

**NASA Technical Paper 1145**

**The N-BOD2 User's  
and Programmer's Manual**

**Harold P. Frisch**

**CASE FILE  
COPY**

**FEBRUARY 1978**

**NASA**

NASA Technical Paper 1145

# The N-BOD2 User's and Programmer's Manual

Harold P. Frisch  
Goddard Space Flight Center  
Greenbelt, Maryland

**NASA**

National Aeronautics  
and Space Administration

**Scientific and Technical  
Information Office**

1978

All measurement values are expressed in the International System of Units (SI) in accordance with NASA Policy Directive 2220.4, paragraph 4.

THE CHOICE OF PLANE OF ORBITATION

# CONTENTS

	<i>Page</i>
INTRODUCTION . . . . .	1
THE SIMULATION MODEL . . . . .	3
PROGRAM OVERVIEW, MAIN PROGRAM, AND SUBROUTINE DYN . . . . .	4
PROGRAMMING TECHNIQUES USED . . . . .	8
PROGRAMMER'S GUIDE TO MAIN PROGRAM AND SUBROUTINE DYN . . . . .	8
DATA INPUT, INPUT/OUTPUT CONTROL . . . . .	15
DATA INPUT, SUBROUTINE RSTART . . . . .	15
PROGRAMMER'S GUIDE TO SUBROUTINE RSTART . . . . .	17
DATA INPUT, SUBROUTINE INBS . . . . .	18
PROGRAMMER'S GUIDE TO SUBROUTINE INBS . . . . .	29
PROGRAMMER'S GUIDE TO SUBROUTINE INEROR . . . . .	31
PROGRAMMER'S GUIDE TO SUBROUTINE SETS . . . . .	31
DATA INPUT, SUBROUTINE INOPT . . . . .	33
Terminate Option Input . . . . .	33
1. No Option Required . . . . .	33
2. End of Options . . . . .	35
Optimize Computational Efficiency . . . . .	35
3. Choice of Frame of Computation . . . . .	35
4. Truncate Computation of Pseudo-Inertia Tensors . . . . .	35
5. Truncate Computation of Inertia Cross-coupling Torques . . . . .	36
6. Truncate Computation of Centripetal and Some of the Coriolis Cross-coupling Torques . . . . .	37

## CONTENTS (Continued)

	<i>Page</i>
7. Choice of Kinematic Techniques . . . . .	38
8. Suppress Selected Coordinate Transformations . . . . .	40
9. Suppress Recomputation of Elements of Coefficient Matrix of Inertia Tensors . . . . .	41
10. Suppress Computation of Relative Displacements . . . . .	41
11. Suppress Computation of Symmetric Wheel Angular Position . . . . .	42
12. Computation of Forces and Torques of Constraint . . . . .	43
13. Specification of Constant Speed Wheels . . . . .	43
14. Caging and Uncaging of Degrees of Freedom . . . . .	44
15. Flexible Body Option . . . . .	45
 PROGRAMMER'S GUIDE TO SUBROUTINE INOPT . . . . .	 52
 DATA INPUT, SUBROUTINE INTOR . . . . .	 55
 PROGRAMMER'S GUIDE TO SUBROUTINE TRNSIV . . . . .	 55
 PROGRAMMER'S GUIDE TO SUBROUTINE VDIV . . . . .	 56
 PROGRAMMER'S GUIDE TO SUBROUTINE EQIV . . . . .	 59
 PROGRAMMER'S GUIDE TO SUBROUTINE TRAN . . . . .	 61
 PROGRAMMER'S GUIDE TO SUBROUTINE TRANVD . . . . .	 62
 PROGRAMMER'S GUIDE TO SUBROUTINE RATE . . . . .	 63
 PROGRAMMER'S GUIDE TO SUBROUTINE XDY . . . . .	 66
 PROGRAMMER'S GUIDE TO SUBROUTINE ETA . . . . .	 69
 PROGRAMMER'S GUIDE TO SUBROUTINE TORQUE . . . . .	 74
 PROGRAMMER'S GUIDE TO SUBROUTINE QFDOT . . . . .	 80
 PROGRAMMER'S GUIDE TO SUBROUTINE DCT . . . . .	 84

**CONTENTS (Continued)**

	<i>Page</i>
PROGRAMMER'S GUIDE TO SUBROUTINE ANGLE . . . . .	85
PROGRAMMER'S GUIDE TO SUBROUTINE SETUP . . . . .	87
PROGRAMMER'S GUIDE TO SUBROUTINES OUTPUT AND OUTPSP . . . . .	90
PROGRAMMER'S GUIDE TO THE UTILITY ROUTINES . . . . .	94
NOTATIONAL CROSS-REFERENCE . . . . .	100
REFERENCES . . . . .	106
APPENDIX A: PROGRAM LISTING FOR N-BOD2 . . . . .	A-1
APPENDIX B: SUPPORT PROGRAMS FOR N-BOD2 FLEXIBLE BODY CAPABILITY . . . . .	B-1
REFERENCES . . . . .	B-37

# THE N-BOD2 USER'S AND PROGRAMMER'S MANUAL

Harold P. Frisch  
*Goddard Space Flight Center  
Greenbelt, Maryland*

## INTRODUCTION

The general purpose digital program N-BOD, and its present version N-BOD2, have been developed with the needs of the spacecraft attitude dynamicist foremost in mind. This fact, however, does not detract in any way from its applicability to a much broader class of problems. The program may be used to both derive and output in a vector-dyadic form, and/or to solve numerically the equations of motion of any system that may be adequately modeled, for the purposes of dynamic simulation, as a topological tree of point-connected rigid bodies, flexible bodies, point masses, and symmetrical momentum wheels (commonly called "symmetric wheel" in this document).

The underlying theoretical development upon which the program N-BOD2 is based may be found in References 1 and 2. In the FORTRAN IV coding of this program, a definite attempt has been made to draw a balance between available generality, ease of user operation, and computational speed and efficiency.

The program is structured to expand or contract with the required needs of the user. By an internal interrogation of the input data, the program sets up a series of logic flags that determine the computational options that should be used and those that should not.

In most practical problems, the user will be required to interface with the program. The output routine, which is called every integration step, prints virtually all system state variables along with numerous other physically meaningful parameters that are useful for initial check-out purposes. In setting up the output routine for production runs, the user is expected to literally delete the unwanted print statements from the output subroutine and insert his own.

If external forces and torques act on the system or if nongyroscopic forces and torques act internal to the system at hinge points, the user is required to define them. This will present no particular problem since a number of examples are provided which show exactly how most commonly occurring forces and torques are defined and coded.

Frequently programs already exist that define, for example, the workings of a complex attitude control system. State-variable information is required as input to it and motor torques are outputted from it. By a few simple interface statements, which create the proper state-variable parameters for the existing torque routine and the required force and torque parameters for N-BOD2, the debugged routine may be inserted in total into N-BOD2.

The input of data is provided for in three separate subroutines which are called sequentially. The first is of a rigid format. In this routine a basic coupled rigid body, point mass, and symmetric wheel model of the system is described. Connection topology, degrees of freedom, mass, inertia, and geometric parameters, along with relative body orientation and initial kinematic conditions, are inputted here. The second input routine is more flexible. Numerous options are available and may be called for by use of option cards. For example, some of the bodies may be redefined to be flexible; if so, the flexible body data would be inputted at this point. The third input routine is for the user. Any data that will be required for the torque and output routines, written by the user, are entered here. It may be left empty, or again the user may insert an existing input routine and set up the proper interface statements. It is permissible, and at times is convenient, for the user to use this routine to override some data already inputted in the first input routine.

When the analyst is first presented with a particular problem for dynamic simulation, he is usually provided with just enough information to create a topological model, define the character of each body in the model (rigid body, flexible body, point mass, or symmetric wheel), and to define the number of degrees of relative freedom between contiguous bodies. With this information and a fictitious set of mass and inertia properties, the user may obtain the equations of motion for the simulation model. At times, this is the end product.

With the outputted equations of motion, the user will immediately recognize familiar terms and will need only a cursory glance at the rather lengthy symbol list for exact definitions. There is a one-for-one correspondence between the outputted equations and those provided in References 1 and 2. For example, the outputted statements

$$\text{FOMC}(5) \times (\text{XIC}(5) \cdot \text{FOMC}(5))$$

and

$$\text{FOMC}(3) \times (\text{FOMC}(3) \times \text{CAC}(3))$$

translate to the familiar vector tensor expressions

$$\vec{\omega}_5 \times (\Phi_5 \cdot \vec{\omega}_5)$$

and

$$\vec{\omega}_3 \times (\vec{\omega}_3 \times \vec{\alpha}_3)$$

respectively.



To use this program effectively, the user must become familiar with the array names used to define the various physical characteristics and state variables in the equations of motion. This is most easily done by a study of the computer derived equations of motion. Once this is done the user is in a position to proceed to code the required input, torque, and output routines for his particular problem and to make use of the full power of N-BOD2.

## THE SIMULATION MODEL

The analysis of the dynamic characteristics of a complex system is usually done through the study of a simulation model. The simulation model must possess the same dynamic characteristics as the complex system and at the same time be amenable to mathematical analysis.

Rigid bodies, point masses, and symmetric wheels are idealized bodies that readily lend themselves to mathematical analysis. Flexible bodies are more complex; however, if the deformation is small and the laws of linear elasticity can be applied, the dynamic characteristics can be simulated by the natural modes and frequencies of vibration, the determination of which is a subject in itself. [ N-BOD2 simply assumes that the modes and frequencies of flexible bodies are obtainable; it accepts only resultant mode-dependent parameters in the input data stream (see Appendix B). ]

N-BOD2 *assumes* that the complex system under study may be simulated by a model comprised of rigid bodies, point masses, symmetric wheels, and flexible bodies for which natural modes and frequencies of vibration are available. Contiguous rigid and flexible bodies are *assumed* to be point connected in such a manner that the total system forms a topological tree (no closed paths). Furthermore, point masses are *assumed* to exist only at limb ends while symmetric wheels are *assumed* to be imbedded within either rigid or flexible bodies.

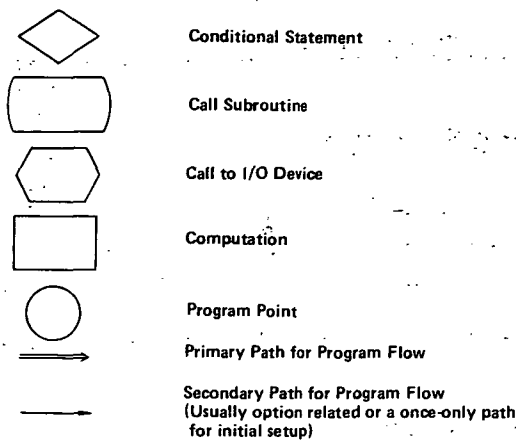
The present version of N-BOD2 has certain *limitations* with respect to the simulation of flexible bodies. The author has not coded the equations required to allow hinge points and wheel attachment points on flexible bodies to be time varying in the flexible body's body-fixed reference frame. This will be done if future simulation demands require it. It should be noted that the equations exist and can be found in Reference 2, in a form amenable for inclusion into N-BOD2.

The simulation model must reflect the fact that relative motion between contiguous bodies may be kinematically constrained. To do this, N-BOD2 *assumes* that contiguous rigid and flexible bodies are connected together by physically realizable 0-, 1-, 2-, or 3-degrees-of-freedom gimbals. The gimbals prohibit relative translational motion and permit, at most, 3 degrees of relative rotational freedom at hinge points. Point masses may have 0, 1, 2, or 3 degrees of relative translational freedom and symmetric wheels may have 1 degree of rotational freedom relative to the body in which they are imbedded. Flexible bodies may possess several degrees of vibrational freedom in addition to the degrees of freedom associated with the connection gimbals. Practical computer storage and speed limitations put a limit on the total number of vibrational modes allowed for the total system of coupled bodies.

## PROGRAM OVERVIEW, MAIN PROGRAM, AND SUBROUTINE DYN

The basic flow logic of the program N-BOD2 is shown on the flowchart in figures 1 and 2 using the symbology listed below. Immediately upon entry into the MAIN program a single control card is read. The data on this card sets several logic flags that control the mode of data input and output.

### Flow Chart Symbology



The input data may either define the physical and kinematic parameters for a new job that will start at time zero, or define the parameters needed to restart the program at the termination time of a previous run.

The output data may either be a listing of the system equations of motion or the output of selected system state variables. Upon reaching the termination time of a particular job, a restart tape may be requested. This tape contains all data required to restart the job at a later date.

If a restart run is to be made, the restart tape created at the end of a previous run becomes the input tape for the present run. It contains the magnitude of every computer-generated parameter required for a reinitiation of computation. The tape is read in and the program then immediately branches to the start of the integration loop and proceeds as if the previous run had never been terminated.

If a new run starting at time zero is to be made, three input subroutines are sequentially called. The first called is INBS. In this subroutine, the basic system is defined. The data are checked in INEROR for physical realizability, and then subroutine SETS is entered. In SETS, various integer sets are computed that enable the program to avoid redundant and trivial computation. Subroutine INOPT, the second input routine, is programmed to recognize various option codes. The available options fall into two categories: those that expand modeling capability, e.g., flexible body option, caged degree of freedom option; and those that attempt to

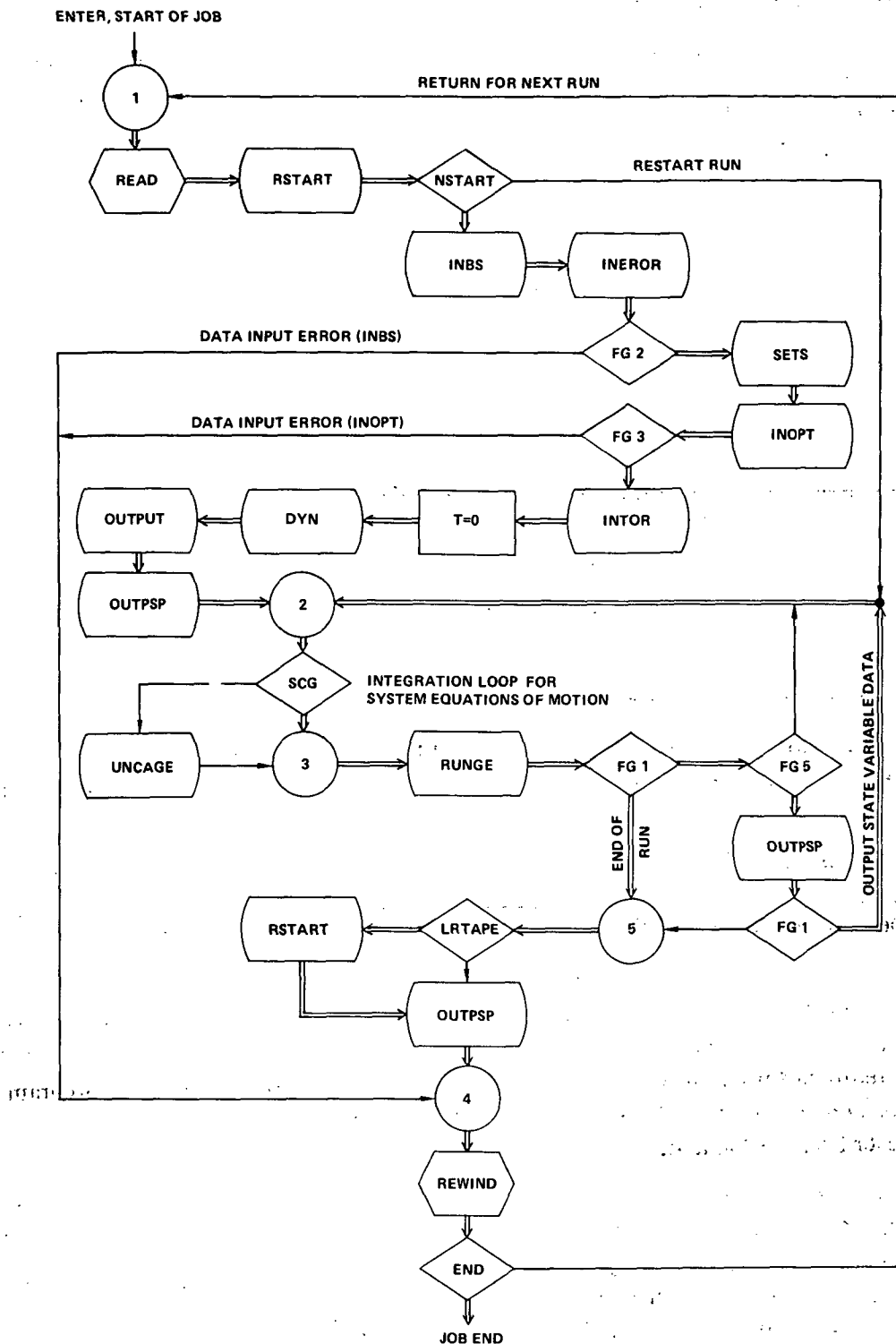


Figure 1. Main program flowchart.

