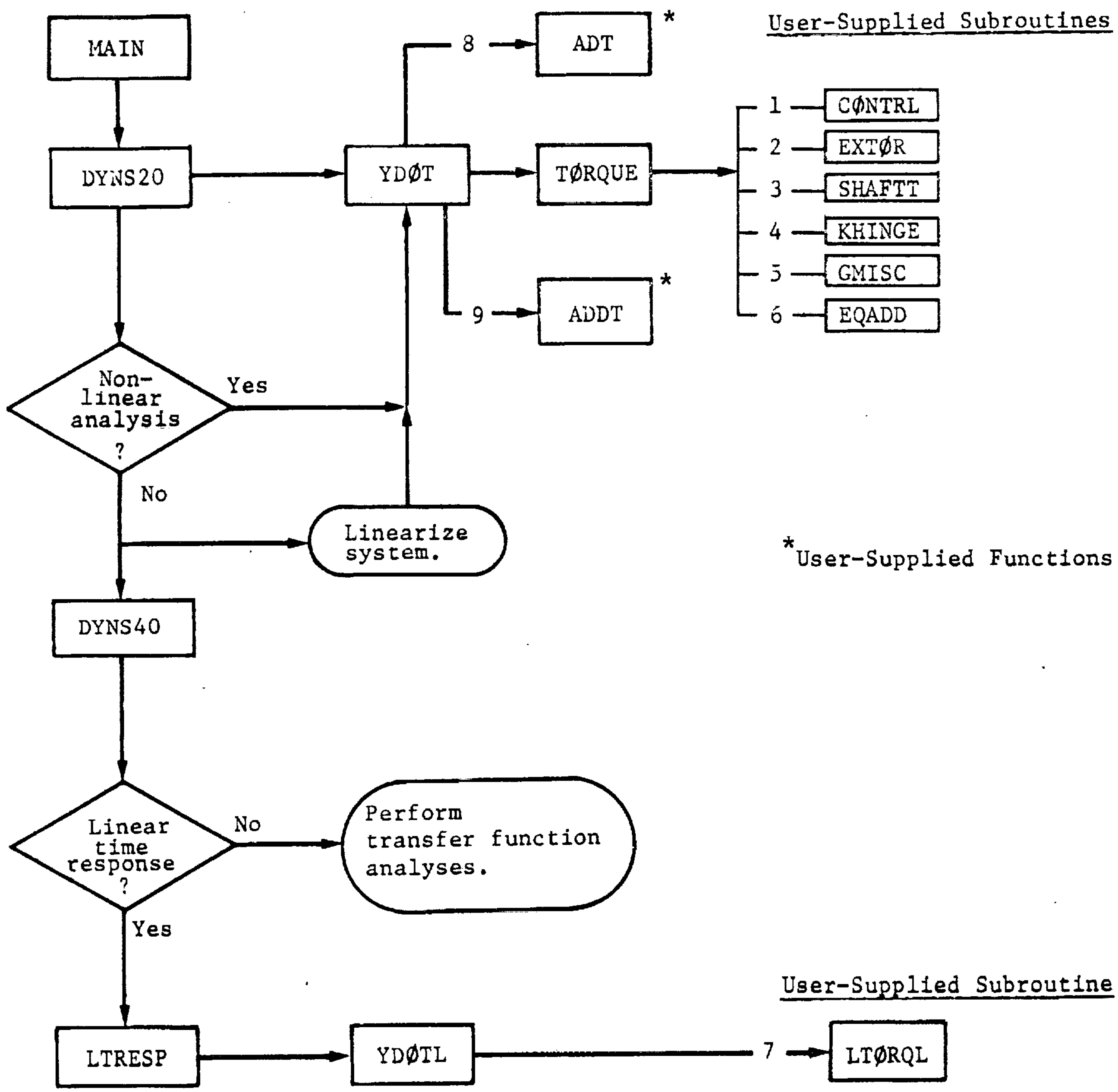


Figure III.A-1 Chronology of Addressing User-Pak

B. SPECIFICS RELATED TO USER-SUPPLIED MODULES

The separate user supplied subprograms each have specific intended purposes and have been coded to fulfill these goals. The user can extend the scope of any of these modules with his own code, but there are certain items that these routines must perform. In any case, the potential user should be very familiar with many of the details of the user supplied package, and it is with this fact in mind that a separate discussion will now be devoted to each of the user supplied modules. Reference will be made to some of the programming logic contained in the DISCOS subroutine TØRQUE, and so this logic has been put into flow chart form as Figure III.B-1. Also, for reference purposes, the seven user-pak subroutines and two subfunctions (ADT, ADDT) are included in Appendix B corresponding to typical situations.

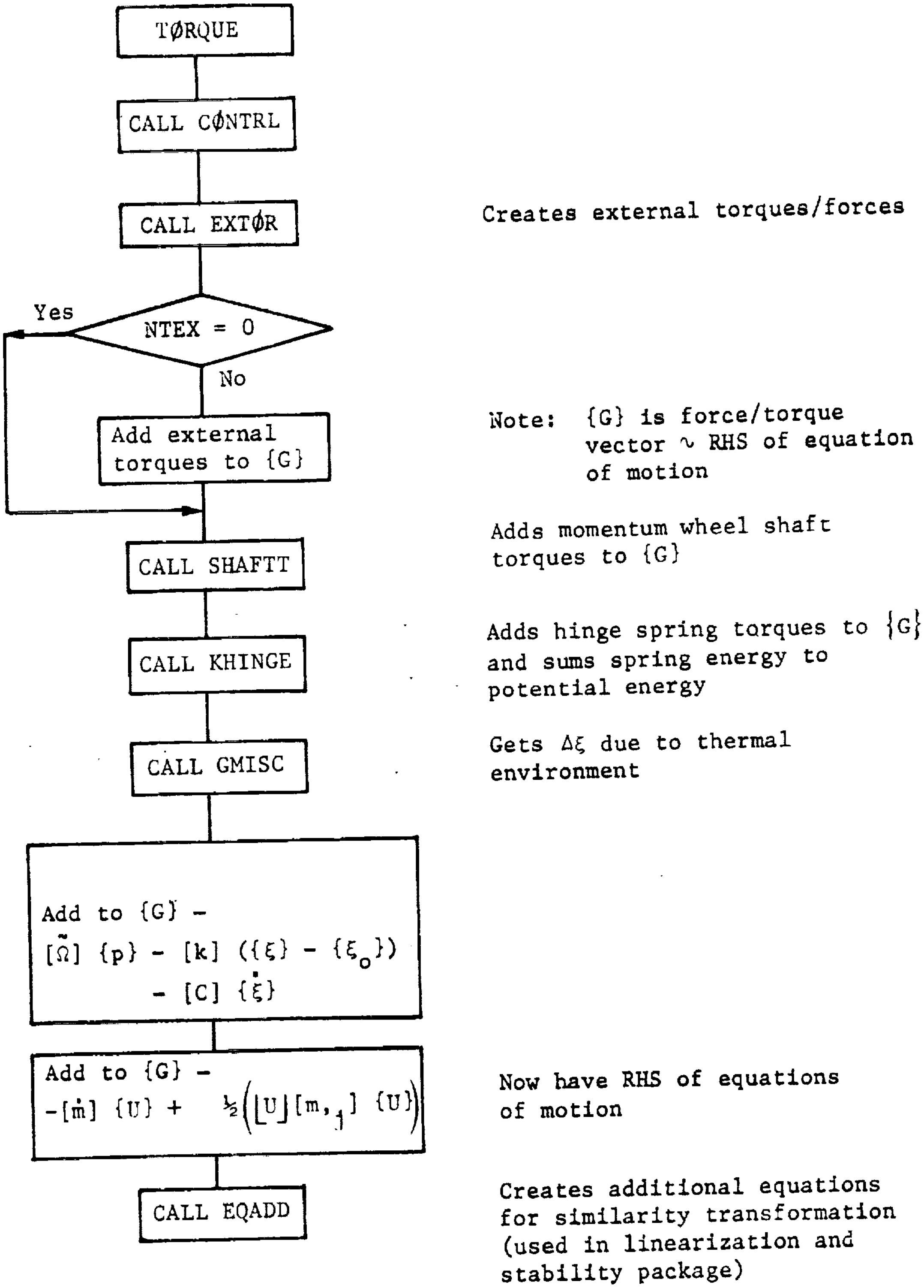


Figure III.B-1 Subroutine TØRQUE Flow Diagram

1. <u>CØNTRL</u>

This is the first of the user-supplied routines and is always called by DYNS20. The primary purpose of CØNTRL is to establish the time derivatives of the control system variables. These variables may be required by some of the other user routines that are activated after CØNTRL has been addressed. The routine must also establish the number of plant sensor signals (NXSS) and the number of control system outputs (NBTQ) which are transmitted through common block (/LDSIZE/) to the remainder of the program. For transfer function studies, the user is also required to identify whether or not transfer function polynominals are to be utilized. This is accomplished in a data statement (in CØNTRL) with the variable NPLY which is the number of polynominal ratio pairs (numerator and denominator) to be utilized. The first call to CØNTRL will read in the polynominal coefficients (for NPLY $\neq$ 0).

Subroutine CONTRL contains a good deal of information pertaining to the simulation by virtue of its common blocks. Section C identifies the constiuents of the common blocks contained in this and other modules. Additional common blocks can be established by the program user to transfer information between the separate user-pak modules.

2. <u>EXTØR</u>

This subroutine establishes the system external torques. Typically, this module can be utilized to accommodate such items as RCS (Reaction Control System) forces and torques, aerodynamics, and/or solar wind. The user can also extend this routine to include the addition of other state dependent torques. In summary, EXTØR, can be used as a "catch all" for inclusion of any additional forces and torques acting on the system. A single call to EXTØR from subroutine TØRQUE establishes an integer array (ISNP) whose elements identify which sensor points are to be used for force/torque inputs. A vector containing torque and force components (ordered: T_x, T_y, T_z, F_x, F_y, F_z) is then established for each of the force/torque sensor points and placed as a column into the array TEX. The vector of discrete forces and torques (referred to the local sensor-axis system) is returned to subroutine TØRQUE from EXTØR. These forces and torques are then transformed and added to the total system external force/torque array $\{G\}$. The user can bypass EXTØR related calculations by setting the variable NTEX equal to zero.

3. SHAFTT

This routine establishes the shaft torque for each of the non-constant speed momentum wheels. Zeros are inserted for the torque contributions to the external torque vector for a constant speed wheel.

4. KHINGE

This routine sets up hinge spring and dashpot torques/forces. It also accounts for potential energy contributions due to hinge spring deflections. The user must identify where spring rates and dashpot constants are to be found. This can easily be handled by a user specified equivalence statement within subroutine KHINGE to locate the leading stiffness and damping elements within the data block identified as CNTDTA. Note: within subroutine KHINGE (see subroutine listing, Appendix B) there are statements of the following form

```
DIMENSION SK(3, NHMAX), DK(3, NHMAX), HNGT(3, NHMAX)
      .
      .
      .
DO 10 I=1,3
      .
      .
DO 15 I=1,3
      .
      .
DO 20 I=1,3
      .
      .
```

where the integer 3 reflects the fact that consideration has been restricted to admitting only rotational springs at each hinge. If the user wants to also include springs/dashpots in relative translation at the hinge points, the three (3) in the statements above must be changed to a six (6), and appropriate spring rates and/or dashpots included within the data input array CNTDTA. Further, the equivalence statement locating the first spring rate (SK(1)) and dashpot constant (DK(1)) reflect an order that is consistent with the hinge order; that is the first three elements in CNTDTA starting with the location corresponding to the leading element of SK represents in order $K\theta_1$, $K\theta_2$, $K\theta_3$ for the first hinge. A similar relationship exists for the array DK.

Example: In KHINGE note that

EQUIVALENCE(CNTDTA(K),SK(1)),(CNTDTA(L),DK(1))

and the array CNTDTA would by

CNTDTA = . . . K_{θ_1} K_{θ_2} K_{θ_3} . . . K_{θ_1} K_{θ_2} K_{θ_3} . . .

element(K) → Hinge 1 (springs) . . . Hinge NH (springs)

element(L) → . . . $C_{\dot{\theta}_1}$ $C_{\dot{\theta}_2}$ $C_{\dot{\theta}_3}$ . . . $C_{\dot{\theta}_1}$ $C_{\dot{\theta}_2}$ $C_{\dot{\theta}_3}$ . . .

Hinge 1 (dashpots) . . . Hinge NH (dashpots)

where the order of $\theta_{1,2,3}$ is consistent with the Euler rotation type (1-12) for the hinge "q" triad.

The remainder of KHINGE is concerned with the proper placement of the spring/dashpot forces and torques onto the composite NB bodies (generalization of forces and torques) and should remain unchanged.

The user can modify the referenced torques and forces immediately after the (DØ 10 L=1,NH) loop if he desires, but care must be taken to assure that the proper force or torque is correctly applied to accomplish the desired result. Several of the demonstration problems (refer to Volume III) employ this process to apply control system outputs.

5. GMISC

This routine is reserved to implement torque/force contributions from thermal gradient effects. The entire state vector, along with component position and attitude information, is available via transfer through labeled common arrays. Section C provides more insight into the information contained in these common blocks.

6. EQADD

This routine establishes additional equations for use in the linearized time domain analyses. It must identify the number of additional equations introduced via the variable NAUX (number of auxiliary equations). These equations relate plant sensor signals, X_{SS}^{i} and control system output forces/torques, B^{i}, to the system state and in the specified order. The additional variables must be placed in the state vector as the last NAUX state variables and in the order, X_{SS}^{i} , then B^{i} and they become an integral part of system transfer function evaluations.

7. LTØRQL

This routine establishes the $b_{ik}U^{k}$ portion of the right hand side of

$$\dot{z}^{i} = A_{ij}z^{j} + b_{ik}U^{k}$$

which is used for the linearized time response. This corresponds to the external excitations for the transformed variables, z^{i}, leading to evaluation of the perturbation response.

8. ADT (Subfunction)

This function is used in conjunction with ADDT to implement prescribed kinematical motion in the hinge coordinates. With reference to Figure I.B-1, the array IHDATA(I,J),I> 1, may have 2 as an entry indicating that the Jth hinge has velocity and acceleration prescribed in that coordinate. The argument of this subfunction is: ADT(IC,T), with IC=6*(J-1)+(I-1) corresponding to IHDATA(I,J)=2. The integer IC and the time T are passed into ADT via argument by the calling subroutine so that the velocity ($\dot{\alpha}$) may be established for the proper hinge coordinate as a function of time.

For a given rheonomic constraint, we note that both $\dot{\alpha}$ and $\ddot{\alpha}$ must be set by subfunction. Now, it is conceivable that the user knows $\ddot{\alpha}$ as the exact mathematical time derivative of $\dot{\alpha}$. It would seem that the natural thing to do would be to create ADT and ADDT subfunctions to return consistent $\dot{\alpha}$ and $\ddot{\alpha}$ respectively. This is not the best thing to do, however, because of numerical integration characteristics. The numerical integration of $\{\dot{U}\}$ reflects the use of $\ddot{\alpha}$. The resulting $\{U\}$ reflects a numerically integrated $\dot{\alpha}$ which cannot be consistent with a value obtained any way other than numerical integration. The consequences of this are seen as slight errors in motion response, but also, there is a large spurious change in system momenta.

The best way to effect rheonomic constraints is to use values of $\dot{\alpha}$ obtained from numerically integrating $\ddot{\alpha}$. This can be easily done by using additional differential equations that are accommodated in the state vector as additional "control variables" or $\{\delta\}$. Thus, after all of the actual control variable rates are established (in subroutine CØNTRL), one need only code additional expressions to set $\dot{\delta}$(additional) = $\ddot{\alpha}$(desired). The statements within subfunction ADT merely return ADT=Y(K); the state vector Y is available in labeled common /VECTOR/, and K corresponds to the location in Y where the $\delta = \dot{\alpha}$ control variable resides. Of course, IC must be tested such that the appropriate $\dot{\alpha} = \delta$ is returned.

9. ADDT (Subfunction)

This function is discussed with regard to its relationship to ADT in Section (8) above. This function has arguments: ADDT(IC,T), exactly the same as ADT, and returns values of $\ddot{\alpha}$ for appropriate IC and T consistent with the $\dot{\alpha}$ returned by ADT. Note in Figure III.A-1, the chronology is such that subroutine CØNTRL is addressed prior to function ADDT. This is so that CØNTRL can establish a value of $\dot{\delta}$(additional) = $\ddot{\alpha}$(desired) to put in the state vector time derivative (YDT, also available in labeled common/VECTOR/). Now, for the appropriate time T and IC, it is only necessary to set ADDT=YDT(K), where again K corresponds to the location in Y where the $\delta = \dot{\alpha}$ auxiliary control variable resides.

C. DISCUSSION OF SELECTED COMMON BLOCK INFORMATION

The program user will very often have a need to access certain information that is calculated and stored within the program in order to compute specific variables required for the user supplied modules. Such information about the simulation is stored in multi-dimensional array form within labeled common blocks. These data provide a good supplement to the state variable content which has been previously discussed in that the user can extract both total and relative positions and rates for any component of the simulated dynamical system once he has a firm understanding of where certain data reside within the program. The following subsections will discuss selected common block arrays to better familiarize the potential user with their content.

1. Common block/BHBSRD/ contains three separate groups of information which the user may need to access. This information is concisely summarized in double and/or triple subscripted arrays as

BS(6,6+NMDBOD,NSPMAX)

ROL(3,3,NBMAX)

DOL(3,NBMAX)

where the following items are noted -

NMDBOD = maximum dimensioned number of modes per body,

NBMAX = maximum dimensioned number of bodies,

NSPMAX = maximum dimensioned number of sensor points.

The array, BS(i,j,k), contains the kinematical coefficients for all of the "sensor" points. The rows (subscript i=1,2...6) of the array refer to (in order: ω_x, ω_y, ω_z, u, v, w) the components of absolute angular and translational velocity (sensor referenced) at sensor point k. The columns of the array (subscript j) refer to the j=1,2,...6 + no. of elastic modes on body containing sensor point k. Thus, in general, if we want to know the projection (the ith velocity component) onto the triad located at sensor point k, the following expression is noted

$$Vel_i = BS(i,j_1,\ldots,j_L,k) \cdot \tilde{U}^j \quad .$$

The array, ROL(i,j,k) contains the rotation transformations relating the body axis systems to the inertial reference. The elements of the array are the direction cosines between the body axes, $\hat{e}_k$, and the fixed inertial system, $\hat{e}_o$. Subscript k denotes the body number.

The array DOL(i,k) contains the three vector components, (X, Y, Z), from the inertial reference to the body axis system, $\hat{e}_k$, for each body.

2. Common block /SPECIF/ contains information which the user may require. These arrays are

BETAH(6,NHMAX)

BETAHD(6,NHMAX)

RS(3,3,2*(NSPMAX))

DS(3,2*(NSPMAX))

where the following items are noted -

NHMAX = maximum dimensioned number of hinges,

NSPMAX = maximum dimensioned number of sensor points.

The arrays BETAH(i,j) and BETAHD(i,j) contain the hinge BETA's and rates respectively (for hinge j). The order (i subscript) is given as

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\Delta}_1 \\ \dot{\Delta}_2 \\ \dot{\Delta}_3 \end{bmatrix}$$

where $\dot{\theta}_i$ is the ith Euler angle rate consistent with ITYPE for hinge j and $\dot{\Delta}_i$ is the ith velocity component of point q relative to point p in the p frame for hinge j.

The array RS(i,j,k) contains the rotation transformations (direction cosines) between the sensor point axis system and the body axis system (body on which sensor is located). Two sets of transformations are identified for a given sensor point. The first represents misalignment of the two triads without elastic deformation and the second includes the elastic deformation. The ordering (subscript k) proceeds as follows: the ℓth sensor rotation (without elastic deformation) is located at $k = 2*\ell-1$. The total rotation transformation for the ℓth sensor is located at $k = 2*\ell$. For a rigid body, these two transformations are identical.

The array DS(i,k) contains the three components of the vector from the body axis system to the body sensor points (in the body axis system). Here again, there are two sets of vectors (rigid body and rigid body + elastic) for each sensor point. The first is for rigid body and the second includes the elastic deformation. The addressing algorithm is the same as for RS(i,j,k).

IV. PROGRAM INPUTS

The dynamic simulation program utilizes some basic data input subroutines in an attempt to standardize a large amount of the bulk data input. Additional formatted inputs have been used where it is more meaningful (and more efficient) to do so. As will be noted in the following section, there is a large amount of data input via subroutines READ and READIM. Therefore, it is useful to familiarize the reader with these two routines prior to describing overall program data input requirements.

A. DISCUSSION OF SUBROUTINES READ AND READIM

These two subprograms are structured to read matrix arrays in floating point (real) notation (subroutine READ) or fixed point (integer) notation (subroutine READIM). A thorough discussion of the routines and their supporting subroutines is contained in Appendix A. The following discussion gives a cursory overview of their usage.

The routines are activated by a FORTRAN call of the form:

CALL READ (A, NR, NC, KR, KC) or

CALL READIM (IA, NR, NC, KR, KC)

where the arguments in the call statement are

A, (IA) = floating (fixed) matrix array of size NR by NC

NR = number of rows in array

NC = number of columns in array

KR = row dimension of array in calling program

KC = column dimension of array in calling program

A call to either of these input routines requires that the data be in the following format:

1. Subroutine READ

First card - matrix name, NR, NC with format (A6,I4,I5)

Middle cards - data with format (2I5, 4D17.8)

first I5 is row number
second I5 is column number of leading D17.8 field
next 4D17.8 are elements of the array

Last card - ten zeros in columns 1 through 10

2. Subroutine READIM

First card - matrix name, NR, NC with format (A6,I4,I5)

Middle cards - data with format (2I5, 14I5)

first I5 is row number
second I5 is column number of leading I5 field
next 14I5 are elements of the array

Last card - ten zeros in columns 1 through 10

B. INPUT DATA STREAM

This section presents the program system input data stream together with the data input control logic. The approach taken herein is to first introduce an overview of the data inputs and program control logic in the form of a flow diagram (Figure IV.B-1) and to then identify the details in much the same way as the FORTRAN code accepts the data inputs. This method of presentation has been chosen as it most closely relates to the actual processing of the user inputs for a given simulation. In addition, the user can follow the program control or switching logic to determine just what data are required to complete a particular simulation.

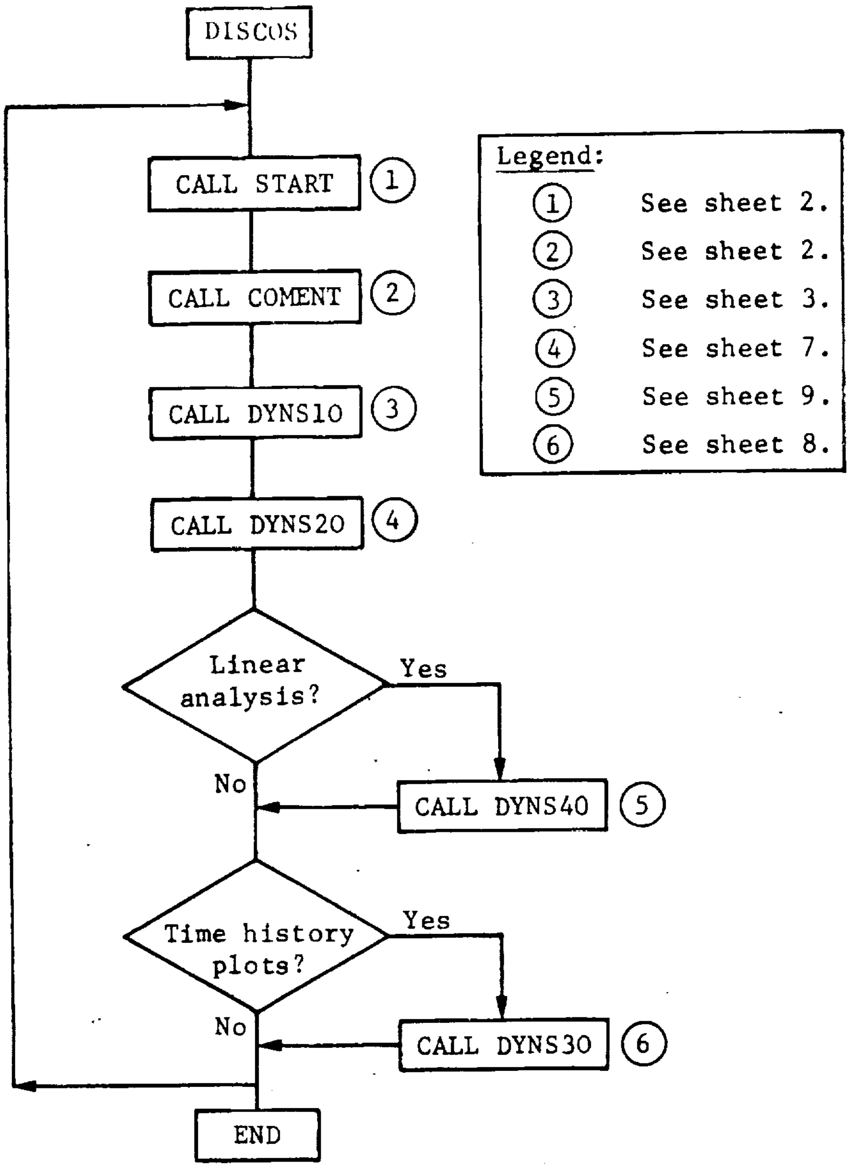


Figure IV. B-1
Program System DISCOS Data Stream Flow (Sheet 1 of 9)