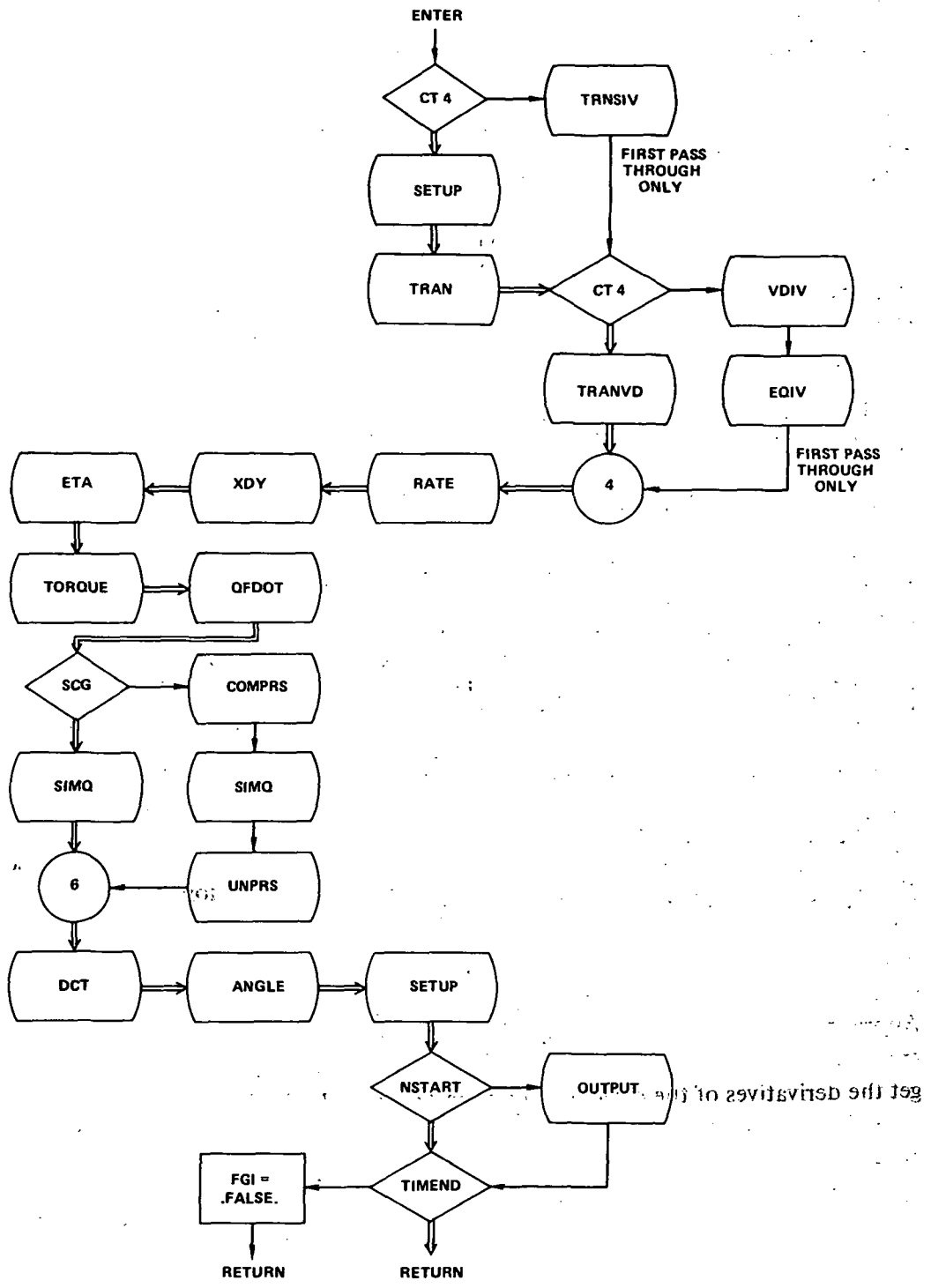


Figure 2. Subroutine DYN flowchart.

improve computational efficiency and run time by allowing the user to introduce engineering judgment in deleting computations that he feels will have a negligible impact upon computed results. The last input routine is INTOR. Normally this routine is empty. It is inserted for the user. Any input data that the user requires for a description of the nongyroscopic forces and torques acting on this system are entered here according to a user written code. The user's input data should be stored in common block /SATELL/ which is reserved for the user. Normally, it contains an empty dummy array of 1000 real double precision words.

With all necessary input data stored in computer memory, subroutine DYN is entered for the first time. The complete state of the system is computed at time zero and outputted. The primary integration loop is then entered. RUNGE is the integration package; it is a fixed-step, four-order Runge Kutta numerical integration routine that calls DYN to compute the equations of motion to be integrated.

At the end of each integration loop, a check is made to determine if the output routine OUTPSP should be entered. At the termination time (a flag is reset in DYN), the program branches out of the integration loop, creates a restart tape if called for, and terminates if there is no succeeding run to follow.

Subroutine DYN is used to call the subroutines in which the equations of motion are coded. Upon entry, a check is made to see if it is the first pass through; if so, certain initialization procedures must be carried out. Initial values for transformation matrices are obtained by TRNSIV; initial values for all vectors and dyads are obtained by VDIV; and initial values for all differential equations are defined in EQIV. If it is not the first pass through DYN, the call for DYN has been made by RUNGE. All of the system state variables obtained from RUNGE are sorted out in SETUP. TRAN computes all transformation matrices; TRANVD transforms all vectors and dyads to a common frame of computation; RATE computes all rate related vectors; XDY computes all inertia and pseudo inertia tensors on the left-hand side of the system equations of motion; and ETA computes all gyroscopic cross coupling torques found on the right-hand side of the equations.

Subroutine TORQUE is user defined; all nongyroscopic forces and torques are defined here in vector form. To reduce the equations to a set of simultaneous scalar equations, QFDOT is called. Upon exit, a check is made in DYN to see if any degrees of freedom are caged. Appropriate action is taken by COMPRS if they are caged, and the simultaneous equations are solved by SIMQ. All acceleration parameters are now at hand. DCT is then entered to get the derivatives of the direction cosine matrices, and ANGLE is entered to get the equations that define relative displacement. All differential equations are then stacked in a one-dimensional array by SETUP in a form acceptable to RUNGE and return to RUNGE is made after a time check.

## PROGRAMMING TECHNIQUES USED

A definite attempt has been made to reflect as much of the subscript notation as possible that is used in References 1 and 2 in the FORTRAN IV coding of the equations. Deviations from this practice occur at points where a more streamlined subscript notation leads to a considerable saving in computer storage and an increase in computational speed. It should be realized that computation speed is severely compromised by the unnecessary use of multi-dimensional arrays.

N-BOD2 is built on the basis that the only labeling which should be done by the user is the labeling of bodies and wheels. All other required labeling is done by the computer. The user, however, must be aware of the labeling rules that are used internally if he is to effectively interface with the program.

Table 1 provides a listing of the symbol names used in the coding of N-BOD2, their dimension, type, storage location, and a very brief description of each along with the subroutine in which they are first defined and/or used extensively. A detailed definition of each symbolic name may be found in the comment cards of the subroutine and/or in the subroutine discussion included in this report.

## PROGRAMMER'S GUIDE TO MAIN PROGRAM AND SUBROUTINE DYN

The purpose of the MAIN program and subroutine DYN is to sequentially call the primary special purpose subroutines that read input data, define and integrate the equations of motion, and write output data. A detailed discussion of the flow logic is provided in the section entitled "Program Overview, MAIN Program, and Subroutine DYN."

The following variables defined in table 1 are either first introduced or used extensively in the MAIN program and subroutine DYN.

CT4 – a counter. Used to count the number of passes through subroutine DYN. It is updated upon entry into DYN. The fourth order fixed step Runge Kutta integration routine calls DYN four times per integration step, consequently

$$CT4 = 1 + 4 * (N - 1) + J$$

during and after the *J*th pass through DYN of the *N*th integration step.

CT4 = 1. During and after the initialization pass through DYN; i.e., when called by MAIN

FG1 – end of job flag. At the end of each integration step its status is checked in MAIN. If .TRUE., computation continues; if .FALSE., an end-of-run condition has been satisfied. If T.GE.TIMEND in DYN, FG1 is reset to .FALSE.. The user may define any other end-of-run condition he desires in subroutines TORQUE or OUTPSP.



Table 1 (Continued)  
Symbol Names Used in N-BOD2

C	CAO	R	3,N	/REAL/	ZERO DEF CM VECT BFC (VDIV)
C	CBDUM,CB	R	3,0:N	/REALZ/	HINGE VECTOR BFC (INBS)
C	CBCDUM,CBC	R	3,0:N	/REALZ/	HINGE VECTOR CFC (VDIV,TRANVD)
C	CBN	R	3	/REALZ/	HINGE VECTOR PART (INBS)
C	CLM	R	N	/REAL/	SCALAR TORQUE ON WHEEL (TORQUE)
C	CNF	R	3,N	EQIV(XMN)	FORCE CENTRIPETAL + CORIOLIS(ETA)
C	COMC	R	3,N+1	/REAL/	ANG RATE TO COMP FRAME (RATE,DCT)
C	CT1	I		/INTG/	COUNTER (INOPT) UNUSED AFTER
C	CT2	I		/INTG/	COUNTER (INOPT) UNUSED AFTER
C	CT3	I		/INTG/	COUNTER (INOPT) UNUSED AFTER
C	CT4	I		/INTG/	COUNTER (INOPT)+PASSES THRU (DYN)
C	CT5	I		/INTG/	COUNTER (INOPT) UNUSED AFTER
C	DOMC	R	3,N+1	/REAL/	PART OF ANG. ACC. VEC. (RATE)
C	DUMMY	R	1000	/SATELL/	STORAGE AREA FOR USER
C	ETC	R	3,N+1	/REAL/	GYRO+EXT.TORQ.ON NEST (ETA,QFDOT)
C	ETIC	R	3,N	EQIV(XMN)	INERT X-COUP TORQ. (ETA)
C	ETM	R	33	/REAL/	SCALAR,GENERALIZED TORQUES (QFDOT)
C	ETMC	R	3,N	EQIV(XMN)	WHEEL X-COUP TORQ. (ETA)
C	FCF	R	3,3,4N	/REAL/	MODAL CENTRIP X-COUP(INOPT,QFDOT)
C	FCK	R	3,4N	/REAL/	MODAL CORIOLIS X-COUP(INOPT,QFDOT)
C	FCON	I	3(N+1)	/INTG/	CODE,FREE VECTORS (INBS)
C	FG1	L		/LOGIC/	END OF RUN FLAG (MAIN,DYN,OUTPSP)
C	FG2	L		/LOGIC/	ERROR INPUT DATA(MAIN,INEROR)
C	FG3	L		/LOGIC/	ERROR INPUT DATA(MAIN,INOPT)
C	FG4	L		/LOGIC/	UNUSED
C	FG5	L		/LOGIC/	OUTPUT DATA ? FLAG (MAIN,TORQUE)
C	FLA	R	3,2N	/REAL/	MODAL CM VECTOR BFC (INOPT)
C	FLAC	R	3,2N	/REAL/	MODAL CM VECTOR CFC (VDIV,TRANVD)
C	FLB	R	3,2N	/REAL/	MODAL MOMENT VECTOR BFC (INOPT)
C	FLC	R	3,2N	/REAL/	MODAL ROTATION MOMENT BFC (INOPT)
C	FLCRC	R	3,N	/REAL/	GYRO FLEXIBILITY FORCE (ETA)
C	FLD	R	3,3,2N	/REAL/	MODAL INERTIA DYAD BFC (INOPT)
C	FLE	R	3,3,2N	EQIV(FLD)	FLD + FLD**T (VDIV)
C	FLH	R	3,3,2N	EQIV(FLJ)	FLD + FLH (VDIV)
C	FLIRC	R	3,N	/REAL/	GYRO FLEXIBILITY TORQUE (ETA)
C	FLJ	R	3,3,2N	/REAL/	MODAL ROTATION DYAD BFC (INOPT)
C	FLQ	R	3,2N	EQIV(FLB)	MODAL MOMENTUM VECTOR BFC (VDIV)
C	FLOC	R	3,2N	/REAL/	FLQ IN CFC (VDIV,TRANVD)
C	FLOM	R	2N	/REAL/	MODAL FREQUENCY (INOPT)
C	FOMC	R	3,N+1	/REAL/	INERTIAL RATE VECTOR (RATE)
C	GAM	R	3, IDEM2	/REAL/	HINGE TO CM VECTOR (VDIV,XDY)
C	H	R		/REAL/	INTEGRATION STEP SIZE (RUNGE,INBS)
C	HM	R	3,N	/REAL/	WHEEL SPIN AXIS BFC (INBS)
C	HMC	R	3,N	/REAL/	WHEEL SPIN AXIS CFC (VDIV,TRANVD)
C	HMODM	R	N	/REAL/	WHEEL ANGULAR MOMENTUM(INBS,SETUP)



Table 1 (Continued)  
Symbol Names Used in N-BOD2

C	IDEM4	I		LOCAL	SIZE OF /LOGIC/ (RSTART)
C	IDEM5	I		LOCAL	SIZE OF /INTG/ (RSTART)
C	IDEM6	I		LOCAL	SIZE OF /INTGZ/ (RSTART)
C	IDEM7	I		LOCAL	SIZE OF /REAL/ (RSTART)
C	IDEM8	I		LOCAL	SIZE OF /REALZ/ (RSTART)
C	IDEM9	I		LOCAL	SIZE OF /SATELL/ (RSTART)
C	IINIT	I	IDEM5	EQIV(AWORK)	ZERO OUT /INTG/ (RSTART)
C	INERF	L		/LOGIC/	FLAG,BFC OR IFC FOR CFC (INOPT)
C	IZINIT	I	IDEM6	EQIV(SCNDUM)	ZERO OUT /INTGZ/ (RSTART)
C	JCON	I	N	/INTG/	BODY CONNECTION MATRIX (INBS)
C	LCON	I	2(N+1)	/INTG/	CODE,LOCKED VECTORS (INBS)
C	L'ANGLE	L		/LDEBUG/	PRINT EQUATIONS IN ANGLE? (MAIN)
C	LDCT	L		/LDEBUG/	PRINT EQUATIONS IN DCT? (MAIN)
C	LEQU	L		EQIV( )	EQIV IN EACH SUB TO PRINT FLAG
C	LEQUIV	L		/LDEBUG/	PRINT EQUATIONS IN EQIV? (MAIN)
C	LETA	L		/LDEBUG/	PRINT EQUATIONS IN ETA? (MAIN)
C	LINIT	L	IDEM4	EQIV(FG1)	ZERO OUT /LOGIC/ (RSTART)
C	LQFDOT	L		/LDEBUG/	PRINT EQUATIONS IN QFDOT? (MAIN)
C	LRATE	L		/LDEBUG/	PRINT EQUATIONS IN RATE? (MAIN)
C	LRTAPE	L		/CHEKS/	CREATE RESTART TAPE? (MAIN)
C	LRUNGE	L		/LDEBUG/	PRINT EQUATIONS IN RUNGE? (MAIN)
C	LSETUP	L		/LDEBUG/	PRINT EQUATIONS IN SETUP? (MAIN)
C	LSIMO	L		/LDEBUG/	PRINT EQUATIONS IN SIMO? (MAIN)
C	LTORQU	L		/LDEBUG/	PRINT EQUATIONS IN TORQUE? (MAIN)
C	LTRAN	L		/LDEBUG/	PRINT EQUATIONS IN TRAN? (MAIN)
C	LTRANV	L		/LDEBUG/	PRINT EQUATIONS IN TRANVD? (MAIN)
C	LTRNSI	L		/LDEBUG/	PRINT EQUATIONS IN TRNSIV? (MAIN)
C	LVDIV	L		/LDEBUG/	PRINT EQUATIONS IN VDIV? (MAIN)
C	LXDY	L		/LDEBUG/	PRINT EQUATIONS IN XDY? (MAIN)
C	MO	I	N	/INTG/	BODY IN WHICH WHEEL IS IN (INBS)
C	NBOD	I		/INTG/	NUMBER OF BODIES (INBS)
C	NB1	I		/INTG/	NUMBER OF BODIES + 1 (INBS)
C	NCTC	I		/INTG/	(INOPT,NA) CONSTRAINT TORQUES
C	NEQ	I		LOCAL	# EQUATIONS SETUP BY N-BOD2 (EQIV)
C	NFER	I		/INTG/	# FREE COORD VECTORS (INBS)
C	NFKC	I		/INTG/	(INOPT,NA) CONSTRAINT FORCES
C	NFLXB	I		/INTG/	# FLEXIBLE BODIES (INOPT)
C	NFRC	I		/INTG/	# RELATIVE ANGLES COMPUTED (INOPT)
C	NLOR	I		/INTG/	# LOCKED COORD VECTORS (INBS)
C	NMO	I		/INTG/	TOTAL NUMBER OF WHEELS (INBS)
C	NMOA	I		/INTG/	# WHEELS TO COMP REL ANGLE (INOPT)
C	NMODS	I		/INTG/	TOTAL # MODES FOR SYSTEM (INOPT)
C	NMV	I		/INTG/	# VARIABLE SPEED WHEELS (INOPT)
C	NSTART	L		/CHEKS/	NEW OR RESTART RUN? (MAIN)
C	NSVP	I		/INTG/	# LOCKED VECTORS TRANSFORM(VDIV)

Table 1 (Continued)  
Symbol Names Used in N-BOD2

C	NSVQ	I		/INTG/	# FREE VECTORS TRANSFORM (VDIV)
C	NTQ	I		/INTG/	# DIFF EQS. IN SUB TORQUE (TORQUE)
C	PCON	I	N+1	/INTG/	# CONSTRAINED AXES AT HINGES (INBS)
C	PHI	R	3,N+1	/REAL/	EXTERNAL TORQUE ON NEST (TORQUE)
C	PLM	R	N	/REAL/	WHEEL SPIN INERTIA (INBS)
C	QF	R	3,3(N+1)	/REAL/	FREE VECTOR BFC (INBS)
C	QFC	R	3,3(N+1)	/REAL/	FREE VECTOR CFC (VDIV,TRANVD)
C	QL	R	3,2(N+1)	/REAL/	LOCKED VECTOR BFC (INBS)
C	QLC	R	3,2(N+1)	/REAL/	LOCKED VECTOR CFC (VDIV,TRANVD)
C	RBLO	L	N	/LOGIC/	RIGID BODY OR POINT MASS? (INBS)
C	RINIT	R	IDEM7	EQIV(CA)	ZERO OUT /REAL/ (RSTART)
C	ROMC	R	3,N+1	/REAL/	RELATIVE RATE VECTOR (RATE)
C	RZINIT	R	IDEM8	EQIV(CBDUM)	ZERO OUT /REALZ/ (RSTART)
C	SC	I	3(N+1)	/INTG/	FREE VECTORS CAGED (INOPT,UNCAGE)
C	SCC	I	N	/INTG/	UNUSED
C	SCG	I		/INTG/	# CAGED DEGREES (INOPT,UNCAGE)
C	SCNDUM,SCN	I	0:N-1	/INTGZ/	CODE,CENTRIPETAL EFFECTS (INOPT)
C	SCRDUM,SCR	I	0:N-1	/INTGZ/	CODE,CORIOLIS EFFECTS (INOPT)
C	SCXC	I	2N	EQIV(TORQ)	CODE,X-COUP. MODES (INOPT,QFOOT)
C	SD	I		/INTG/	CODE,DIRECTION COSINES (INOPT)
C	SEU	I		/INTG/	CODE,EULER ANGLES (INOPT)
C	SFCC	I		/INTG/	CODE,BODIES FLEX X-COUPLING (INOPT)
C	SFKDUM,SFK	I	0:N-1	/INTGZ/	CODE,CONSTRAINT FORCE (INOPT,NA)
C	SFLX	I		/INTG/	CODE,ALL FLEXIBLE BODIES (INOPT)
C	SFR	I	3(N+1)	/INTG/	CODE,COMPUTE FREE VEC ANGLE (INOPT)
C	SFXM	I	N	/INTG/	# MODES EACH BODY (INOPT)
C	SG	I		/INTG/	CODE,ALL GYROSTATS (SETS)
C	SI	I	IDEM3	/INTG/	CODE,BODIES HINGE TO CM (SETS)
C	SIG	I		/INTG/	UNUSED
C	SIXDUM,SIX	I	0:N-1	/INTGZ/	CODE,INERTIA EFFECTS (INOPT)
C	SKDUM,SK	I	0:N-1	/INTGZ/	CODE,BODIES IN EACH NEST (SETS)
C	SL	I		/INTG/	CODE,ALL POINT MASSES (SETS)
C	SLK	I	3(N+1)	/INTG/	CODE,CONSTRAINT TORQUE (INOPT,NA)
C	SMA	I	N	/INTG/	CODE,WHEEL ANGLE COMPUTE (INOPT)
C	SMAL	I		/INTG/	CODE,SMALL ANGLES (INOPT)
C	SMCDUM,SMC	I	0:N-1	/INTGZ/	CODE,ALL WHEELS IN NEST (INOPT)
C	SMV	I		/INTG/	CODE,VARIABLE SPEED WHEELS (INOPT)
C	SOK	I	N+1	/INTG/	CODE,BODIES HINGE 0 - CM (VDIV)
C	SPIDUM,SPI	I	0:N-1	/INTGZ/	CODE,PSUEDO INERTIA TENSORS (INOPT)
C	SQF	I	N+1	/INTG/	CODE,FREE VECTOR AT HINGE (SETS)
C	SQL	I	N+1	/INTG/	CODE,LOCKED VECTOR AT HINGE (SETS)
C	SR	I		/INTG/	CODE,ALL RIGID BODIES (SETS)
C	SSCN	I		/INTG/	CODE,UNION OF ALL SCN (VDIV)
C	SSIX	I		/INTG/	CODE,UNION OF ALL SIX (VDIV)
C	SVA	I		/INTG/	CODE,CM VECTORS TRANSFORM (VDIV)

Table 1 (Continued)  
Symbol Names Used in N-BOD2

C	SVB	I	/INTG/	CODE, HINGE VECTORS TRANSFORM(VDIV)
C	SVD	I	/INTG/	CODE, DON'T TRANSFORM (INOPT)
C	SVI	I	/INTG/	CODE, INERTIA DYAD TRANSFORM (VDIV)
C	SVM	I	/INTG/	CODE, SPIN VECTORS TRANSFORM(VDIV)
C	SVP	I	2(N+1)	CODE, LOCKED VECTORS TRANSFORM(VDIV)
C	SVQ	I	3(N+1)	CODE, FREE VECTORS TRANSFORM(VDIV)
C	SXM	I	3,N	CODE, SMALL ANGLE KINEMATICS(INOPT)
C	SXT	I	/INTG/	CODE, TIME VARY COL INER MAT(INOPT)
C	T	R	/REAL/	TIME (MAIN)
C	TEM	R	2,160	LOCAL TEMP STORAGE AREA (RUNGE)
C	THA	R	33	/REAL/ GENERALIZED COORDINATES(INBS, SETUP)
C	THAD	R	33	/REAL/ GENERALIZE COORD RATE (INBS, SETUP)
C	THADD	R	33	EQIV(ETM) GENERALIZE COORD ACC (SETUP, SIMQ)
C	THADW	R	N	/REAL/ WHEEL RATE (INBS, SETUP)
C	THAW	R	N	/REAL/ WHEEL ANGLE (INBS, SETUP)
C	TIMEND	R		/REAL/ TIME TO END RUN (INBS, DYN)
C	TORQ	I	97	/INTG/ UNUSED STORAGE AREA FOR USER
C	TUG	R	3(N+1)	/REAL/ TIME TO UNCAGE (INOPT, UNCAGE)
C	XDIC	R	3,3, IDEM2	/REAL/ MATRIX OF INERTIA TENSORS(VDIV, XDY)
C	XI	R	3,3,N	/REAL/ INERTIA DYAD BFC (INBS)
C	XIC	R	3,3,N	/REAL/ INERTIA DYAD CFC (VDIV, TRANVD)
C	XIO	R	3,3,N	/REAL/ ZERO INERTIA DYAD BFC(VDIV)
C	XMAS	R	N	/REAL/ BODY MASS (INBS)
C	XMCDUM, XMC	R	3,3,0:N	/REALZ/ TRANSFORM BFC TO CFC (TRANSIV, TRAN)
C	XMN	R	33,33	/REAL/ SCALAR INERTIA MATRIX (VDIV, QFDOT)
C	Y	R	160	LOCAL SYSTEM STATE (EQIV, SETUP, TORQUE)
C	YD	R	160	LOCAL SYSTEM STATE DERIV (SETUP, TORQUE)
C	YMCD	R	3,2,N+1	EQIV(XMN) DIRECTION COSINE RATES (DCT)
C	XMT	R	3,3,N	/REAL/ ZERO STATE TRANSFORMATION MAT(INBS)
C	ZETA	R	2N	/REAL/ MODAL DAMPING RATIO (INOPT)

FG2 – input error flag. If .TRUE., the input data defines a physically realizable system. If .FALSE., a criteria for physical realizability in subroutine INEROR has not been satisfied and the job will terminate with an error message.

FG3 – input error flag. If .FALSE., a code word on an option card read by subroutine INOPT has not been recognized and the job will terminate with an error message.

FG5 – output data flag. Its status is checked at the end of each integration step. If .TRUE., a call to the data output subroutine OUTPSP is made. If .FALSE., it is not. To change the frequency of data output, the user may change the status of FG5 in subroutine TORQUE. The default status of FG5 is .TRUE..

- LANGLE –
- LDCT –
- LEQUIV –
- LETA –
- LQFDOT –
- LRATE –
- LRTAPE –
- LRUNGE –
- LSETUP –
- LSIMQ –
- LTORQU –
- LTRAN –
- LTRANV –
- LTRNSI –
- LVDIV –
- LXDY –
- NSTART –

see section "Data Input, Input/Output Control."

T – independent variable, defines actual simulation time. Evaluated in subroutine RUNGE.

TIMEND – termination time. At T.GE.TIMEND, the flag FG1 is reset to .FALSE. in subroutine DYN.

For additional discussion, see comment cards inserted into the MAIN program and subroutine DYN.

## DATA INPUT, INPUT/OUTPUT CONTROL

The first card of the data deck is called from the MAIN program by the statement:

```
READ 102, NSTART, LRUNGE, LTRNSI, LVDIV, LEQUIV, LTRAN, LTRANV, LRATE, LXDY  
LETA, LTORQU, LQFDOT, LDCT, LANGLE, LSETUP, LSIMQ, LRTAPE,
```

```
102 FORMAT (4X, 17L1)
```

The data on the card set 17 logic flags that are used to control the mode of input and output. The following definitions apply to the logic variables set by the data on this card:

The first logic variable controls the mode of data input.

NSTART = .FALSE. A new job: data input will be from cards. The starting time for the simulation run will be time zero (T.EQ.0).  
= .TRUE. A restart job: data input will be from tape. The input tape for a restart job is the output tape of a previous job. It is generated by N-BOD2. The starting time for the simulation will be the termination time of the job for which the restart tape was written.

The next 15 logic variables control the print commands programmed in the various subroutines. For example:

LRUNGE = .FALSE. All print statements coded in subroutines RUNGE will be bypassed.  
= .TRUE. All equations coded in subroutine RUNGE will be printed along with the numerical magnitudes of the associated variables at the time of execution.

Table 2 lists the subroutines in which the system equations of motion are coded and the associated logic variables that control whether or not the equations coded therein should be outputted on the line printer.

The last logic variable of card 1 controls whether or not a restart tape should be written.

LRTAPE = .FALSE. Create a restart tape at the termination time of this job.  
= .TRUE. Do not create a restart tape at the termination time.

If a restart tape is created, the restart data are put into file 2 of the output tape referred to as data set 11 by the WRITE statements.

## DATA INPUT, SUBROUTINE RSTART

After the first data card is read, a CALL to subroutine RSTART is made. The precautionary step of zeroing all storage locations, except those already loaded by the data on the first input

Table 2  
 Logic Variables Controlling the Printing  
 of the Equations of Motion Coded in the Subroutines

Logic Variable	Subroutine
LRUNGE	RUNGE
LTRNSI	TRNSIV
LVDIV	VDIV
LEQUIV	EQIV
LTRAN	TRAN
LTRANV	TRANVD
LRATE	RATE
LXDY	XDY
LETA	ETA
LTORQU	TORQUE
LQFDOT	QFDOT
LDCT	DCT
LANGLE	ANGLE
LSETUP	SETUP
LSIMQ	SIMQ

card, is performed upon entry. A check on the status of the logic variable NSTART is then made:

IF (NSTART .EQ. FALSE),

a return to the main program is immediately made. The data to be read in will be from cards and will define a new job.

IF (NSTART .EQ. TRUE),

the data to restart an old job will be read according to the read statements coded in subroutine RSTART. The restart tape is read according to the following statements:

```

READ (10, 102) Y
READ (10, 102) YD
READ (10, 101) NEQ
READ (10, 102) (DUMMY (I), I = 1, IDEM9)
READ (10, 102) (RINIT (I), I = 1, IDEM7)
```

```

      READ (10, 102) (RZINIT (I), I = 1, IDEM8)
      READ (10, 101) (IINIT (I), I = 1, IDEM5)
      READ (10, 101) (IZINIT (I), I = 1, IDEM6)
      READ (10, 101) (LINIT (I), I = 1, IDEM4)
101  FORMAT (16 Z 8)
102  FORMAT (8 Z 16)

```

These statements read the restart tape exactly as it was written. The user has no control over the writing or reading of the restart tape other than by the logic control variable LRTAPE that defines whether or not it should be created at the time of run termination.

At the end of any job that calls for a restart tape to be generated, the restart data are put in file 2 of the output tape which is data set 11. To execute a restart job, the program assumes that the appropriate job control statements have been provided that will position the restart tape (now data set 10) at the start of file 2 before the execution of the first READ input tape instruction. (See the comment cards in subroutine RSTART and the job control requirements discussion in the following section.)

After the restart tape has been read, the last input card required for a restart run is read by the statement:

```

      READ 103, TIMEND
103  FORMAT (D15.5)

```

in RSTART where

TIMEND = termination time for present job.

This read statement is executed only on a restart job. Upon exit from RSTART, return is made to the start of the numerical integration loop in the MAIN program. The job then proceeds as if it had never been previously terminated. There are no more data input cards required for a restart job.

## PROGRAMMER'S GUIDE TO SUBROUTINE RSTART

There is no actual computation carried out in subroutine RSTART. Its functions are:

- Zero all storage locations used by N-BOD2 before any computation is done.
- Read the restart tape so that computation may be reinitiated at the termination time of a previous run.
- Write the restart tape so that computation may be restarted at a later date.

For further detail, see the comment cards inserted in the subroutine RSTART.

To zero out all common blocks, the size of each must be defined. Thus:

IDEM4 = number of logical words in /LOGIC/

IDEM5 = number of integer words in /INTG/

IDEM6 = number of integer words in /INTGZ/

IDEM7 = number of real words in /REAL/

IDEM8 = number of real words in /REALZ/

IDEM9 = number of real words in /DUMMY/

These parameters are defined in subroutine RSTART and must be changed if any of the primary labeled common blocks are extended or modified. The user is referred to the program listing to note the method used to zero the common blocks.

To make use of the restart capability, the user is required to provide appropriate machine-dependent job control statements.

N-BOD2 is programmed to put all user-desired output data in file 1 of data set 11 and to put all of the restart data in file 2 of data set 11. Data set 11 becomes the restart tape.

To restart the job, the restart tape becomes data set 10. The appropriate job control statements must be provided to position it at the start of file 2 prior to the execution of the first read tape instruction of the program.

For a restart job, the author has found it convenient to provide the job control statements that copy, prior to job execution, file 1 of data set 10 onto file 1 of data set 11. Data set 10 will then be positioned at the start of file 2 and, at the end of the restart job, file 1 of data set 11 will contain a continuous record of the output data from time zero. Furthermore, the old restart tape, data set 10, will not have been destroyed.

## DATA INPUT, SUBROUTINE INBS

IF (NSTART .EQ. FALSE)\*,

a call to the subroutine INBS is made immediately after the program returns to the MAIN program from RSTART. In INBS, the input data required to describe the basic simulation model are read in from cards.

As in References 1 and 2, the labeling of a system of N bodies (rigid bodies + flexible bodies + point masses) and M symmetric wheels must adhere to the following rules:

- Bodies must be given distinct integer labels ranging from 1 to N inclusive. The labeling must be such that, along any topological path extending from body 1 to a limb end, the body labels are of increasing magnitude.

---

\*Condition code for a new job.



- Symmetric wheels must be given distinct integer labels ranging from 1 to M inclusive. These may be randomly assigned.
- Body 1 is treated as the principal body of the system.

The labeling of hinge points between contiguous bodies is done by the computer according to the following rule:

- Hinge point K-1 is the point of connection between body K and the chain of bodies extending to it from body 1.

Figures 3 and 4 are provided as visualization aids. Figure 3 is the example used in Reference 1 to illustrate the defined labeling convention. Figure 4 shows the exact position and labeling of each of the position vectors defined for the system.

The following definitions are used in the coding of subroutine INBS. The total number of bodies and wheels that make up the simulation model are defined by:

NBOD – total number of rigid bodies, flexible bodies, and point masses

NMO – total number of symmetric wheels

also defined is the constant

$$NB1 = NBOD + 1$$

As a direct consequence of the established labeling convention, the system topology can be uniquely defined by the one dimensional arrays:

JCON(K)                      Body label of the body connected to body K at hinge  
for K = 2,3, . . . , NBOD      point K-1. Relative to body 1, it is the body inboard of  
body K.

JCON(1) = 0                      implies body 1 is contiguous to the inertial frame of  
reference.

MO(M)                         Body label of the body in which symmetric wheel M is  
for M = 1,2, . . . , NMO        is imbedded.

The physical nature of each body, its mass and the number of kinematic constraints between contiguous bodies, are defined by:

RBLO(K) = .TRUE.              Body K is either a rigid or a flexible body. For N-BOD2,  
body 1 and all bodies not at a limb end must be rigid.

= .FALSE.                        Body K is a point mass.

for K = 1,2, . . . , NBOD

XMAS(K)                        Total mass of body K including mass of all symmetric  
for K = 1,2, . . . , NBOD        wheels which are imbedded within it.