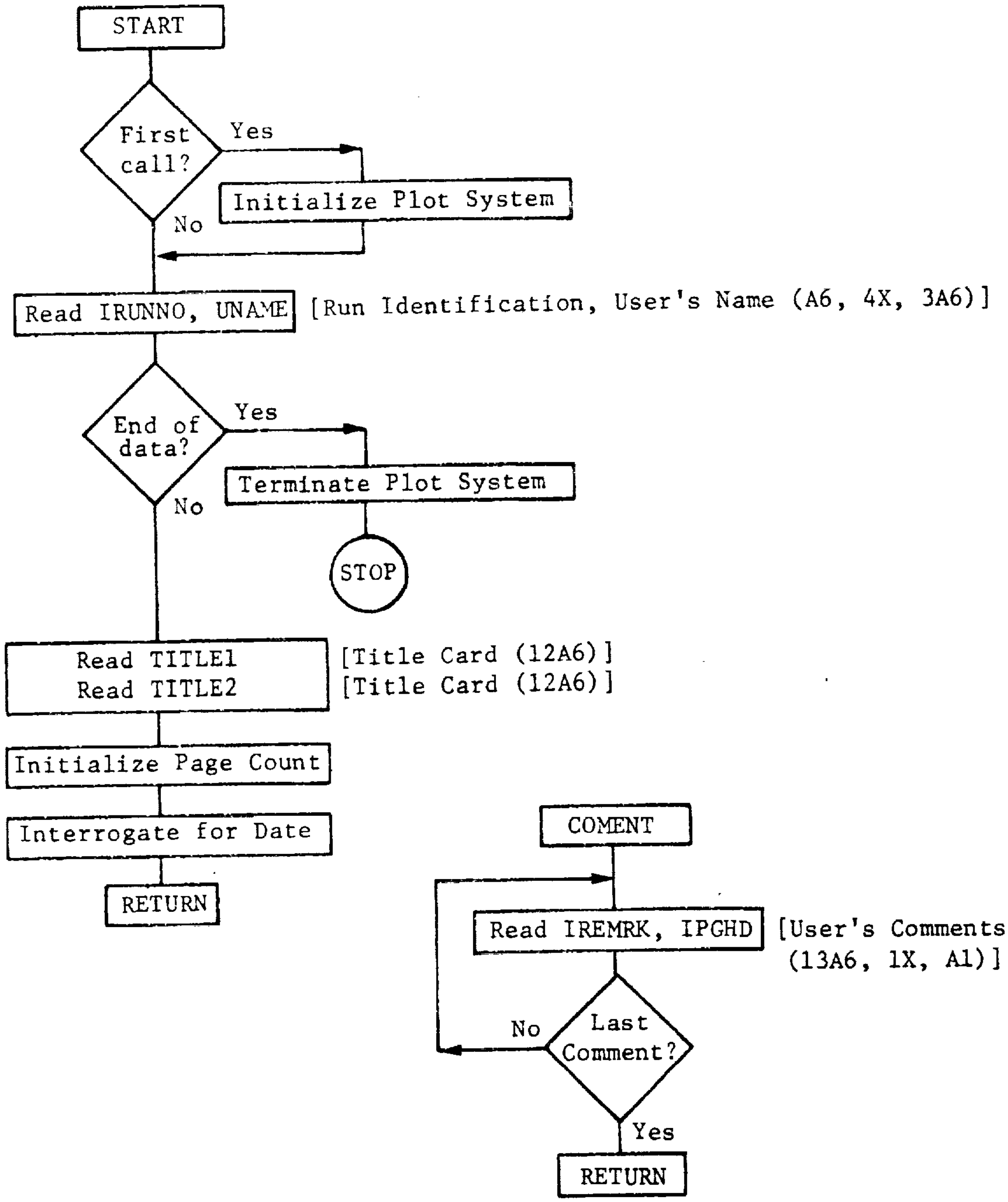
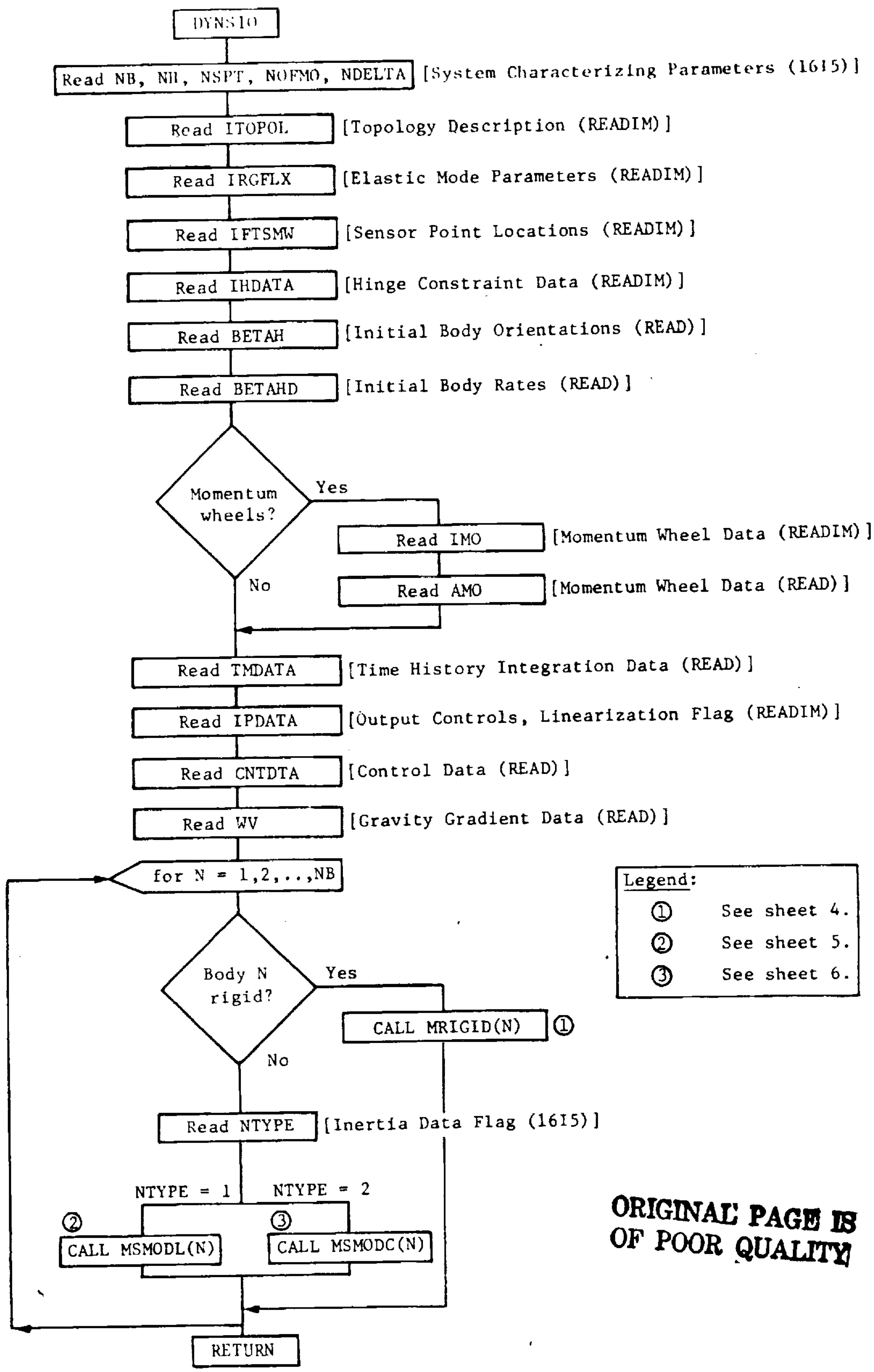


Figure IV.B-1
Program System DISCOS Data Stream Flow (Sheet 2 of 3)



ORIGINAL PAGE IS
OF POOR QUALITY

Figure IV.B-1
Program DISCOS Data Stream Flow (Sheet 3 of 9)

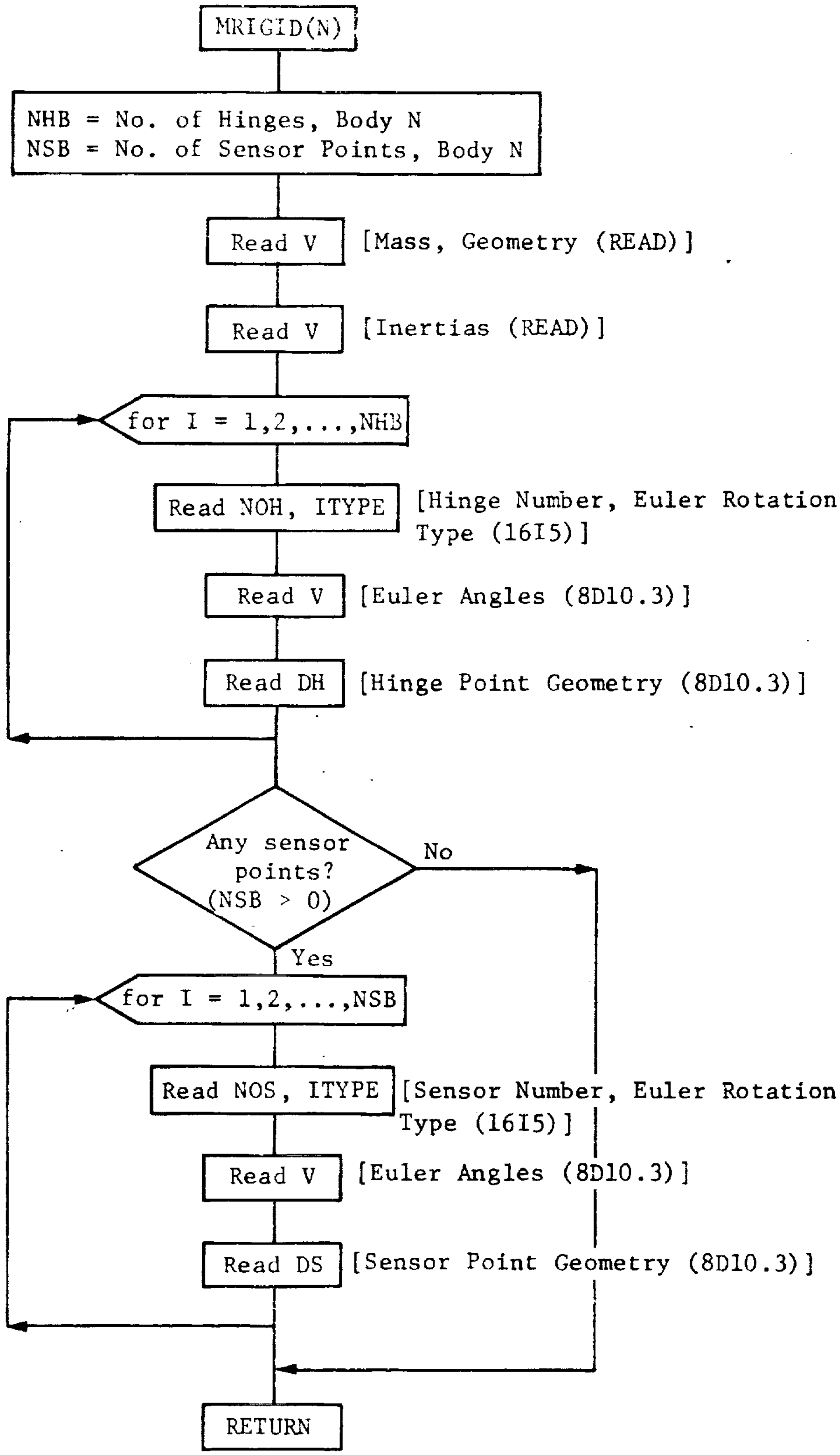


Figure IV.B-1
Program System DISCOS Data Stream Flow (Sheet 4 of 9)

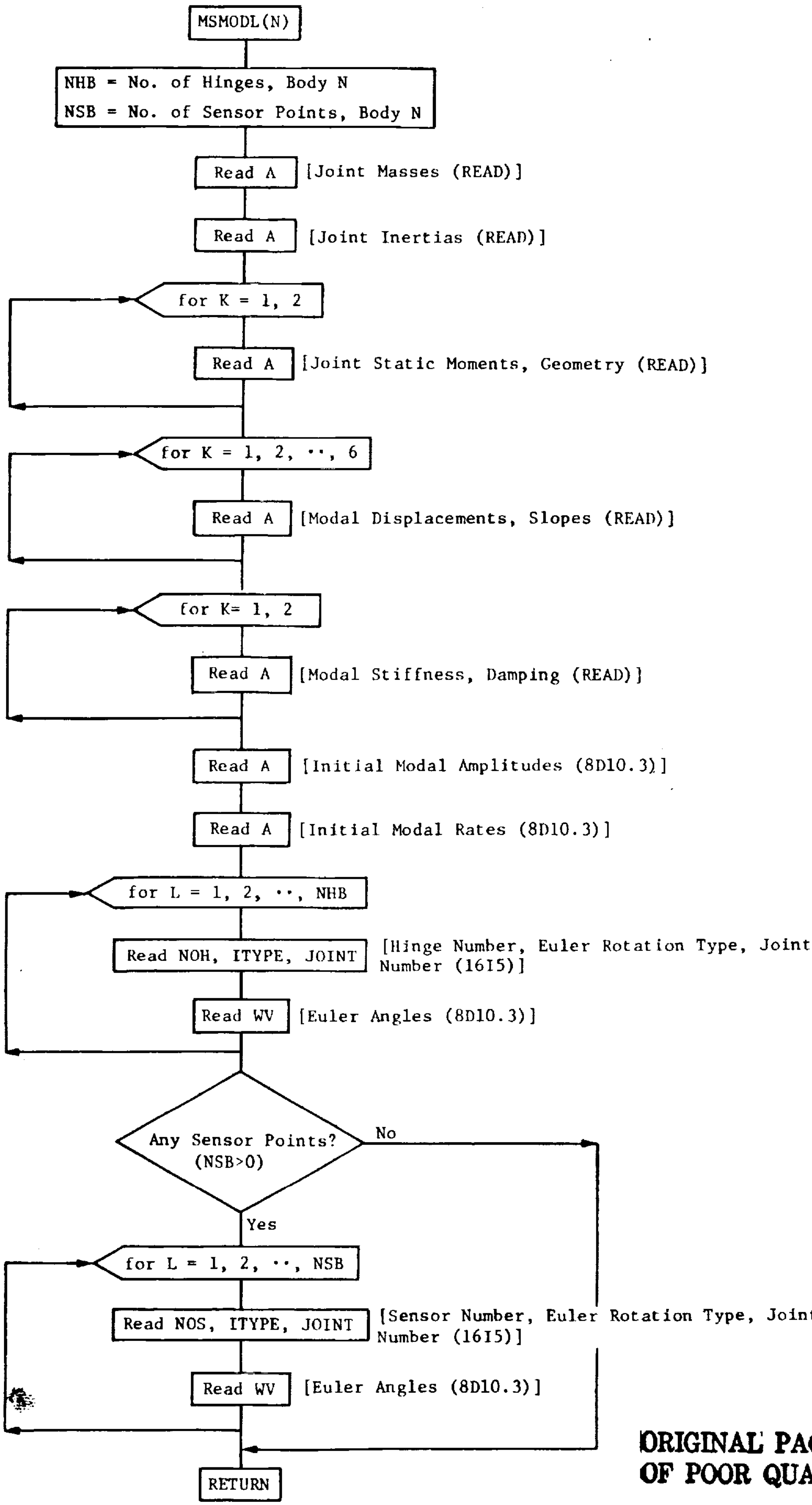


Figure IV.B-1
Program System DISCOS Data Stream Flow (Sheet 5 of 9)

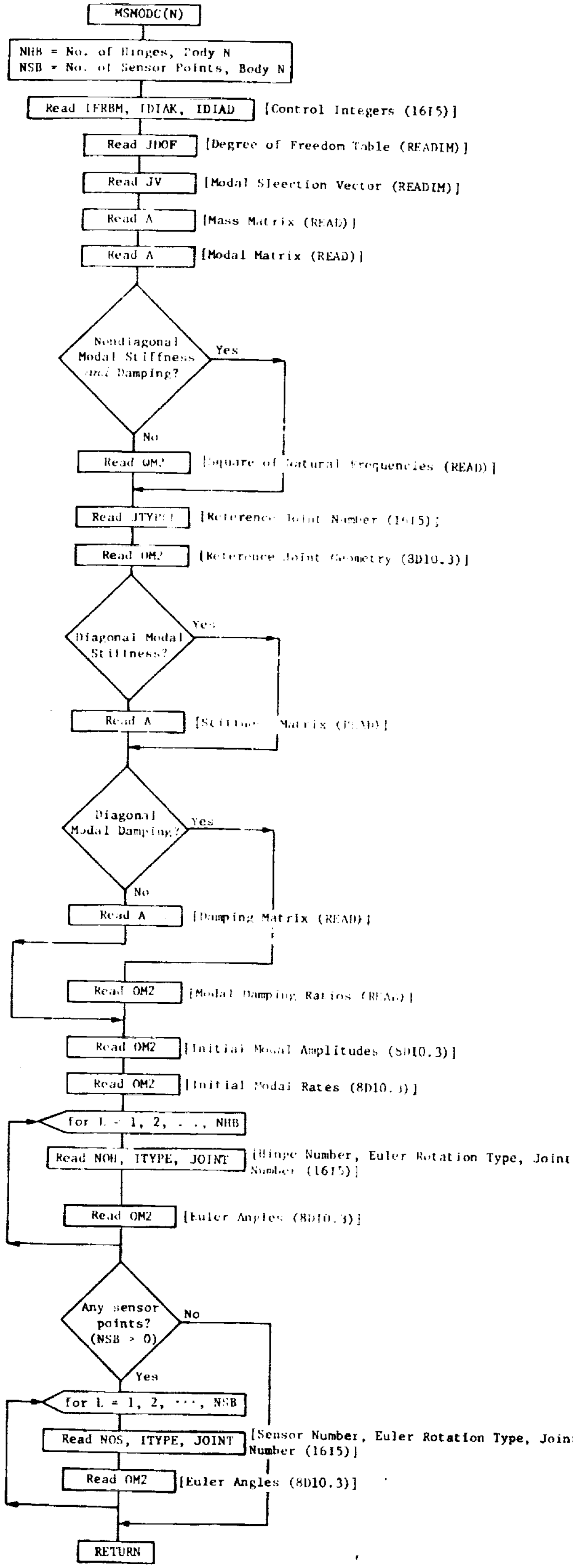


Figure [illegible] Program [illegible] DISCOS Data Stream Flow (Sheet 6 of 9)

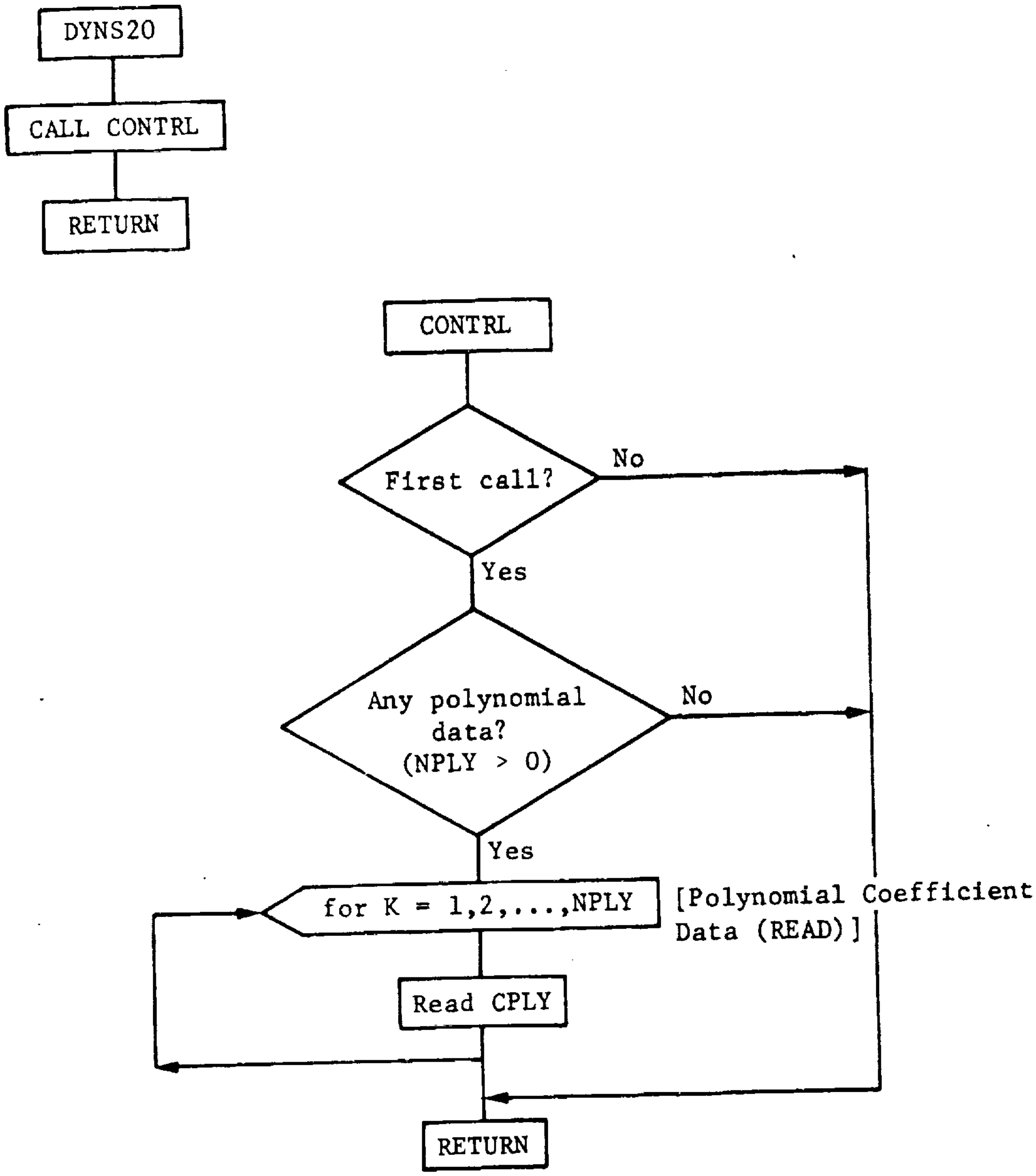


Figure IV.B-1
Program System DISCOS Data Stream Flow (Sheet 7 of 9)

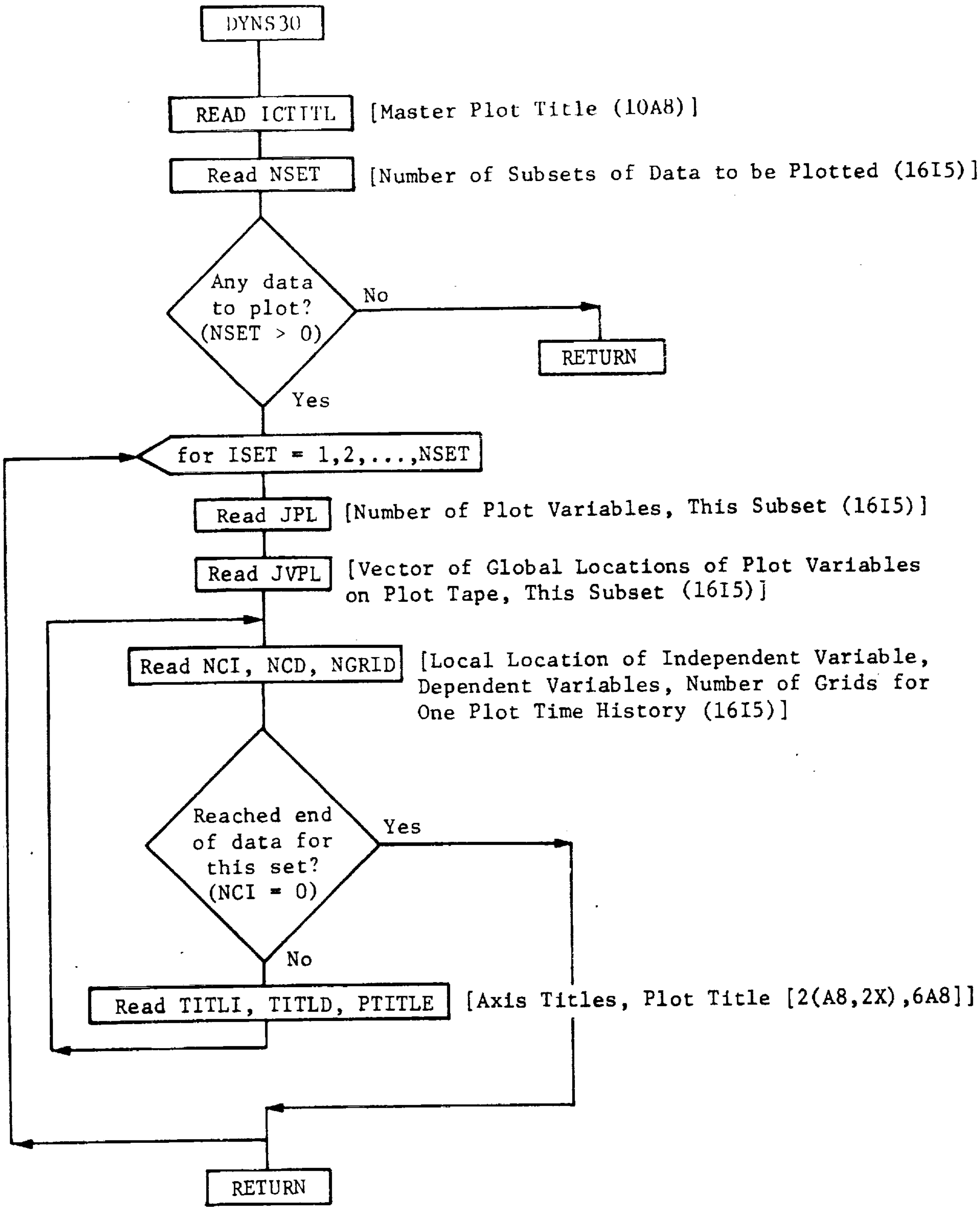


Figure IV.B-1
Program System DISCOS Data Stream Flow (Sheet 8 of 9)