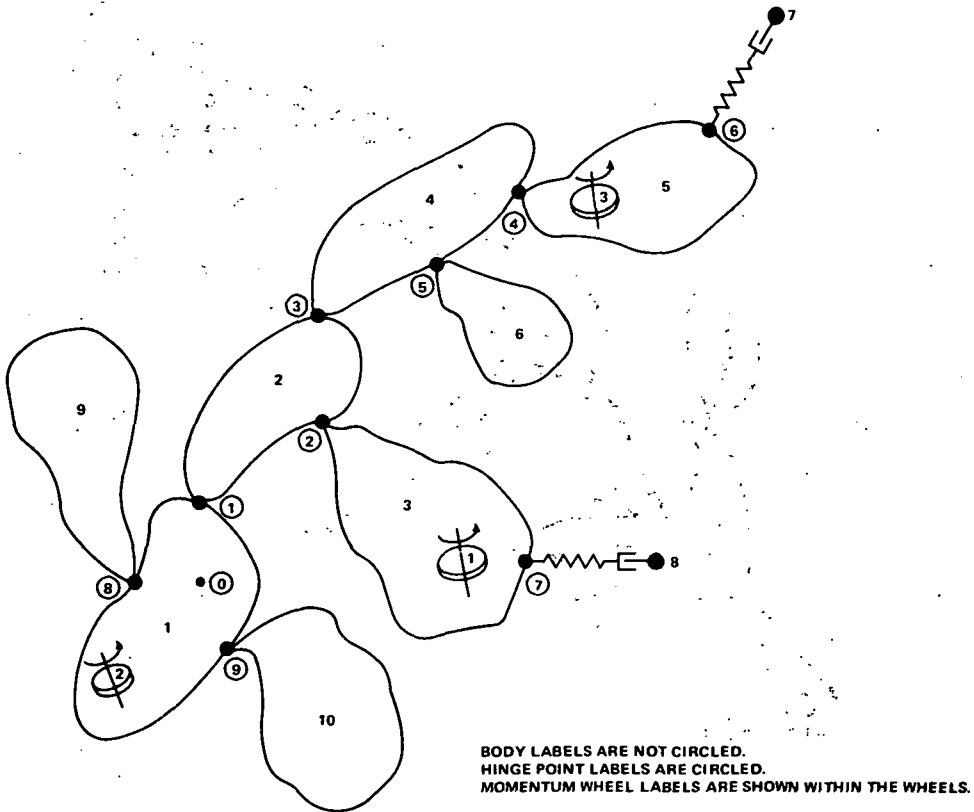


Figure 3. Labeling scheme for 10-body example.

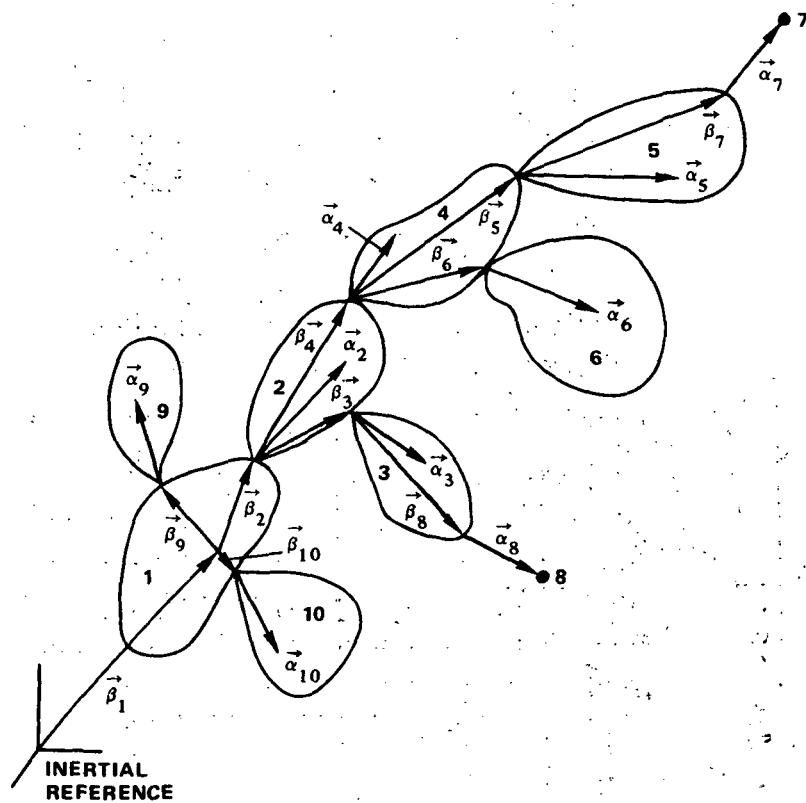


Figure 4. Positions and labeling of hinge-point and center-of-mass location vectors.

PCON(K)

for K = 1

Number of kinematically constrained rotational degrees of freedom between body 1 and an inertially fixed frame of reference.

for K = 2,3, ..., NBOD

Number of kinematically constrained rotational or translational degrees of freedom between body K and body JCON(K).

for K = NBOD+1

Number of kinematically constrained translational degrees of freedom between body 1 and the inertially fixed frame of reference.

$$\text{PCON(K)} \quad \text{for K = 1,2, \dots, NBOD+1} = \begin{cases} 0 & \text{implies 3 degrees of relative freedom} \\ 1 & \text{implies 2 degrees of relative freedom} \\ 2 & \text{implies 1 degree of relative freedom} \\ 3 & \text{implies no relative motion allowed} \end{cases}$$

The equations of motion have been derived in a vector-dyadic format and are valid for all frames of reference. To numerically solve these equations, it must be possible to express any vector or tensor relative to any desirable frame of reference.

The equations of motion are derived relative to an origin that is fixed in an inertially fixed-reference system. The user is required to pick an inertial origin and to define an orthogonal reference frame there. All translational and rotational motion of body 1 is computed relative to this inertially fixed-reference frame.

Let

$(X_0, Y_0, Z_0)$  = coordinate axes of the inertially fixed-reference frame

N-BOD2 assumes that if:

PCON (NBOD + 1) = 0 Body 1 unconstrained in translation

PCON (NBOD + 1) = 1 Body 1 constrained to translate in the  $X_0, Y_0$  plane

PCON (NBOD + 1) = 2 Body 1 constrained to translate in the  $X_0$  direction

PCON (NBOD + 1) = 3 Body 1 totally constrained in translation

In addition to the inertial reference frame, body fixed-reference frames must also be defined. Consequently, a body K fixed-reference frame is defined at hinge point K-1; K = 1, 2, ..., NBOD. The orientation of the body K fixed-coordinate axes is arbitrary and to be defined by the user. To simplify the input of initial kinematic conditions, it is desirable to define a zero state for the system. All relative motion between contiguous bodies can then be measured relative to this state in terms of a few physically meaningful relative displacement parameters.

The following set of transformation matrices completely defines the relative orientation of all reference frames when the system is in the zero relative displacement state.

- [XMT (I, J, K)] — transformation matrix used to transform vectors from body K to body JCON(K) fixed coordinates when the system is in its zero relative displacement state

$$[XMT (I,J,K)] = \begin{bmatrix} XMT (1,1,K) & XMT (1,2,K) & XMT (1,3,K) \\ XMT (2,1,K) & XMT (2,2,K) & XMT (2,3,K) \\ XMT (3,1,K) & XMT (3,2,K) & XMT (3,3,K) \end{bmatrix}$$

where

$$K = 1, 2, \dots, NBOD$$

If K = 1, the inputted transformation matrix transforms vectors from body 1 fixed coordinates to inertial fixed coordinates.

The origin of the body K fixed-reference frame is always at hinge point K-1; K = 1, 2, . . . , NBOD. If body K, K = 2, 3, . . . , NBOD, is a point mass, a transformation matrix is not inputted. The axes of the body fixed-reference frames are defined by the computer to be respectively parallel to the axes of the body JCON(K) fixed-reference frame.

The components of all inertia tensors and position vectors required to describe the physical size and rotational inertia characteristics of the simulation model are inputted relative to the body fixed-reference frame in which they are fixed. That is:

- [XI (I, J, K)] — components of the inertia tensor for body K and all despun wheels contained therein about the body's center of mass, relative to the body K fixed-reference frame (assumed to be zero if body K is a point mass).  
for K = 1, 2, . . . , NBOD

$$[XI (I,J,K)] = \begin{bmatrix} XI (1,1,K) & XI (1,2,K) & XI (1,3,K) \\ XI (2,1,K) & XI (2,2,K) & XI (2,3,K) \\ XI (3,1,K) & XI (3,2,K) & XI (3,3,K) \end{bmatrix}$$

- CA (I, K) — components of the vector from hinge point K-1 to the center of mass of body K (zero state location if body K is a point mass), relative to the body K fixed-reference frame.  
for K = 1, 2, . . . , NBOD

$$\{CA (I,K)\} = \begin{Bmatrix} CA (1,K) \\ CA (2,K) \\ CA (3,K) \end{Bmatrix}$$

CB(I, K)

for K = 0

— reserved for the components of the vector from the inertial origin to the composite system center of mass. (Not needed for computation and hence not computed, may be user-defined if desirable.)

for K = 1, 2, ..., NBOD

— components of the vector from hinge point JCON(K)-1 to hinge point K-1, (from inertial origin if K = 1), relative to the body JCON(K) fixed-reference frame

$$\{CB(I,K)\} = \begin{Bmatrix} CB(1,K) \\ CB(2,K) \\ CB(3,K) \end{Bmatrix}$$

Note: Special provision has been made in the coding of N-BOD2 to accept the subscript 0 in certain arrays such as CB. Furthermore, to avoid numerical computation problems, it is advisable to avoid placing the inertial origin at a distance that is many orders of magnitude greater than the diameter of a sphere which could contain the entire simulation model.

The previously defined integer array PCON(K), K = 1, ..., NBOD+1, specifies the number of kinematic constraints that exist between contiguous bodies and between body 1 and the inertial reference. The logical array RBLO(K), K = 1, 2, ..., NBOD, can be used to determine whether relative motion between the bodies K and JCON(K) is rotational or translational. Body 1 may have both rotational and translational degrees of freedom relative to the inertial reference. Rigid and flexible bodies may have only relative rotational degrees of freedom while point masses may have only relative translational degrees of freedom. N-BOD2 *assumes* that relative rotational motion may be adequately modeled by either a 0-, 1-, 2-, or 3-degree-of-freedom gimbal, while relative translational motion may be either 0, 1, 2, or 3 dimensional. Both the directions about which, or along which, motion is free of kinematic constraints and the directions about which, or along which, motion is kinematically constrained must be defined.

These directions are defined by a set of exactly NBOD+1 triads of unit coordinate vectors. These coordinate vectors are defined to be either free coordinate vectors (implying that motion is free about or along the directions defined by them) or locked coordinate vectors (implying that motion is locked or kinematically constrained about or along the directions defined by them). The user will be required to define two of the three coordinate vectors; the third will be internally generated by a vector cross product based upon conditions of orthogonality.

The integer array PCON(K), K = 1, 2, ..., NBOD+1, defines the mix of free and locked coordinate vectors in each of the NBOD+1 triads. It also provides a convenient tool for creating a labeling sequence for the free and locked coordinate vectors. The labeling will be done by the computer.

It is a sequential numbering algorithm that starts with the coordinate vectors associated with relative rotational motion of body 1, proceeds sequentially up to body NBOD, and ends with the coordinate vectors associated with the description of the translational motion of body 1.

Let

$\{QF(I, M)\}$  = components of the unit free coordinate vector  $M$  relative to the reference frame in which it is fixed.

$\{QL(I, L)\}$  = components of the unit locked coordinate vector  $L$  relative to the reference frame in which it is fixed.

The mix of free and locked coordinate vectors at hinge point  $K-1$  and their labeling sequence are defined as follows for  $K = 1, 2, \dots, NBOD+1$ :

- For  $PCON(K) = 0$  the mix of coordinate vectors at hinge point  $K-1$  is

$$\{QF(I, M)\}, \{QF(I, M+1)\}, \{QF(I, M+2)\}$$

- For  $PCON(K) = 1$  the mix of coordinate vectors at hinge point  $K-1$  is

$$\{QF(I, M)\}, \{QF(I, M+1)\}, \{QL(I, L)\}.$$

- For  $PCON(K) = 2$  the mix of coordinate vectors at hinge point  $K-1$  is

$$\{QF(I, M)\}, \{QL(I, L)\}, \{QL(I, L+1)\}.$$

- For  $PCON(K) = 3$  the mix of coordinate vectors at hinge point  $K-1$  is

$$\{QL(I, L)\}, \{QL(I, L+1)\}, \{QL(I, L+2)\}$$

where for any  $K = 1, 2, \dots, NBOD + 1$

$$M = 1 + \sum_{\substack{i=1 \\ K \neq i}}^{K-1} (3 - PCON(i))$$

$$L = 1 + \sum_{\substack{i=1 \\ K \neq i}}^{K-1} PCON(i)$$

One of several different kinematic conditions can exist at each hinge point. The user must adhere to the following rules when specifying the coordinate of the free and locked coordinate vectors in each of the  $NBOD+1$  triads:

- If  $RBLO(K) = .TRUE.$  for any  $K = 1, 2, \dots, NBOD$ , the coordinate vectors are associated with gimbal axes. Table 3 defines for this case the coordinate frames in which the free and locked coordinate vectors are defined.

- If  $RBLO(K) = .FALSE.$  for any  $K = 2, 3, \dots, NBOD$ , the coordinate vectors define the directions along which relative translational motion of point mass  $K$  is free and along which it is constrained. The origin of this triad of coordinate vectors is the zero position of the point mass  $K$  that is defined by the components of the center of mass vector  $\{CA(I, K)\}$ . Table 4 defines the coordinate frame in which the free and locked coordinate vectors are defined.
- If  $K = NBOD+1$ , the coordinate vectors define the direction along which the center of mass of body 1 is free to translate. These directions are defined by the computer based upon the number of kinematic constraints defined by  $PCON(NBOD+1)$ . Table 5 defines the coordinates of the unit coordinate vectors relative to the inertially fixed reference. These are set by the computer.

The initial state of the simulation model can now be defined by specification of initial displacements and rates about or along the free coordinate vectors whose direction defines the directions of positive displacement. Let

$THA(M) =$  initial relative displacement about or along the free coordinate vector  $M$

$THAD(M) =$  initial relative rate about or along the free coordinate vector  $M$

The user need not be concerned with defining initial values for transformation matrices since it is done by an internal computer computation.

Table 3  
Coordinate Frames in Which Free and Locked  
Coordinate Vectors are to be Defined for  
 $RBLO(K) = .TRUE.; K = 1, 2, \dots, NBOD$

Vector	PCON (K)			
	0	1	2	3
{QF (I, M)}	JCON (K)	JCON (K)	JCON (K)	
{QF (I, M + 1)}	Intermediate	K		
{QF (I, M + 2)}	K			
{QL (I, L)}		Intermediate	K	K
{QL (I, L + 1)}			K	K
{QL (I, L + 2)}				K



Table 4  
 Coordinate Frames in Which Free and Locked  
 Coordinate Vectors are to be Defined for  
 RBLO (K) = .FALSE.; K = 2, 3, . . . , NBOD

Vector	PCON (K)			
	0	1	2	3
{QF (1, M)}	K	K	K	
{QF (1, M + 1)}	K	K		
{QF (1, M + 2)}	K			
{QL (1, L)}		K	K	K
{QL (1, L + 1)}			K	K
{QL (1, L + 2)}				K

Table 5  
 Coordinates of the Unit Coordinate Vectors Defined  
 at the Inertial Origin Along Which Translational  
 Motion of Body 1 is Measured;  
 K = NBOD + 1

Vector	PCON (K)			
	0	1	2	3
{QF (I, M)}	L 1, 0, 0 J	L 1, 0, 0 J	L 1, 0, 0 J	
{QF (I, M + 1)}	L 0, 1, 0 J	L 0, 1, 0 J		
{QF (I, M + 2)}	L 0, 0, 1 J			
{QL (I, L)}		L 0, 0, 1 J	L 0, 1, 0 J	L 1, 0, 0 J
{QL (I, L + 1)}			L 0, 0, 1 J	L 0, 1, 0 J
{QL (I, L + 1)}				L 0, 0, 1 J

To define the characteristics of each of the symmetric wheels imbedded within the simulation model, the following definitions are made:

- $\{HM(I, M)\}$  — components of a unit vector along the spin axis of symmetric wheel M relative to the body MO(M) fixed-reference frame. Its direction defines the direction of positive rotation.
- PLM (M) — rotational inertia of the symmetric wheel M about its spin axis.
- THAW (M) — initial angular position of the wheel M about its spin axis.
- THADW (M) — initial angular rate of the wheel M about its spin axis relative to body MO(M).

The integration step size for the fourth order Runge-Kutta integration package and the time for job termination are defined by:

- H — integration step size
- TIMEND — job termination time

To completely specify the initial state of the simulation model, the above parameters must be defined by the user or be internally generated. The following is a listing of the READ statements coded in subroutine INBS in order of execution. The data deck must be prepared accordingly.

```

***** READ STATEMENTS FOR SUBROUTINE INBS IN ORDER OF EXECUTION *****
C
C INITIALIZE FREE AND LOCKED COORDINATE VECTOR LABELS
M = 1
L = 1
***** FIRST CARD READ BY INBS IS SECOND CARD OF DATA DECK
READ 100, NR0D
C
***** READ EXACTLY NR0D SETS OF BODY DATA, ONE PER BODY
DO 1 K=1, NR0D
  READ 105, N, (MESS(J), J=1, 18)
  IF(N.NE.K) ERROR, CARD OUT OF SEQUENCE, PRINT MESSAGE, HALT
C MESS(J), J=1, 18 ALPHANUMERIC DESCRIPTION OF BODY N, PRINTED WITH DATA ECHO
  READ 101, RRL0(K), JCON(K), PCON(K), XMAS(K)
  IF(RHLO(K)) GO TO 2 BODY K IS A RIGID OR FLEXIBLE BODY
  GO TO B BODY K IS A POINT MASS
2 READ 102, ((XI(I, J, K), J=1, 3), I=1, 3)
  READ 102, ((XMT(I, J, K), J=1, 3), I=1, 3)
  B READ 102, (CA(J, K), J=1, 3)
  READ 102, (CB(J, K), J=1, 3)
C COMPUTE LABELS FOR FREE AND LOCKED COORDINATE VECTORS INTERNALLY
  IF(K.EQ.1) GO TO 12
  M = M + 3 - PCON(K-1)
  L = L + PCON(K-1)
12 GO TO (13, 14, 15, 17) PCON(K)+1
13 CONTINUE THREE DEGREES OF RELATIVE FREEDOM FOR BODY K
  READ 102, (QF(J, M), J=1, 3)
  READ 102, (QF(J, M+2), J=1, 3)
  READ 102, (THA(J), J=M, M+2)
  READ 102, (THAD(J), J=M, M+2)
  GO TO 1 END OF DATA FOR BODY K
14 CONTINUE TWO DEGREES OF RELATIVE FREEDOM FOR BODY K
  READ 102, (QF(J, M), J=1, 3)
  READ 102, (QF(J, M+1), J=1, 3)
  READ 102, (THA(J), J=M, M+1)
  READ 102, (THAD(J), J=M, M+1)
  GO TO 1 END OF DATA FOR BODY K

```

```

15 CONTINUE      ONE DEGREE OF RELATIVE FREEDOM FOR BODY K
  READ 102, (QF(J,M),J=1,3)
  RFAD 102, (QL(J,L),J=1,3)
  READ 102, THA(M)
  READ 102, THAD(M)
  GO TO 1  END OF DATA FOR BODY K
17 CONTINUE      ZERO DEGREES OF RELATIVE FREEDOM FOR BODY K
  READ 102, (QL(J,L),J=1,3)
  RFAD 102, (QL(J,L+1),J=1,3)
  GO TO 1  END OF DATA FOR BODY K
1  CONTINUE
***** ALL BODY DESCRIPTIVE DATA IS IN MEMORY NOW
C
***** READ TRANSLATION CONDITIONS FOR BODY 1
  READ 100, PCON(NBOD+1),
  M = M + 3 - PCON(NBOD)
  READ 102, (THAD(J),J=M,M+2-PCON(NBOD+1))
C
***** READ ALL SYMMETRIC WHEEL DATA
  READ 100, NMO
  DO 16 I=1,NMO  GO TO 16 IF NMO.EQ.0
  RFAD 104, MI(I),(HM(J,I),J=1,3)
  READ 106, PLM(I),(HESS(J),J=1,16)
  RFAD 102, THAW(I),THADW(I)
16 CONTINUE
***** ALL WHEEL DATA IS IN MEMORY
C
***** READ STEP SIZE AND TIME TO END SIMULATION
  READ 102, H,TIMEND
C  END OF DATA TO BE READ BY SUBROUTINE INBS
C
***** FORMAT STATEMENTS USED
100 FORMAT (I5)
101 FORMAT (L5,2I5,D15.5)
102 FORMAT (3D15.5)
104 FORMAT (I5,3D15.5)
105 FORMAT (I5,18A4)
106 FORMAT (D15.5,16A4)

```

## PROGRAMMER'S GUIDE TO SUBROUTINE INBS

The primary function of subroutine INBS is to read the data cards that define the basic simulation model and its initial kinematic conditions. This has been thoroughly discussed in the section "Data Input, Subroutine INBS." A cursory glance at the subroutine listing will show that it is also programmed to provide an echo of the input data and to compute a few additional variables that will be needed for the succeeding computations.

The following variables defined in table 1 are either first introduced or used extensively in subroutine INBS. The programmer is referred to the comment cards inserted therein and the following for detailed definitions:

- CA (I, K) = see section "Data Input, Subroutine INBS."
- CB (I, K) – see section "Data Input, Subroutine INBS."
- CBN (I) – computed variable. Used to compute body 1 center of mass location relative to inertial origin

$$\begin{Bmatrix} \text{CBN}(1) \\ \text{CBN}(2) \\ \text{CBN}(3) \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix} \quad \text{if PCON}(NB1) = 0$$

$$\begin{aligned}
&= \begin{Bmatrix} 0 \\ 0 \\ \text{CB}(3, 1) \end{Bmatrix} && \text{if PCON (NB1) = 1} \\
&= \begin{Bmatrix} 0 \\ \text{CB}(2, 1) \\ \text{CB}(3, 1) \end{Bmatrix} && \text{if PCON (NB1) = 2} \\
&= \begin{Bmatrix} \text{CB}(1, 1) \\ \text{CB}(2, 1) \\ \text{CB}(3, 1) \end{Bmatrix} && \text{if PCON (NB1) = 3}
\end{aligned}$$

FCON (M) — code used to define the body label of the body in which free coordinate vector M is fixed.

FCON (M) = K      if free coordinate vector M is fixed in body K

FCON (M) = -(M-1)    if free coordinate vector M is the intermediate axis of a three axis gimbal

H — see section "Data Input, Subroutine INBS."

{HM (I, M)} — see section "Data Input, Subroutine INBS."

HMOM (M) — relative angular momentum of momentum wheel M.

$$\text{HMOM (M)} = \text{PLM (M)} * \text{THADW (M)}$$

JCON (K) — see section "Data Input, Subroutine INBS."

LCON (L) — code used to define the body label of the body in which locked coordinate ordinate vector L is fixed

LCON (L) = K      if locked coordinate vector L is fixed in body K

= -M      if locked coordinate vector L is the axis about which relative motion is kinematically constrained on a two axis gimbal defined by the free vector M and M+1

MO (M) —

NBOD —

NB1 —

} see section "Data Input, Subroutine INBS."

NFER — total number of free coordinate vectors

NLOR — total number of locked coordinate vectors

NMO —

PCON (K) —

PLM (M) —

} see section "Data Input, Subroutine INBS."

{QF (I, M)}	-	}	see section "Data Input, Subroutine INBS"
{QL (I, L)}	-		
RBLO (K)	-		
THA (M)	-		
THAD (M)	-		
THAW (M)	-		
THADW (M)	-		
TIMEND	-		
[XI(I, J, K)]	-		
XMAS (K)	-		
[XMT (I, J, K)]	-		

### PROGRAMMER'S GUIDE TO SUBROUTINE INEROR

The main function of subroutine INEROR is to check the data inputted by subroutine INBS for obvious errors.

The following error checks are made:

- All bodies must have positive mass greater than zero.
- The topological tree must be properly labeled; i.e., JCON (K) must be less than K for all K where  $K = 1, 2, \dots, \text{NBOD}$ .
- The number of kinematic constraints must be 0, 1, 2, or 3.
- Inertia tensors must be symmetric and have a positive nonzero determinant.
- Zero-state transformation matrices must be orthonormal.

If any of the above tests are not satisfied, the flag FG2 is set .FALSE.. Appropriate error messages are printed to assist the user in locating the error source, and the job will terminate after a return to the MAIN program.

### PROGRAMMER'S GUIDE TO SUBROUTINE SETS

The main function of subroutine SETS is to interrogate the data inputted by subroutine INBS and compute various sets of integers which will enable the computer to avoid redundant and trivial computation. To save computer storage, code words are created that compact the desired integer arrays into a single integer code word. Triangular arrays are stored as one-dimensional arrays in the memory of the computer. The following variables, defined

in table 1, are either first introduced or used extensively in subroutine SETS. The programmer is referred to the comment cards inserted therein and the following for detailed definitions.

SG — code word used to define the body labels of all gyrostats; i.e., all bodies that have one or more imbedded symmetric wheels.

SI (KN) — one dimensional array of code words used to define the body labels of all bodies on the topological path extending from hinge point K-1 outward to the center of mass of body N, where

$$KN = KT0(NBOD, K-1, N)$$

$$K = 1, 2, \dots, NBOD$$

$$N \geq K$$

The utility function KT0 is used to store the triangular array of code words in a one-dimensional array.

SK (K-1) — code word used to define the body labels of those bodies contained in the nest of bodies K-1;  $K = 1, 2, \dots, NBOD$ . Due to the setup of labeled common block /INTGZ/, SK(0) is stored in the location SKDUM.

SL — code word used to define the body labels of all point masses.

SQF (K) — lowest magnitude free coordinate vector label at hinge point K-1;  $K = 1, 2, \dots, NBOD+1$ , where

$$K = NBOD+1 \quad \text{implies the inertial origin}$$

and

$$SQF(K) = 0 \quad \text{if } PCON(K) = 3$$

SQL (K) — lowest magnitude locked coordinate vector label at hinge point K-1;  $K = 1, 2, \dots, NBOD+1$ , where

$$K = NBOD+1 \quad \text{implies the inertial origin}$$

and

$$SQL(K) = 0 \quad \text{if } PCON(K) = 0$$

SR — code word used to define the body labels of all rigid and flexible bodies.

## DATA INPUT, SUBROUTINE INOPT

Subroutine INOPT is the second input subroutine called by the MAIN program. The input cards to be read by it follow immediately after those read by subroutine INBS in the data deck.

The program N-BOD2 has been coded to accept a description of the basic simulation model by subroutine INBS. Amplifications or modifications to the model or to the equations of motion that describe it are defined by subroutine INOPT.

In many problems of practical interest, the user will not have to exercise any of the available options. In such cases all parameters required for computation, which could have been user defined, are set by default to the values that, in the author's opinion, are most commonly desired. If no options are to be used, INOPT will read only one data card.

INOPT is programmed to recognize different code words that are used to define which of the available options is to be used. The code words are punched on what will be referred to as "option cards." All option cards are read at program point 12 in INOPT by the statement

```
12 READ 100, ICD1, ICD2, ICD3, I, NSET, (S1 (J), J=1, 10)
100 FORMAT (3A4, I3, 11I5)
```

Columns 1 through 12 of each option card contain the code word. Table 6 contains a listing of all code words, the columns in which the alphanumeric characters must be punched, and a brief description of the function of the option.

Upon recognition of a code word, the program branches to the appropriate point in the program where all required computational and additional READ data operations are carried out. Upon completion, the program branches back to program point 12 to read the next option card. The cycling back to program point 12 continues until an END OPTIONS option card is recognized. The END OPTIONS card is the last card to be read by INOPT.

If a code word is not recognized, the flag FG3 will be set .FALSE., an error message will be printed, and return to the MAIN program will be immediately made.

The following subsections define in detail the various option codes recognized by INOPT and their purpose.

### \*\*\* Terminate Option Input \*\*\*

#### 1. No Option Required

If no options are required for the simulation model, the code word NO OPTIONS must be punched on the data card immediately following the data cards read by subroutine INBS. In this case, all modifiable parameters are set by default. The NO OPTIONS card would, in this case, be the only card read by INOPT.

Table 6.  
Option Codes Recognized by INOPT and Function

GODDARD SPACE FLIGHT CENTER  
FORTRAN CODING RECORD

PROGRAM PROGRAMMER		DATE	PUNCHING INSTRUCTIONS	GRAPHIC PUNCH	PAGE	OF	CARD ELECTRO NUMBER*	
1	OPTION CODE	FUNCTION OF OPTION	FORTRAN STATEMENT					IDENTIFICATION SEQUENCE
2	NO OPTIONS	NO OPTION CARDS TO BE READ						
3	END OPTIONS	END OF OPTION CARDS						
4	INERTIAL	USE INERTIAL COORDINATES FOR COMPUTATION						
5	BODY I	USE BODY I COORDINATES FOR COMPUTATION						
6	ASETX	MODIFY ARRAY SPI						
7	ASETXC	MODIFY ARRAY SIX						
8	ASETGEN	MODIFY ARRAY SGN						
9	ASETCOR	MODIFY ARRAY SCR						
10	EULERANG	MODIFY ARRAY SEU						
11	SMA LLANG	MODIFY ARRAY SMAL						
12	VTLNBOD	MODIFY ARRAY SYD						
13	FREE	MODIFY ARRAY SFR						
14	LOCK	MODIFY ARRAY SLK						
15	MOMANGLE	MODIFY ARRAY SMA						
16	CFORCE	MODIFY ARRAY SFX						
17	VARWHEEL	MODIFY ARRAY SMV						
18	COLX	MODIFY ARRAY SXT						
19	CAGE	SOME DEGREES OF FREEDOM CAGED						
20	FLEXIBLE	SOME BODIES ARE TO BE FLEXIBLE						
21	MODE CUP	SOME VIBRATION MODES CROSS-COUPLED						

OSFC 6-11/2/69



## 2. End Options

If the user is to make use of one or more options, the last card of the cards to be read by INOPT must have the code word END OPTIONS punched on it. If it does not, an input error will result and the program will terminate with an appropriate error message.

### \*\*\* Optimize Computation Efficiency \*\*\*

In order to optimize computation speed, all unnecessary, trivial, and redundant computation must be avoided. By an interrogation of the input data, numerous sets of integers can be defined that can be used to carry out the above task. It is not possible for the computer to pick out "almost trivial" computations; this the engineer must do. By use of particular option cards, he can literally insert engineering judgment into N-BOD2.

In practical application, the author has found that usually there are not enough "almost trivial" terms that can be justifiably deleted to make a significant impact on computation speed. Consequently, while the options are available for the user, the author has rarely made practical use of most of them.

## 3. Choice of Frame of Computation

The equations of motion, as derived, are in a vector-dyadic form and are valid for all frames of reference. The user may choose the frame of computation to be either the inertially fixed-reference frame or the body 1 fixed-reference frame. All vectors and tensors required for computation are stored in memory in computing frame coordinates. The user should define the frame of computation to be the reference frame in which he wishes to output most of the computed system state variable vectors.

The logic variable INERF is used to define which frame of computation is to be used for internal computation.

INERF = .TRUE. if the frame of computation is to be the inertially fixed-reference frame. Recognition of the code word INERTIAL on an option card will set INERF equal to .TRUE.

INERF = .FALSE. if the frame of computation is to be the body 1 fixed-reference frame. Recognition of the code word BODY 1 on an option card will set INERF equal to .FALSE.

Default INERF = .FALSE.

Body 1 will be the frame of computation unless otherwise directed.

## 4. Truncate Computation of Pseudo-inertia Tensors

The vector-dyadic form of the simulation model's equations of motion is given in Reference 1 by equation 88 and in Reference 2 by equation 111. The coefficient matrix on the left-hand side of the equation contains numerous inertia and pseudo-inertia tensors that are

defined by equations 63 through 70 of Reference 1. If the simulation model has both large and small bodies, some of the small bodies may contribute a nearly trivial amount to the pseudo-inertia tensors associated with the nests that contain both the large and the small bodies.

To compute the inertia and pseudo-inertia tensors defined by equations 63 through 70 of Reference 1, the integer code words SPI(I), I = 0, 1, . . . , NBOD-1, are used in subroutine XDY.

**SPI(I)** — code word. Used to define the set of bodies contained in the nest I that contribute significantly in the computation of the inertia and pseudo-inertia tensors associated with the nest I; I = 0, 1, . . . , NBOD-1.

Upon recognition of the code word ASETX, INOPT makes use of the data in columns 13 through 70 of the option card to redefine the code word SPI(I). The following definitions apply for this option:

I = nest number

NSET = number of body labels to be defined to be in the set defined by SPI(I)

(S1 (J), J = 1, NSET) = the body labels defined to be in the set defined by SPI(I).

The code word SPI(I) is created by the operation

CALL COMPAC (S1, NSET, SPI(I)).

Default SPI(I) = SK(I); I = 0, 1, . . . , NBOD-1

Unless otherwise directed, all bodies contained in the nest I are assumed to contribute significantly to the inertia and pseudo-inertia tensors associated with the nest I.

## 5. Truncate Computation of Inertia Cross-Coupling Torques

In Reference 1, the inertia cross-coupling torque acting on the nest I can be found from equations 79 and 80. In Reference 2, it can be found in equation 92. It is given in the notation of these references as

$$\sum_{\lambda \in S_I} \left[ \dot{\Phi}_\lambda \cdot \vec{\omega}_\lambda + \vec{\omega}_\lambda \times \left( \Phi_\lambda \cdot \vec{\omega}_\lambda + \frac{\Delta \vec{L}_{\lambda,\lambda}}{m_{\lambda,\lambda}} \right)_i \right]$$

where the summation extends over all bodies contained in the nest I. In subroutine ETA, the integer code words SIX(I), I = 0, 1, . . . , NBOD-1, are used to define which bodies contribute significantly to the above summation.

SIX(I) — code word. Used to define the set of bodies contained in the nest I that contribute significantly to the computation of the inertia cross-coupling torque acting on the nest I; I = 0, 1, . . . , NBOD-1.

Upon recognition of the code word ASETIXC, INOPT makes use of the data in columns 13 through 70 of the option card to redefine the code word SIX(I). The following definitions apply for this option:

I = nest number

NSET = number of body labels to be defined to be in the set defined by SIX(I).

(S1 (J), J = 1, NSET) = the body labels defined to be in the set defined by SIX(I).

The code word SIX(I) is created by the operation

CALL COMPAC (S1, NSET, SIX(I)).

Default Unless otherwise directed, all rigid and flexible bodies contained in the nest I are assumed to contribute significantly to the inertia cross-coupling torque acting on the nest I.

NOTE: Point masses have zero contribution and hence their body labels are not contained in any of the sets defined by the code words SIX(I); I = 0, 1, . . . , NBOD-1.

## 6. Truncate Computation of Centripetal and some of the Coriolis Cross-Coupling Torques

In Reference 1, the centripetal and Coriolis cross-coupling forces and torques appearing on the right-hand side of equation 88 are defined by equations 75 through 78. In Reference 2, the analogous terms are found in equations 100, 107, and 108.