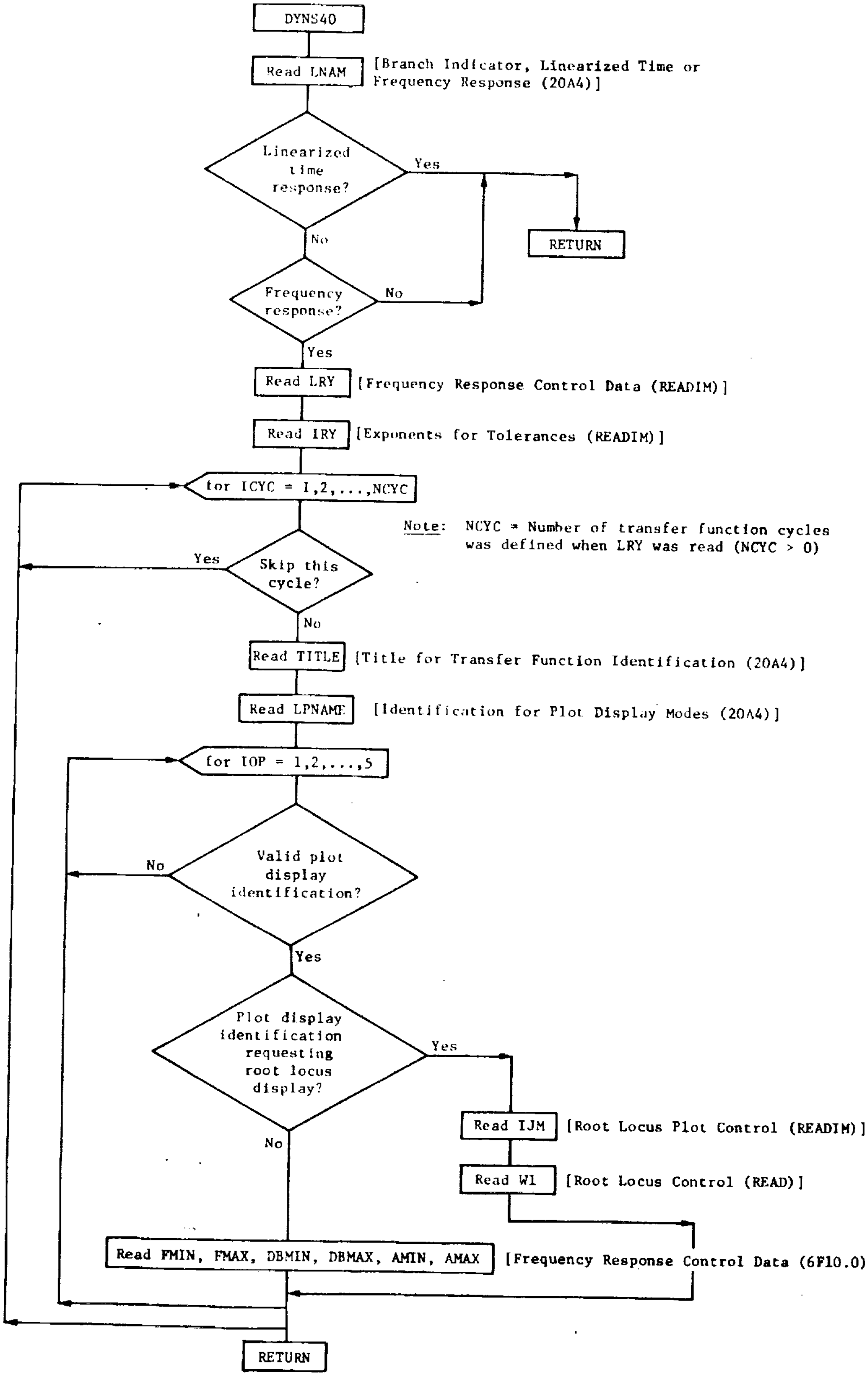


Figure IV.B-1
Program System DISCOS Data Stream Flow (Sheet 3 of 3)

```
C
C
C
C
C
C
C         *****  DISCOS PROGRAM DATA STREAM  *****
C
C                        **************
C
C
C**********************************************************************
C*
C* MAIN PROGRAM
C*
C**********************************************************************
C
C          NIT = INPUT TAPE NUMBER
C
C-------------------- DENOTES WHERE DATA
C-------------------- IS READ INTO PROGRAM
C
C
C-----999 CALL START
C
C---------CALL COMENT
C
C---------CALL DYNS10
C
C---------CALL DYNS20
C
C             IFLNER = LINEARIZATION FLAG = IPDATA(3)
C             NOPLOT = PLOT CONTROL FLAG  = IPDATA(2)
C
C---------IF (IFLNER .EQ. 1) CALL DYNS40
C
C---------IF (NOPLOT .GT. 0) CALL DYNS30
C
C          GO TO 999
C
C          END
```

```
C
C
C
C
C
C***********************************************************************
C*
C* SUBROUTINE START  - INPUT IDENTIFICATIONS
C*
C***********************************************************************
C
C
C---------READ(NIT,FORMAT = A6,4X,3A6) IRUNNO, (UNAME(I),I=1,3)
C
C                 IRUNNO = RUN IDENTIFICATION NUMBER (6 CHARACTERS)
C                 UNAME  = USERS NAME (18 CHARACTERS)
C
C                 IRUNNO EQ 4HSTOP - TERMINATE THE RUN
C                 IRUNNO NE 4HSTOP - CONTINUE THE RUN
C
C
C---------READ(NIT,FORMAT = 12A6) (TITLE1(I),I=1,12)
C---------READ(NIT,FORMAT = 12A6) (TITLE2(I),I=1,12)
C
C                 TITLE1 = 72 CHARACTER USER SUPPLIED TITLE
C                 TITLE2 = 72 CHARACTER USER SUPPLIED TITLE
C
C         RETURN
C         END
```

```
C
C
C
C
C
C***********************************************************************
C*
C* SUBROUTINE COMENT - INPUT USER SUPPLIED COMMENTS
C*
C***********************************************************************
C
C
C---------READ(NIT,FORMAT = 13A6,1X,A1) (IREMRK(I),I=1,13), IPGHD
C
C                    IREMRK = 78 CHARACTER USER SUPPLIED COMMENT
C                    IPGHD  = NEW PAGE FLAG
C
C                         IPGHD = 1HP -- NEW PAGE BEFORE PRINTING
C                         THERE IS NO LIMIT TO THE NUMBER OF COMMENT CARDS PUT
C                         LAST COMMENT CARD MUST CONTAIN ZERO IN
C                         COLUMNS 1 THRU 10
C
C         RETURN
C         END
```

```
C
C
C
C
C
C*********************************************************************
C*
C* SUBROUTINE DYNS1C - INPUT PRIMARY DATA
C*
C*********************************************************************
C
C
C----------READ(NIT,FORMAT = 5I5) NB, NH, NSPT, NCFMC, NDELTA
C
C                  NB     = NUMBER OF BODIES
C                  NH     = NUMBER OF HINGES
C                  NSPT   = NUMBER OF SENSOR POINTS
C                  NCFMC  = NUMBER OF MOMENTUM WHEELS
C                  NDELTA = NUMBER OF ADDITIONAL DIFFERENTIAL EQS
C                           REQUIRED - CONTAINS CONTROL SYSTEM
C                           VARIABLES
C
C
C----------CALL READIM (ITCPOL, 2, NH, 2, NHMAX)
C
C                  MATRIX SIZE 2 BY NH TO DESCRIBE TOPOLOGY
C
C                  FOR THE JTH HINGE -
C
C                  ITCPOL(1,J) = BODY N
C                  ITCPOL(2,J) = BODY M
C                    BODY N CONNECTED TO BODY M AT HINGE J
C                        ITCPOL(1,1) = 1 BY DEFINITION (BODY 1)
C                        ITCPOL(2,1) = 0 BY DEFINITION (INERTIAL REF)
C
C
C----------CALL READIM (IRGFLX, 1, NB, 1, NBMAX)
C
C                  VECTOR SIZE 1 BY NB TO DEFINE NUMBER OF ELASTIC MODES
C
C                  FOR THE JTH BODY -
C
C                  IRGFLX(J) = 0  - RIGID BODY
C                  IRGFLX(J) = N  - FLEXIBLE BODY WITH N MODES
C
C
C----------CALL READIM (IFTSMW, 1, NSPT, 1, NSPMAX)
C
C                  VECTOR SIZE 1 BY NSPT TO DEFINE SENSOR POINT LOCATIONS
C
C                  FOR THE JTH SENSOR POINT -
C
C                  IFTSMW(J) = BODY NUMBER FOR SENSOR POINT J
C
```

```
C
C
C
C
C
C                  *** SUMMARY OF EULER ROTATION TYPES ***
C
C
C                      ITYPE   PERMUTATION ORDER
C
C                        1         (1,2,3)
C
C                        2         (1,2,1)
C
C                        3         (1,3,1)
C
C                        4         (1,3,2)
C
C                        5         (2,3,1)
C
C                        6         (2,3,2)
C
C                        7         (2,1,2)
C
C                        8         (2,1,3)
C
C                        9         (3,1,2)
C
C                       10         (3,1,3)
C
C                       11         (3,2,3)
C
C                       12         (3,2,1)
C
C
C---------CALL READIN (IHDATA, 7, NH, 7, NHMAX)
C
C              MATRIX SIZE 7 BY NH TO DEFINE HINGE POINT
C              CONSTRAINT DATA
C
C              FOR THE JTH HINGE -
C
C              IHDATA(1,J) = EULER ROTATION TYPE TO ORIENT Q-TRIAD
C                            WRT TO P-TRIAD AT HINGE J (ITYPE)
C              IHDATA(2,J) = HINGE CONSTRAINT TYPE - THETA 1 ROTATION
C              IHDATA(3,J) = HINGE CONSTRAINT TYPE - THETA 2 ROTATION
C              IHDATA(4,J) = HINGE CONSTRAINT TYPE - THETA 3 ROTATION
C              IHDATA(5,J) = HINGE CONSTRAINT TYPE - X TRANSLATION
C              IHDATA(6,J) = HINGE CONSTRAINT TYPE - Y TRANSLATION
C              IHDATA(7,J) = HINGE CONSTRAINT TYPE - Z TRANSLATION
C
C                ROTATIONS REFER TO ORDER DEFINED BY ITYPE
C
C                TRANSLATIONS REFER TO P-TRIAD
C
C                IHDATA(2-7,J) = 0 - NO CONSTRAINT
C                IHDATA(2-7,J) = 1 - FIXED CONSTRAINT
C                IHDATA(2-7,J) = 2 - RHEONOMIC CONSTRAINT
C
C
```

```
C
C
C
C
C
C               NOTE -- NUMBER OF BETA STATE VARIABLES COMPUTED FROM
C                       IHDATA AS SUM OF NUMBER OF ZEROS + SUM OF
C                       NUMBER OF TWOS IN ROWS 2 THRU 7
C
C                       NUMBER OF CONSTRAINTS COMPUTED FROM
C                       IHDATA AS SUM OF NUMBER OF ONES + SUM OF
C                       NUMBER OF TWOS IN ROWS 2 THRU 7
C
C
C---------CALL READ (BETAH, 6, NH, 6, NHMAX)
C
C               MATRIX SIZE 6 BY NH TO DEFINE INITIAL VALUES
C               OF BETA WHICH ORIENT TWO BODIES ASSOCIATED
C               WITH EACH HINGE
C
C               FOR THE JTH HINGE -
C
C               BETAH(1,J) = THETA 1 ROTATION
C               BETAH(2,J) = THETA 2 ROTATION
C               BETAH(3,J) = THETA 3 ROTATION
C               BETAH(4,J) = X TRANSLATION
C               BETAH(5,J) = Y TRANSLATION
C               BETAH(6,J) = Z TRANSLATION
C
C
C---------CALL READ (BETAHD, 6, NH, 6, NHMAX)
C
C               MATRIX SIZE 6 BY NH TO DEFINE INITIAL VALUES
C               OF BETA DOT - TIME DERIVATIVE OF BETAH
C               DESCRIBED PREVIOUSLY
C
C               FOR THE JTH HINGE -
C
C               BETAHD(1,J) = THETA 1 ROTATION RATE
C               BETAHD(2,J) = THETA 2 ROTATION RATE
C               BETAHD(3,J) = THETA 3 ROTATION RATE
C               BETAHD(4,J) = X TRANSLATION RATE
C               BETAHD(5,J) = Y TRANSLATION RATE
C               BETAHD(6,J) = Z TRANSLATION RATE
C
C
C               NOTE -- IF THE CORRESPONDING CONSTRAINT TYPE
C                       IS 1 OR 2, THE INITIAL BETAHD(1-6,J),
C                       INPUT HERE, WILL BE IGNORED AND SET TO
C                       ZERO OR TO THE RHEONOMICALLY PRESCRIBED
C                       VALUE, RESPECTIVELY
C
C
C
```

```
C
C
C
C
C
C      NOFMO = NUMBER OF MOMENTUM WHEELS
C
C           IF (NOFMO .EQ. 0) GO TO 41
C
C
C----------CALL READIM (IMO, 3, NOFMO, 3, NMWMAX)
C
C                    MATRIX SIZE 3 BY NOFMO TO DEFINE BASIC
C                    MOMENTUM WHEEL DATA
C
C                    FOR THE JTH MOMENTUM WHEEL -
C
C                    IMO(1,J) = SENSOR POINT NUMBER FOR THE WHEEL
C                    IMO(2,J) = SPIN AXIS NUMBER FOR THE WHEEL (1,2 OR 3)
C                    IMO(3,J) = 1 ACTIVE WHEEL
C                             = 0 CONSTANT SPEED WHEEL
C
C
C----------CALL READ (AMO, 2, NOFMO, 2, NMWMAX)
C
C                    MATRIX SIZE 2 BY NOFMO TO DEFINE MOMENTUM
C                    WHEEL DATA
C
C                    FOR THE JTH MOMENTUM WHEEL -
C
C                    AMO(1,J) = INITIAL WHEEL SPIN RATE
C                    AMO(2,J) = SPIN INERTIA
C
C      41 CONTINUE
C
C
C----------CALL READ (TMDATA, 1, 3, 1, 3)
C
C                    VECTOR SIZE 1 BY 3 CONTAINING TIME HISTORY
C                    INTEGRATION CONTROLS
C
C                    TMDATA(1) = INITIAL TIME      (STARTT)
C                    TMDATA(2) = TIME INTERVAL     (DELTAT)
C                    TMDATA(3) = TERMINATION TIME (ENDT  )
C
C
```

```
C
C
C
C
C
C---------READIM (IPDATA, 1, 3, 1, 3)
C
C                VECTOR SIZE 1 BY 3 CONTAINING INTEGER
C                CONTROL DATA
C
C                IPDATA(1) = PRINT CONTROL FOR TIME RESPONSE - PRINT
C                            EVERY IPDATA(1) MULTIPLES OF DELTAT
C                IPDATA(2) = PLOT CONTROL FOR TIME RESPONSE - SAVE
C                            EVERY IPDATA(2) MULTIPLES OF DELTAT
C                IPDATA(3) = 0 PERFORM NONLINEAR TIME RESPONSE
C                          = 1 PERFORM LINEAR TIME OR FREQUENCY
C                              RESPONSE
C
C
C---------CALL READ (CNTDTA, 1, NCNPAR, 1, KCONT)
C
C                VECTOR SIZE 1 BY NCNPAR FOR CONTROL SYSTEM VARIABLES
C                AND USER SUPPLIED VARIABLES
C
C                THIS IS A CATCH-ALL VECTOR FOR USER PAK DATA
C
C                INFORMATION IS PUT INTO COMMON /CCNTRL/ AND IS
C                IDENTIFIED BY USER SUPPLIED EQUIVALENCE MAP
C                IN USER PAK
C
C                FIRST NDELTA ELEMENTS MUST CONTAIN INITIAL VALUES
C                FOR DELTA VARIABLES IN STATE VECTOR
C
C                ADDITIONAL SPACE IS AVAILABLE TO THE USER
C
C
C---------CALL READ (WV, 1, 4, 1, 5)
C
C                VECTOR SIZE 1 BY 4 FOR GRAVITY GRADIENT DATA
C
C                WV(1) = PROJECTION OF GRAVITY VECTOR ON X INERTIAL AXIS
C                WV(2) = PROJECTION OF GRAVITY VECTOR ON Y INERTIAL AXIS
C                WV(3) = PROJECTION OF GRAVITY VECTOR ON Z INERTIAL AXIS
C                WV(4) = RADIUS VECTOR FROM GRAVITY SOURCE TO GENERAL
C                        VICINITY OF BODY CLUSTER
C
C
C                NOTE -- IF(WV(1)**2 + WV(2)**2 + WV(3)**2)
C                                  EQ 0, WV(4) IS IGNORED
C                                  NE 0, WV(4) MUST BE GT 1
C
C                NOTE -- GRAVITY VECTOR COMPONENTS IN
C                        UNITS OF ACCELERATION
```

```
C
C
C
C
C
C
C     NB = NUMBER OF BODIES
C
C          DO 20 N=1,NB
C
C                IRGFLX(N) EQ 0 -  RIGID BODY
C                IRGFLX(N) GT 0 -  FLEXIBLE BODY
C
C          IF (IRGFLX(N) .EQ. 0) GO TO 25
C
C----------READ(NIT,FORMAT = I5) NTYPE
C
C                NTYPE = 1 - LUMPED MASS REPRESENTATION
C                NTYPE = 2 - CONSISTENT MASS REPRESENTATION
C
C          IF (NTYPE .EQ. 1) CALL MSMCOL(N)    --- SEE FOLLOWING
C
C          IF (NTYPE .EQ. 2) CALL MSMCDC(N)    --- SEE FOLLOWING
C
C          GO TO 20
C
C       25 CALL MRIGID(N)    --- SEE FOLLOWING
C
C       20 CONTINUE
C
C          RETURN
C          END
```

```
C
C
C
C
C
C*****************************************************************************
C*
C* SUBROUTINE MRIGID - INPUT FOR RIGID BODY
C*
C*****************************************************************************
C
C
C---------CALL READ (V, 1, 4, 1, 6)
C
C              VECTOR SIZE 1 BY 4
C
C              V(1) = MASS OF BODY
C              V(2) = X (BODY REF POINT TO BODY CG, BODY TRIAD)
C              V(3) = Y (BODY REF POINT TO BODY CG, BODY TRIAD)
C              V(4) = Z (BODY REF POINT TO BODY CG, BODY TRIAD)
C
C
C---------CALL READ (V, 1, 6, 1, 6)
C
C              VECTOR SIZE 1 BY 6
C
C              V(1) = JXX
C              V(2) = JYY
C              V(3) = JZZ (BODY REFERENCED INERTIAS - BODY TRIAD)
C              V(4) = JXY      (INERTIAS ABOUT REF POINT, NOT BODY CG)
C              V(5) = JXZ
C              V(6) = JYZ
C
C        NHB = NUMBER OF HINGES ON BODY N - EXCLUSIVE OF HINGE 1, BODY 1
C
C        DO 10 I=1,NHB
C
C
C---------READ(NIT,FORMAT = 2I5) NOH, ITYPE
C
C              NOH   = HINGE NUMBER
C              ITYPE = EULER ROTATION TYPE TO ORIENT HINGE
C                      TRIAD WRT BODY TRIAD
C
C
```

NOTE -- $J_{xy} = \int_{Vol} xy\, dm$, ect.

```
C
C
C
C
C
C---------READ(NIT,FORMAT = 3D10.3)  (V(J),J=1,3)
C
C                   EULER ANGLES TO ORIENT HINGE TRIAD - PERMUTATION
C                   ORDER DEFINED BY ITYPE
C
C                   V(1) = THETA 1 (FIRST ROTATION)
C                   V(2) = THETA 2 (SECND ROTATION)
C                   V(3) = THETA 3 (THIRD ROTATION)
C
C
C---------READ(NIT,FORMAT = 3D10.3)  (DH(J),J=1,3)
C
C                   VECTOR TO POSITION HINGE TRIAD WRT BODY TRIAD
C
C                   DH(1) = X (BODY REF POINT TO HINGE POINT, BODY TRIAD)
C                   DH(2) = Y (BODY REF POINT TO HINGE POINT, BODY TRIAD)
C                   DH(3) = Z (BODY REF POINT TO HINGE POINT, BODY TRIAD)
C
C       10 CONTINUE
C
C          NSP = NUMBER OF SENSOR POINTS ON BODY N
C
C          IF (NSP .EQ. 0) RETURN
C
C          DO 20 I=1,NSP
C
C
C---------READ(NIT,FORMAT = 2I5) NOS, ITYPE
C
C                   NOS   = SENSOR POINT NUMBER
C                   ITYPE = EULER ROTATION TYPE TO ORIENT SENSOR POINT
C                           TRIAD WRT BODY TRIAD
C
C
C---------READ(NIT,FORMAT = 3D10.3)  (V(J),J=1,3)
C
C                   EULER ANGLES TO ORIENT SENSOR POINT TRIAD - PERMUTATION
C                   ORDER DEFINED BY ITYPE
C
C                   V(1) = THETA 1 (FIRST ROTATION)
C                   V(2) = THETA 2 (SECND ROTATION)
C                   V(3) = THETA 3 (THIRD ROTATION)
C
C
```

```
C
C
C
C
C
C
C---------READ(NIT,FORMAT = 3D10.3) (DS(J),J=1,3)
C
C                  VECTOR TO POSITION SENSOR POINT TRIAD WRT BODY TRIAD
C
C                  DH(1) = X (BODY REF POINT TO SENSOR POINT, BODY TRIAD)
C                  DH(2) = Y (BODY REF POINT TO SENSOR POINT, BODY TRIAD)
C                  DH(3) = Z (BODY REF POINT TO SENSOR POINT, BODY TRIAD)
C
C      20 CONTINUE
C
C         RETURN
C         END
```

```
C
C
C
C
C
C*************************************************************************
C*
C* SUBROUTINE MSMODL - INPUT FOR FLEXIBLE BODY, LUMPED MASS MATRIX
C*
C*************************************************************************
C
C
C---------CALL READ (A, NJ, 1, KJOINT, KMODE)
C
C                   MATRIX SIZE NJ BY 1 WHERE NJ = NUMBER OF JOINTS
C                   ON BODY N
C
C                   FOR THE ITH JOINT -
C
C                   A(I,1) = JOINT MASS
C
C
C---------CALL READ (A, NJ, 6, KJOINT, KMODE)
C
C                   MATRIX SIZE NJ BY 6
C
C                   FOR THE ITH JOINT -
C
C                   A(I,1) = JOINT INERTIA, JXX
C                   A(I,2) = JOINT INERTIA, JYY
C                   A(I,3) = JOINT INERTIA, JZZ
C                   A(I,4) = JOINT INERTIA, JXY
C                   A(I,5) = JOINT INERTIA, JXZ
C                   A(I,6) = JOINT INERTIA, JYZ
C
C
C         DO 5 K=1,2
C
C
C---------CALL READ (A, NJ, 3, KJOINT, KMODE)
C
C                   MATRIX SIZE NJ BY 3
C
C                   FOR THE ITH JOINT -
C
C              K=1  A(I,1) = JOINT STATIC MASS MOMENT, SX
C                   A(I,2) = JOINT STATIC MASS MOMENT, SY
C                   A(I,3) = JOINT STATIC MASS MOMENT, SZ
C
C              K=2  A(I,1) = X (BODY REF POINT TO JOINT, BODY TRIAD)
C                   A(I,2) = Y (BODY REF POINT TO JOINT, BODY TRIAD)
C                   A(I,3) = Z (BODY REF POINT TO JOINT, BODY TRIAD)
C
C       5 CONTINUE
C
C
C
```