In the notation of Reference 2, the gyroscopic torque associated with these effects acting on the nest I is given by

$$\sum_{\lambda \in S_I} \vec{\gamma}_{I,\lambda} \times \vec{C}_\lambda \quad ; I = 0, 1, \dots, \text{NBOD}-1$$

where

$$\vec{C}_\lambda = m_\lambda \left[ \sum_{\substack{i \in S_{0,\lambda-1} \\ i \neq 1}} \left( \vec{\omega}_{J(i)} \times \left( \vec{\omega}_{J(i)} \times \vec{\beta}_i \right) + 2\vec{\omega}_{J(i)} \times \dot{\vec{\beta}}_i \right) + \vec{\omega}_\lambda \times \left( \vec{\omega}_\lambda \times \vec{\alpha}_\lambda \right) + 2\vec{\omega}_\lambda \times \dot{\vec{\alpha}}_\lambda \right]$$

and the summation extends over all bodies contained in the nest I.

In subroutine ETA, the integer code words SCN(I) and SCR(I),  $I = 0, 1, \dots, \text{NBOD}-1$ , are used to define which bodies contribute significantly to the above summation.

SCN(I) — code word. Used to define the set of bodies contained in the nest I which contribute significantly to the computation of the centripetal and Coriolis cross-coupling torques acting in the nest I;  $I = 0, 1, \dots, \text{NBOD}-1$ .

SCR(I) — code word. Used to define the set of point masses contained in the nest I for which their Coriolis acceleration effects contribute significantly to the computation of the centripetal and Coriolis cross-coupling torques acting on the nest I;  $I = 0, 1, \dots, \text{NBOD}-1$ .

Upon recognition of the code words ASETCEN or ASETCOR, INOPT makes use of the data in columns 13 through 70 of the option card to redefine the code words SCN(I) or SCR(I) respectively. The following definitions apply for these options:

I = nest number

NSET = number of body labels to be defined by SCN(I) or SCR(I), respectively.

(S1 (J),  $J = 1, \text{NSET}$ ) = the body labels defined to be in the set defined by SCN(I) or SCR(I), respectively.

The code word SCN(I) is created by the data on the ASETCEN option card by the operation

CALL COMPAC (S1, NSET, SCN(I)).

The code word SCR(I) is created by the data on the ASETCOR option card by the operation

CALL COMPAC (S1, NSET, SCR(I)).

Default Unless otherwise directed, all bodies contained in the nest I are assumed to contribute to the centripetal and Coriolis cross-coupling torques acting on the nest I;  $I = 0, 1, \dots, \text{NBOD}-1$ .

NOTE: The elements of sets defined by the code words SCR(I),  $I = 0, 1, \dots, \text{NBOD}-1$ , are only point-mass body labels.

## 7. Choice of Kinematic Techniques

The user has the choice of which type of kinematic formulation should be used to define the transformation matrices. In the first version of N-BOD2, direction cosine techniques were used exclusively. In an attempt to improve computation speed, it was noted that the

integration of the equations of motion yielded Euler angles directly. These could be used to directly obtain the required transformation matrices and eliminate the need for setting up and integrating six direction cosine equations per transformation matrix. It was further reasoned that time-consuming trigonometric operations could also be dropped if it was known that the Euler angles would be small angles. Practical application has shown that the computational operations required to go from Euler angles or small Euler angles to the transformation matrices takes about the same amount of computer time as the operations required to set up and integrate the direction cosine equations.

These three approaches are available for the user in N-BOD2. The author personally uses direction cosine techniques for most simulation work; however, if it is critical that the transformation matrices maintain orthonormality to a high degree of accuracy, Euler angle techniques are preferable. There is no significant computational speed advantage associated with any of the techniques. The Euler angle and small angle approach requires that the body fixed free vectors, about which the Euler angles are measured, be aligned with the coordinate axes of their respective body fixed-reference frames; a restriction not imposed in the direction cosine approach.

In subroutine DCT, the direction cosine equations are set up and in subroutine TRAN, transformation matrices are computed. The code words SD, SEU, and SMAL are used in these subroutines to define the particular kinematic approach that is to be used.

SD = code word. Used to define all contiguous pairs of bodies (JCON(I), I) for which direction cosine techniques are used to compute transformation matrices.

SMAL = code word. Used to define all contiguous pairs of bodies (JCON(I), I) for which small angle techniques are used to compute transformation matrices.

SEU = code word. Used to define all contiguous pairs of bodies (JCON(I), I) for which Euler angle techniques are used to compute transformation matrices.

Upon recognition of the code word EULERANG, the data in columns 13 through 70 are used to create the code word SEU by the operation

CALL COMPAC (S1, NSET, SEU).

Upon recognition of the code word SMALLANG, the data in columns 13 through 70 are used to create the code word SMAL by the operation

CALL COMPAC (S1, NSET, SMAL).

NOTE: Redundancy is not permitted. Pairs of bodies defined by the user to be in either of the sets defined by SEU or SMAL will be subtracted from the set defined by SD

In columns 13 through 17 of the option cards,

I = unused

NSET = the number of pairs of bodies for which the special kinematic techniques are to be used.

(S1 (J), J = 1, NSET) = the labels defining the pairs. The *J*th pair is defined as body JCON(SI(J)) and body SI(J).

**Default** Unless otherwise directed, all contiguous pairs of rigid or flexible bodies are contained in the set defined by SD. Both SEU and SMAL are empty sets unless otherwise directed.

### 8. Suppress Selected Coordinate Transformations

All nontrivial body fixed vectors and tensors are transformed to computing frame coordinates four times per integration step. Under certain conditions, this can be a wasteful operation. If it can be determined that the components of all body fixed vectors and tensors of certain bodies are essentially time invariant in computing frame coordinates, it makes no sense to retransform them at every integration step.

The avoidance of these redundant transformations is controlled by the code word SVD in VDIV and TRANVD.

SVD — code word. Used to define the set of all bodies for which body-fixed vectors and tensors are time varying in the frame of computation.

Upon recognition of the code word VTINBOD, the data in columns 13 through 70 are used to create the code word SVD by the operation

CALL COMPAC (S1, NSET, SVD).

In columns 13 through 70 of the option card,

I = unused

NSET = the number of bodies for which body-fixed vectors and tensors are time varying in the frame of computation.

(S1 (J), J = 1, NSET) = the body labels defined to be in the set defined by SVD.

**Default** Unless otherwise defined, all body labels are contained in the set SVD. If the computing frame is the body 1 fixed-coordinate frame, body label 1 is automatically subtracted from SVD.

**NOTE:** Body labels associated with point masses are included in SVD since their nominal center-of-mass positions are given in their respective body-fixed coordinate system.

## 9. Suppress Recomputation of Elements of Coefficient Matrix of Inertia Tensors

In References 1 and 2, the matrix  $\mathbf{X}$  is a symmetric  $(NB1) \times (NB1)$  matrix of inertia and pseudo-inertia tensors, the elements of which are given by equations 63 through 70 in Reference 1. Under certain conditions, one or more of the columns (down to the diagonal) of this matrix will have nonzero entries that are nearly constant in time. Unless otherwise directed, the program recomputes these terms four times per integration step. The avoidance of these redundant computations in subroutine XDY is controlled by the code word SXT.

**SXT** — code word. Used to define the columns of the matrix  $\mathbf{X}$  of pseudo-inertia tensors that have time-varying elements.  $\mathbf{X}$  is symmetric and columns referred to extend down only to the diagonal element. Column NB1 always is taken as having time-varying elements. The elements of column  $K$ ,  $K = 2, \dots, NBOD$ , give the inertia contributions of the next  $K-1$  to the system's equations of motion.

Upon recognition of the code word COLX, the data in columns 13 through 70 are used to create the code word SXT by the operation

```
CALL COMPAC (S1, NSET, SXT).
```

In columns 13 through 70 of the option card

I = unused

NSET = the number of columns to have time-varying elements (not including column NB1,  $NB1 = NBOD + 1$ ).

(S1 (J), J = 1, NSET) = the column numbers of the columns of  $\mathbf{X}$  that will have time-varying elements.

Default Unless otherwise defined, all body labels are contained in the set SXT. This implies that all nonzero elements of  $\mathbf{X}$  will be assumed to be time varying.

## 10. Suppress Computation of Relative Displacements

A redundancy exists in the computation of kinematics. All transformation matrices may be computed by a solution of direction cosine equations. Relative displacement angles, which provide basically the same information, are found by an integration of Euler rate equations. If direction cosine methods are used, it usually is not necessary to compute all relative displacement angles since their only use is for a description of torques acting between bodies that are a function of relative displacement angle (springs) and for output display. Accordingly, the user may selectively suppress their computation.

The avoidance of the computation of relative displacement angles is controlled by the integer array SFR in subroutines SETUP and ANGLE.

NFRC — total number of free coordinate vectors about or along which relative displacement is to be computed.

SFR (I) = the labels associated with the free coordinate vectors about or along which relative displacement is to be computed.  $I = 1, 2, \dots, \text{NFRC}$

Upon recognition of the code word FREE, the data in columns 13 through 70 are used to define the elements of the set SFR. In columns 13 through 70 of the option card,

I = unused

NSET = the number of free coordinate labels to be defined on this card for inclusion in the set SFR.

(SI (J),  $J = 1, \dots, \text{NSET}$ ) = the free coordinate labels to be defined to be in the set SFR.

The input data format statement allows a maximum of 10 free coordinate vector labels to be defined per option card. Several option cards with the code word FREE may be used if required.

Default Unless otherwise defined, all relative displacement rate equations will be set up and integrated.

NOTE: INOPT is programmed to prohibit the user from suppressing the computation of angles required for use in conjunction with either the Euler angle or the small angle option. It also prohibits the suppression of the computation of the coordinates that define body 1 translational displacement or point mass displacement.

### 11. Suppress Computation of Symmetric Wheel Angular Position

In spacecraft application, the integration of the symmetric wheel rate equation to ascertain the angular position of a mark on a symmetric wheel is usually a time-consuming and wasteful practice since its rates are usually high relative to the spacecraft frequencies of interest. At times, however, it may be important to know the wheel's angular position.

The determination as to which wheel rate equation should be integrated is controlled in subroutine SETUP by the integer array SMA.

NMOA — total number of symmetric wheel rate equations to be integrated.

SMA(I) — the labels of the symmetric wheels for which wheel angular position is to be computed.  $I = 1, 2, \dots, \text{NMOA}$

Default Unless otherwise directed, none of the wheel rate equations will be integrated.

Upon recognition of the code word MOMANGLE, the data in columns 13 through 70 are used to define the elements of the set SMA. In columns 13 through 70 of the option card,

I = unused



NSET = the number of symmetric wheel labels to be defined on this card for inclusion in the set SMA.

(S1 (J), J = 1, NSET) = the symmetric wheel labels to be defined to be in the set

The input data format statement allows a maximum of 10 wheel labels to be defined per option card. Several option cards with the code word MOMANGLE may be used if desired.

## 12. Computation of Forces and Torques of Constraint

The code words CFORCE and LOCK are recognized and may be used to redefine the elements in the integer array SFK and SLK, respectively. These arrays were to be used to define which forces and torques of constraint should be computed. This option has not been coded and is not available for users of N-BOD2.

### Default

NCTC = 0

NFKC = 0

SFK = 0

SLK = 0

### \*\*\* Expand Modeling Capability \*\*\*

## 13. Specification of Constant Speed Wheels

Wheels that are controlled to spin at a constant speed should be modeled as constant speed wheels. Computationally, this reduces the order of the equations of motion of the system and eliminates the need for defining a control system that will maintain constant wheel rate.

The code word SMV is used in subroutines QFDOT and SETUP to define which wheels are to be treated as variable speed wheels.

SMV — code word. Used to define which of the symmetric wheels are to be treated as variable speed wheels; all others are treated as constant speed wheels.

NMV — total number of variable speed wheels.

Upon recognition of the code word VARWHEEL, the data in columns 13 through 70 are used to create the code word SMV by the operation

CALL COMPAC (S1, NSET, SMV).

In columns 13 through 70 of the option card,

I = unused

NSET = number of wheels to be defined as variable speed wheels.

(S1 (J), J = 1, NSET) = the labels of the variable speed wheels.

Only one VARWHEEL option card is permitted.

Default Unless otherwise directed, all symmetric wheels are assumed to be variable speed wheels.

#### 14. Caging and Uncaging of Degrees of Freedom

In order to simulate the deployment of spacecraft appendages, it is desirable to have the capability to increase the number of allowable degrees of freedom during a simulation run. A "caged degree of freedom" is a degree of freedom that is not allowed for a predefined time. The uncaging of degrees of freedom at predefinable times allows the user to simulate deployment transients.

The following definitions are used:

SCG — total number of caged degrees of freedom at time zero.

SC (I) — labels of the free coordinate vectors about or along which relative motion will be constrained for a predefined time. I = 1, 2, . . . , SCG

TUG (I) — time at which motion about or along free coordinate vector SC(I) will be allowed (uncaged). I = 1, 2, . . . , SCG

NOTE: All uncaging takes place with zero initial relative rate. The coding to account for the impulse effect that is associated with uncaging with an initial relative rate has not been included in N-BOD2.

Upon recognition of the code word CAGE, the procedure for inputting the data required to use this option is started. In columns 13 through 70 of the option card,

I = unused

NSET = SCG, the total number of degrees of freedom caged at time zero.

(S1 (J), J = 1, NBOD) = unused.

Their labels and times for uncaging are then read according to the following:

READ 101, (SC(I), TUG(I), I=1, SCG)

101 FORMAT (I5, D 15.5)

After these data cards are read, a check is made to make sure that the initial rates associated with the caged degrees of freedom are zero. If not, they are reset to zero.

Default Unless otherwise directed, all degrees of freedom are assumed uncaged at time zero.

## 15. Flexible Body Option

The user is encouraged to first refer to the section entitled "Flexible versus Rigid Body Modeling" in Reference 2 before making use of the flexible body option. Contained therein are equations that provide a guide for estimating the required number of vibration modes and a discussion of the assumptions implicit in the formulation of the coupled flexible body equations of motion.

N-BOD2 accepts only resultant mode dependent parameters. Their description is based upon the assumption that a finite element model of the flexible body can be defined.\*

The following definitions are taken directly from Reference 2 and are used to define the required resultant mode dependent parameters.

$m_{i,\lambda}$  = mass of element  $i$  of body  $\lambda$

${}^o\Phi_{i,i,\lambda}$  = inertia tensor of element  $i$  of body  $\lambda$  about its own center of mass in the undeformed state of body  $\lambda$

$\vec{r}_{i,\lambda}$  = position vector from hinge point  $\lambda-1$  to the undeformed center-of-mass position of the element  $i$  of body  $\lambda$

${}^o\vec{a}_\lambda$  = position vector from hinge point  $\lambda-1$  to the undeformed center-of-mass position of body  $\lambda$

$\vec{\phi}_{n,i,\lambda}^T$  =  $n$ th normal mode displacement vector for the element  $i$  of body  $\lambda$

$\vec{\phi}_{n,i,\lambda}^R$  =  $n$ th normal mode rotation vector for the element  $i$  of body  $\lambda$

$m_\lambda$  = total mass of body  $\lambda$

$\delta_{m,n}$  = Kronecker delta function

$N_{i,\lambda}$  = total number of elements  $i$  used in the finite element model of body  $\lambda$

\*The user is also referred to Appendix B in which a NASTRAN DMAP program and a preprocessor program are listed. The output of these two programs define the resultant mode dependent parameter required.