NOTE -- $J_{xy} = \int_{Vol} xy \, dm$, ect.

```
C
C
C
C
C
C         DO 10 K=1,6
C
C
C---------CALL READ (A, NJ, NE, KJOINT, KMODE)
C
C                   MATRIX SIZE NJ BY NE WHERE NE = NUMBER OF ELASTIC
C                   MODES RETAINED FOR BODY N
C
C                   FOR THE ITH JOINT -
C
C              K=1  A(I,J) = X DISPLACEMENT AT JOINT, MODE J
C              K=2  A(I,J) = Y DISPLACEMENT AT JOINT, MODE J
C              K=3  A(I,J) = Z DISPLACEMENT AT JOINT, MODE J
C              K=4  A(I,J) = THETA X ROTATION AT JOINT, MODE J
C              K=5  A(I,J) = THETA Y ROTATION AT JOINT, MODE J
C              K=6  A(I,J) = THETA Z ROTATION AT JOINT, MODE J
C
C      10 CONTINUE
C
C
C         DO 20 K=1,2
C
C
C---------CALL READ (A, NE, NE, KJOINT, KMODE)
C
C                   MATRIX SIZE NE BY NE
C
C              K=1  A = MODAL STIFFNESS
C
C              K=2  A = MODAL DAMPING
C
C      20 CONTINUE
C
C
C---------READ(NIT,FORMAT = 8D10.3) (A(J),J=1,NE)
C
C                   VECTOR OF INITIAL MODAL DEFLECTION COORDINATES
C
C---------READ(NIT,FORMAT = 8D10.3) (A(J),J=1,NE)
C
C                   VECTOR OF INITIAL MODAL VELOCITY COORDINATES
C
C
C
```

```
C                NCTE -- FCLLOWING EULER ANGLES MEASURED
C                        IN UNDEFORMED CONFIGURATION
C
C           NHB = NUMBER CF HINGES CN BODY N - EXCLUSIVE CF HINGE 1, BODY 1
C
C           DO 150 L=1,NHB
C
C----------READ(NIT,FORMAT = 3I5) NOH, ITYPE, JCINT
C
C                NOH   = HINGE NUMBER
C                ITYPE = EULER ROTATICN TYPE TO ORIENT HINGE
C                        TRIAD WRT BODY TRIAD
C                JCINT = JCINT NUMBER CORRESPONDING TC HINGE POINT
C
C----------READ(NIT,FORMAT = 3D10.3) (HV(J),J=1,3)
C
C                EULER ANGLES TO CRIENT HINGE TRIAD - PERMUTATION
C                ORDER DEFINED BY ITYPE
C
C                HV(1) = THETA 1 (FIRST ROTATION)
C                HV(2) = THETA 2 (SECND RCTATICN)
C                HV(3) = THETA 3 (THIRD ROTATION)
C
C       150 CCNTINUE
C
C           NSB = NUMBER OF SENSOR POINTS CN BCDY N
C
C           IF (NSB .EQ. 0) RETURN
C
C           DO 160 L=1,NSB
C
C----------READ(NIT,FORMAT = 3I5) NOS, ITYPE, JCINT
C
C                NOS   = SENSOR POINT NUMBER
C                ITYPE = EULER ROTATICN TYPE TO CRIENT SENSCR POINT
C                        TRIAD WRT BODY TRIAD
C                JCINT = JCINT NUMBER CORRESPCNDING TC SENSCR POINT
C
C
C
```

```
C
C
C
C
C
C---------READ(NIT,FORMAT = 3E10.3)  (WV(J),J=1,3)
C
C                EULER ANGLES TO ORIENT SENSOR POINT TRIAD - PERMUTATIO
C                ORDER DEFINED BY ITYPE
C
C                WV(1) = THETA 1 (FIRST ROTATION)
C                WV(2) = THETA 2 (SECND ROTATION)
C                WV(3) = THETA 3 (THIRD ROTATION)
C
C     160 CONTINUE
C
C         RETURN
C         END
```

```
C
C
C
C
C
C*************************************************************************
C*
C* SUBROUTINE MSMODC - INPUT FOR FLEXIBLE BODY, CONSISTENT MASS MATRIX
C*
C*************************************************************************
C
C
C---------READ(NIT,FORMAT = 3I5) IFORM, IDIAK, IDIAD
C
C                       IFORM = RIGID BODY MODE CONTROL VARIABLE
C
C                             = 0  WHEN RIGID BODY MODES CAN BE
C                                  OPERATED ON VIA ROW-COL
C                                  INTERCHANGES TO OBTAIN A RIGID
C                                  BODY MODAL MATRIX OF THE FORM-
C
C                           **   **     **                       ** **     **
C                           * HX *      * 1   0   0   0   Z  -Y  * * CX    *
C                           *    *      *                        * *       *
C                           * HY *      * 0   1   0  -Z   0   X  * * CY    *
C                           *    *      *                        * *       *
C                           * HZ *      * 0   0   1   Y  -X   0  * * CZ    *
C                           *    *  =   *                        * *       *
C                           * TX *      * 0   0   0   1   0   0  * * CTX   *
C                           *    *      *                        * *       *
C                           * TY *      * 0   0   0   0   1   0  * * CTY   *
C                           *    *      *                        * *       *
C                           * TZ *      * 0   0   0   0   0   1  * * CTZ   *
C                           **   **     **                       ** **     **
C
C
C                             = 1  WHEN ABOVE IS NOT TRUE, SUCH
C                                  AS FOR AN ORTHONORMAL SET OF
C                                  RIGID BODY  VECTORS. DATA WILL
C                                  BE READ IN LATER AND THE ABOVE
C                                  FORM WILL BE CREATED WITHIN
C                                  THIS SUBROUTINE
C
C
```

```
C
C
C
C
C
C       IDIAK = 0  IF MODAL STIFFNESS MATRIX
C                  IS NOT DIAGONAL
C
C             = 1  IF MODAL STIFFNESS MATRIX
C                  IS DIAGONAL
C
C
C       IDIAD = 0  IF MODAL DAMPING MATRIX
C                  IS NOT DIAGONAL
C
C             = 1  IF MODAL DAMPING MATRIX
C                  IS DIAGONAL
C
C
C---------CALL READIM (JDOF, NX, N6, KJDOF, 6)
C
C          MATRIX SIZE NX BY 6 DEFINING JOINT DEGREES
C          OF FREEDOM FOR NX JOINTS ON BODY N
C
C          FOR THE ITH JOINT -
C
C           JDOF(I,1) = X DEG. OF FREEDOM FOR POINT I.
C           JDOF(I,2) = Y DEG. OF FREEDOM FOR POINT I.
C           JDOF(I,3) = Z DEG. OF FREEDOM FOR POINT I.
C           JDOF(I,4) =TX DEG. OF FREEDOM FOR POINT I.
C           JDOF(I,5) =TY DEG. OF FREEDOM FOR POINT I.
C           JDOF(I,6) =TZ DEG. OF FREEDOM FOR POINT I.
C
C                 POINT I IS CONSISTANT WITH THE
C                 MODAL MATRIX ROW I, BEFORE REORDING.
C
C                 JDOF WILL BE INTERROGATED AND THE
C                 COORDINATES WILL BE ORDERED AS --
C
C                   HX(I) FOR JOINTS 1 THRU NX
C                   HY(I) FOR JOINTS 1 THRU NX
C                   HZ(I) FOR JOINTS 1 THRU NX
C                   TX(I) FOR JOINTS 1 THRU NX
C                   TY(I) FOR JOINTS 1 THRU NX
C                   TZ(I) FOR JOINTS 1 THRU NX
C
C
```

```
C
C
C
C
C
C---------CALL READIM (JV, 1, NMODT, 1, KAR)
C
C                  VECTOR SIZE 1 BY NMODT = NUMBER OF RIGID BODY MODES +
C                                            NUMBER OF ELASTIC MODES
C
C                  JV(J) = IND    WHERE IND IS COLUMN NO. WHICH COL(J) OF ORIG
C                                 MODAL MATRIX WILL APPEAR IN
C                                 REVISED MODE MATRIX
C                          IND GT 0   REPLACE COLUMN
C                          IND EQ 0   DELETE COLUMN
C                          IND LT 0   REPLACE COLUMN, CHANGE SIGNS
C
C
C---------CALL READ (A, NRA, NCA, KAR, KAR)
C
C                  MATRIX SIZE 6*NX BY 6*NX CONTAINS CONSISTENT
C                  MASS REPRESENTATION
C
C                  ROW-COLUMN COORDINATE ORDER MUST BE CONSISTENT
C                  WITH THE DEGREE OF FREEDOM TABLE, JDOF
C
C
C---------CALL READ (A, NRA, NMODT, KAR, KAR)
C
C                  MATRIX SIZE 6*NX BY NMODT CONTAINS MODAL DEFINITION
C
C                  THE COLUMNS (MODE ORDER) WILL BE REORDERED BY
C                  THE INPUT VECTOR JV
C
C
C          IF (IDIAK .EQ. 0  .AND.  IDIAR .EQ. 0) GO TO 11
C
C---------CALL READ (OM2, 1, NMODT, 1, KAR)
C
C                  VECTOR SIZE 1 BY NMODT CONTAINING SQUARES OF NATURAL
C                  FREQUENCIES
C
C                  OM2(J) = SQUARE OF JTH NATURAL FREQUENCY CORRESPONDING
C                  TO JTH INPUT MODE SHAPE
C
C      11 CONTINUE
C
C
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
C
C
C
C
C
C         IF (IFRAM .EQ. 0) GO TO 5
C
C
C---------READ(NIT,FORMAT = I5) JTYPCL
C
C                  JTYPCL = REFERENCE JOINT NUMBER WHOSE GEOMETRIC
C                           POSITION COORDINATES WILL BE USED TO
C                           ESTABLISH RIGID BODY MODAL MATRIX
C
C---------READ(NIT,FORMAT = 3D10.3) (OM2(J),J=1,3)
C
C                  OM2(1) = X COMPONENT OF VECTOR THAT LOCATES JTYPCL PT
C                  OM2(2) = Y COMPONENT OF VECTOR THAT LOCATES JTYPCL PT
C                  OM2(3) = Z COMPONENT OF VECTOR THAT LOCATES JTYPCL PT
C
C                  VECTOR COMPONENTS REFERED TO BODY TRIAD
C
C        5 CONTINUE
C
C
C         IF (IDIAK .EQ. 1) GO TO 50
C
C---------CALL READ (A, NRA, NCA, KAB, KAB)
C
C                  STIFFNESS MATRIX -- SEE NOTE BELOW
C
C       50 CONTINUE
C
C         IF (IDIAD .EQ. 1) GO TO 60
C
C
C---------CALL READ (A, NRA, NCA, KAB, KAB)
C
C                  DAMPING MATRIX -- SEE NOTE BELOW
C
C         GO TO 61
C
C       60 CONTINUE
C
C
C---------READ(NIT,FORMAT = 8D10.3) (OM2(J),J=1,NE)
C
C                  VECTOR SIZE 1 BY NE = NUMBER OF ELASTIC MODES
C                  RETAINED VIA INPUT JV SELECTION VECTOR
C
C                  OM2(J) = MODAL DAMPING RATIO FOR JTH ELASTIC MODE
C
C       61 CONTINUE
C
C
```

```
C
C
C
C
C
C                 NOTE FOR STIFFNESS AND DAMPING MATRICES
C
C                    COORDINATE ORDER ASSUMED CONSISTENT WITH
C                        THE REORDERED COORDINATE DESCRIPTION
C                        AFTER THE DEGREE OF FREEDOM TABLE (JDOF)
C                        HAS BEEN INTERROGATED AND THE FOLLOWING
C                        ORDER ESTABLISHED
C
C
C                            HX(I),I=1,2,...,N
C                             .
C                             .
C                            HY(I),I=1,2,...,N
C                             .
C                             .
C                            HZ(I),I=1,2,...,N
C                             .
C                             .
C                            TX(I),I=1,2,...,N
C                             .
C                             .
C                            TY(I),I=1,2,...,N
C                             .
C                             .
C                            TZ(I),I=1,2,...,N
C                             .
C                             .
C
C---------READ(NIT,FORMAT = 8D10.3)  (OM2(J),J=1,NE)
C
C                VECTOR SIZE 1 BY NE CONTAINING INITIAL
C                MODAL DEFLECTION COORDINATES
C
C
C---------READ(NIT,FORMAT = 8D10.3)  (OM2(J),J=1,NE)
C
C                VECTOR SIZE 1 BY NE CONTAINING INITIAL
C                MODAL VELOCITY COORDINATES
C
C
```

```
C
C
C
C
C
C                 NOTE -- FOLLOWING EULER ANGLES MEASURED
C                         IN UNDEFORMED CONFIGURATION
C
C
C         NHB = NUMBER OF HINGES ON BODY N - EXCLUSIVE OF HINGE 1, BODY 1
C
C         DO 110 L=1,NHB
C
C
C---------READ(NIT,FORMAT = 3I5) NOH, ITYPE, JOINT
C
C                 NOH   = HINGE NUMBER
C                 ITYPE = EULER ROTATION TYPE TO ORIENT HINGE
C                         TRIAD WRT BODY TRIAD
C                 JOINT = JOINT NUMBER CORRESPONDING TO HINGE POINT
C
C
C---------READ(NIT,FORMAT = 3D10.3) (OM2(J),J=1,3)
C
C                 EULER ANGLES TO ORIENT HINGE TRIAD - PERMUTATION
C                 ORDER DEFINED BY ITYPE
C
C                 OM2(1) = THETA 1 (FIRST ROTATION)
C                 OM2(2) = THETA 2 (SECOND ROTATION)
C                 OM2(3) = THETA 3 (THIRD ROTATION)
C
C      110 CONTINUE
C
C
C         NSB = NUMBER OF SENSOR POINTS ON BODY N
C
C         IF (NSB .EQ. 0) RETURN
C
C         DO 120 L=1,NSB
C
C
C---------READ(NIT,FORMAT = 3I5) NOS, ITYPE, JOINT
C
C                 NOS   = SENSOR POINT NUMBER
C                 ITYPE = EULER ROTATION TYPE TO ORIENT SENSOR POINT
C                         TRIAD WRT BODY TRIAD
C                 JOINT = JOINT NUMBER CORRESPONDING TO SENSOR POINT
C
C
```

```
C
C
C
C
C
C
C
C---------READ(NIT,FORMAT = 3D10.3) (OM2(J),J=1,3)
C
C                    EULER ANGLES TO ORIENT SENSOR POINT TRIAD - PERMUTATION
C                    ORDER DEFINED BY ITYPE
C
C                    OM2(1) = THETA 1 (FIRST ROTATION)
C                    OM2(2) = THETA 2 (SECOND ROTATION)
C                    OM2(3) = THETA 3 (THIRD ROTATION)
C
C      120 CONTINUE
C
C          RETURN
C          END
```

```
C
C
C
C
C
C*********************************************************************
C*
C* SUBROUTINE DYNS2C - INPUT USER PAK DATA (IF ANY)
C*
C*********************************************************************
C
C
C                    THE ONLY DATA READ HERE ARE THOSE ASSOCIATED
C                    WITH POLYNOMIAL DEFINITION OF TRANSFER
C                    FUNCTIONS
C
C                    DATA (IF ANY) IS READ ON THE FIRST CALL TO
C                    SUBROUTINE CONTRL WHERE THE INTEGER VARIABLE
C                    NPLY IS DEFINED BY A USER SUPPLIED DATA
C                    STATEMENT
C
C                    NPLY = NUMBER OF SETS OF POLYNOMIAL TRANSFER
C                           FUNCTION COEFFICIENTS TO READ
C
C           IF (NPLY .EQ. 0) GO TO 10
C
C           DO 20 K=1,NPLY
C
C           K2 = 2*K-1
C
C
C---------CALL READ (CPLY(1,K2), KPLY(K), 2, KPY, KCY)
C
C       20 CONTINUE
C
C                    MATRIX SIZE KPLY(K) BY 2 OF POLYNOMIAL COEFFICIENTS
C
C                    CPLY(I,1) = DENOMINATOR COEFFICIENTS IN
C                                ASCENDING ORDER
C                    CPLY(I,2) = NUMERATOR COEFFICIENTS IN
C                                ASCENDING ORDER
C
C                    NOTE -- KPLY(K) IS DEFINED AT INPUT (SUBROUTINE READ)
C                            AND IS ONE GREATER THAN THE ORDER OF THE
C                            DENOMINATOR POLYNOMIAL
C
C                            ORDER OF NUMERATOR POLYNOMIAL MUST NOT EXCEED
C                            ORDER OF DENOMINATOR POLYNOMIAL
C
C       10 CONTINUE
C
C           RETURN
C           END
```

```
C
C
C
C
C
C*****************************************************************************
C*
C* SUBRCUTINE DYNS30 - INPUT FOR TIME HISTCRY PLOT OUTPUT
C*
C*****************************************************************************
C
C
C---------READ(NIT,FORMAT = 10A8) (ICTITL(I),I=1,10)
C
C                 80 CHARACTER MASTER PLOT TITLE WILL APPEAR
C                 ON ALL FRAMES
C
C
C---------READ(NIT,FORMAT = I5) NSET
C
C                              NSET = NO. OF PLOT SETS TO BE
C                                     CYCLED THROUGH. EACH SET
C                                     IS LIMITED TO 16 SEPARATE
C                                     VARIABLES TO BE SELECTED
C                                     FROM THE SET OF VARIABLES
C                                     WRITTEN BE SUBROUTINE PLOTWR
C                                     OR LPLTWR.
C
C
C         DO 1000 ISET=1,NSET
C
C
C---------READ(NIT,FORMAT = I5) JPL
C
C                              JPL = NO. OF VARIABLES TO BE
C                                    SELECTED FROM THE NOPLOT
C                                    VARIABLES PREVIOUSLY WRITTEN
C                                    ON NTAPE3 FOR PLOTTING.
C                                   (JPL .LE. 16) FOR A SET
C
C                              NOPLOT = 1+2*NEQ+NLAM +NU+JR+3+3+2*NB+5
C                                       (NONLINEAR ANALYSIS)
C
C
C                              NOPLOT = 1+2*NEQ
C                                       (LINEAR ANALYSIS)
C
C                                 WHERE ----
C
C                                   NEQ = NO. OF EQUATIONS
C                                         INTEGRATED.
C
C                                  NLAM = NO. OF LAMBDA VARIABLES
C
C                                    NU = NO. OF U VARIABLES
C
C                                    JR = 6*NB
C
C                                    NB = NO. OF BODIES.
C
C
```

```
C
C
C
C
C
C     NOTE ---              NP
C                          *****
C                            *
C               NEQ     =    *  IRGFLX(J) + NU + NBETA + NDELTA
C                            *
C                          *****
C                           J=1
C
C
C
C
C                           NE                  NOEMC
C                          *****                *****
C                            *                    *
C               NU      =    *  IRGFLX(J) +       *  IMC(K) + 6*NP
C                            *                    *
C                          *****                *****
C                           J=1                  K=1
C
C
C
C               NBETA   = SUM OF NUMBER OF ZEROS + SUM OF
C                         NUMBER OF TWOS IN ROWS 2 THRU 7
C                         OF ARRAY IHDATA
C
C
C
C               NLAM    = SUM OF NUMBER OF ONES + SUM OF
C                         NUMBER OF TWOS IN ROWS 2 THRU 7
C                         OF ARRAY IHDATA
C
C
C
C
C
```

```
C
C
C
C
C
C                         ORDER OF VARIABLES AND
C                         SIZE FOR A SINGLE RECORD
C                         (FOR A SINGLE TIME, T).
C
C
C                          VARIABLE ID.        SIZE
C
C           NONLINEAR        TIME                1
C           ANALYSIS
C                            YDOT               NEQ
C
C                             Y                 NEQ
C
C                          LAMBDA               NLAM
C
C                             U                 NU
C
C                          HX, HY, HZ,
C                          PX, PY, PZ          6*NB
C
C                          TOTAL ANGULAR
C                          MOMENTUM VECTOR
C                          COMPONENTS (X,Y,Z).    3
C                                                          INERTIAL
C                          TOTAL LINEAR                    FRAME.
C                          MOMENTUM VECTOR
C                          COMPONENTS (X,Y,Z).    3
C
C                          BODY KINETIC
C                          ENERGIES.              NB
C
C                          BODY POTENTIAL
C                          ENERGIES.              NB
C
C                          TOTAL ANGULAR MOM.,
C                          TOTAL LINEAR MOM.,
C                          TOTAL K.E.,
C                          TOTAL P.E.,
C                          TOTAL ENERGY           5
C
C
C
C           LINEAR
C           ANALYSIS       TIME                   1
C
C                          YDOT                  NEQ
C                          Y                     NEQ
C
C
```

```
C
C
C
C
C
C----------READ(NIT,FORMAT = 16I5) (JVPL(J),J=1,JPL)
C
C
C                             JVPL(J) = INTEGER DENOTING GLOBAL
C                                       LOCATION OF JTH SELECTED
C                                       VARIABLE FROM THE NCPLOT
C                                       LONG ARRAY.
C
C      20 CONTINUE
C
C
C----------READ(NIT,FORMAT = 5I5) NCI, (NCD(I),I=1,3), NGRID
C
C                             NCI = ELEMENT LOCATION (LOCAL WRT
C                                   JVPL ARRAY) FOR THE INDEPEN-
C                                   DENT PLOT VARIABLE.
C
C                             NCD = ELEMENT LOCATION (LOCAL WRT
C                                   JVPL ARRAY) FOR UP TO 3
C                                   DEPENDENT VARIABLES TO PLOT
C                                   SIMULTANEOUSLY VERSES THE
C                                   NCI DEPENDENT VARIABLE.
C
C                             NGRID = NO. OF PLOT FRAMES TO USE
C                                     FOR PLOTTING THE NCI-NCD
C                                     GROUP. IE, THE NO. OF FRAMES
C                                     TO USE SIDE BY SIDE TO
C                                     EXHAUST THE RANGE OF THE
C                                     INDEPENDENT VARIABLE.
C
C
C         IF (NCI .EQ. 0) GO TO 1000
C
C              THIS IS CUE TO PROCEDE TO NEXT SET
C
C              THIS OPEN ENDED LOOP PERMITS MANY SELECTIONS OF THE
C              JVPL DATA VARIABLES WITH REGARD TO INDEPENDENT AND
C              DEPENDENT VARIABLES IN ORDER TO FORM CROSS-PLOTS.
C
C
C
```

```
C
C
C
C
C
C---------READ(NIT,FCRMAT = A8,2X,A8,2X,6A8) TITLI, TITLD,
C        *                                   (PTITL(I),I=1,6)
C
C                                ALPHANUMERIC TITLING INFORMATION
C                                TO INCLUDE ON PLOTTED OUTPUT.
C
C                                TITLI = INDEPENDENT VARIABLE TITLE.
C
C                                TITLD =   DEPENDENT VARIABLE TITLE.
C
C                                PTITL =  OVERALL TITLE FOR PARTICULAR
C                                         IDENTIFICATION OF THIS FRAME.
C
C
C        GO TO 20
C
C   1000 CONTINUE
C
C        RETURN
C        END
```

```
C
C
C
C
C
C*****************************************************************
C*
C* SUBROUTINE DYNS4C - INPUT FOR LINEARIZED SYSTEM ANALYSES
C*
C*****************************************************************
C
C
C---------READ(NIT,FORMAT = A4) LNAM
C
C                         LNAM = 4HTIME FOR LINEARIZED
C                                TIME RESPONSE
C
C                              = 4HFREQ FOR LINEARIZED
C                                FREQUENCY DOMAIN STUDY
C
C         IF (LNAM .EQ. 4HTIME) RETURN
C
C         IF (LNAM .EQ. 4H    ) RETURN
C
C
C---------CALL READIN (LRY, 9, NCYC, 9, KR)
C
C                 MATRIX SIZE  9 BY NCYC DEFINING FREQUENCY
C                 ANALYSIS CONTROL DATA
C
C                      NCYC = NO. OF SEPARATE TRANSFER
C                             FUNCTION CYCLES TO CONSIDER
C                             FOR THIS SIMULATION
C
C                 FOR THE JTH CYCLE -
C
C                 LRY( 1,J) = ITYPE --- INTEGER CLASSIFYING THE
C                                       TRANSFER FUNCTION TYPE.
C
C                             ITYPE = 1 PLANT ONLY (G)
C                                   = 2 CONTROLLER (H)
C                                   = 3 OPEN LOOP (GH)
C                                   = 4 OPEN LOOP (HG)
C                                   = 5 CLOSED LOOP (GH/(1+GH))
C                                   = 6 CLOSED LOOP
C                                   = 7 PSEUDO OPEN LOOP,
C                                     PERMITS OPENING OF A
C                                     SINGLE RETURN LOOP.
C
C                                     A MINUS SIGN ON ITYPE
C                                     EQUAL (3,4,5,OR 7) INDICATES
C                                     NEGATIVE CONTROLLER FEEDBACK.
C
C
C
```

```
LRY( 2,J) = ITFIN --- TRANSFER FUNCTION INPUT
                      VARIABLE IDENTIFICATION.
                      THIS INTEGER IS A LOCAL
                      IDENTIFICATION INTEGER,
                      REFERENCING (DEPENDING
                      UPON ITYPE) EITHER A
                      SENSOR SIGNAL OR A
                      CONTROLLER OUTPUT VARIABLE
                      WHICH IS THE  V(IN)  OF
                      THE EXPRESSION --

                           V(OUT)
                         ----------  =  TF
                           V(IN)


LRY( 3,J) = JTFOUT --- TRANSFER FUNCTION OUTPUT
                       VARIABLE IDENTIFICATION.
                       THIS INTEGER IS A LOCAL
                       IDENTIFICATION INTEGER,
                       REFERENCING (DEPENDING
                       UPON ITYPE) EITHER A
                       SENSOR SIGNAL OR A
                       CONTROLLER OUTPUT VARIABLE
                       WHICH IS THE  V(OUT)  OF
                       THE EXPRESSION --

                           V(OUT)
                         -----------  = TF
                           V(IN)


LRY( 4,J) = KPLOT

            KPLOT = 0 NO PLOTS
            KPLOT = 1 PLOTS WILL BE MADE

LRY( 5,J) = IAFLG

            THIS INTEGER CONTROL PARAMETER
            PERMITS THE USER TO SELECT
            THE CHARACTERISTIC ROOTS FOR A
            GIVEN TRANSFER FUNCTION FROM
            EITHER THE CHARACTERISTIC MATRIX,
            AR, OR ITS TRANSPOSE.

            IAFLG = 1  PROGRAM WILL USE ROOTS
                       FROM  AR TRANSPOSE.

                  DEFAULT VALUE IS 0 AND
                  ROOTS FROM AR WILL BE USED.
```

```
C
C
C
C
C
C                       NOTE - PROGRAM EXTRACTS ROOTS FOR BOTH
C                              AF AND ITS TRANSPOSE. THIS SERVES
C                              AS A SORT OF SELF CHECK ON THE
C                              ROOT QUALITY. ALTHOUGH IT IS A
C                              RARE OCCURRANCE, ROOTS FROM AF
C                              TRANSPOSE CAN BE -CLEANER- THAN
C                              THOSE OBTAINED FROM AF.
C
C                LRY( 6,J) = NO. OF B VARIABLES TO FEED BACK -- ITYPE = 7
C                              MAX OF 3 B VARIABLES CAN BE FED BACK FOR
C                              THE TYPE 7 PSEUDO OPEN LOOP
C                              TRANSFER FUNCTION.
C
C                LRY( 7,J) = LOCAL ID. OF FIRST  B TO RETAIN.
C
C                LRY( 8,J) = LOCAL ID. OF SECOND B TO RETAIN.
C
C                LRY( 9,J) = LOCAL ID. OF THIRD  B TO RETAIN.
C
C
C---------CALL READIM (IRY, 3, NCYC, 3, KR)
C
C                MATRIX SIZE 3 BY NCYC DEFINING EXPONENT FOR
C                TOLERANCES    TOL = (10.)**EXP
C
C                FOR THE JTH CYCLE -
C
C                IRY(1,J) = ROOT TOLERANCE EXPONENT
C                IRY(2,J) = GAIN TOLERANCE EXPONENT
C                IRY(3,J) = ROOT TOLERANCE EXPONENT USED TO
C                           REMOVE SHIFT FREQUENCY (SUBROUTINE NUMS)
C
C
C                NOTE - IF ROOT OR GAIN LE TOL, SET
C                       ROOT OR GAIN EQ O.
```

```
C
C
C
C
C
C          DO 500 ICYC=1,NCYC
C
C          IF (ITYPE .EQ. 0) GO TO 500
C
C
C---------READ(NIT,FORMAT = 20A4) (TITLE(I),I=1,20)
C
C                   80 CHARACTER TITLE FOR TRANSFER FUNCTION
C                   IDENTIFICATION
C
C
C---------READ(NIT,FORMAT = 5A4) (LPNAME(I),I=1,5)
C
C                   LPNAME(I) PERMITS UP TO 5 FOUR CHARACTER
C                             IDENTIFICATIONS WHICH SELECT THE
C                             PLOT DISPLAY MODE.
C
C                   LPNAME(I) = 4H       (ALL BLANK) NO DISPLAYS
C                                        ARE IMPLEMENTED --- GO TO 500
C                                        THE CHARACTERISTIC ROOTS
C                                        FOR THE SYSTEM ARE FOUND.
C
C                   LPNAME(I) = 4HBODE   ONLY A BODE DISPLAY.
C
C                             = 4HNICH   ONLY A NICHOLS DISPLAY.
C
C                             = 4HNYQU   ONLY A NYQUIST DISPLAY.
C
C                             = 4HNTNY   BOTH NICHOLS AND NYQUIST.
C
C                             = 4HBONN   GIVES BODE, NICHOLS, NYQUIST.
C
C                             = 4HROOT   GIVES A ROOT LOCUS DISPLAY.
C
C
```

```
C
C
C
C
C
C       DO 500 IOP=1,5
C
C
C        IF   (LPNAME(IOP) .EQ. 4H    ) GO TO 500
C
C        IF   (LPNAME(IOP) .EQ. 4HBODE
C       *.OR. LPNAME(IOP) .EQ. 4HNICH
C       *.OR. LPNAME(IOP) .EQ. 4HNYQU
C       *.OR. LPNAME(IOP) .EQ. 4HNINY
C       *.OR. LPNAME(IOP) .EQ. 4HBONN) GO TO 200
C
C        IF   (LPNAME(IOP) .EQ. 4HROOT) GO TO 300
C
C
C   200 CONTINUE
C
C
C             *** FREQUENCY RESPONSE SECTION ***
C
C
C---------READ(NIT,FORMAT = 6F10.0) FMIN, FMAX, DBMIN, DBMAX, AMIN, AMAX
C
C                 FMIN  = FREQUENCY SWEEP LOWER LIMIT
C                 FMAX  = FREQUENCY SWEEP UPPER LIMIT
C                 DBMIN = MINIMUM DB AMPLITUDE FOR BODE, NICHOLS PLOTS
C                 DBMAX = MAXIMUM DB AMPLITUDE FOR BODE, NICHOLS PLOTS
C                 AMIN  = MINIMUM AMPLITUDE FOR NYQUIST PLOTS
C                 AMAX  = MAXIMUM AMPLITUDE FOR NYQUIST PLOTS
C
C
C       GO TO 500
C
C
C   300 CONTINUE
C
C
C
```

```
C
C
C
C
C
C               *** ROOT LOCUS SECTION ***
C
C
C---------CALL READIM (IJM, 2, NRLC, 2, KR)
C
C                  MATRIX SIZE 2 BY NRLC FOR ROOT LOCUS PLOT CONTROL
C
C                  NRLC = NUMBER OF ROOT LOCI TO PERFORM
C
C                  FOR THE JTH ROOT LOCI -
C
C                    IJM(1,J) = ISNIM     = 1  STARTING POINT IS
C                                              OPEN LOOP ZERO.
C                                         = 2  STARTING POINT IS
C                                              OPEN LOOP POLE.
C                                         = 3  STARTING POINT IS
C                                              CLOSED LOOP POLE.
C
C
C                    IJM(2,J) = ELEMENT LOCATION IN ROOT ARRAY
C                               FOR STARTING ROOT LOCI.
C
C
C---------CALL READ (W1, 6, NRLC, K5, KR)
C
C                  MATRIX SIZE 6 BY NRLC FOR ROOT LOCUS CONTROL DATA
C
C                  FOR THE JTH ROOT LOCI -
C
C                    W1(1,J) = THETAC(J)  INITIAL SEARCH ANGLE,
C                              NORMALLY -180. (DEGREES)
C
C                    W1(2,J) = SCL  SCALE FACTOR, NORMALLY
C                                   SCL = 1.0
C
C                    W1(3,J) = ALOC  PHASE CONTROL PARAMETER.
C                                    ALOC = +1. --- 180. DEG. PHASE.
C                                    ALOC = -1. ---  0   DEG. PHASE.
C
C                    W1(4,J) = XMIN  MIN REAL VALUE TO PLOT.
C
C                    W1(5,J) = XMAX  MAX REAL VALUE TO PLOT.
C
C                    W1(6,J) = YMAX  MAX IMAG VALUE TO PLOT AND
C                                    YMIN SET TO - YMAX.
C
C      500 CONTINUE
C
C          RETURN
C          END
```

V. PROGRAM OUTPUTS

This section discusses the various program output information and correlates the output data with both the input data and the problem simulation. This information is presented in much the same fashion as was the input data stream of the previous chapter so as to better acquaint the reader with the actual formated output as it is presented by the program.