The normalization condition imposed upon the modes is

$$\sum_{i \in \lambda} \left[ m_{i,\lambda} \overset{\rightarrow T}{\varphi}_{m,i,\lambda} \cdot \overset{\rightarrow T}{\varphi}_{n,i,\lambda} + \overset{\rightarrow R}{\varphi}_{m,i,\lambda} \cdot \left( \overset{\circ}{\Phi}_{i,i,\lambda} \cdot \overset{\rightarrow R}{\varphi}_{n,i,\lambda} \right) \right] = m_{\lambda} \delta_{m,n}$$

where

$$\lim_{N_{i,\lambda} \rightarrow \infty} \sum_{i \in \lambda} \overset{\rightarrow R}{\varphi}_{m,i,\lambda} \cdot \left( \overset{\circ}{\Phi}_{i,i,\lambda} \cdot \overset{\rightarrow R}{\varphi}_{n,i,\lambda} \right) = 0$$

The resultant mode dependent parameters required as input data to N-BOD2 are given by equations 48, 50, 51, 54, 55, 67, 70, and 72 in Reference 2. They are

$$\overset{\rightarrow}{A}_{n,\lambda} = \frac{1}{m_{\lambda}} \sum_{i \in \lambda} m_{i,\lambda} \overset{\rightarrow T}{\varphi}_{n,i,\lambda}$$

$$\overset{\rightarrow}{D}_{n,\lambda} = \sum_{i \in \lambda} m_{i,\lambda} \left[ \left( \overset{\circ}{\Gamma}_{i,\lambda} - \overset{\circ}{\alpha}_{\lambda} \right) \cdot \overset{\rightarrow T}{\varphi}_{n,i,\lambda} \mathbf{1} - \left( \overset{\circ}{\Gamma}_{i,\lambda} - \overset{\circ}{\alpha}_{\lambda} \right) \overset{\rightarrow T}{\varphi}_{n,i,\lambda} \right]$$

$$\overset{\rightarrow}{B}_{n,\lambda} = \frac{1}{m_{\lambda}} \sum_{i \in \lambda} m_{i,\lambda} \overset{\circ}{\Gamma}_{i,\lambda} \times \overset{\rightarrow T}{\varphi}_{n,i,\lambda}$$

$$\overset{\rightarrow}{C}_{n,\lambda} = \sum_{i \in \lambda} \overset{\circ}{\Phi}_{i,i,\lambda} \cdot \overset{\rightarrow R}{\varphi}_{n,i,\lambda}$$

$$\overset{\rightarrow}{J}_{n,\lambda} = \sum_{i \in \lambda} \overset{\rightarrow R}{\varphi}_{n,i,\lambda} \times \overset{\circ}{\Phi}_{i,i,\lambda}$$

$$\overset{\rightarrow}{F}_{m,n,\lambda} = \sum_{i \in \lambda} m_{i,\lambda} \left[ \overset{\rightarrow T}{\varphi}_{m,i,\lambda} \cdot \overset{\rightarrow T}{\varphi}_{n,i,\lambda} \mathbf{1} - \overset{\rightarrow T}{\varphi}_{m,i,\lambda} \cdot \overset{\rightarrow T}{\varphi}_{n,i,\lambda} \right]$$

$$\overset{\rightarrow}{K}_{m,n,\lambda} = \sum_{i \in \lambda} \left\{ m_{i,\lambda} \overset{\rightarrow T}{\varphi}_{m,i,\lambda} \times \overset{\rightarrow T}{\varphi}_{n,i,\lambda} + \frac{1}{2} \left( \overset{\circ}{\Phi}_{i,i,\lambda} \cdot \overset{\rightarrow R}{\varphi}_{m,i,\lambda} \right) \times \overset{\rightarrow R}{\varphi}_{n,i,\lambda} \right\}$$

If mode shapes are obtained by a continuum analysis, the rotational inertia effects of the infinitesimal elements about their respective centers of mass is negligible; hence

$$\lim_{N_{i,\lambda} \rightarrow \infty} \vec{C}_{n,\lambda} = 0$$

$$\lim_{N_{i,\lambda} \rightarrow \infty} \mathbf{J}_{n,\lambda} = 0$$

$$\lim_{N_{i,\lambda} \rightarrow \infty} \vec{K}_{m,n,\lambda} = \sum_{i \in \lambda} m_{i,\lambda} \vec{\varphi}_{m,i,\lambda}^T \times \vec{\varphi}_{n,i,\lambda}^T$$

and

$$\lim_{N_{i,\lambda} \rightarrow \infty} \left[ \sum_{i \in \lambda} m_{i,\lambda} \vec{\varphi}_{m,i,\lambda}^T \cdot \vec{\varphi}_{n,i,\lambda}^T - m_{\lambda} \delta_{m,n} \right] = 0$$

The other mode-dependent parameters that are used or computed are:

- $\omega_{n,\lambda}$  — natural frequency of mode n of body  $\lambda$  (rad/...)
- $\zeta_{n,\lambda}$  — damping ratio for mode n of body  $\lambda$
- $a_{n,\lambda}(t)$  — generalized displacement coordinate for mode n of body  $\lambda$
- $\dot{a}_{n,\lambda}(t)$  — generalized displacement coordinate rate for mode n of body  $\lambda$

The position vector from the undeformed to the deformed center-of-mass position of the element i of body  $\lambda$  is

$$\vec{e}_{i,\lambda} = \sum_{n,\lambda} a_{n,\lambda}(t) \vec{\varphi}_{n,i,\lambda}^T$$

where the summation is overall modes defined for body  $\lambda$ .

The following parameters are defined in subroutine INOPT and used within the various subroutines to compute the effects of body flexibility:

- NFLXB — total number of flexible bodies
- SFXM (I) — total number of flexible body modes to be used to describe the flexible body characteristics of body I (zero if body I is a rigid body or a point mass);  $I = 1, 2, \dots, \text{NBOD}$ .
- SFLX — code word used to define the body labels of those bodies that are to be treated as flexible bodies.
- NMODS — total number of flexible body modes used for entire simulation model.

Upon recognition of the code word FLEXIBLE, the procedure for inputting uncoupled modal data starts. In columns 13 through 70 of the option card,

I = unused

NSET = total number of bodies to be defined to be flexible.

(S1 (I), I = 1, NBOD) = the number of flexible body modes to be used to describe the flexible body characteristics of body I; I = 1, 2, . . . , NBOD. (Zero if body I is a rigid body or point mass).

The parameters NFLXB, SFXM, SFLX, and NMODS are computed by an internal process that makes use of the above data provided on the option card.

To conserve computer storage and increase computation speed, all mode-dependent parameters are sequentially numbered from 1 to NMODS by an internally defined algorithm that makes use of the SFXM integer array. All parameters associated with mode M of body K are given the integer label MN which is computed by the algorithm

$$MN = M + \sum_{I=1}^{K-1} SFXM(I)$$

where

$$K = 2, 3, \dots, NBOD$$

and

$$M = 1, 2, \dots, SFXM(K)$$

MN will be referred to as the "mode number."

All uncoupled mode-dependent data required by N-BOD2 are read immediately after recognition of the code word FLEXIBLE on an option card. For mode M of flexible body K, the following data must be defined. Its mode number MN is defined as

$$MN = M + \sum_{I=1}^{K-1} SFXM(I)$$

and recall that

NFER = total number of free coordinate vectors.

Then

FLOM (MN) =  $\omega_{M,K}$  the modal natural frequency (rad/. . .)

ZETA (MN) =  $\zeta_{M,K}$  the modal damping ratio

- THA (NFER + MN) =  $a_{M,K}(t)$  generalized displacement coordinate (initial value to be inputted)
- THAD (NFER + MN) =  $\dot{a}_{M,K}(t)$  the generalized displacement coordinate rate (initial value to be inputted)
- {FLA (I, MN)} = components of the resultant mode-dependent vector  $\vec{A}_{M,K}$  relative to the body K fixed reference frame.
- {FLB (I, MN)} = components of the resultant mode-dependent vector  $\vec{B}_{M,K}$  relative to the body K fixed reference frame.
- {FLC (I, MN)} = components of the resultant mode-dependent vector  $\vec{C}_{M,K}$  relative to the body K fixed reference frame.
- [FLD (I, J, MN)] = components of the resultant mode-dependent dyad  $D_{M,K}$  relative to the body K fixed reference frame.
- [FLJ (I, J, MN)] = components of the resultant mode-dependent dyad  $J_{M,K}$  relative to the body K fixed reference frame.

The data required to describe the uncoupled modal data for each flexible body are read according to the following sequence of read statements.

```

***** READ STATEMENTS FOR OPTION 'FLEXIBLE' IN INOPT *****
C
C   INITIALIZE MODE LABELS
C   MN = 0
C
***** READ UNCOUPLED MODE DATA FOR ALL FLEXIBLE BODIES *****
DO 1 K=1,NBOD
IF(SFXM(K).EQ.0) GO TO 1 BODY K IS NOT A FLEXIBLE BODY.
READ 103, N,(MESS(J),J=1,18)
IF(K.NE.N) ERROR,CARDS OUT OF SEQUENCE, PRINT MESSAGE HALT
C   N = BODY LABEL FOR WHICH FLEXIBLE BODY DATA TO FOLLOW APPLIES
C   MESS = ALPHANUMERIC DESCRIPTION OF BODY N, PRINTED WITH DATA ECHO
C
***** READ EXACTY SFXM(K) SETS OF MODAL DATA FOR BODY K *****
DO 2 M =1,SFXM(K)
MN = MN + 1
READ 104, FLOM(MN),ZETA(MN)
READ 104, THA(NFER+MN),THAD(NFER+MN)
READ 104, (FLA(I,MN),I=1,3)
READ 104, (FLB(I,MN),I=1,3)
READ 104, (FLC(I,MN),I=1,3)
READ 104, ((FLD(I,J,MN),J=1,3),I=1,3)
READ 104, ((FLJ(I,J,MN),J=1,3),I=1,3)
2 CONTINUE
***** ALL UNCOUPLED MODAL DATA FOR BODY K HAS BEEN READ *****
C
C
1 CONTINUE

```

```

***** ALL UNCOUPLED MODAL DATA HAS BEEN READ *****
C
C
*****      END OF DATA FOR OPTION 'FLEXIBLE'      *****
C
C
C      FORMAT STATEMENTS
103 FORMAT (15,18A4)
104 FORMAT (3D15.5)

```

The input of the coupled modal data follows the option card 'MODE CUP.'

In the process of developing the equations that define centripetal and Coriolis related loads associated with flexible bodies, certain mode-dependent, cross-coupling parameters are derived; namely  $F_{m,n,\lambda}$  and  $\bar{K}_{m,n,\lambda}$  (previously defined in this section). In many practical applications, most, if not all, of these terms are identically or nearly equal to zero. Computer time and storage can be conserved by developing a scheme which will only use nontrivial cross-coupling terms.

The following parameters are defined in subroutine INOPT and used in subroutine QFDDOT to compute the modal cross-coupling effects.

SFCC = code word. Used to define the body labels of the flexible bodies that have flexible body modes that are cross coupled (significantly).

SCXC (MN) = code word. Used to define the modes M of body K which significantly cross couple in the generalized displacement coordinate equation for the mode having mode number MN

$$MN = N + \sum_{I=1}^{K-1} SFXM(I)$$

From equation 95 of Reference 2, the cross-coupling terms in the equation for the generalized displacement coordinate associated with mode n of body  $\lambda$  are

$$\vec{\omega}_\lambda \cdot \left[ \sum_{m,\lambda} F_{m,n,\lambda} \dot{a}_{m,\lambda}(t) \right] \cdot \vec{\omega}_\lambda - 2\vec{\omega}_\lambda \cdot \sum_{m,\lambda} \dot{a}_{m,\lambda}(t) \vec{K}_{m,n,\lambda}$$

More simply expressed, the SCXC array defines which modes m of body  $\lambda$  shall be used in evaluating the above summations.

In the input deck, the code word MODE CUP must follow FLEXIBLE. Upon recognition of the code word MODE CUP, the procedure for inputting coupled modal data starts. In columns 13 through 70 of the option card,

I = unused

NSET = unused



(S1 (I), I = 1, NBOD) = number of blocks of mode dependent cross coupling data to be read in for body I; a block of data consists of the three integers (M, N, K), the components of the tensor  $F_{M,N,K}$ , and the components of the vector  $\vec{K}_{M,N,K}$

The parameter SFCC is computed by an internal process which makes use of the above data provided on the option card.

To conserve computer storage and increase computational speed, all coupled mode-dependent data are sequentially numbered. Strict adherence to the following input logic is required.

```

MN = 0
KF = 0
DO 1 K = 1, NBOD
  DO 1 N = 1, SFXM(K)
    MN = MN + 1
    DO 1 M = 1, SFXM(K)
      IF [FM,N,K INSIGNIFICANT .AND.  $\vec{K}_{M,N,K}$  INSIGNIFICANT] GO TO 1
      READ M, N, K
      KF = KF + 1
      READ FM,N,K and  $\vec{K}_{M,N,K}$ 
      LOAD FM,N,K into FCF array, starting at location (1,1,KF)
      LOAD  $\vec{K}_{M,N,K}$  into FCK array, starting at location (1,KF)
      ADD integer M to code word SCXC(MN)
    1 CONTINUE

```

where

- MN — mode number for mode N of body K
- KF — storage location integer designator for modal cross-coupling parameters
- [FCF(II, JJ, KF)] — components of the resultant coupled mode-dependent tensor  $F_{M,N,K}$  relative to the body K fixed reference frame (if M = N, significant terms usually exist)
- [FCK(II, KF)] — components of the resultant coupled mode dependent vector  $\vec{K}_{M,N,K}$  relative to the body K fixed reference frame.

All coupled mode-dependent data required by N-BOD2 are read immediately after recognition of the code word MODE CUP on an option card. If flexible bodies are defined and the 'MODE CUP' option card and associated data are not in the option deck, all computation required to define modal cross-coupling effects is skipped.

The data required to describe the coupled modal data for each flexible body are read according to the following sequence of read statements:

```

READ STATEMENTS FOR OPTION 'MODE CUP' IN INOPT
C
C INITIALIZE COUPLED MODE LABELS
  KF = 0
C
  READ ALL NON-TRIVIAL COUPLED MODE PARAMETERS
C
  DO 1 K = 1, NBOD
C
C   S1(K) = NUMBER OF DATA BLOCKS FOR BODY K
C   SFXM(K) = NUMBER OF MODES USED FOR BODY K
  IF(SFXM(K).EQ.0) GO TO 1 (BODY K IS RIGID)
  IF(S1(K).NE.0.) GO TO 2 (SOME CROSS COUPLING FOR BODY K)

  GO TO 1
2 DO 1 M = 1, S1(K)
  READ 105 MB, NB, KB
  KF = KF + 1
  READ 104, (FCF(IJ,KF),IJ = 1, 3),II = 1, 3)
  READ 104, (FCK(II,KF),II = 1, 3)
1 CONTINUE
C
  ALL COUPLED MODAL DATA IN
C
C
C FORMAT STATEMENTS
104 FORMAT (3E15.5)
105 FORMAT (3I5)

```

**Default** Unless otherwise directed, the simulation model will consist only of rigid bodies, point masses, and symmetric wheels. All parameters associated with flexible bodies will be set equal to zero.

### PROGRAMMER'S GUIDE TO SUBROUTINE INOPT

The main function of subroutine INOPT is to accept the data that will define the parameters associated with the various computational options available to the user. It is structured to

grow. As new modeling requirements are defined and their associated equations developed, INOPT is the subroutine in which the basic data required to define the new option are to be entered. This is done by simply defining another code word that will be recognized along with the FORTRAN coding required to set up the basic data necessary to describe the new option.

Subroutine INOPT has the following structure:

- Define by default all parameters associated with available options.
- Read option card and compare option code with library of recognizable codes.
- Branch to appropriate program point, redefine option-related parameters, input any additional data required.
- Branch back to program point 12 to read another option card.
- Upon recognition of the 'END OPTIONS' code word, check all option data for basic incompatibilities.
- Perform any final setup work that must be done after all option data have been inputted.
- Print an echo of all option related data defined in INOPT.

The following variables, defined in table 1, are either first introduced or used extensively in subroutine INOPT. The programmer is referred to the comment cards inserted therein, and the following, for detailed definition.

CT1	—	counter. Zeroed before exit from INOPT
CT2	—	counter. Zeroed before exit from INOPT
CT3	—	counter. Zeroed before exit from INOPT
CT4	—	counter. Zeroed before exit from INOPT (See DYN)
CT5	—	counter. Zeroed before exit from INOPT
[FCF(I,J,K)]	—	See section "Data Input/INOPT," subsection 15
{FCK(I,K)}	—	See section "Data Input/INOPT," subsection 15
FG3	—	input error flag. If the code word on any option card is not recognized, it is set to .FALSE.
{FLA(I,MN)}	—	See section "Data Input/INOPT," subsection 15
{FLB(I,MN)}	—	See section "Data Input/INOPT," subsection 15
{FLC(I,MN)}	—	See section "Data Input/INOPT," subsection 15
[FLD(I,J,MN)]	—	See section "Data Input/INOPT," subsection 15

- [FLJ(I,J,MN)] — See section “Data Input/INOPT,” subsection 15
- FLOM(MN) — See section “Data Input/INOPT,” subsection 15
- INERF — See section “Data Input/INOPT,” subsection 3
- NCTC — See section “Data Input/INOPT,” subsection 12
- NFKC — See section “Data Input/INOPT,” subsection 12
- NFLXB — See section “Data Input/INOPT,” subsection 15
- NFRC — See section “Data Input/INOPT,” subsection 10
- NMOA — See section “Data Input/INOPT,” subsection 11
- NMODS — See section “Data Input/INOPT,” subsection 15
- NMV — See section “Data Input/INOPT,” subsection 13
- SC(I) — See section “Data Input/INOPT,” subsection 14
- SCXC(K) — See section “Data Input/INOPT,” subsection 15
- SCG — See section “Data Input/INOPT,” subsection 14
- SCN(I) — See section “Data Input/INOPT,” subsection 6
- SCR(I) — See section “Data Input/INOPT,” subsection 6
- SD — See section “Data Input/INOPT,” subsection 7
- SEU — See section “Data Input/INOPT,” subsection 7
- SFCC — See section “Data Input/INOPT,” subsection 15
- SFK(I) — See section “Data Input/INOPT,” subsection 12
- SFLX — See section “Data Input/INOPT,” subsection 15
- SFR(I) — See section “Data Input/INOPT,” subsection 10
- SFXM(I) — See section “Data Input/INOPT,” subsection 15
- SIX(I) — See section “Data Input/INOPT,” subsection 5
- SLK(I) — See section “Data Input/INOPT,” subsection 12
- SMA(I) — See section “Data Input/INOPT,” subsection 11
- SMAL — See section “Data Input/INOPT,” subsection 7
- SMC(K) — code word. Used to define the symmetric wheel labels associated with all wheels contained in the nest K-1; K = 1, 2, . . . , NBOD
- SMV — See section “Data Input/INOPT,” subsection 13

- SPI(I)            — See section "Data Input/INOPT," subsection 4
- SVD              — See section "Data Input/INOPT," subsection 8
- SXM(I,K)        — computed integer array. Used in subroutine TRAN if the transformation matrices for body K are to be computed using Euler-angle techniques. SXM(I,K) defines the coordinate axis ( $\pm 1, \pm 2, \pm 3$ ) in body K along which the free coordinate vector  $M+(I-1)$ ,  $I = 1, \dots, 3$ -PCON(K), is aligned in the zero state (sign implies direction). It is computed only if needed; otherwise it is set to zero.
- SXT              — See section "Data Input/INOPT," subsection 9
- TUG(I)           — See section "Data Input/INOPT," subsection 14
- ZETA(MN)        — See section "Data Input/INOPT," subsection 15

### DATA INPUT, SUBROUTINE INTOR

The third and last input data subroutine to be entered is INTOR. Normally this routine contains no executable instruction. It is for the user. The user may insert any desired input statements and store the resultant data in common block /SATELL/. The data inputted here usually consist of all data needed to define the parameters in the user-defined description of the forces and torques acting on the simulation model.