It is pointed out that the output stream will not reflect certain outputs that occur from routines that identify troublesome areas such as matrix singularities. Recall also that the basic input routines READ and READIM can also print out input matrix data as dictated by the user. These printouts will not be included either. Reference is made to the theoretical volume (Vol. I) and to the input data stream (Chapter IV) to correlate certain outputs with both the theory and the user input requirements.

```
C
C
C
C
C
C*****************************************************************
C*****************************************************************
C*
C*    DESCRIPTION OF PROGRAM DISCOS OUTPUT DATA
C*
C*****************************************************************
C*****************************************************************
C
C
C-------- COMMENTS (FORMAT 13A6)
C
C                   COMMENT CARDS READ INTO THE
C                   PROGRAM WITH SUBROUTINE COMENT
C                   ARE PRINTED HERE.
C
C
C
C*****************************************************************
C*
C*    SUBROUTINE DYNS10 OUTPUT
C*
C*****************************************************************
C
C
C
C
C--------- SUMMARY OF DYNAMIC-SIMULATION INPUT DATA * * * * * * * * * *
C
C
C--------- ACTUAL SIZES  MAXIMUM SIZES  INTEGRATION DATA  GRAVITY GRADIENT DATA
C          ------------  -------------  ----------------  ---------------------
C
C--------- NB     =      NBMAX   =       STARTT =          G1    =     GAM1   =
C
C--------- NH     =      NHMAX   =       DELTAT =          G2    =     GAM2   =
C
C--------- NSPT   =      NSPMAX  =       ENDT   =          G3    =     GAM3   =
C
C--------- NOFMO  =      NMWMAX  =                         GMAG  =     RCMAG  =
C
C--------- NDELTA =      NMWBOD  =
C
C--------- NU     =      NMDBOD  =                         MISC.  DATA
C                                                          -----------
C--------- NBETA  =      KMU     =
C
C--------- NLAM   =      KY      =                         NOPRNT =
C
C--------- NEQ    =      KU      =                         NOPLOT =
C
C                                                          IFLNER =
```

```
C
C
C
C
C
C                     THE FOLLOWING LIST IDENTIFIES THE
C                     OUTPUT VARIABLES SUMMARIZED ON THE
C                     PREVIOUS PAGE.
C
C
C*************************************************************************
C
C
C                     NB     = NO. OF BODIES.
C
C                     NH     = NO. OF HINGES.
C
C                     NSPT   = NO. OF SENSOR POINTS.
C
C                     NDELTA = NO. OF CONTROL SYSTEM DELTAS.
C
C                     NU     = NO. OF U'S.
C
C                     NBETA  = NO. OF BETA'S.
C
C                     NLAM   = NO. OF LAMBDA'S (CONSTRAINTS).
C
C                     NEQ    = NO. OF STATE EQUATIONS.
C
C                     NBMAX  = MAXIMUM DIMENSIONED NO. OF BODIES.
C
C                     NHMAX  = MAXIMUM DIMENSIONED NO. OF HINGES.
C
C                     NSPMAX = MAXIMUM DIMENSIONED NO. OF SENSOR POINTS.
C
C                     NMWMAX = MAXIMUM DIMENSIONED NO. OF MOM. WHEELS.
C
C                     NMWBOD = MAXIMUM DIMENSIONED NO. OF MOM. WHEELS
C                                                         PER BODY.
C
C                     NMDBOD = MAXIMUM DIMENSIONED NO. OF MODES PER BODY.
C
C                     KMU    = MAXIMUM DIMENSIONED NO. OF U'S PER BODY.
C                            = 6 + NMDBOD + NMWBOD.
C
C                     KY     = MAXIMUM DIMENSIONED SIZE FOR STATE VECTOR.
C
C                     KU     = MAXIMUM DIMENSIONED NO. OF U'S.
C                            = NBMAX*(6 + NMDBOD) + NMWMAX.
C
C
```

```
C
C
C
C
C                   OUTPUT VARIABLE IDENTIFICATION SUMMARY
C                                (CONT'D)
C
C
C************************************************************************
C
C
C
C                   STARTT = START TIME FOR TIME RESPONSE.
C
C                   DELTAT = INTEGRATION STEP SIZE.
C
C                   ENDT   = END TIME FOR TIME RESPONSE.
C
C
C
C
C
C
C
C
C                   G1     = X COMPONENT OF GRAVITY VECTOR. (INPUT)
C
C                   G2     = Y COMPONENT OF GRAVITY VECTOR. (INPUT)
C
C                   G3     = Z COMPONENT OF GRAVITY VECTOR. (INPUT)
C
C                   GMAG   = SQRT(G1**2 + G2**2 + G3**2) -- ACC. OF GRAVITY.
C
C                   GAM1   = DIRECTION COSINE (GRAVITY VECTOR AND X).
C
C                   GAM2   = DIRECTION COSINE (GRAVITY VECTOR AND Y).
C
C                   GAM3   = DIRECTION COSINE (GRAVITY VECTOR AND Z).
C
C                   RCMAG  = REFERENCE RADIUS FOR ACTING GRAVITY VECTOR. (INPUT)
C
C                   NOPRNT = MULTIPLE OF DELTAT TO PRINT TIME RESPONSE.
C
C                   NOPLOT = MULTIPLE OF DELTAT TO WRITE PLOT TAPE.
C
C                   IFLNER = 0  (NON LINEAR TIME RESPONSE)
C                          = 1  (LINEAR ANALYSIS OR FREQ RESPONSE)
C
```

```
C
C
C
C
C
C                   THE FOLLOWING SERIES OF ARRAYS PERTAIN
C                   TO THE MODEL SIMULATION AND ARE PRINTED
C                   OUT AT THE BEGINNING OF EACH SIMULATION
C
C*****************************************************************
C
C
C---------THE TOPOLOGY ARRAY (ITOPOL) FOR THIS CASE FOLLOWS
C
C                     (1)    (2)   ...     (NH)
C
C---------     1    1
C---------     2    1
C
C                            THIS IS THE INPUT INTEGER ARRAY ITOPOL
C                            (SEE PREVIOUS CHAPTER -- INPUT DATA)
C
C
C---------THE CONSTRAINT SPECIFICATIONS FOR THIS CASE FOLLOW
C
C                     (1)     (2)   ...     (NH)
C
C---------     1    1
C---------     2    1
C---------     3    1         THIS IS THE INPUT INTEGER ARRAY THDATA
C---------     4    1         (SEE PREVIOUS CHAPTER -- INPUT DATA)
C---------     5    1
C---------     6    1
C---------     7    1
C
C
C---------THE SPECIFIED INITIAL HINGE ANGLES
C         AND DISPLACEMENTS (BETAH) FOLLOW
C
C                      (1)    (2)    ...     (NH)
C
C---------     1     1
C---------     2     1     ROWS 1-3 = HINGE  ANGLES (CONSISTENT WITH ITYPE).
C---------     .     .
C---------     .     .     ROWS 4-6 = HINGE DISPLACEMENTS- 0 RELATIVE TO P.
C---------     .     .
C---------     6     1
C
C
```

```
C
C
C
C
C
C---------THE SPECIFIED INITAL HINGE RATES (BETAHD) FOLLOW
C
C                        (1)    (2)    ...     (NH)
C
C---------     1     1
C---------     2     1      ROWS 1-3 = ANGULAR RATES.
C---------     .     .
C---------     .     .      ROWS 4-6 = DISPLACEMENT RATES- Q RELATIVE TO P.
C---------     .     .
C---------     6     1
C
C
C---------THE NO. OF ELASTIC MODES/BODY ARRAY (IRGFLX) FOLLOWS
C
C                      (1)    (2) ...  (NB)
C
C---------    1     1  THE JTH ENTRY IS THE NO. OF ELASTIC
C                      MODES FOR BODY J.
C
C
C---------THE NO. OF P/Q HINGE POINTS/BODY ARRAY (NHPOI) FOLLOWS
C
C                      (1)    (2) ...  (NB)
C
C---------     1     1         ELEMENTS ARE THE NO. OF P/Q
C                              HINGE POINTS ON EACH BODY.
C
C
C---------THE NO. OF SENSOR POINTS/BODY ARRAY (NSPOI) FOLLOWS
C
C                      (1)    (2) ...  (NB)
C
C---------     1     1         ELEMENTS ARE THE NO. OF SENSOR
C                              POINTS ON EACH BODY.
C
C
C---------THE MOM. WHEEL/BODY TABLE (NMOW) FOLLOWS
C
C                        (1)   (2)    ...  (NB)
C
C---------     1     1 COL J = JTH BODY
C---------     2     1 ROW 1 = NO. OF MOM. WHEELS ON BODY J.
C---------     .     . ROW 2 = NO. OF VARIABLE SPEED WHEELS ON BODY J.
C---------     .     . ROW 3 = SUCCESSIVE ROWS ARE THE MOMENTUM WHEEL NUMBERS
C---------     .     .  .        ON BODY J (IN ASCENDING ORDER).
C---------2+NMWBOD  1  .
C
C
C
```

```
C
C
C
C
C
C---------THE STATE VECTOR LENGTH ARRAY (LENU) FOLLOWS
C
C                    (1)   (2)   ...  (2*NB + 2)
C
C---------      1      1
C
C                 THE ELEMENTS ARE THE LENGTHS OF SEGMENTS
C                 OF THE STATE VECTOR.
C
C                 ORDER IS
C
C                      U(1),U(2),...,U(NB),XI(1),...,XI(NB),BETA,DELTA
C
C
C
C---------THE STATE VECTOR LOCATION ARRAY (LOCU) FOLLOWS
C
C                    (1)   (2)   ...  (2*NB + 2)
C
C---------      1      1
C
C                   LEADING ELEMENT LOCATION IN THE STATE VECTOR
C                   FOR THE SEGMENTS DESCRIBED IN ARRAY (LENU).
C
C
C---------THE SPECIFIED SENSOR POINT/BODY CORRELATION ARRAY (IFTSMW) FOLLOWS
C
C                    (1)   (2)   ...  (NSPT)
C
C---------      1      1     THE JTH ELEMENT IS THE BODY NO. ON
C                            WHICH SENSOR POINT J IS LOCATED.
C
C
C
```

```
C
C
C
C
C
C---------THE FOLLOWING DATA IS SPECIFIED MOM. WHEEL INFORMATION (IF ANY)
C                             AND CONTROLLER INFORMATION
C
C
C---------THE SPECIFIED MOM. WHEEL CONTROL ARRAY (IMO) FOLLOWS
C
C                     (1)      (2)     ...   (NOFMO)
C
C                       COL J = JTH MOMENTUM WHEEL
C---------      1    1     ROW 1 = WHEEL SENSOR POINT NO.
C---------      2    1     ROW 2 = SPIN AXIS
C---------      3    1     ROW 3 = 1  ACTIVE
C                                = 0 CONSTANT SPEED
C
C
C---------THE SPECIFIED MOM. WHEEL RATES AND INERTIAS (AMO) FOLLOW
C
C                     (1)      (2)     ...   (NOFMO)
C
C                       COL J = JTH MOMENTUM WHEEL
C---------      1    1     ROW 1 = INITIAL WHEEL SPIN RATE
C---------      2    1     ROW 2 = SPIN INERTIA
C
C
C---------THE SPECIFIED CONTROLLER INITIAL CONDITIONS
C          AND CHARACTERISTICS FOLLOW
C            (THE FIRST NDELTA ARE INITIAL CONTROLLER STATE
C             VARIABLES, THERE ARE   K ADDITIONAL PARAMETERS)
C
C                    THIS IS THE USER INPUT ARRAY CNTDTA.
C
C                    THE ADDITIONAL K PARAMETERS, IF ANY, ARE
C                    AVAILABLE TO THE USER FOR USER-PAK DATA.
C
C                    THE FIRST NDELTA ENTRIES IN THIS ARRAY
C                    ARE THE INITIAL CONDITIONS FOR THE
C                    CONTROL VARIABLES.
C
C
C
C*****************************************************************
C*****************************************************************
```

```
C
C
C
C
C
C*****************************************************************
C*
C*    SUBROUTINE MRIGID -- OUTPUT FOR RIGID BODY
C*
C*****************************************************************
C
C                   THE FOLLOWING IS TYPICAL FOR THE ITH BODY
C
C---------          SUMMARY OF INPUT DATA FOR BODY  I  WHICH IS RIGID.
C
C---------THE 6X6 INERTIA MATRIX IS --
C
C                    (1)     (2)     (3)     (4)     (5)     (6)
C
C------     1    1   IXX    -IXY    -IXZ      0      -SZ      SY
C------     2    1 -IYX     IYY    -IYZ     SZ       0      -SX
C------     3    1 -IZX    -IZY     IZZ    -SY      SX       0
C------     4    1   0       SZ     -SY      M        0       0
C------     5    1  -SZ      0       SX      0        M       0
C------     6    1   SY     -SX      0       0        0       M
C
C
C---------FOR BODY  I THE P-Q HINGE NO. AND THE EULER ROTATION TYPE
C         APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED BY
C         AN ARRAY CONTAINING EULER ANGLES (1,2,3), AND POSITION
C         VECTOR COMPONENTS (4,5,6) THAT POSITION THE HINGE TRIAD
C         WRT THE BODY TRIAD
C
C                             -----  DATA HERE  -----
C
C
C         IF BODY I HAS ANY SENSOR POINTS,
C         THE FOLLOWING WILL BE PRINTED
C
C
C---------FOR BODY  I THE SENSOR POINT NO. AND THE EULER ROTATION TYPE
C         APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED
C         BY AN ARRAY CONTAINING EULER ANGLES (1,2,3), AND POSITION
C         VECTOR COMPONENTS (4,5,6) THAT POSITION THE SENSOR TRIAD
C         WRT THE BODY TRIAD
C
C
C                             -----  DATA HERE  -----
C
C
C*****************************************************************
C*****************************************************************
```

```
C
C
C
C
C
C*********************************************************************
C*
C*     SUBROUTINE MSMODC - OUTPUT FOR FLEXIBLE BODY, CONSISTENT MASS MATRIX
C*
C*********************************************************************
C
C
C                        TYPICAL OUTPUT FOR ITH BODY
C
C
C          THIS SUBROUTINE PRINTS OUT SEVERAL MATRICES THAT ARE
C          RELATED TO THE FORM OF THE GOVERNING EQUATIONS.
C
C          THE READER WILL BE REFERRED TO VOL I AND THE INPUT
C          DATA STREAM FOR FURTHER CLARIFICATION.
C
C----------SUMMARY OF INPUT DATA FOR BODY I WHICH IS
C          FLEXIBLE W/CONSISTENT MASS MATRIX.
C
C
C----------THE INPUT PARAMETERS--- IFRBM, IFDIAK, IFDIAD ARE
C
C                              SEE INPUT DATA FOR MSMODC
C
C
C----------THE JDOF TABLE FOLLOWS---
C
C                              DEGREE OF FREEDOM AS INPUT
C
C                              SEE INPUT DATA FOR MSMODC
C
C
C----------THE MODE SELECTION VECTOR FOLLOWS
C
C                              MODE SELECTION VECTOR AS INPUT
C
C                              SEE INPUT DATA FOR MSMODC
C
C
C----------FOR BODY NO.  I     THE POSITION VECTOR FROM THE BODY ORIGIN
C                              TO JOINT K IS
C
C                              X =        Y =        Z =
C
C                              WHERE   K IS JOINT COORDINATES
C                              USED TO DEVELOP RIGID BODY MODES
C
C                              SEE INPUT DATA FOR MSMODC
C
```

```
C
C
C
C
C
C---------THE CONSISTENT, REPARTITIONED MASS MATRIX IS--
C
C                          THIS IS THE REPARTITIONED MASS
C                          MATRIX AND IS CONSISTENT WITH
C                          THE IDOF TABLE.
C
C
C---------THE REPARTITIONED MODAL MATRIX IS---
C
C                          THIS IS THE REPARTITIONED MODAL
C                          MATRIX. THE ROWS ARE CONSISTENT
C                          WITH THE REPARTITIONED MASS MATRIX
C                          AND THE COLS ARE CONSISTENT WITH
C                          THE ELEMENTS OF THE MODE SELECTION
C                          VECTOR.
C
C
C
C---------THE  -UNDEFORMED-  INERTIA MATRIX (MU) IS---
C
C                          THIS IS THE MO MATRIX NOTED AS
C                          EQUATION II-87 (VOL I)  .
C
C
C
C---------
C         THERE THEN FOLLOWS MATRICES
C
C                        A       COEFFICIENTS
C                        B       COEFFICIENTS
C                      COFXY     COEFFICIENTS
C                      COFXZ     COEFFICIENTS
C                      COFYZ     COEFFICIENTS
C
C
C         WHICH ARE THE  ALPHA, B, AND C COEFFICIENTS
C         GIVEN IN THE EXPRESSION
C         EQUATION II-88 (VOL I)
C
C
C---------
C         THERE THEN FOLLOWS MATRICES
C
C                        C11, C22, C33,
C                        C12, C13, C23
C
C
C         WHICH ARE IDENTIFIED IN
C           EQUATION II-89 (VOL I)
C
C
```

```
C
C
C
C
C
C
C---------THE MODAL STIFFNESS IS --
C
C         -----  DATA HERE  -----
C
C
C---------THE MODAL DAMPING MATRIX IS --
C
C         -----  DATA HERE  -----
C
C
C---------
C          THERE FOLLOWS TWO ARRAYS CONTAINING THE
C
C                          INITIAL MODAL DISPLACEMENTS AND
C                          INITIAL MODAL VELOCITIES RESPECTIVELY.
C
C
C---------FOR BODY I THE P-Q HINGE NO., THE EULER ROTATION TYPE
C         AND THE JOINT NO. CORRESPONDING TO THE P-Q HINGE
C         APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED BY AN
C         ARRAY CONTAINING EULER ANGLES THAT POSITION THE HINGE
C         TRIAD WRT THE BODY TRIAD
C
C                    ----  DATA HERE  -----
C
C
C         IF BODY I HAS ANY SENSOR POINTS,
C         THE FOLLOWING WILL BE PRINTED
C
C
C---------FOR BODY I THE SENSOR POINT NO., THE EULER ROTATION TYPE
C         AND THE JOINT NO. CORRESPONDING TO THE SENSOR POINT
C         APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED
C         BY AN ARRAY CONTAINING EULER ANGLES THAT POSITION THE
C         SENSOR TRIAD WRT THE BODY TRIAD
C
C                    ----  DATA HERE  -----
C
C
C
C*****************************************************************
C*****************************************************************
```

```
C
C
C
C
C
C*******************************************************************
C*
C*    SUBROUTINE MSMODL - OUTPUT FOR FLEXIBLE BODY, LUMPED MASS MATRIX
C*
C*******************************************************************
C
C
C                      TYPICAL OUTPUT FOR THE ITH BODY
C
C
C          THIS SUBROUTINE PRINTS OUT SEVERAL MATRICES THAT ARE
C          RELATED TO THE FORM OF THE GOVERNING EQUATIONS.
C
C
C          THE READER WILL BE REFERRED TO VOL I AND THE INPUT
C          DATA STREAM FOR FURTHER CLARIFICATION.
C
C
C---------OUTPUT MATRICES  INERO, STATO, MASSO, DOCOFF, AOCOFF, FCOFF
C
C                          ARE THE J, -S, M, D, A, F, PARTITIONS
C                          RESPECTIVELY OF THE MATRIX MO OF EQUATION II-87
C                          IN VOL I.
C
C
C---------OUTPUT MATRIX    MUO IS MATRIX MO OF EQUATION II-87 (VOL I)  .
C
C
C---------OUTPUT MATRICES  ACOF, BCOF, CXY, CXZ, CYZ,
C
C                          ARE THE  ALPHA, B, AND C
C                          COEFFICIENTS IN THE MATRIX
C                          GIVEN AS EQUATION II-88 (VOL I)  .
C
C
```

```
C
C
C
C
C
C
C
C---------OUTPUT MATRICES  C11, C22, C33,
C                          C12, C13, C23
C
C                          ARE THE CONSTITUENTS OF THE
C                          MATRIX GIVEN AS  EQUATION II-89 (VOL I)
C
C
C---------OUTPUT MATRIX XF0
C
C                          CONTAINS THE INITIAL MODAL
C                          DEFLECTIONS. (AS INPUT)
C
C
C---------OUTPUT MATRIX XF0D
C
C                          CONTAINS THE INITIAL MODAL
C                          VELOCITIES.  (AS INPUT)
C
C
C---------FOR BODY I THE P-Q HINGE NO., THE EULER ROTATION TYPE
C         AND THE JOINT NO. CORRESPONDING TO THE P-Q HINGE
C         APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED BY AN
C         ARRAY CONTAINING EULER ANGLES THAT POSITION THE HINGE
C         TRIAD WRT THE BODY TRIAD
C
C                  ----  DATA HERE  -----
C
C
C         IF BODY I HAS ANY SENSOR POINTS,
C         THE FOLLOWING WILL BE PRINTED
C
C
C---------FOR BODY I THE SENSOR POINT NO., THE EULER ROTATION TYPE
C         AND THE JOINT NO. CORRESPONDING TO THE SENSOR POINT
C         APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED
C         BY AN ARRAY CONTAINING EULER ANGLES THAT POSITION THE
C         SENSOR TRIAD WRT THE BODY TRIAD
C
C                  ----  DATA HERE  -----
C
C
C
C****************************************************************
C****************************************************************
```

```
C
C
C
C
C
C          IF THE USER UTILIZES POLYNOMIAL INPUT FOR
C          CONTROL SYSTEM TRANSFER FUNCTIONS, THE FOLLOWING
C          MATRIX WILL BE PRINTED.
C
C
C---------OUTPUT MATRIX CPLY(KPLY,2*NPLY)
C
C               WHERE KPLY = ROW DIMENSION SIZE OF CPLY IN SUBROUTINE CONTRL
C
C                     NPLY = NO. OF INPUT POLYNOMIAL RATIOS
C
C               AND FOR I = ODD INTEGER --
C
C                    COL I   = DENOMINATOR POLYNOMIAL COEFFICIENTS
C                              IN ASCENDING ORDER
C
C                    COL I+1 = NUMERATOR POLYNOMIAL COEFFICIENTS
C                              IN ASCENDING ORDER
C
C
C---------THE FOLLOWING INTEGER ARRAY (INDEP) PRESCRIBES INDEPENDENT
C          VARIABLES (1) AND DEPENDENT VARIABLES (0)
C
C                             THE ELEMENTS OF THIS ARRAY IDENTIFY
C                             WHICH VARIABLES SURVIVE IN THE FINDU
C                             SEARCH TO DETERMINE AN INDEPENDENT
C                             SET TO BE INTEGRATED.
C
C                             THE SURVIVING VARIABLES WILL ALSO
C                             REPRESENT THE FIRST NX ROWS OF
C                             THE LINEARIZED MATRIX, A, USED
C                             BY SUBROUTINE DYNS40.
C
C                             NOTE -  NX = NO. OF NON-ZERO ENTRIES IN ARRAY INDEP
C
C
C
C
C
C*****************************************************************
C*****************************************************************
```

```
C
C
C
C
C
C*******************************************************************
C*
C*     SUBROUTINE DYNS20 OUTPUT
C*
C*******************************************************************
C
C
C                  THE PRINTOUT IS TYPICAL FOR A GIVEN SIMULATION TIME, T.
C
C                  THE T=0  PRINT OUT IS ALWAYS GIVEN
C                           (EVEN FOR A LINEARIZED ANALYSIS)
C
C
C                  THE DATA ARE PRESENTED IN VECTOR FORM AND
C                  ORDERED AS FOLLOWS---
C
C----------THE STATE VECTOR Y =
C
C----------THE STATE VECTOR TIME DERIVATIVE YDT =
C
C----------THE BETAS (EULER ANGLES, POSITION COORDINATES) ARE
C
C----------THE BETA TIME DERIVATIVES ARE
C
C----------THE DELTAS (CONTROL SYSTEM VARIABLES) ARE
C
C----------THE DELTA TIME DERIVATIVES ARE
C
C
C
```

```
C
C
C
C
C
C---------
C          THE FOLLOWING INFORMATION IS TYPICAL FOR BODY I.
C
C
C---------FOR BODY I  THE VELOCITIES ARE
C
C          ORDER IS    OMGX
C                      OMGY    BODY ANGULAR VELOCITY
C                      OMGZ
C
C                        U
C                        V     BODY TRANSLATIONAL VELOCITY
C                        W
C
C                      XTD(1)
C                      XTD(2)
C                        .     BODY MODAL VELOCITIES
C                        .
C                        .
C                      XTD(NF)
C
C                     THETAD(1)
C                        .     MOMENTUM WHEEL ANGULAR VELOCITY
C                        .     (RELATIVE TO SENSOR POINT TRIAD)
C                        .          (ACTIVE WHEEL)
C
C
C---------FOR BODY I THE CORRESPONDING MOMENTA ARE
C
C            ORDER IS    HX
C                        HY  BODY AXES REF. ANGULAR MOMENTUM
C                        HZ  (INCLS CONTRIBUTION OF CONST. SPEED WHEEL)
C
C                        LX
C                        LY  BODY AXES REF. LINEAR MOMENTUM
C                        LZ
C
C                      P XT(1)
C                      P XT(2)
C                        .   BODY AXES REF. MODAL MOMENTUM
C                        .
C                        .
C                      P XT(NF)
C
C                      H MW(1)
C                        .   BODY AXES REF. MOM WHEEL MOMENTUM
C                        .   (ACTIVE WHEEL)
C                        .
C
C
```

```
C
C
C
C
C
C
C----------FOR BODY I ITS CONTRIBUTION TO TOTAL ANGULAR
C                      AND LINEAR MOMENTUM IS
C
C               ORDER IS     HX
C                            HY   ANGULAR MOMENTUM
C                            HZ
C                            (REFFERENCED TO INERTIAL ORIGIN)
C                            LX
C                            LY   LINEAR MOMENTUM
C                            LZ
C
C
C----------FOR BODY I ITS CONTRIBUTION TO TOTAL KINETIC AND
C                      POTENTIAL ENERGIES IS
C
C
C----------FOR BODY I  THE ELASTIC DEFLECTIONS ARE
C
C
C----------THE INTERCONNECTION CONSTRAINT FORCES (LAMBDAS) ARE
C
C
C----------THE TOTAL ANGULAR MOMENTUM VECTOR IS
C
C                     X,Y,Z  COMPONENTS IN INERTIAL REF. AXIS SYSTEM.
C
C
C----------THE TOTAL LINEAR MOMENTUM VECTOR IS
C
C                     X,Y,Z  COMPONENTS IN INERTIAL REF. AXIS SYSTEM.
C
C
C----------THE TOTAL ANGULAR MOMENTUM =
C----------THE TOTAL LINEAR MOMENTUM  =
C----------THE TOTAL KINETIC ENERGY   =
C----------THE TOTAL POTENTIAL ENERGY =
C----------THE TOTAL ENERGY (T+V)     =
C
C
C
C*******************************************************************
C*******************************************************************
```

```
C
C
C
C
C
C
C*****************************************************************************
C*
C*    SUBROUTINE DYNS30 OUTPUT
C*
C*****************************************************************************
C
C
C
C---------              SUMMARY OF PLOTTING INFORMATION
C
C
C                       THE OUTPUT SUMMARIZES THE INPUT DATA
C                       WHICH CONTROLLED THE TIME RESPONSE
C                       PLOT LOGIC.
C
C                           SEE INPUT DATA FOR DYNS30 FOR
C                           DETAILED DESCRIPTIONS.
C
C
C                   NOTE- THE OUTPUT VARIABLE NRPLOT
C                         IS THE NO. OF LOGICAL RECORDS
C                         WRITTEN ON THE PLOT TAPE.
C
C
C
C
C****************************************************************************
C****************************************************************************
```

```
C
C
C
C
C
C
C*********************************************************************
C*
C*   SUBROUTINE DYNS40 OUTPUT - LINEARIZED SYSTEM ANALYSIS
C*
C*********************************************************************
C
C
C----------OUTPUT MATRIX -A- (NJQ,NX)
C
C                          NJQ = TOTAL NO. OF EQUATIONS
C                                LINEARIZED (INCLUDING
C                                AUXILIARY EQUATIONS FROM
C                                SUBROUTINE EQADD)
C
C                          NX  = NO OF INDEPENDENT STATE EQUATIONS
C                                DETERMINED BY FINDU IN DYNS20.
C
C
C                          THIS IS THE MATRIX OF PARTIAL DERIVATIVES
C                          WHICH ARE THE LINEARIZED COMPONENTS OF THE
C                          STATE VARIABLES AS DETERMINED BY
C                          SUBROUTINE LINEAR.
C
C
C                          THE ORDER OF THE VARIABLES FOR THE
C                          ROWS IS
C
C                                                SIZE
C
C                              PLANT VARIABLES    NY
C
C                              CONTROL VARIABLES  NDELTA
C
C                              CONTROLLER OUTPUTS NBTQ
C
C                              SENSOR SIGNALS     NXSS
C
C
C
C                          THE COLUMN ORDER IS
C
C                              PLANT VARIABLES    NY
C
C                              CONTROL VARIABLES  NDELTA
C
```

```
C
C
C
C
C
C
C---------OUTPUT MATRIX -T- (NX,NX)
C
C                               THIS IS THE SIMILARITY
C                               TRANSFORMATION MATRIX
C                               THAT INTRODUCES THE (NBTQ + NXSS)
C                               AUXILIARY VARIABLES INTO THE
C                               TRANSFORMED STATE EQUATIONS.
C
C                               REF. MATRIX R IN EQUATION III-24 (VOL I)
C
C
C---------OUTPUT MATRIX  Y* (1,NX)
C
C                              THIS IS THE TRANSFORMED STATE
C                              VECTOR INITIAL CONDITIONS.
C
C                              REF. VECTOR Z IN EQUATION III-24 (VOL I)
C
C
C---------OUTPUT MATRIX A* (NX,NX)
C
C                              THIS IS THE TRANSFORMED, LINEARIZED
C                              STATE VARIABLE COEFFICIENT MATRIX
C                              THAT IS THE BASIS FOR THE ENTIRE
C                              LINEARIZATION PACKAGE.
C
C                              REF. EQUATION III-28 (VOL I)
C
C                              THE ROW/COL VARIABLE ORDERING AND
C                              SIZES ARE:
C
C                                VARIABLE ID.                  SIZE
C                                ------------                  ----
C
C                                PLANT VARIABLES                NY2
C
C                                PLANT SENSOR SIGNALS           NXSS
C
C                                CONTROL SYSTEM VARIABLES       ND2
C
C                                CONTROL OUTPUTS (B'S)          NBTQ
C
C
C                          NOTE-  NY2 = NY - NXSS
C
C                                 ND2 = NDELTA - NBTQ
C
```

```
C
C
C
C
C
C---------              RT  A                              RTA*
C
C            REAL PART  IMAGINARY PART       REAL PART  IMAGINARY PART
C
C
C                             COMPLEX ROOTS OBTAINED FROM
C                                A AND A* RESPECTIVELY.
C
C
C                                THESE ARE THE POLES OF
C                                THE CLOSED LOOP SYSTEM.
C
C
C***********************************************************************
C
C
C             THE FOLLOWING OUTPUTS ARE CHARACTERISTIC OF
C             A SINGLE TRANSFER FUNCTION FREQUENCY RESPONSE.
C
C
C---------OUTPUT MATRIX -AR-
C
C                             THIS IS THE REDUCED A* MATRIX
C                             FOR A PARTICULAR USER SPECIFIED
C                             TRANSFER FUNCTION TYPE.
C
C                             REF. INPUT DATA LIST AND
C                             SECTION III. D-2. IN VOL I.
C
C                             THE ROOTS OF AR  ARE THE
C                             TRANSFER FUNCTION POLES.
C
C
C---------OUTPUT MATRIX BCOL
C
C                             THIS IS THE VECTOR (COL) WITH WHICH
C                             AR IS AUGMENTED TO DETERMINE THE
C                             TRANSFER FUNCTION ZEROS.
C
```

```
C
C
C
C
C
C
C---------          R AR                         RART
C
C         REAL PART  IMAGINARY PART     REAL PART  IMAGINARY PART
C
C
C                       THESE ARE THE COMPLEX ROOT ARRAYS
C                       AS EXTRACTED FROM MATRICES
C                       AR AND AR TRANSPOSE RESPECTIVELY.
C
C                       THE USER SELECTS VIA INPUT WHICH
C                       SET OF ROOTS TO USE FOR THE POLES.
C
C
C---------          NUM                          DEN
C
C         REAL PART  IMAGINARY PART     REAL PART  IMAGINARY PART
C                    .                             .
C                    .                             .
C                    .                             .
C
C            NUMERATOR ROOTS                 DENOMINATOR ROOTS
C
C          (TRANSFER FUNCTION ZEROS)       (TRANSFER FUNCTION POLES)
C
C                    .                             .
C                    .                             .
C                    .                             .
C
C
```

```
C
C
C
C
C
C
C----------OUTPUT MATRIX RRED
C
C
C              TRANSFER FUNCTION ROOT ARRAY CONTAINING ROOT
C              COUNTS, TIME CONSTANTS, DAMPING, AND FREQUENCY
C              FOR ZEROS AND POLES.
C
C
C                   ELE 1 = NO. OF NUMERATOR REAL ROOTS.        NNR
C
C                   ELE 2 = NO. OF NUMERATOR COMPLEX PAIRS,     NNC
C
C                   ELE 3 = NO. OF NUMERATOR FREE S'S,          NNZ
C
C                   ELE 4 = NO. OF DENOMINATOR REAL ROOTS.      NDR
C
C                   ELE 5 = NO. OF DENOMINATOR COMPLEX PAIRS,NDC
C
C                   ELE 6 = NO. OF DENOMINATOR FREE S'S,        NDZ
C
C                   ELE 7 = BODE GAIN, KB.
C
C
C                          THE ROOTS FOLLOW IN THE ORDER
C
C
C            ELEMENT LOCATION          DESCRIPTION
C            ----------------          -----------
C
C              8 THRU NNR+7            NUMERATOR TIME CONSTANTS
C
C           NNR+8 THRU 2*NNC + 7       NUMERATOR DAMPING AND FREQUENCIES --
C
C                                      ZETA1, OMEGA1, ZETA2, OMEGA2 ---
C                          FREE S'S ARE NOT INCLUDED
C
C
C              REMAINING ELEMENTS ARE DEMONINATOR ROOTS IN SAME
C              ORDER AS NUMERATOR ROOTS, IE.,
C
C              TAU(1),...,TAU(NDR),ZETA(1),OMEGA(1),...,ZETA(NDC),OMEGA(NDC)
C
C
```

```
C
C
C
C
C
C                  THE TRANSFER FUNCTION FREQUENCY RESPONSE FOLLOWS
C
C
C---------- FREQ/RAD/SEC    FREQ/HERTZ  REAL   IMAG   AMP  DECIBELS    RAD    DEG
C                    .                   .                   .
C                    .                   .                   .
C                    .                   .                   .
C
C
C                  THE DAMPED RESONANCES (BASED ON A
C                  POLE OR ZERO) ARE IDENTIFIED
C                  BY  *****  IN BOTH THE LEFT AND RIGHT MARGINS.
C
C
C                  ***************************************
C
C
C                 THE FOLLOWING OUTPUTS ARE TYPICAL OF A ROOT
C                 LOCUS INVESTIGATION.
C
C
C----------OUTPUT MATRICES    PDEN
C                             PNUM
C
C                            THE DENOMINATOR AND NUMERATOR
C                            TRANSFER FUNCTION POLYNOMINAL
C                            COEFFICIENTS --- ASCENDING POWERS OF S.
C----------
C              P(S)  =       NUMERATOR
C                              AND
C              Q(S)  =       DEMONINATOR
C
C                  PREPROCESSED
C                  POLYNOMINAL COEFFICIENTS AS USED BY RLOCUS.
C
C
C----------  STARTING POINT =  USER IDENTIFIED STARTING POINT
C                              FOR THE LOCI.
C
C----------  SCAN LIMITS    =   LIMITS ON REAL AND IMAGINARY
C                               COMPONENTS FOR THE LOCI.
C
C
C----------    GAIN              ROOTS              ERROR
C
C
C                             ROOT LOCUS OUTPUT
C
C
C
C*****************************************************************
C*****************************************************************
```

VI. AUXILIARY PROGRAMS

This section describes two auxiliary digital codes that have been developed to aid the DISCOS program system user. The first code is a FORTRAN program which accepts the DISCOS code as input and, based upon some additional user-supplied input, automatically redimensions the DISCOS source program to minimize core storage requirements. The second code is a DISCOS/NASTRAN interface which processes user-supplied NASTRAN generated data into the required DISCOS input formats.

A. REDIM - THE REDIMENSION PROGRAM

This code was developed to aid the user in the efficient use of available digital computer core storage locations. Examination of existing digital computer codes for generalized analyses of (possibly) large systems indicates that very frequently the nature of the code dictates that a great deal of core storage locations are required (due to the sizes of program DIMENSION and COMMON blocks). This often leads to inefficient use of core storage as the user must have available sufficient core storage locations so as to satisfy the program size. As a large percentage of program executions probably don't require the maximum dimension sizes of program storage blocks, it is obvious that a automatic procedure to alter the program code to meet a user's specific requirements would be desirable. Program REDIM was developed to satisfy these requirements.

REDIM is a self-contained code that contains an extensive list of format statements. The code reads the DISCOS source code from tape as coded data and reproduces it on the tape unless it finds an identifying format number in columns 73-75. In this case, it rewrites the source code according to the format corresponding to the identification. REDIM, therefore, provides an efficient and foolproof method of recasting the source input code to meet the user's requirements.

Following is a more detailed explanation of the manner in which the REDIM program can be used.

```
C
C
C
C
C
C*********************************************************************
C*********************************************************************
C
C
C                PROGRAM REDIM INPUT DATA STREAM
C
C
C                             ********
C
C*********************************************************************
C*********************************************************************
C
C
C                THERE IS NO CARD INPUT DATA REQUIRED FOR THIS
C                PROGRAM.  IF A USER WISHES TO REDIMENSION THE
C                DISCOS PROGRAM VIA THE REDIMENSION PROGRAM
C                HE MUST MAKE THE NECESSARY ADJUSTMENTS TO
C                THE VARIABLES INDICATED BELOW (AND DEFINED BY
C                A PROGRAM DATA STATEMENT), RECOMPILE AND
C                EXECUTE THE REDIM PROGRAM
C
C
C
C
C
C                THE FOLLOWING PROGRAM DATA VARIABLES CAN BE
C                ADJUSTED
C
C                                                       CURRENT VALVE
C
C                NBMAX      = MAXIMUM NUMBER OF BODIES              6
C                NHMAX      = MAXIMUM NUMBER OF HINGES              6
C                NSPMAX     = MAXIMUM NUMBER OF SENSOR POINTS      15
C                NMWMAX     = MAXIMUM NUMBER OF MOMENTUM WHEELS     5
C
C                NMWBOD     = MAXIMUM NUMBER OF MOMENTUM
C                               WHEELS PER BODY                     4
C
C                NMDBOD     = MAXIMUM NUMBER OF ELASTIC
C                               MODES  PER BODY                    12
C
C                KY         = MAXIMUM NUMBER OF STATE VARIABLES   250
C
C                JMAXC      = MAXIMUM NUMBER OF JOINTS ON A
C                               BODY - CONSISTENT MASS              7
C
C                JMAXL      = MAXIMUM NUMBER OF JOINTS ON A
C                               BODY - LUMPED MASS                105
C
C
```

```
C
C
C
C
C
C
C                                                              CURRENT VALVE
C
C                MAXZP      = ROW DIMENSION OF TIME HISTORY
C                             PLOT DATA ARRAY                       1000
C
C                MXJVPL     = SIZE OF TIME HISTORY PLOT
C                             VARIABLE SELECTION VECTOR -- MUST
C                             NOT BE CHANGED --                       16
C
C                MAXDUM     = SIZE OF DUMMY VECTOR IN TIME
C                             HISTORY PLOT SECTION -- MUST
C                             NOT BE LESS THAN NCPLOT --            1500
C
C                MAXCNT     = SIZE OF CONTROL DATA VECTOR            100
C
C                L1         = ROW DIMENSION OF WORK SPACE
C                             THAT WILL ACCOMMODATE THE
C                             LINEARIZED COEFFICIENT MATRIX
C                             (A) IN DYNS40 SEGMENT                  100
C
C                L5         = SIZE OF FREQUENCY RESPONSE
C                             DATA VECTORS AND CORRESPONDS
C                             TO MAXIMUM NO. OF POINTS TO
C                             BE PLOTTED                             500
C
C
C
C*****************************************************************************
C*****************************************************************************
C
```

B. NASFOR - THE NASTRAN INTERFACE PROGRAM

The multi-purpose programming system described herein can be made more versatile if a reliable and efficient means to process input data arising from other sources can be provided. One other source of input data is NASTRAN, a digital code that is gaining wide acceptance in the aerospace and other industries. Program NASFOR, described in this section, was developed to provide an interface between NASTRAN and DISCOS. The program assumes NASTRAN generated structural data is available in a prescribed format and transforms these data to a format acceptable to the DISCOS system. Either tape or punch card data may be processed.

NASFOR is a self-contained code; it processes data from one source (NASTRAN) and generates data for application in DISCOS. Originally, it was felt that this interface should be an integral part of the DISCOS code. However, during development, it was realized that this would impose a large overhead on the dynamic response program and it was, therefore, decided that NASFOR should be a stand-alone program.

NASFOR has the capability to process either tape or punch card input and create either tape or punch card output. The output formats are consistent with the input requirements of the DISCOS program system. The input formats assume that NASTRAN generated data is double precision and in OUTPUT2 format if on tape or is single precision and of the format (24X,3F8.0) if on punched cards. Reference to the example following indicates the format requirements for all other date required to exercise the program.

The code was designed to process NASTRAN generated structural data for a series of bodies and to create DISCOS input compatible with subroutine MSMODL, the lumped mass input routine. It is assumed that the available data is in a specific format as follows:

For a body whose inertial and geometric characteristics are defined at a set of NJ discrete joints,

1. Inertial Properties

$$m_i = \begin{bmatrix} m & & & & S_z & -S_y \\ & m & & -S_z & & S_x \\ & & m & S_y & -S_x & \\ & & & J_{xx} & -J_{xy} & -J_{xz} \\ & (Sym) & & & J_{yy} & -J_{yz} \\ & & & & & J_{zz} \end{bmatrix}_{6x6}$$

where m = mass

$S_{x,y,z}$ = static mass moments

$J_{xx,\ldots,zz}$ = inertias

and the total assembled mass matrix is of the quasi-diagonal form

$$M = \begin{bmatrix} m_1 & & & & & \\ & m_2 & & & & \\ & & \cdot & & & \\ & & & \cdot & & \\ & & & & \cdot & \\ & & & & & m_{NJ} \end{bmatrix}_{6NJx6NJ} .$$

2. Joint Coordinate Locations

i) card input data

$$G = \begin{bmatrix} x & y & z \end{bmatrix}_{NJx3}$$

with $x = \begin{bmatrix} x_1 & x_2 & \cdot\ \cdot\ \cdot\ \cdot & x_{NJ} \end{bmatrix}^T_{1xNJ}$

$y = \begin{bmatrix} y_1 & y_2 & \cdot\ \cdot\ \cdot\ \cdot & y_{NJ} \end{bmatrix}^T_{1xNJ}$

$$z = \lfloor z_1 \; z_2 \; . \; . \; . \; z_{NJ} \rfloor^T_{1xNJ}$$

or ii) tape input data

$$G = G_1 - G_o$$

with $G_1 = \begin{bmatrix} n & x & y & z \end{bmatrix}_{NJx4}$

where $n = \lfloor n_1 \; n_2 \; . \; . \; . \; n_{NJ} \rfloor^T_{1xNJ}$ are joint numbers and will be ignored by the tape reading section and x, y and z are as above

and $G_o = \begin{bmatrix} x_o & y_o & z_o \end{bmatrix}_{NJx3}$

where x_o, y_o and z_o are user-supplied card inputs and may be null.

3. Modal Properties

$$\Phi = \begin{bmatrix} \Phi_R & \Phi_E \end{bmatrix}_{6NJxNM}$$

where Φ_R = NR rigid body modes

Φ_E = NE elastic modes and NM = NR + NE

and where

$$\phi_j = \lfloor \phi_1 \; \phi_2 \; . \; . \; . \; \phi_i \; . \; . \; . \; \phi_{NJ} \rfloor^T_{1x6NJ}$$

and $\phi_i = \lfloor h_x \; h_y \; h_z \; \sigma_x \; \sigma_y \; \sigma_z \rfloor^T_{1x6}$

with $h_{x,y,z}$ = modal displacement amplitude

$\sigma_{x,y,z}$ = modal rotation amplitude.

4. Generalized Stiffness and Damping

$$K = \begin{bmatrix} K_{gen} \end{bmatrix}_{NExNE} \quad \text{and} \quad C = \begin{bmatrix} C_{gen} \end{bmatrix}_{NExNE}$$

The data input as previously noted, are manipulated within the program and the results are written on tape and/or provided as punch card output as follows:

1. inertial properties

$$M = \begin{bmatrix} m_1 & m_2 & \cdot & \cdot & \cdot & \cdot & \cdot & m_{NJ} \end{bmatrix}^T_{1xNJ}$$

$$S = \begin{bmatrix} S_{x_1} & S_{y_1} & S_{z_1} \\ \cdot & & \\ \cdot & & \\ \cdot & & \\ S_{x_{NJ}} & S_{y_{NJ}} & S_{z_{NJ}} \end{bmatrix}_{NJx3}$$

$$J = \begin{bmatrix} J_{xx_1} & J_{yy_1} & J_{zz_1} & J_{xy_1} & J_{xz_1} & J_{yz_1} \\ \cdot & \cdot & \cdot & & & \\ \cdot & \cdot & \cdot & & & \\ \cdot & \cdot & \cdot & & & \\ J_{xxNJ} & J_{yyNJ} & J_{zzNJ} & J_{xyNJ} & J_{xzNJ} & J_{yzNJ} \end{bmatrix}_{NJx6} .$$

2. joint coordinate locations

$$G = \begin{bmatrix} x_1 & y_1 & z_1 \\ \cdot & & \\ \cdot & & \\ \cdot & & \\ x_{NJ} & y_{NJ} & z_{NJ} \end{bmatrix}_{NJx3} .$$

3. modal properties

$$h_{x,y,z} = \begin{bmatrix} h^{1,1}_{x,y,z} & h^{1,2}_{x,y,z} & \cdots & h^{1,NE}_{x,y,z} \\ \vdots & & & \\ h^{NJ,1}_{x,y,z} & \cdots & \cdots & h^{NJ,NE}_{x,y,z} \end{bmatrix}_{NJxNE}$$

$$\sigma_{x,y,z} = \begin{bmatrix} \sigma^{1,1}_{x,y,z} & \sigma^{1,2}_{x,y,z} & \cdots & \sigma^{1,NE}_{x,y,z} \\ \vdots & & & \\ \sigma^{NJ,1}_{x,y,z} & \cdots & \cdots & \sigma^{NJ,NE}_{x,y,z} \end{bmatrix}_{NJxNE}$$

4. generalized stiffness and damping

$$K_{gen} = \begin{bmatrix} K_{11} & K_{12} & \cdots & K_{1,NE} \\ \vdots & & & \\ K_{NE,1} & \cdots & \cdots & K_{NE,NE} \end{bmatrix}_{NExNE}$$

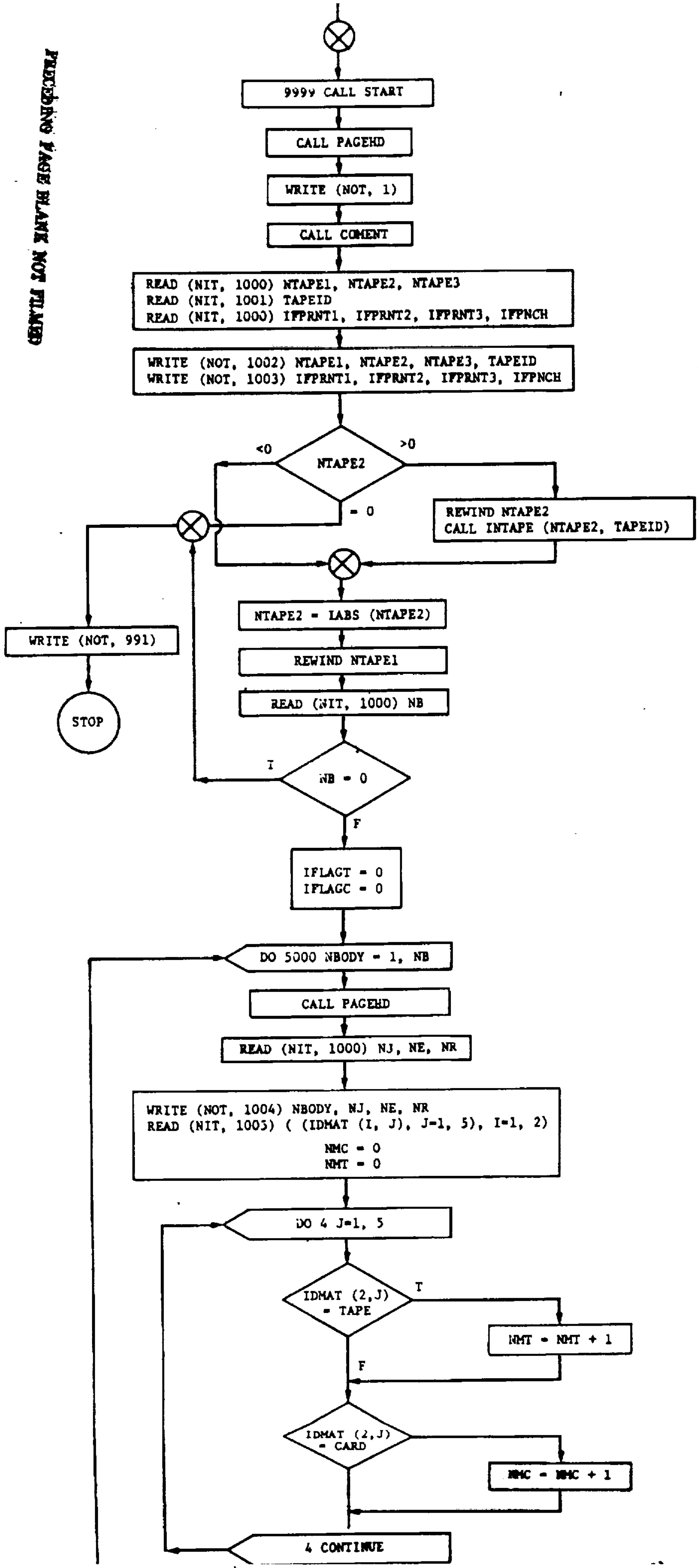
$$C_{gen} = \begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1,NE} \\ \vdots & & & \\ C_{NE,1} & \cdots & \cdots & C_{NE,NE} \end{bmatrix}_{NExNE}$$

NASFOR consists of a main program, 8 program subroutines to process the data and 10 auxiliary input/output routines. The function of the program routines is indicated in Table VI.B-1. and a logic flow diagram appears as Figure VI.B-1.

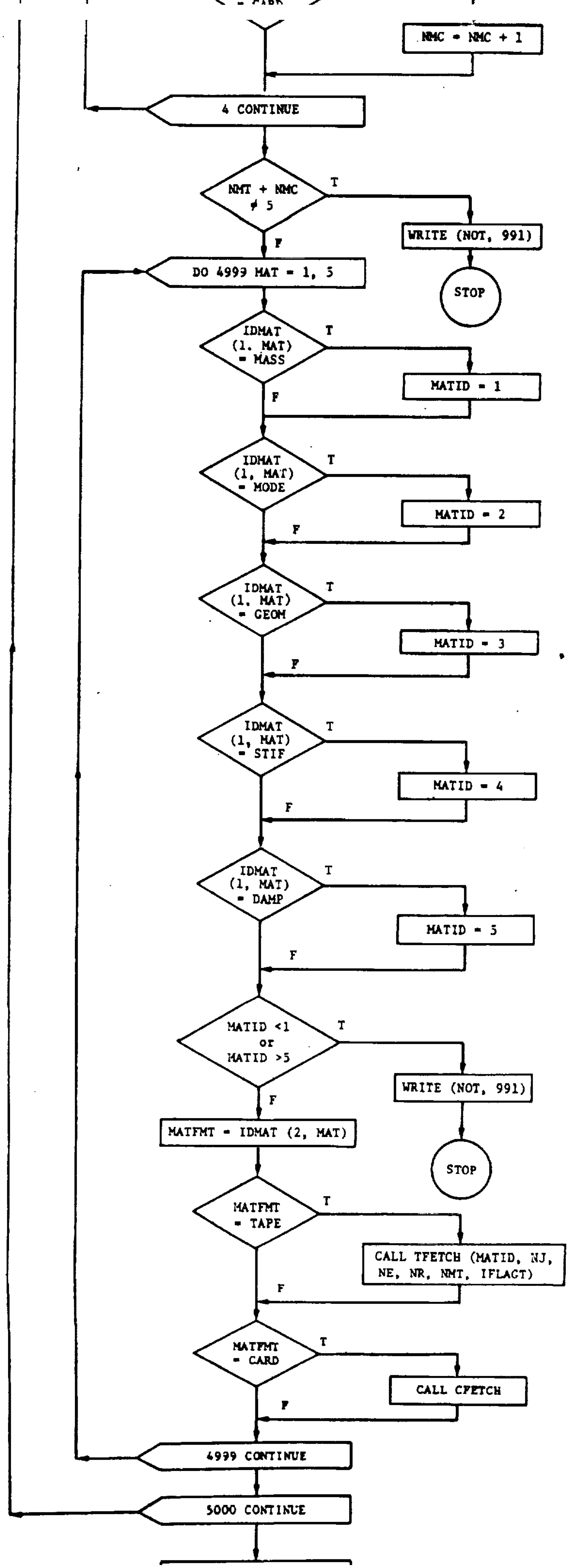
Table VI.B-1. Description of NASFOR Subroutines

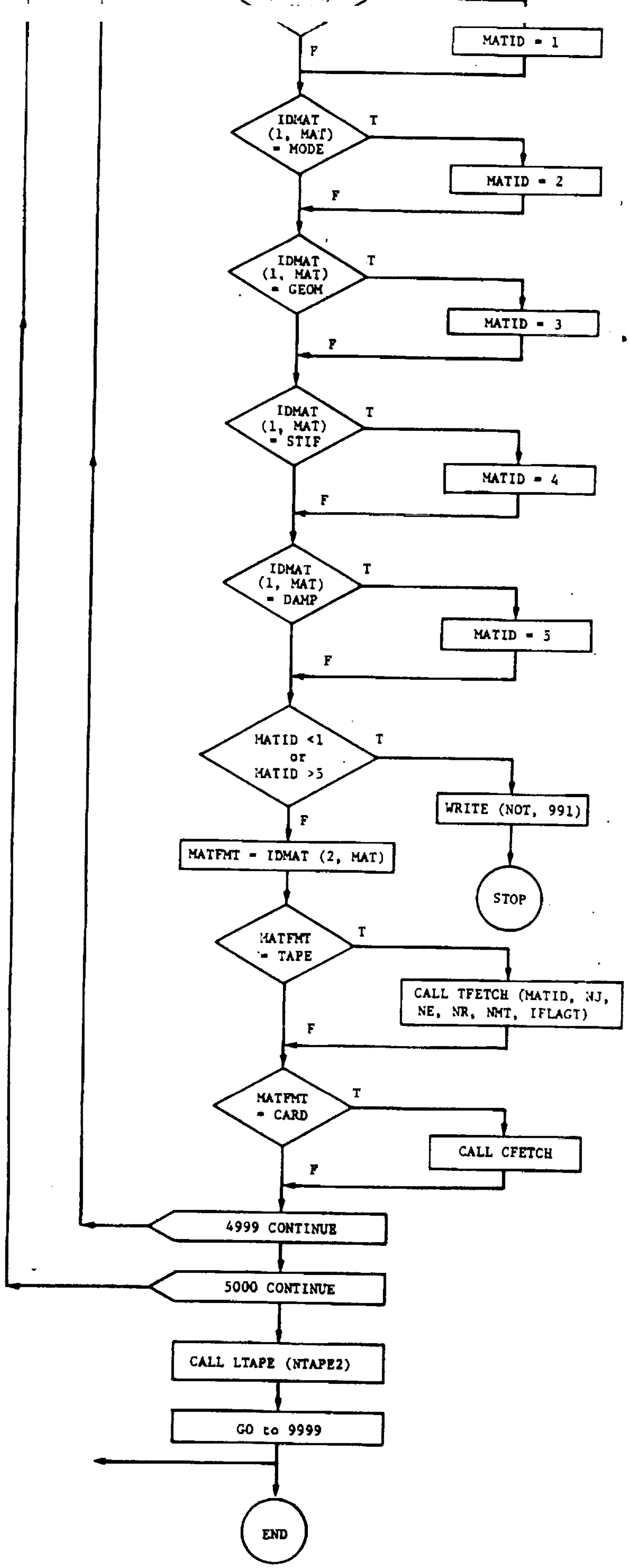
Subroutine	Function
MAIN	Program control
TFETCH	Fetch a matrix from NASTRAN tape
CFETCH	Fetch a matrix from cards
GMASS	Generate mass data
GMODE	Generate modal displacement data
GGEOM	Generate geometric data
GSTIF	Generate stiffness data
GDAMP	Generate damping data
PRINTT	Print entire input tape (on option)

The code has been designed to minimize input data requirements yet provide a high degree of flexibility. Following is a detailed explanation of the input data stream requirements.


9999 CALL START
CALL PAGEHD
WRITE (NOT, 1)
CALL COMENT
READ (NIT, 1000) NTAPE1, NTAPE2, NTAPE3
READ (NIT, 1001) TAPEID
READ (NIT, 1000) IFPRNT1, IFPRNT2, IFPRNT3, IFPNCH
WRITE (NOT, 1002) NTAPE1, NTAPE2, NTAPE3, TAPEID
WRITE (NOT, 1003) IFPRNT1, IFPRNT2, IFPRNT3, IFPNCH
NTAPE2
<0
>0
= 0
REWIND NTAPE2
CALL INTAPE (NTAPE2, TAPEID)
WRITE (NOT, 991)
STOP
NTAPE2 = IABS (NTAPE2)
REWIND NTAPE1
READ (NIT, 1000) NB
NB = 0
T
F
IFLAGT = 0
IFLAGC = 0
DO 5000 NBODY = 1, NB
CALL PAGEHD
READ (NIT, 1000) NJ, NE, NR
WRITE (NOT, 1004) NBODY, NJ, NE, NR
READ (NIT, 1005) ((IDMAT (I, J), J=1, 5), I=1, 2)
NMC = 0
NMT = 0
DO 4 J=1, 5
IDMAT (2,J) = TAPE
T
NMT = NMT + 1
F
IDMAT (2,J) = CARD
NMC = NMC + 1
4 CONTINUE

FOLDOUT FRAME 3





1000 FORMAT (16I5)
1001 FORMAT (12A6)
1005 FORMAT (5 (A4, 6X))

Figure VI.B-1. Logic Flow, Program MASPOR

```
C
C
C
C
C
C**** ******************************************************************
C**** ******************************************************************
C
C
C                 PROGRAM NASFOR INPUT DATA STREAM
C
C
C                             *******
C
C**********************************************************************
C**********************************************************************
C
C
C
C          PROGRAM READS NASTRAN OUTPUT TAPE AND/OR CARDS,
C          MANIPULATES THE DATA AND WRITES (PUNCHES) TAPE
C          (CARDS) TO BE USED AS INPUT TO PROGRAM DISCOS
C
C
C          PURPOSE -- PROCESS NASTRAN STRUCTURAL DATA
C                     FOR A SERIES OF FLEXIBLE BODIES
C                     AND GENERATE DATA CONSISTENT
C                     WITH DISCOS SUBROUTINE MSMODL
C                     INPUT REQUIREMENTS
C
C
C          ASSUMES -- FOLLOWING NASTRAN DATA AVAILABE
C                     FOR ALL FLEXIBLE BODIES
C
C                       MATRIX                 SIZE
C                    -----------            ---------
C
C                    MASS - LUMPED           6NJ X 6NJ
C                    MODES                   6NJ X(NE+NR)
C                    GEOMETRY                 NJ X 4
C                    STIFFNESS - MODAL        NE X NE
C                    DAMPING   - MODAL        NE X NE
C
C
```

```
C
C
C
C
C
C
C
C                        WHERE  NJ = NO. OF JOINTS ON BODY
C                               NE = NO. OF ELASTIC MODES RETAINED FOR BODY
C                               NR = NO. OF RIGID BODY MODES
C
C
C          ASSUMES --
C
C                    1.  NASTRAN MATRICES WRITTEN DOUBLE PRECISION
C                        (REAL*8 OR ITYPE = 2) IF ON TAPE
C
C                    2.  NASTRAN MATRICES WRITTEN SINGLE PRECISION
C                        (FORMAT = 24X,3F8.0) IF ON CARDS
C
C                    3.  MAX(NJ) = 1000
C                        MAX(NE) = 10     -- REDIMENSION IF SO REQUIRED
C                        MAX(NR) = 6
C
C
C          NCTE -
C
C              C-----------INDICATES WHERE DATA INPUT
C                          IS REQUIRED
C
C                          NIT = INPUT TAPE NUMBER
C
C
C
C
C---------9999  CALL START   (SEE FOLLOWING)
C
C
C-------------- CALL COMENT  (SEE FOLLOWING)
C
C
C-------------- READ(NIT,FORMAT = 3I5)  NTAPE1, NTAPE2, NTAPE3
C
C                    NTAPE1 = LOGICAL UNIT NASTRAN INPUT TAPE (EG, 20)
C                    NTAPE2 = LOGICAL UNIT OUTPUT TAPE        (EG, 30)
C                    NTAPE3 = SCRATCH                         (EG, 40)
C
C
C-------------- READ(NIT,FORMAT = A6) TAPEID
C
C                    TAPEID = 6 CHARACTER TAPE IDENTIFICATION (EG, L12345)
C
```

```
C
C
C
C
C
C------------- READ(NIT,FORMAT = 4I5) IFPRT1, IFPRT2, IFPRT3, IFPNCH
C
C                    IFPRT1 NE 0   PRINT NASTRAN DATA
C                    IFPRT2 NE 0   PRINT OUTPUT DATA
C                    IFPRT3 NE 0   PRINT NASTRAN HEX TAPE IF ERROR FOUND
C                    IFPNCH NE 0   PUNCH OUTPUT DATA
C
C
C------------- READ(NIT,FORMAT = I5) NB
C
C                    NB = NO. OF BODIES FOR WHICH DATA AVAILABLE
C                         ON THE NASTRAN TAPE (EG, 2)
C
C
C              IF(NB .EQ. 0) STOP      WRITE ERROR MESSAGE
C                                      AND TERMINATE
C
C
C              THE FOLLOWING LOOP IS EXECUTED FOR EACH BODY
C
C
C              DO 5000  NBODY = 1,NB
C
C
C------------- READ(NIT,FORMAT = 3I5) NJ, NE, NR
C
C                    NJ = NO. OF JOINTS ON BODY (EG, 11)
C                    NE = NO. OF ELASTIC MODE RETAINED FOR BODY (EG, 5)
C                    NR = NO. OF RIGID BODY MODES (EG, 0)
C
C
C------------- READ(NIT,FORMAT = 5(A4,6X)) ((IDMAT(I,J),J=1,5),I=1,2)
C
C                    IDMAT IS AN INPUT ARRAY OF SIZE 2 X 5 THAT
C                    DEFINES THE ORDER OF INPUT DATA TO BE READ
C                    FROM TAPE OR CARDS BY MATRIX FUNCTION AND
C                    ALSO SPECIFIES WHETHER DATA IS TO BE READ
C                    FROM TAPE OR CARDS
C
C                    THE DATA WILL BE PROCESSED IN THE ORDER J=1,2,...,5
C
C
```

```
C
C
C
C
C
C
C                  AN EXAMPLE IDMAT ARRAY IS
C
C                 ***                                                   ***
C         IDMAT = *                                                       *
C                 *GEOM      MASS       MODE       STIF       DAMP*
C                 *                                                       *
C                 *CARD      TAPE       TAPE       TAPE       TAPE*
C                 *                                                       *
C                 ***                                                   ***
C
C
C                  AND INDICATES DATA TO BE PROCESSED AS
C
C                       1. JOINT GEOMETRIC DEFINITION FROM CARDS
C                       2. JOINT LUMPED MASS DATA FROM TAPE
C                       3. ELASTIC MODAL DATA FROM TAPE
C                       4. MODAL STIFFNESS FROM TAPE
C                       5. MODAL DAMPING FROM TAPE
C
C
C              THE FOLLOWING LOOP IS EXECUTED 5 TIMES PER BODY
C
C
C              DO 4999 MAT = 1,5
C
C              MATFMT = IDMAT(2,MAT)
C
C------------- IF(MATFMT .EQ. TAPE) CALL TFETCH(...,....,....,...,..,....)
C
C------------- IF(MATFMT .EQ. CARD) CALL CFETCH
C
C                  TFETCH IS TAPE PROCESSING CONTROL SUBROUTINE
C                  CFETCH IS CARD PROCESSING CONTROL SUBROUTINE
C
C                  SEE FOLLOWING FOR DESCRIPTION
C
C        4999  CONTINUE
C
C        5000  CONTINUE
C
C              GO TO 9999
C
C              END
C
C
C
C
```

```
C
C
C
C
C
C***********************************************************************
C*
C*              SUBROUTINE START  - INPUT IDENTIFICATIONS
C*
C***********************************************************************
C
C-------------- READ(NIT,FORMAT = A6,4X,3A6) IRUNNO (UNAME(I),I=1,3)
C
C                    IRUNNO = RUN IDENTIFICATION (6 CHARACTERS)
C                    UNAME  = USERS NAME (18 CHARACTERS)
C
C                   IRUNNO EQ 4HSTOP - TERMINATE THE RUN
C                   IRUNNO NE 4HSTOP - CONTINUE THE RUN
C
C
C-------------- READ(NIT,FORMAT = 12A6) (TITLE1(I),I=1,12)
C-------------- READ(NIT,FORMAT = 12A6) (TITLE2(I),I=1,12)
C
C                   TITLE1 = 72 CHARACTER TITLE
C                   TITLE2 = 72 CHARACTER TITLE
C
C               RETURN
C               END
C
C
C
C
C
C***********************************************************************
C*
C*              SUBROUTINE COMENT  - INPUT USER SUPPLIED COMMENTS
C*
C***********************************************************************
C
C-------------- READ(NIT,FORMAT = 13A6,1X,A1) (IREMRK(I),I=1,13), IPGHD
C
C                    IREMRK = 78 CHARACTER COMMENT
C                    IPGHD  = NEW PAGE FLAG
C
C                        IPGHD = 1HP -- NEW PAGE BEFORE PRINTING
C                        THERE IS NO LIMIT TO THE NUMBER OF COMMENT
C                        CARDS BUT THE LAST ONE MUST CONTAIN ZERO
C                        IN COLUMNS 1 THRU 10
C
C               RETURN
C               END
C
C
```

```
C
C
C***********************************************************************
C*
C*             SUBROUTINE TFECTH
C*
C***********************************************************************
C
C                   THIS SUBROUTINE READS NO INPUT DATA
C                   DIRECTLY BUT DOES CALL AUXILIARY
C                   SUBROUTINES DEPENDING UPON THE VALUE
C                   OF THE INPUT ARGUMENT -MATID-
C
C                   THESE AUXILIARY SUBROUTINES READ
C                   DATA AS DESCRIBED LATER
C
C-------------- IF  (MATID  .EQ.  1) CALL GMASS(..)    LUMPED MASS DATA
C-------------- IF  (MATID  .EQ.  2) CALL GMODE(..)    MODAL DATA
C-------------- IF  (MATID  .EQ.  3) CALL GGEOM(..)    GEOMETRIC DATA
C-------------- IF  (MATID  .EQ.  4) CALL GSTIF(..)    STIFFNESS DATA
C-------------- IF  (MATID  .EQ.  5) CALL GDAMP(..)    DAMPING DATA
C
C              RETURN
C              END
C
C
C
C
C
C***********************************************************************
C*
C*             SUBROUTINE CFETCH
C*
C***********************************************************************
C
C                   THIS SUBROUTINE READS CARD INPUT DATA AS FOLLOWS --
C
C-------------- READ(NIT,FORMAT = A6,4X,2I5) ANAME, NR, NC
C
C                    ANAME = 6 CHARACTER MATRIX IDENTIFICATION
C                    NR    = NO. OF ROWS IN MATRIX
C                    NC    = NO. OF COLS IN MATRIX
C
C              DO 100 I = 1,NR
C
C-------------- READ(NIT,FORMAT = 24X,3F8.0) (WR(J),J=1,NC)
C
C                   (WR(J),J=1,2,...,NC) = INPUT ROW I OF MATRIX
C
C          100 CONTINUE
C
C              RETURN
C              END
C
```

```
C
C
C
C
C
C*************************************************************************
C*
C*               SUBROUTINE GMASS
C*
C*************************************************************************
C
C                      SUBROUTINE READS OUTPUT TITLE CARD FOR
C                      EACH BODYS MASS, STATIC MASS MOMENT AND
C                      INERTIAL DATA
C
C                      DATA READ ON FIRST CALL FOR EACH BODY
C
C-------------- READ(NIT,FORMAT = 3(A6,4X)) AN1, AN2, AN3
C
C                    AN1 = 6 CHARACTER OUTPUT TITLE FOR MASSES
C                    AN2 = 6 CHARACTER OUTPUT TITLE FOR STATIC MOMENTS
C                    AN3 = 6 CHARACTER OUTPUT TITLE FOR INERTIAS
C
C               RETURN
C               END
C
C
C
C
C
C*************************************************************************
C*
C*               SUBROUTINE GMODE
C*
C*************************************************************************
C
C                      SUBROUTINE READS OUTPUT TITLE CARD FOR
C                      EACH BODYS MODAL DATA
C
C                      DATA READ ON FIRST CALL FOR EACH BODY
C
C-------------- READ(NIT,FORMAT = 6(A6,4X)) AN1, AN2, AN3, AN4, AN5, AN6
C
C                    AN1 = 6 CHARACTER OUTPUT TITLE FOR HX MODAL AMPS
C                    AN2 = 6 CHARACTER OUTPUT TITLE FOR HY MODAL AMPS
C                    AN3 = 6 CHARACTER OUTPUT TITLE FOR HZ MODAL AMPS
C                    AN4 = 6 CHARACTER OUTPUT TITLE FOR SIGX MODAL SLOPES
C                    AN5 = 6 CHARACTER OUTPUT TITLE FOR SIGY MODAL SLOPES
C                    AN6 = 6 CHARACTER OUTPUT TITLE FOR SIGZ MODAL SLOPES
C
C               RETURN
C               END
C
C
```

```
C
C
C
C
C***********************************************************************
C*
C*              SUBROUTINE GGEOM
C*
C***********************************************************************
C
C                    SUBROUTINE READS INITIAL GEOMETRIC DATA TO
C                    BE MANIPULATED WITH INPUT TAPE GEOMETRIC
C                    DATA AND OUTPUT TITLE CARD FOR EACH
C                    BODIES GEOMETRIC DATA
C                    DATA READ ON FIRST CALL FOR EACH BODY
C                    INPUT FORMAT FOR SUBROUTINE READ EXPLAINED
C                    ELSEWHERE IN THIS DOCUMENT
C
C                    INITIAL GEOMETRIC DATA (MAY BE NULL) IS
C                    PROVIDE TO ACCOUNT FOR POSSIBLE
C                    GEOMETRIC OFFSETS
C
C------------- CALL READ (WS,NR,NC,KWS,LWS)
C
C                     NR = INPUT NO. OF ROWS (MUST = NJ)
C                     NC = INPUT NO. OF COLS (MUST = 3 )
C                     WS = INPUT MATRIX OF INITIAL
C                          GEOMETRIC COORDINATES -- SIZE = NJ X 3
C
C                          THE FINAL GEOMETRY FOR NJ JOINTS
C                          WILL BE COMPUTED AS
C
C               ***                   ***     ***       ***     ***       ***
C               * X(1)    Y(1)    Z(1) *     *           *     *           *
C               *   .       .       .  *     * X   Y   Z *     * X   Y   Z *
C               *   .       .       .  *  =  *           *  -  *           *
C               *   .       .       .  *     * (TAPE)    *     * (CARD)    *
C               *   .       .       .  *     *           *     *           *
C               *   .       .       .  *     *           *     *           *
C               * X(NJ)   Y(NJ)   Z(NJ)*     *           *     *           *
C               ***                   ***     ***       ***     ***      ****
C
C                     SO THAT, FOR THE ITH JOINT
C
C                          X(I) = X(I) FROM TAPE INPUT - X(I) READ ABOVE
C                          Y(I) = Y(I) FROM TAPE INPUT - Y(I) READ ABOVE
C                          Z(I) = Z(I) FROM TAPE INPUT - Z(I) READ ABOVE
C
C------------- READ(NIT,FORMAT = A6) AN1
C
C                   AN1 = 6 CHARACTER OUTPUT TITLE FOR GEOMETRIC DATA
C
C              RETURN
C              END
C
```

```
C
C
C
C
C
C***********************************************************************
C*
C*              SUBROUTINE GSTIF
C*
C***********************************************************************
C
C
C                     SUBROUTINE READS OUTPUT TITLE CARD FOR
C                     EACH BODYS MODAL STIFFNESS DATA
C
C                     DATA READ ON FIRST CALL FOR EACH BODY
C
C
C-------------- READ(NIT,FORMAT = A6) AN1
C
C                  AN1 = 6 CHARACTER OUTPUT TITLE FOR
C                        MODAL STIFFNESS DATA
C
C
C               RETURN
C               END
C
C
C
C
C
C
C***********************************************************************
C*
C*              SUBROUTINE GDAMP
C*
C***********************************************************************
C
C
C                     SUBROUTINE READS OUTPUT TITLE CARD FOR
C                     EACH BODYS MODAL DAMPING DATA
C
C                     DATA READ ON FIRST CALL FOR EACH BODY
C
C
C-------------- READ(NIT,FORMAT = A6) AN1
C
C                  AN1 = 6 CHARACTER OUTPUT TITLE FOR
C                        MODAL DAMPING DATA
C
C
C               RETURN
C               END
C
C
C
```

APPENDIX A--INPUT/OUTPUT SUBROUTINE EXPLANATIONS

This appendix presents excerpts from *Synthesis of Dynamic Systems Using FORMA--Fortran Matrix Analysis*, MCR-71-75, Martin Marietta Corporation, Denver, Colorado, May 1971, that explain the subroutines from the FORMA library used in the digital computer program.

Subroutine COMENT reads input comment cards and reproduces each card in the printed output of the computer run. Each comment card may have any keypunch symbol in card columns 1 thru 78. A use of COMENT is to print an explanation of coordinates used in a computer run. Thus, this information is always retained with a run to correlate matrix location numbers with physical coordinates.

R. L. Wohlen
May 1971

Last Revision: R. L. Wohlen
Sept. 1971

Subroutine INTAPE initializes a tape (a disk is preferred. See writeup of Subroutine WTAPE.) for the FORMA tape system by writing EOT (end of tape) at the beginning of the tape (disk). All FORMA tape subroutines recognize this EOT as being the end of written data. Each "new" tape (disk) must be initialized with this Subroutine INTAPE to make the tape (disk) compatible with the other FORMA tape subroutines (LTAPE, RTAPE, WTAPE, and UPDATE).

A "new" tape (disk) is defined as a tape (disk) for which it is desired to start writing matrix data at the front of the tape (disk). Thus, a "new" tape (disk) could be one with obsolete FORMA matrix data on it as well as one that has never been written on by the FORMA system.

As an example, pertinent statements from a program containing INTAPE could be:

```
      DATA  NIT,NOT/5,6/
 1001 FORMAT (12A6)
      NRTAPE = 10
      READ (NIT,1001) IFINIT, TAPEID
      IF (IFINIT  .EQ. 6HINITIL) CALL INTAPE(NRTAPE,TAPEID)
          .
          .
          .
```

The input data (starting in card column 1) to this example program would be:

either INITILTXXXX, if the tape is to be initialized. (TXXXX represents the particular tape number used, e.g., T1234);

or NOINIT, if the tape is *not* to be initialized. The tape identification is not needed.

R. L. Wohlen
May 1971

Subroutine LTAPE lists the matrix headings (see Subroutine WTAPE writeup) written on a FORMA tape (or disk). These matrix headings were written by Subroutine WTAPE and consist of:

NO. = Matrix number on tape;

RUN NO. = Run number of problem when matrix was written on tape;

NAME = Matrix name;

NROWS = Number of rows of matrix;

NCOLS = Number of columns of matrix;

DATE = Date when matrix was written on tape;

NNZ = Number of nonzeros (just used in sparse FORMA where only nonzeros are used);

PARTITION = Partition number of sparse matrix.

R. L. Wohlen
May 1971

Subroutine READ reads a matrix of real numbers (a FORTRAN term for numbers with a decimal point) from either cards or tape into the computer. The matrix is then printed so that these input data are recorded with the answers of a run. A print suppression option is available for a matrix read from tape. On option, the matrix read from either cards or tape may be written on a tape (by Subroutine WTAPE).

The first data card read by Subroutine READ contains the information to indicate whether cards or tape will be used. The information entered on this card (and subsequent cards for card input) is given below.

Card Data Input Form

Required entries are denoted by an * symbol below. Any other entry is optional.

	Card Columns	Format Type (1)		Entry
First Card	1-6	A		*Matrix Name. Will appear in printout.
	7-10	I		*Matrix Row Size.
	11-15	I		*Matrix Column Size.
	16-69	A		Any remarks to further identify the input matrix.
Write-Tape Options	72			$. Only if the Write-Tape is to be initialized by Subroutine INTAPE. The Write-Tape identification will be from card columns 73-78.
	72		or	Anything other than $ is the Write-Tape is not to be initialized.
	73-78	A		The Write-Tape identification. (e.g., T1234). Use with $ in card column 72.
	73-78		or	REWIND. The Write-Tape will be rewound before being used.
	73-76		or	LIST. The Write-Tape will be listed by Subroutine LTAPE after the matrix has been written on the Write-Tape.
	73-78		or	Anything else will be ignored.
	79-80	I		The Write-Tape Number. (e.g., 21).
			or	Blank if the matrix is not to be written on tape.

	Card Columns	Format Type (1)	Entry
Middle Cards	1-5	I	*Row Number of matrix elements on card.
	6-10	I	*Column Number of matrix element in first data field.
	11-27	E	*First data field with matrix elements. (2)
	28-44	E	*Second data field with matrix elements. (2)
	45-61	E	*Third data field with matrix elements. (2)
	62-78	E	*Fourth data field with matrix elements. (2)
Last Card	1-10	I	*Ten zeroes.

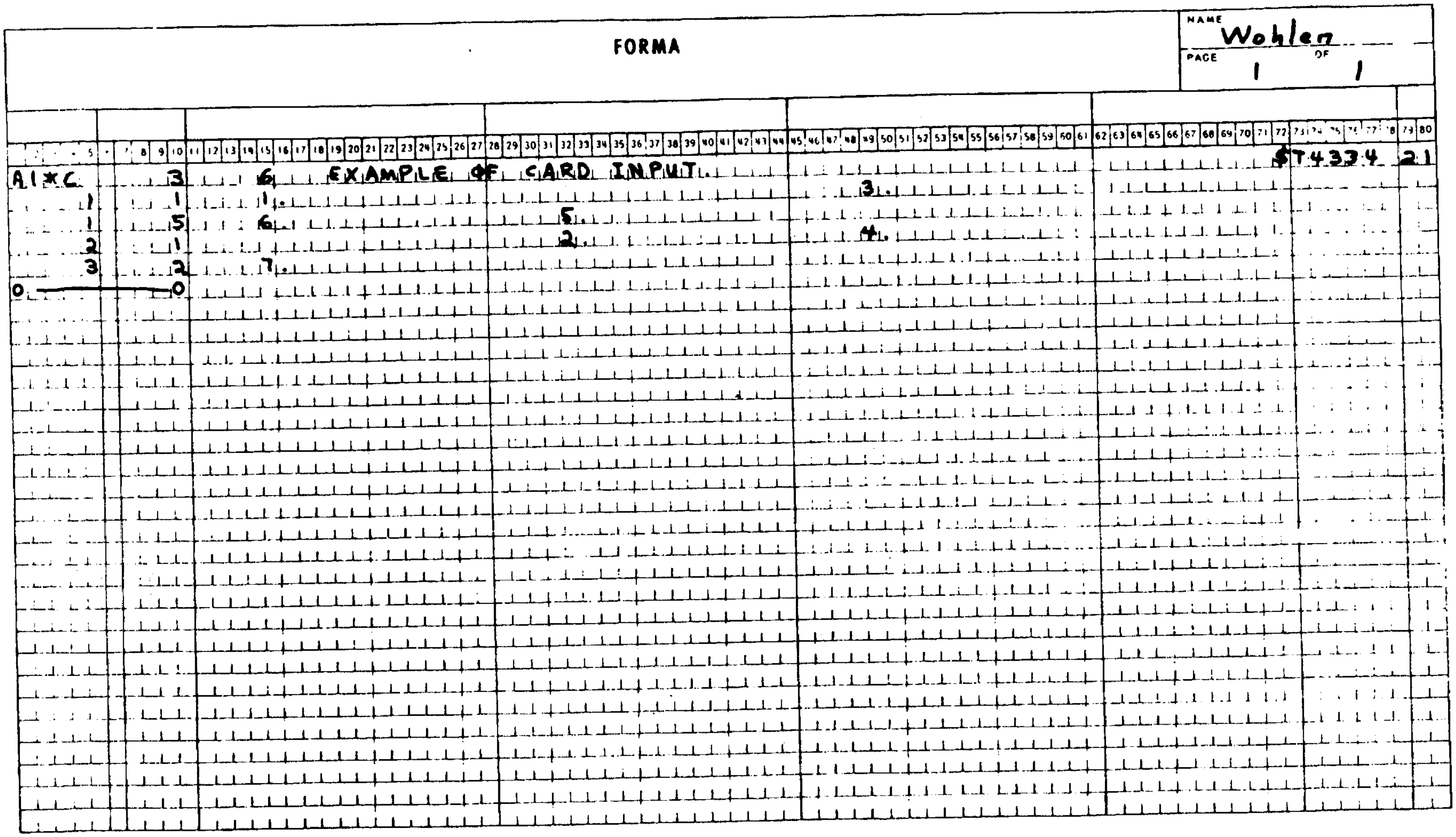
Note (1) Format Type A allows any keypunch symbol. Format Type I allows only integer numbers right justified in the field. Format Type E allows only real numbers (a FORTRAN term for numbers with a decimal point) anywhere in the field.

Note (2) Only nonzero elements need be entered.

As an example of card input to Subroutine READ consider the following matrix:

$$[A1*C]_{3x6} = \begin{bmatrix} 1. & 0. & 3. & 0. & 6. & 5. \\ 0. & 2. & 4. & 0. & 0. & 0. \\ 0. & 7. & 0. & 0. & 0. & 0. \end{bmatrix}.$$

This matrix is also to be written on tape number 21 that is to be initialized and identified as T4334. Figure 1 demonstrates how this information could be written on a coding form to facilitate keypunching to cards.



FORMA

NAME Wohlen
PAGE 1 OF 1

```
A1*C     3    6   EXAMPLE OF CARD INPUT.                                   $T4334 21
    1    1    1.                                3.
    1    5    6.               5.
    2    1                     2.               4.
    3    2    7.
0        0
```

Figure 1 Example of Card Input for Subroutine READ

Tape Data Input Form

Required entries are denoted with an * symbol below. Any other entry is optional. Only one card is used for each matrix read.

	Card Columns	Format Type (1)	Entry
One Card	1-6	A	*Name of matrix to be read from the Read-Tape.
	10		Zero. The Read-Tape will move forward from its present position and search to the end of the tape. If the matrix is not found upon the first end-of-tape encounter, the tape will automatically rewind and make one more pass. If it is not found on the second end-of-tape encounter, an error message will be printed and the program will stop.
	7-10	I or	Minus the location number of matrix on the Read-Tape. Tape will be positioned at the beginning of the location specified and then continue as described above for a zero in column 10.
	11-15	I	*The Read-Tape Number. (e.g., 11). If positive, the matrix read will be printed in the output. If negative, the matrix read will not be printed in the output.
	16-21	A	*Run number of matrix to be read from the Read-Tape.
	22-27		REWIND. The Read-Tape will be rewound before being used.
	22-25	or	LIST. The Read-Tape will be listed by Suoroutine LTAPE.
	22-27	or	Anything else will be considered as part of the remarks described below.

	Card Columns	Format Type (1)	Entry
	28-69	A	Any remarks to further identify the input matrix.
Write-Tape Options	72		$. Only if the Write-Tape is to be initialized by Subroutine INTAPE. The Write-Tape identification will be from card columns 73-78.
	72	or	Anything other than $ if the Write-Tape is not to be initialized.
	73-78	A	The Write-Tape identification. (e.g., T1234). Use with $ in card column 72.
	73-78	or	REWIND. The Write-Tape will be rewound before being used.
	73-76	or	LIST. The Write-Tape will be listed by Subroutine LTAPE after the matrix has been written on the Write-Tape.
	73-78	or	Anything else will be ignored.
	79-80	I	The Write-Tape Number. (e.g., 21).
		or	Blank if the matrix is not to be written on tape.

Note (1) Format Type A allows any keypunch symbol.
Format Type I allows only integer numbers right justified in the field.

As examples of tape input to Subroutine Read consider:

Example 1. A matrix named AB2 with run number of RUN-46 is to be read from tape number 11 into the computer and printed. This matrix is also to be written on tape number 22 that is to be initialized and identified as T4321.

Example 2. A matrix named XYZ4 with run number of TKD is on tape number 13 twice. The first time is at location 29 and the second time is at location 54. It is desired to read the second matrix.

Figure 2 demonstrates how these two examples would be written on a coding form to facilitate keypunching to cards.

FORMA

NAME: Wohlen
PAGE 1 OF 1

1–5	6	7–10	11–27	28–44	45–61	62–72	73–78	79–80
AP2		0	11RUN-46	EXAMPLE 1.			$T4321	22
XY24		-54	12TKD	EXAMPLE 2.				

Figure 2 Examples of Tape Input for Subroutine READ

Subroutine READIM reads a matrix of integer numbers from either cards or tape into the computer. The matrix is then printed so that these input data are recorded with the answers of a run. A print suppression option is available for a matrix read from tape. On option, the matrix read from either cards or tape may be written on a tape (by Subroutine WTAPE).

The first data card read by Subroutine READIM contains the information to indicate whether cards or tape will be used. The information entered on this card (and subsequent cards for card input) is given below.

Card Data Input Form

Required entries are denoted by an * symbol below. Any other entry is optional.

	Card Columns	Format Type (1)		Entry
First Card	1-6	A		*Matrix Name. Will appear in printout.
	7-10	I		*Matrix Row Size.
	11-15	I		*Matrix Column Size.
	16-69	A		Any remarks to further identify the input matrix.
Write-Tape Options	72			$. Only if the Write-Tape is to be initialized by Subroutine INTAPE. The Write-Tape identification will be from card columns 73-78.
Write-Tape Options	72		or	Anything other than $ if the Write-Tape is not to be initialized.
Write-Tape Options	73-78	A		The Write-Tape identification. (e.g., T1234). Use with $ card column 72.
Write-Tape Options	73-78		or	REWIND. The Write-Tape will be rewound before being used.
Write-Tape Options	73-76		or	LIST. The Write-Tape will be listed by Subroutine LTAPE after the matrix has been written on the Write-Tape.
Write-Tape Options	73-78		or	Anything else will be ignored.
Write-Tape Options	79-80	I		The Write-Tape Number. (e.g., 21).
Write-Tape Options			or	Blank if the matrix is not to be written on tape.

	Card Columns	Format Type (1)	Entry
Middle Cards	1-5	I	*Row Number of matrix elements on card.
	6-10	I	*Column Number of matrix element in first data field.
	11-15	I	*First data field with matrix elements. (2)
	16-20	I	*Second data field with matrix elements. (2)
	etc 76-80	I	*Fourteenth data field with matrix elements. (2)
Last Card	1-10	I	*Ten zeroes.

Note (1) Format Type A allows any keypunch symbol. Format Type I allows only integer numbers right justified in the field.

Note (2) Only nonzero elements need be entered.

As an example of card input to Subroutine READIM consider the following matrix:

$$[A1*C]_{3x6} = \begin{bmatrix} 1 & 0 & 3 & 0 & 6 & 5 \\ 0 & 2 & 4 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

This matrix is also to be written on tape number 21 that is to be initialized and identified as T4334. Figure 1 demonstrates how this information could be written on a coding form to facilitate keypunching to cards.

FORMA

NAME Wohlen

PAGE 1 OF 1

```
         1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890
AIKC     3    6   EXAMPLE OF CARD INPUT.                                 $T4334 21
    1    1    1           3       6    5
    2    1        2       4
    3    2    7
0        0
```

Figure 1 Example of Card Input for Subroutine READIM

Tape Data Input Form

Required entries are denoted with an * symbol below. Any other entry is optional. Only one card is used for each matrix read.

	Card Columns	Format Type (1)	Entry
One Card	1-6	A	*Name of matrix to be read from the Read-Tape.
	10		Zero. The Read-Tape will move forward from its present position and search to the end of the tape. If the matrix is not found upon the first end-of-tape encounter, the tape will automatically rewind and make one more pass. If it is not found on the second end-of-tape encounter, an error message will be printed and the program will stop.
	7-10	I or	Minus the location number of matrix on the Read-Tape. Tape will be positioned at the beginning of the location specified and then continue as described above for a zero in column 10.
	11-15	I	*The Read-Tape Number. (e.g., 11). If positive, the matrix read will be printed in the output. If negative, the matrix read will not be printed in the output.
	16-21	A	*Run number of matrix to be read from the Read-Tape.
	22-27		REWIND. The Read-Tape will be rewound before being used.
	22-25	or	LIST. The Read-Tape will be listed by Subroutine LTAPE.
	22-27	or	Anything else will be considered as part of the remarks described below.

	Card Columns	Format Type (1)	Entry
	28-69	A	Any remarks to further identify the input matrix.
Write-Tape Options	72		$. Only if the Write-Tape is to be initialized by Subroutine INTAPE. The Write-Tape identification will be from card columns 73-78.
Write-Tape Options	72	or	Anything other than $ if the Write-Tape is not to be initialized.
Write-Tape Options	73-78	A	The Write-Tape identification. (e.g., T1234). Use with $ in card column 72.
Write-Tape Options	73-78	or	REWIND. The Write-Tape will be rewound before being used.
Write-Tape Options	73-76	or	LIST. The Write-Tape will be listed by Subroutine LTAPE after the matrix has been written on the Write-Tape.
Write-Tape Options	73-78	or	Anything else will be ignored.
Write-Tape Options	79-80	I	The Write-Tape Number. (e.g., 21).
Write-Tape Options		or	Blank if the matrix is not to be written on tape.

Note (1) Format Type A allows any keypunch symbol.
Format Type I allows only integer numbers right justified in the field.

As examples of tape input to Subroutine READIM consider:

Example 1. A matrix named AB2 with run number of RUN-46 is to be read from tape number 11 into the computer and printed. This matrix is also to be written on tape number 22 that is to be initialized and identified as T4321.

Example 2. A matrix named XYZ4 with run number of TKD is on tape number 13 twice. The first time is at location 29 and the second time is at location 54. It is desired to read the second matrix.

Figure 2 demonstrates how these two examples would be written on a coding form to facilitate keypunching to cards.

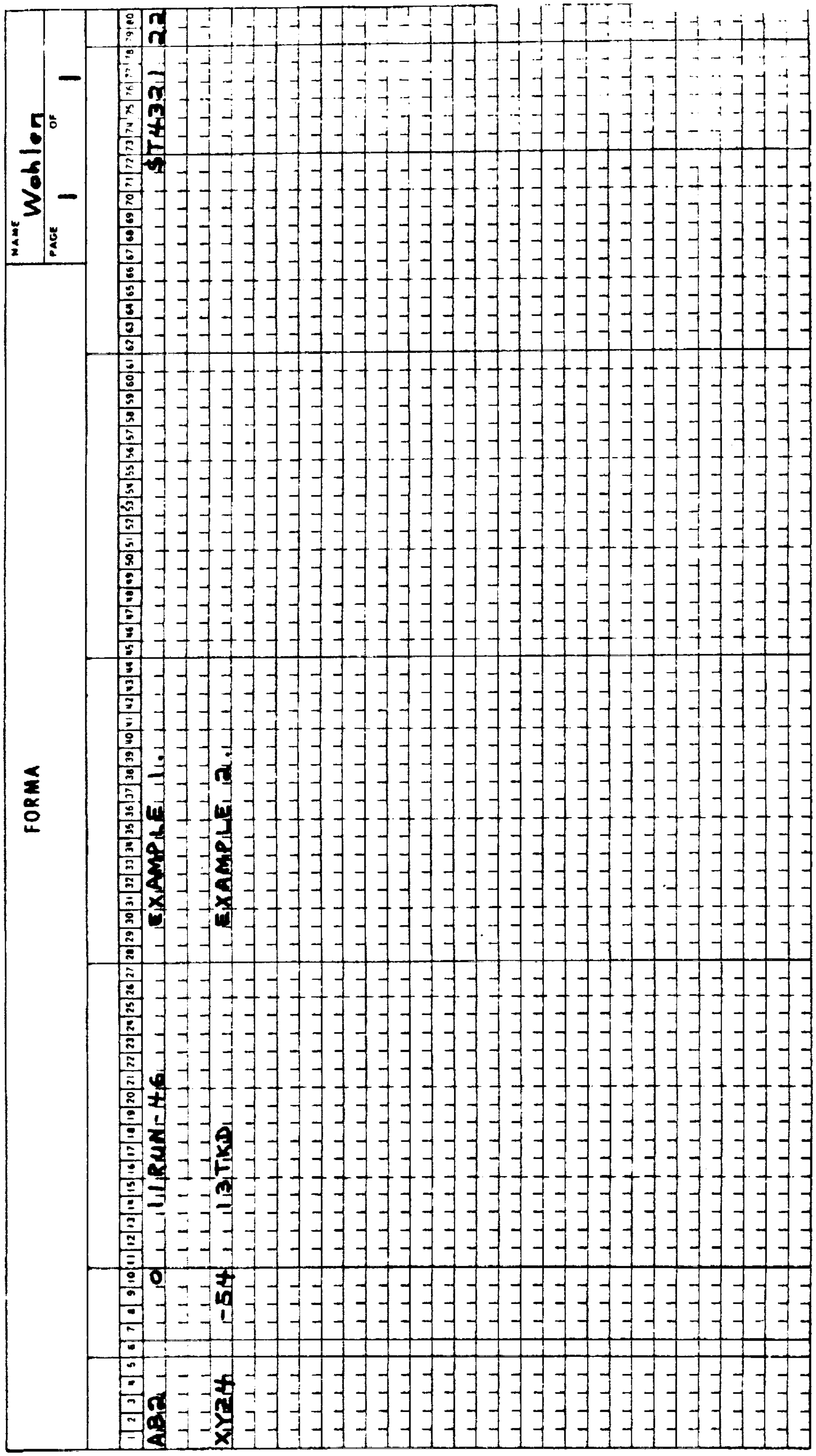


Figure 2 Examples of Tape Input for Subroutine READIM

R. L. Wohlen
May 1971

Subroutine RTAPE reads a selected matrix from tape (disk) into the computer core. The matrix to be selected is identified by the desired run number and matrix name. This procedure is accomplished by searching the matrix headings (see Subroutine WTAPE writeup) until a match with the desired run number and matrix name is obtained and then reading the matrix elements from tape (disk) into the computer core. The search starts from the current position (does not rewind) of the tape (disk) and proceeds to the EOT (end of tape defined in Subroutine WTAPE writeup). If the desired matrix was not found upon reaching the EOT, a rewind is performed and one more search to the EOT is made. If the desired matrix is again not found, (1) an error message is printed, (2) a listing of the matrix headings is printed (see Subroutine LTAPE writeup), and (3) transfer is made to Subroutine ZZBOMB where the program is terminated.

Subroutine START performs the following operations:

1) Reads Input Card 1 for the run number (any keypunch symbol in card columns 1 thru 6) and the user's name (any keypunch symbol in card columns 11 thru 28).

   If the run number is equal to STOP (Card columns 1 thru 4), the run is terminated.

   If the run number is not equal to STOP, the run continues in Subroutine START as follows.

2) Reads Input Card 2 for Title Card 1. Any keypunch symbols may be used in Card columns 1 thru 72.

3) Reads Input Card 3 for Title Card 2. Any keypunch symbols may be used in Card columns 1 thru 72.

4) Initializes page number as zero for use in Subroutine PAGEHD.

5) Interrogates computer for the date.

Run number, date, page number, user's name, Title Card 1, and Title Card 2 are transferred by a COMMON block labeled LSTART for use in other subroutines PAGEHD, PLOT1, PLOT2, PLOT3, and WTAPE.

Subroutine START is used to start each computer run in the FORMA system and will normally be the first subroutine called in a computer program. As an example, pertinent statements from a program using START could be:

```
1 CALL START
      .
      .
      .
  GO TO 1
  END
```

R. L. Wohlen
May 1971

Subroutine WRITE writes a matrix of real numbers (a Fortran term for numbers with a decimal point) on paper. A group of up to ten consecutive elements from a row of the matrix are printed on each line. If all of the elements of a group are zero, printing of this line is suppressed.

Each matrix printed begins on a new page. On each page of printout is the page heading given by Subroutine PAGEHD, the name of the matrix, and the row size and column size of the matrix. This is followed by the matrix data. On any line of matrix data the first integer number is the row number of the matrix elements on that line. The second integer number is the column number of the matrix element in the first data field. The next group of real numbers (up to ten) are the values of the matrix elements. This group of matrix elements is given in consecutive column order.

Subroutine WRITIM writes a matrix of integer numbers on paper. A group of up to twenty consecutive elements from a row of the matrix are printed on each line. If all of the elements of a group are zero, printing of this line is suppressed.

Each matrix printed begins on a new page. On each page of printout is the page heading given by Subroutine PAGEHD, the name of the matrix, and the row size and column size of the matrix. This is followed by the matrix data. On any line of matrix data the first integer number is the row number of the matrix elements on that line. The second integer number is the column number of the matrix element in the first data field. The next group of integer numbers (up to twenty) are the values of the matrix elements. This group of matrix elements is given in consecutive column order.

R. L. Wohlen
May 1971

Subroutine WTAPE writes matrix data at the end of existing written matrix data on a FORMA tape (disk is preferred, see below). Each set of matrix data consists of two logical records. The first record contains the matrix heading (tape identification, location number, run number, matrix name, number of rows of matrix, number of columns of matrix, date, and the word "dense"). The second record consists of the matrix elements.

A schematic representation of the tape (disk) is given by the following sketch.

Beginning
of
tape (disk)

H_1	E_1	H_2	E_2	...	H_n	E_n	EOT

where

H_i = Matrix heading of the i^{th} written matrix,

E_i = Matrix elements of the i^{th} written matrix,

EOT = End of Tape. Data written by Subroutine WTAPE or INTAPE that all FORMA tape subroutines recognize as being the end of written data.

Each vertical line is an end of logical record put on by computer system's routines. The tape is written in binary form as opposed to binary coded decimal (BCD) form.

To find the end of written matrix data, a search is made of the matrix headings until the EOT is found. For this reason, a "new" tape (disk) must be initialized with Subroutine INTAPE so that the tape (disk) contains an EOT. A "new" tape (disk) is defined to be a tape (disk) for which it is desired to start writing matrix data at the front of the tape (disk). Thus, a "new" tape (disk) could be one with obsolete FORMA matrix data on it as well as one that has never been written on by the FORMA system. When the EOT is found, a backspace operation is done over the EOT, and then the current matrix heading, current matrix elements, and a new EOT is written.

A disk is preferred to a tape for the following reason. Because of the physical separation of the read and write heads on most tape drives there may be tape tolerance problems thus backspacing over the EOT is usually not successful. Instead of ending up positioned in front of the EOT, the write head is often positioned in front of the previous matrix elements (E_n in the above sketch). The current matrix heading will be written over the previous matrix elements. This causes problems later when trying to read the records written on the tape. To alleviate this problem, it is strongly recommended that all FORMA tape subroutines (INTAPE, LTAPE, RTAPE, WTAPE, and UPDATE) use an intermediate device such as a disk. At the start of a computer run, the existing tape should be copied onto the disk by using computer control cards. Likewise, at the end of the run, the disk should be copied back onto tape by using computer control cards.

R. L. Wohlen
May 1971

APPENDIX B -- BASELINE USER-PAK DEFINITION

This appendix presents a listing of the seven subroutines and two functions which comprise the basic user-supplied packages required for any simulation.

```
      SUBROUTINE CONTRL
      IMPLICIT REAL*8 (A-H,O-Z)
C
            COMMON /PHRSRD/
     *      PP(6,18,11),BS(6,18,15),ROL(3,3, 6),DOL(3, 6)
            COMMON /CONPAR/
     *      CNTDTA(100)
      COMMON /LOSIZE/ NX,NY,NDLTA,NXSS,NETC,NJG,NY2,NG2
            COMMON /SPECIF/
     *      FETAH(6, 6),FETAHD(6, 6),AMC(2, 5),RH(3,3,30),RS(3,3,30),
     *      DH(3,55),DS(3,20),IMD(2, 5),NMOW(6, 6),IFTSMW(15),
     *      NF,NF,NSFT,NOFMD,NDELTA,ITOFOL(2, 6),IRGFLX( 6),IHDATA(7, 6),
     *      LOCU(14),LEND(14),NU,NEETA,NLAM,NEG
            COMMON /TIMESS/
     *      STARTT,DELTAT,T,ENDT,TMST
            COMMON /VECTOR/
     *      Y(250),YDT(250)
CCCCCCC  THIS COMMON IS TRANSFER BETWEEN CONTRL AND SHAFTT ONLY ----
            COMMON /WHEEL /
     *      CLM(4)
C
      DIMENSION  TG(6),TOC(6),PHE(3),THADW(3)
      DIMENSION CPLY(10,4), KPLY(2), UI(7)
      DATA ICT4/0/, PED / 0.D0, 0.D0, 0.D0 /
      DATA T1,T2,T3,T4,DTHL/
     *    .2D0, 1.2D0, .7D0, 1.7D0, 1.047197551D0 /
      DATA NPLY, KRY, KCY/ 0, 10, 4/
      DATA IIST/ 0 /
      ALIM(U,V) = DMAX1(-V,DMIN1(U,V))
C
CCCCCCCCCC
CCCCCCCCCC
CCC  THE FOLLOWING STATEMENTS MUST ALWAYS BE IN CONTRL..
      IF (IIST .NE. 0)GO TO 110
      IIST = 1
      IF (NPLY .EQ. 0) GO TO 106
      CALL ZERO (CPLY,KRY,KCY,KRY)
      DO 105 K=1,NPLY
         K2=2*K-1
  105 CALL READ (CPLY(1,K2),KPLY(K),NZ,KRY,KCY)
      CALL WRITE (CPLY,KRY,KCY,4HCPLY,KRY)
  106 CONTINUE
      NDLTA = NDELTA
      LDEL = LOCU(2*NB+2) - 1
  110 CONTINUE
      NXSS = 3
      NETC = 3
      IF (NDELTA .EQ. 0) RETURN
```

ORIGINAL PAGE IS OF POOR QUALITY

```
CCCC----NOTE--THIS SUBROUTINE MUST ESTABLISH  NDLTA,NXSS AND NPTG
CCCCCCCCCCC
C
CCCC  ESTABLISH THE D/DT(DELTAS)
C
CCCCCCCCCC
CCCC---NOTE--THIS SECTION IS TYPICAL OF USE OF TFPLY.
CCCCCCCCCC
C
      IF (NPLY .EQ. 0) GO TO 116
      L = LDEL+1
      DO 115 K=1,NPLY
         K2 = 2*K-1

      CALL TFPLY (CPLY(1,K2),CPLY(1,K2+1),UI(K),X,KPLY(K),L)
      L = L+KPLY(K) - 1
  115 CONTINUE
  116 CONTINUE
C
CCCCCCCCCC
      ICT4 = ICT4 + 1
      IA = (ICT4-1)/4
      IAA = (ICT4-2)/4
      IFLAG = IA - IAA
      DO 6 I=1,2
    6 THADW(I) = Y(6+I)
      DO 5 I=1,6
    5 TQ(I) = Y(LDEL+I)
C
C          WHEEL 1 (ROLL INERTIA WHEEL CONTROL TORQUE)
C          DEFINE DIFFERENTIAL EQUATIONS FOR ROLL CONTROL LOOP
C
      U1 = 57.2958D0*POL(3,2,1)/POL(3,3,1)
      U5 = ALIM(TQ(5),25.D0)
      U2 = 2.17D0*U1 - U5
      U3 = ALIM(1.1D0*U2,1.17D0)
      TQD(5) = (1.D0/86.D0)*(-TQ(5) + (9/1.1D0)*U3)
      U6 = ALIM(5*U5,1.65D0)
      U8 = ALIM(TQ(6),1.9D0)
      IF (IFLAG .EQ. 0) GO TO 32
      UU = DABS(U8)
      IF (UU.GT.1.D0) GO TO 30
      IF (UU.LT.0.5D0) GO TO 31
      U9 = RHO(1)
      GO TO 10
   30 U9 = U8/UU
      GO TO 10
   31 U9 = 0.D0
      GO TO 10
   32 U9 = RHO(1)
      GO TO 33
   10 RHO(1) = U9
   33 CONTINUE
      TQD(6) = (-TQ(6) + 2.5D0*(U6-U9))/.5D0
C
C   1500 RPM = 157.0795 RAD/SEC
C    6 INCH*D7 = .03125 FT*LBS
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      IF (DABS(THADW(1)).GT. 157.0795D0) U9 = 0.D0
      CLM(1) = .0317D0*U9 - 5.D-05*THADW(1)
C
C          WHEEL 2 (PITCH INERTIA WHEEL CONTROL TORQUE)
C          DEFINE DIFFERENTIAL EQUATIONS IN PITCH CONTROL LOOP
C
      U1 = -57.2958D0*ROL(3,1,1)/ROL(3,3,1)
      U5 = ALIM(TQ(1),16.4D0)
      U2 = 1.17D0*U1 - U5
      U3 = ALIM(.62D0*U2,1.17D0)
      TQP(1) = (-TQ(1) + U3*(7/.62D0))/80.D0
      U6 = ALIM(5*U3,1.68D0)
      U8 = ALIM(TQ(2),1.9D0)
      IF (IFLAG.EQ.0) GO TO 14
      UU = DABS(U8)
      IF (UU.GT. 1.D0) GO TO 15
      IF (UU.LT.0.5D0) GO TO 16
      U9 = FHD(2)
      GO TO 13



   15 U9 = U8/UU
      GO TO 12
   16 U9 = 0.D0
      GO TO 12
   14 U9 = FHD(2)
      GO TO 13
   12 FHD(2) = U9
   13 CONTINUE
      TQP(2) = (-TQ(2) + 2.5D0*(U6 - U9))/.5D0
      IF (DABS(THADW(2)).GE. 157.0796D0) U9 = 0
      CLM(2) = .0315D0*U9 - 5.D-05*THADW(2)
C
C          WHEEL 3 (YAW INERTIA WHEEL CONTROL TORQUE)
C          DEFINE DIFFERENTIAL EQUATIONS FOR YAW CONTROL LOOP
C
      U1 = 57.2958D0*ROL(2,1,1)/ROL(2,2,1)
      U2 = ALIM(U1,2.D0)
      U6 = ALIM(TQ(3),29.5D0)
      U3 = 2.17D0*U2 - U6
      U4 = ALIM(1.47D0*U3,1.17D0)
      TQP(3) = (1.D0/85.D0)*(-TQ(3) + (9/1.47D0)*U4)
      U7 = ALIM(5*U4,1.68D0)
      U9 = ALIM(TQ(4),1.9D0)
      IF (IFLAG.EQ.0) GO TO 20
      UU = DABS(U9)
      IF (UU.GT.1.D0) GO TO 21
      IF (UU.LT. 0.5D0) GO TO 22
      U10 = FHD(3)
      GO TO 18
   21 U10 = U9/UU
      GO TO 18
   22 U10 = 0.D0
      GO TO 13
   20 U10 = FHD(3)
```

```
      GO TO 24
   18 RHD(3) = U10
   24 CONTINUE
      TCD(4) = (-TC(4) + 2.5D0*(U7 - U10))/.5D0
      IF (DABS(THADW(3)) .GT. 157.0795D0) U10 = 0.D0
      CLM(3) = .02125D0*U10 - 5.D-06*THADW(3)
C
      DO 34 I=1,6
   34 YDT(LDEL+I) = TKC(I)
      YDT(LDEL+7) = Y(16)
      SK4 = CNTDTA(NDELTA+1)
      DK4 = CNTDTA(NDELTA+2)
      CLM(4) =-(SK4*Y(LDEL+7) + DK4*YDT(LDEL+7))
C
      RETURN
      END
```

```
      SUBROUTINE EXTLF   (TEX,ISPN,NTEX)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION TEX(6,1), ISPN(1)
C
            COMMON /MAXNUM/
     *     NMXAY,NFRAX,NOPMAX,NMWMAX,NMWMOD,NMDBCE,KMD,KY,KL
            COMMON /SPECIF/
     *     ETTAF(6, 6),EFTAHD(6, 6),AHC(3, 5),RH(3,3,30),FS(3,3,30),
     *     RH(3,35),RS(3,30),IMC(3, 5),NNCW(6, 6),IFTSMW(15),
     *     NS,NH,NSFT,NCFAL,NDELTA,ITOPCL(2, 6),IHGFLX( 6),IHDATA(7, 6),
     *     LOCL(14),LENU(14),NC,NEFTA,NLAM,NEQ
            COMMON /VECTOR/
     *     Y(250),YDT(250)
C
      DATA IIST / 0 /
C
CCC ESTABLISH THE EXTERNAL FORCE/TORQUE (6-LONG VECTOR) AND NUMBER
CCC  THE CORRESPONDING SENSOR POINTS. ALSO ESTABLISH THE NUMBER OF
CCC  SIX-LONG VECTORS (NTEX).
C
      IF (IIST .EQ. 1) GO TO 5
      IIST = 1
      DO 10 I=1,6
      DO 10 J=1,NSPMAX
   10 TEX(I,J) = 0.D0
C
    5 NTEX = 0
C
      RETURN
      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      SUBROUTINE SHAFTT (TSHFT)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION TSHFT(1)
C
            COMMON /MAXNUM/
     *      NBMAX,NPMAX,NSPMAX,NMAMAX,NMWEGS,NMDECC,NML,KY,KU
            COMMON /SPECIF/
     *      BETAH(6, 6),BETAHD(6, 6),AM(2, 5),RM(3,3,30),RS(3,3,30),
     *      DH(3,35),DS(3,30),IMC(5, 5),NMCW(6, 6),IPTSMW(15),
     *      NS,NP,NSPT,NCRML,NDELTA,ITOFUL(2, 6),IRGFLX( 5),INDATA(7, 6),
     *      LOCU(14),LENU(14),NU,NBETA,NLAM,NEQ
            COMMON /VECTOR/
     *      Y(250),YDT(250)
CCCCCCC  THIS COMMON IS TRANSFER BETWEEN CENTRL AND SHAFTT ONLY ----
           COMMON /WHEEL /
     *    CLM(4)
C
      DATA IIST / 0 /
C
      IF (IIST .EQ. 1) GO TO 10
      IIST = 1
      DO 5 I=1,NMWMAX
    5 TSHFT(I) = 0.D0
C
   10 DO 15 I=1,4
   15 TSHFT(I) = CLM(I)
C
      RETURN
      END
```

```
      SUBROUTINE KHINGE (G)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION G(1)
      DIMENSION SK(3,6),DK(3,6),HNGT(3,6)
C
                COMMON /FHBSKD/
     *     EH(6,18,11),BS(6,16,15),ROL(3,3, 6),DOL(3, 6)
                COMMON /CONPAP/
     *     CNTDTA(100)
                COMMON /MAXMUM/
     *     NFMAX,NHMAX,NSPMAX,NMWMAX,NMWECD,NMDECD,KMU,KY,KU
                COMMON /MOMENG/
     *     P(113),PMOM(36),HTOT(3),TOTL(3),ENGKE( 6),ENGPE( 6),
     *     TOTKE, TOTPE, TOTENG, AHTOT,ATOTL
                COMMON /SPECIF/
     *     BETAH(6, 6),BETAHD(6, 6),AMC(2, 5),RH(3,3,30),RS(3,3,30),
     *     DH(3,30),DS(3,30),IMC(3, 5),NMCW(6, 6),IFTSMW(15),
     *     NF,NH,NSPT,NOFNO,NDELTA,ITOPOL(2, 6),IFGFLX( 6),IHDATA(7, 6),
     *     LOCU(14),LFNU(14),NU,NEETA,NLAM,NE6
C
      EQUIVALENCE (CNTDTA(12),SK(1)), (CNTDTA(30),DK(1))
C
      TOTPE = 0.D0
C
      DO 10 L=1,NH
      DO 10 I=1,3
      HNGT(I,L) = -(SK(I,L)*BETAH(I,L) + DK(I,L)*BETAHD(I,L))
   10 TOTPE = TOTPE + 0.5D0*SK(I,L)*BETAH(I,L)**2
C
      LE6 = IFGFLX(1) + 6
      DO 15 I=1,3
      F = HNGT(I,1)
      DO 16 J=1,LE6
   16 G(J) = G(J) + F*EH(I,J,1)
   15 CONTINUE
C
      DO 20 L=2,NH
      NOFC = ITOPOL(1,L)
      NOFP = ITOPOL(2,L)
      LC = 2*L - 2
      LP = LC + 1
      LOC = LOCU(NOFC) - 1
      LOP = LOCU(NOFP) - 1
      LEC = IFGFLX(NOFC) + 6
      LEP = IFGFLX(NOFP) + 6
      DO 20 I=1,3
      F = HNGT(I,L)
      DO 25 J=1,LEC
      LOCJ = LOC + J
   25 G(LOCJ) = G(LOCJ) + F*EH(I,J,LC)
      DO 26 J=1,LEP
      LOPJ = LOP + J
   26 G(LOPJ) = G(LOPJ) + F*EH(I,J,LP)
   20 CONTINUE
C
      RETURN
      END
```

```
      SUBROUTINE CMISC (N,LL,LC,V2)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION V2(1)
C
           COMMON /MAXNUM/
     *     NDMAX,NHMAX,NSHMAX,NMWMAX,NMWSOL,NMDFLD,NML,KY,KU
           COMMON /SPECIF/
     *     IFTAH(6, 6),IFTAHS(6, 6),ANC(2, 5),RP(3,3,30),RS(3,3,30),
     *     RH(3,35),DS(3,30),IMC(3, 5),NMLW(6, 6),IFTCMB(15),
     *     N2,NF,NSFT,NCFNU,NDELTA,ITCPOL(2, 6),IRGFLX( 6),IHDATA(7, 6),
     *     LPOL(14),LENU(14),NU,NDETA,NLAM,NEG
           COMMON /VECTOR/
     *     Y(300),YDT(300)
C
      DATA IIST / 0 /
C
CCC   USER SUPPLIED SUBROUTINE TO CREAT MISC. CONTRIBUTIONS TO R.H.S.
CCC   INCLUDING  THE THERMAL GRADIENT ENVIRONMENT.
C
      IF (IIST .EQ. 1) GO TO 5
      IIST = 1
C
    5 CONTINUE
      LCM1 = LC - 1
      DO 10 I=1,LE
   10 V2(I) = Y(LCM1 + I) - 0.0
C
      RETURN
      END
```

```
      SUBROUTINE EQADD
      IMPLICIT REAL*8 (A-H,O-Z)

         COMMON /EHPSRD/
     *   EH(6,18,11),ES(5,19,15),ROL(3,3, 6),DOL(3, 6)
         COMMON /ENAUX /
     *   NAUX
         COMMON /MAXMUM/
     *   NEMAX,NHMAX,NSHMAX,NMWMAX,NMWCOD,NMDEOD,KMU,KY,KC
         COMMON /SPECIF/
     *   EETAH(6, 6),EETAHS(6, 6),AMD(2, 5),RH(3,3,30),RS(3,3,30),
     *   DH(3,35),DS(3,30),IMD(3, 5),NMOW(6, 6),IFTSMW(15),
     *   NE,NH,NSFT,NOFMO,NDELTA,ITCPUL(2, 6),IFGFLY( 6),IHDATA(7, 6),
     *   LOCU(14),LENU(14),NU,NEETA,NLAM,NEQ
         COMMON /VECTOR/
     *   Y(250),YDT(250)
C
      NAUX = 0
      LDEL = LOCU(2*NE+2) - 1
      ACON = 57.2958D0
      YDT(NEQ+1) =  ACON*ROL(3,2,1)/ROL(3,3,1)
      YDT(NEQ+2) = -ACON*ROL(3,1,1)/ROL(3,3,1)
      YDT(NEQ+3) =  ACON*ROL(2,1,1)/ROL(2,2,1)
      YDT(NEQ+4) = Y(LDEL+2)
      YDT(NEQ+5) = Y(LDEL+4)
      YDT(NEQ+6) = Y(LDEL+6)
      RETURN
      END




      SUBROUTINE LTORQE (VTORQ)
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION VTORQ(1)
      COMMON /KDSIZE/
     1               KR, KFT, KRX, KV1, KV2, KVX
      COMMON /VECTOR/
     F               Y    (250), YD (250)
      COMMON /TIMESS/
     C               ST, FT, T, ET, TMST
C
      DATA I1ST/0/
C
      TLMT = 10.00*DT
      IF (I1ST .NE. 0) GO TO 10
      IF (T .GT. TLMT) I1ST = 1
      CALL ZERO (VTORQ,1,KVX,1)
      VTORQ(34) = 1.D0
      VTORQ(35) = 1.D0
      RETURN
   10 CONTINUE
      CALL ZERO (VTORQ,1,KVX,1)
      RETURN
      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      FUNCTION ADOT (IC,T)
      IMPLICIT REAL*8 (A-H,O-Z)
                COMMON /VECTOR/
     *      Y(250),YDT(250)
      IF (IC .EQ. 19) GO TO 20
      ADOT = YDT(65)
      RETURN
   20 ADOT = YDT(66)
      RETURN
C
      END
```

```
      FUNCTION ACT (IC,T)
      IMPLICIT REAL*8 (A-H,O-Z)
                COMMON /VECTOR/
     *      Y(250),YDT(250)
C
      IF (IC .EQ. 19) GO TO 20
      ACT  = Y(65)
      RETURN
   20 ACT  = Y(66)
      RETURN
C
      END
```