There are no new variables introduced in this subroutine other than those that are user defined.

### PROGRAMMER'S GUIDE TO SUBROUTINE TRNSIV

The primary purpose of the computation coded in subroutine TRNSIV is to compute initial values for all required transformation matrices. The user need never interface with this subroutine.

All required computation carried out in N-BOD2 is done relative to the frame of computation (see section "Data Input/INOPT," subsection 3). Consequently, transformation matrices which take vectors from body fixed coordinates to computing frame coordinates must be evaluated. TRNSIV computes the initial values of these matrices based upon the data inputted in subroutine INBS. Quaternion techniques are applied since gimbal axes need not be aligned with body fixed coordinate axes (see Reference 1, section "Initial Orientation of the N-Body System" and Appendix "Quaternion Techniques").

The following variables, defined in table 1, are first introduced in subroutine TRNSIV. All other variables used therein have been previously defined. The programmer is referred to the comment cards inserted in TRNSIV, and the following, for detailed definitions:

[XMC(I,J,K)]

for  $K = 0$

- transformation matrix used to transform vectors from inertially fixed coordinates to the frame of computation.

for  $K = 1, 2, \dots, \text{NBOD}$  — transformation matrix used to transform vectors from body  $K$  fixed coordinates to the frame of computation.

where

$$[\text{XMC}(I,J,K)] = \begin{bmatrix} \text{XMC}(1,1,K) & \text{XMC}(1,2,K) & \text{XMC}(1,3,K) \\ \text{XMC}(2,1,K) & \text{XMC}(2,2,K) & \text{XMC}(2,3,K) \\ \text{XMC}(3,1,K) & \text{XMC}(3,2,K) & \text{XMC}(3,3,K) \end{bmatrix}$$

For any particular problem, all computation carried out in TRNSIV can be outputted on the line printer by defining

LTRNSI = .TRUE.

on card 1 of the data deck. There is a one-for-one correspondence between the line printer output and the theoretical notation in Reference 1.

### PROGRAMMER'S GUIDE TO SUBROUTINE VDIV

The initial values for all body fixed vectors and tensors, relative to the frame of computation, are computed in subroutine VDIV. The user need never interface with this subroutine.

All vector-tensor operations required to define the system equations of motion are carried out in computing frame coordinates. N-BOD2 attempts to avoid redundant and trivial computation by transforming all body fixed vectors and tensors, used more than once, into computing frame coordinates at time zero. Thereafter, integer sets, computed in VDIV, are used to transform, at every integration step, only those vectors and tensors that have non-trivial time-varying components relative to computing frame coordinates.

The following variables, defined in table 1, are first introduced in subroutine VDIV. All other variables used therein have been previously defined. The programmer is referred to the comment cards inserted in VDIV, and the following, for detailed definitions:

$\{\text{CAC}(I,K)\}$   
for  $K = 1, 2, \dots, \text{NBOD}$  — components of the vector from hinge point  $K-1$  to the deformed center-of-mass position of body  $K$ , relative to the frame of computation (see Reference 2, equation 47).

$\{\text{CAO}(I,K)\}$   
for  $K = 1, 2, \dots, \text{NBOD}$  — components of the vector from hinge point  $K-1$  to the undeformed center-of-mass position of body  $K$ , relative to body- $K$  fixed coordinates.

$\{\text{CBC}(I,K)\}$   
for  $K = 0$  — unused. Reserved for the components of the vector from the inertial origin to the composite system center-of-mass, if needed.

- för  $K = 1, 2, \dots, \text{NBOD}$  — components of the vector from hinge point  $\text{JCON}(K)-1$  to hinge point  $K-1$ , relative to the frame of computation ( $\text{N-BOD2}$  assumes zero elastic deformation at hinge points).
- $\{\text{FLAC}(I, \text{MN})\}$  — components of the mode-dependent vector defined by equation 48 of Reference 2 relative to the frame of computation. It is used in equation 47 to define the contribution of mode  $M$  deformation of body  $K$  to the vector which defines the deformed center-of-mass location of body  $K$ , where mode number  $\text{MN}$  is defined as

$$\text{MN} = M + \sum_{I=1}^{K-1} \text{SFXM}(I)$$

$$K = 2, 3, \dots, \text{NBOD}$$

$$M = 1, 2, \dots, \text{SFXM}(K)$$

- $[\text{FLE}(I, \text{J}, \text{MN})]$  — components of the mode-dependent tensor defined by equation 52 of Reference 2 relative to body  $K$  fixed coordinates. It is used in equation 49 to define the contribution of mode  $M$  deformation of body  $K$  to the deformed state inertia tensor of body  $K$ , where  $\text{MN}$  is the mode number defined as above.
- $[\text{FLH}(I, \text{J}, \text{MN})]$  — components of the mode-dependent tensor defined by equation 68 of Reference 2 relative to body  $K$  fixed coordinates. It is used in equation 65 of Reference 2 to define a component of the generalized force acting on mode  $M$  of body  $K$ , where  $\text{MN}$  is the mode number defined as above.
- $\{\text{FLQ}(I, \text{MN})\}$  — components of the mode-dependent vector defined by equation 56 of Reference 2 relative to the body  $K$  fixed reference. It is used in equation 53 of Reference 2 to compute the contribution of mode  $M$  deformation rate to the angular momentum of body  $K$ , where  $\text{MN}$  is the mode number as defined above.
- $\{\text{FLQC}(I, \text{MN})\}$  — components of the mode-dependent vector defined by equation 56 of Reference 2 relative to the computing frame.

{GAM(I,KL)}	– unused in VDIV; however, all elements of the array GAM are set equal to zero in VDIV.
{HMC(I,M)} for M = 1, 2, . . . , MNO	– components of the unit vector directed along the spin axis of symmetric wheel M relative to the frame of computation (N-BOD2 assumes zero elastic deformation at wheel attachment points).
NSVP	– total number of locked coordinate vectors to be transformed to computing frame coordinates at every integration step.
NSVQ	– total number of free coordinate vectors to be transformed to computing frame coordinates at every integration step.
{QFC(I,M)} for M = 1, 2, . . . , NFER	– components of free coordinate vector M relative to the frame of computation.
{QLC(I,L)} for L = 1, 2, . . . , NLOR	– components of locked coordinate vector L relative to the frame of computation.
SOK(K) for K = 1, 2, . . . , NBOD	– code word used to define the body labels of those bodies on the topological path from hinge point 0 to the center-of-mass of body K.
SOK(NBI) for NBI = NBOD+1	– code word used to define the set of integers from 1 to NBOD+1 inclusive.
SSCN	– code word used to define the union of all labels contained in the sets of integers defined by the code words SCN(K); K = 0, 1, . . . , NBOD-1.
SSIX	– code word used to define the union of all labels contained in the sets of integers defined by the code words SIX(K); K = 0, 1, . . . , NBOD-1.
SVA	– code word used to define the set of center-of-mass vectors that have nontrivial time-varying components in the frame of computation.
SVB	– code word used to define the set of hinge-point vectors that have nontrivial time-varying components in the frame of computation.
SVI	– code word used to define the set of inertia tensors that have nontrivial time-varying components in the frame of computation.



SVM	— code word used to define the set of symmetric-wheel spin vectors that have nontrivial time-varying components in the frame of computation.
SVP(L) for L = 1, 2, . . . , NSVP	— the labels associated with the locked coordinate vectors that must be transformed at every integration step.
SVQ(M) for M = 1, 2, . . . , NSVQ	— the labels associated with the free coordinate vectors that must be transformed at every integration step.
[XDIC(I,J,IK)]	— unused in VDIV; however, all elements of the array XDIC are set equal to zero in VDIV.
[XIC(I,J,K)] for K = 1, 2, . . . , NBOD	— components of the inertia tensor of body K about its center-of-mass in the deformed state relative to the frame of computation (see Reference 2, equation 49).
[XIO(I,J,K)] for K = 1, 2, . . . , NBOD	— components of the inertia tensor of body K about its undeformed center-of-mass position in the undeformed state relative to the body K fixed reference.

For any particular problem, all computation carried out in VDIV can be outputted on the line printer by defining

LVDIV = .TRUE.

on card 1 of the data deck.

## PROGRAMMER'S GUIDE TO SUBROUTINE EQIV

The complete description of the motion of the coupled-body system is defined by the simultaneous solution of a set of first-order, nonlinear, ordinary differential equations. Subroutine EQIV is used to compute the total number of first-order equations to be integrated (excluding those that are user defined) and to set up an array of initial conditions in a form compatible with the integration subroutine RUNGE. The user need never interface with this subroutine.

The following variables, defined in table 1, are first introduced in subroutine EQIV. All other variables used therein have been previously defined. The programmer is referred to the comment cards inserted in EQIV, and the following, for detailed definitions:

NEQ	— total number of first-order differential equations that will be set up and numerically integrated via RUNGE (excluding those that are user defined in TORQUE).
-----	--

In the coding of the equations of motion, the system state variables and their associated time derivatives are stored in various arrays which are given different symbolic names.

To effectively use the integration subroutine, these must all be reloaded into two one-dimensional arrays so that the equations to be integrated take on the general form

$$\dot{y}_N(t) = f_N [\dot{y}_1(t), \dots, \dot{y}_{NEQ}(t), y_1(t), \dots, y_{NEQ}(t), t]$$

where

$$N = 1, 2, \dots, NEQ.$$

$Y(N)$  — one dimensional array in which all state variables that are determined by integration are stored for entry into the integration package. In EQIV, all initial values are defined

$$N = 1, 2, \dots, NEQ.$$

The following sequential ordering scheme is used to define the location of each system state variable in the array Y.

- Angular and linear rates about or along all of the free-coordinate vectors

$$THAD(N); N = 1, 2, \dots, NFER.$$

- Generalized elastic coordinate rates for all modes of vibration

$$THAD(N); N = NFER + 1, \dots, NFER + NMODS.$$

- Angular rates for all variable-speed wheels

$$THADW(M); M \text{ defined by code word SMV.}$$

- Displacements about or along selected free-coordinate vectors

$$THA(SFR(N)); N = 1, 2, \dots, NFRC.$$

- Generalized elastic coordinates, all of them

$$THA(N); N = NFER + 1, \dots, NFER + NMODS.$$

- Angular position of selected wheels

$$THAW(SMA(M)); M = 1, 2, \dots, NMOA.$$

- Elements of the first two columns of each transformation matrix that is obtained by direction-cosine technique

$$XMC(I,J,M); I = 1, 2, 3$$

$$J = 1, 2$$

$$M \text{ defined by code word SD.}$$

Refer to subroutine listing for exact details if needed. For any particular problem, all computation carried out in EQIV can be outputted on the line printer by defining

LEQIV = .TRUE.

on card 1 of the data deck.

## PROGRAMMER'S GUIDE TO SUBROUTINE TRAN

Numerical solution of the vector-dyadic equations of motion requires that the orientation of all body fixed-reference frames relative to each other and to the inertially fixed frame be obtainable. A minimal set of transformation matrices is the set that defines the relative orientation of each body fixed and inertially fixed reference frame to the frame of computation. These transformation matrices are evaluated in subroutine TRAN. The user need never interface with this subroutine.

As discussed in section "Data Input Subroutine INOPT," subsection 7, three kinematic methods for obtaining transformation matrices are available for the user. TRAN makes use of the data stored in the code words SD, SMAL, and SEU to determine the kinematic technique to be used for the computation of each respective transformation matrix.

The following variable, defined in table 1, is recomputed in subroutine TRAN. All other variables used have been previously defined. The programmer is referred to the comment cards inserted in TRAN, and the following, for detailed definitions:

[XMC(I,J,K)] — see section "Programmer's Guide to Subroutine TRNSIV." In subroutine TRAN, which is entered four times per integration step, the elements of all transformation matrices that have time-varying coefficients are recomputed.

If direction-cosine techniques are used, the first two columns of the transformation matrix [XMC(I,J,K)], namely

$\{XMC(I,1,K)\}$  and  $\{XMC(I,2,K)\}$

are computed by integration of direction-cosine rate equations. The third column

$\{XMC(I,3,K)\}$

is computed in TRAN by a simple vector cross-product operation. No attempt is made to ensure that the resultant matrix is orthonormal.

The author is of the opinion that since it generally is impossible to determine which of the six integrated direction cosine parameters is contributing most strongly to any error buildup, it is also impossible to define an orthonormalization procedure which would be satisfactory for all possible situations. If orthonormalization of transformation matrices must be maintained to a high degree of accuracy, then Euler angle techniques should be called for in INOPT.

If Euler angles are used, the appropriate free coordinate vectors must be aligned with body fixed coordinate axes in the nominal zero displacement state. Euler angle techniques are used to obtain only the transformation matrix between pairs of contiguous bodies. The transformation matrix to the computing frame is obtained by an appropriate matrix multiplication. The same logic is used for the small-angle technique with the exception that the approximation

$$\theta_M = \sin \theta_M$$

is employed.

For any particular problem, all computation carried out in TRAN can be outputted on the line printer by defining

LTRAN = .TRUE.

on card 1 of the data deck.

### PROGRAMMER'S GUIDE TO SUBROUTINE TRANVD

The primary purpose of TRANVD is to transform to computing frame coordinates all body fixed vectors and tensors, which are used more than once in the evaluation of the equations of motion. The user need never interface with this subroutine.

The following variables defined in table 1 are recomputed each time TRANVD is called by DYN:

{CAC (I,K)}	{HMC (I,M)}
{CBC (I,K)}	{QFC (I,M)}
{FLAC (I, MN)}	{QLC (I,L)}
{FLQC (I,MN)}	[XIC (I,J,K)]

Each of the above variables is defined in the section "Programmers Guide to Subroutine VDIV." In subroutine TRANVD, which is entered four times per integration step, the transformation matrices evaluated in TRAN are used to obtain the components of each of the above body fixed vectors and tensors in computing frame coordinates. Only those that have time-varying components in the computing frame are recomputed; all others retain the initial value that was set in VDIV. The components of the above variables are recomputed only if the corresponding components of the transformation matrix are recomputed.

For any particular problem, all computation carried out in TRANVD can be outputted on the line printer by defining

LTRANV = .TRUE.

on card 1 of the data deck.

## PROGRAMMER'S GUIDE TO SUBROUTINE RATE

The primary purpose of subroutine RATE is to compute all linear and angular velocity vectors that will be required by the succeeding subroutines. The user need never interface with this subroutine.

In the coding of N-BOD2, all hinge points of the system are assumed to be either on rigid bodies or at node points of flexible bodies. Consequently, all rate-related effects associated with the deformation dependent motion of hinge points are identically equal to zero. That is, in Reference 2, equation 126, all elements of the partition  $\varphi_H^R$ ,  $\varphi_W^R$ , and  $\varphi_H^T$  equal zero.

The following variables are defined in table 1 and recomputed each time subroutine RATE is called by DYN:

$$\begin{array}{ll} \{\text{ROMC (I,K)}\} & \{\text{COMC (I,K)}\} \\ \{\text{FOMC (I,K)}\} & \{\text{DOMC (I,K)}\} \end{array}$$

In Reference 1, equation 141, the partition that defines relative rate between contiguous bodies is

$$\{\tilde{\omega}\} = [q] \{\dot{\theta}\}$$

In Reference 2, equation 126, it is

$$\{\tilde{\omega}\} = \{^{\circ}\tilde{\omega}\} + [\varphi_H^R] \{\ddot{a}\}$$

where, by equation 133, Reference 2,

$$\{^{\circ}\tilde{\omega}\} = [q] \{\dot{\theta}\}$$

and under the restrictions of N-BOD2

$$[\varphi_H^R] = [0]$$

The components, relative to the computing frame, of the NB1 vectors contained in the column matrix  $\{\tilde{\omega}\}$  are stored in the array ROMC according to the following storage convention:

- Body K is a rigid or flexible body;  $K = 1, 2, \dots, \text{NBOD}$

$$\{\text{ROMC (I,K)}\} = \{\tilde{\omega}_K\}_c = \sum_{M@K-1} \dot{\theta}_M \{\vec{q}_M\}_c$$

- Body K is a point mass;  $K = 2, 3, \dots, \text{NBOD}$

$$\{\text{ROMC (I,K)}\} = \{\dot{\vec{\alpha}}_K\}_c = \sum_{M@K-1} \dot{\theta}_M \{\vec{q}_M\}_c$$

- $K = \text{NB1}$  implies linear velocity of the center-of-mass of body 1 relative to inertial origin

$$\{\text{ROMC (I,K)}\} = \{\dot{\vec{\beta}}_1\}_c = \sum_{M@K-1} \dot{\theta}_M \{\vec{q}_M\}_c$$

Note that by applying equation 134 of Reference 1, namely

$$\vec{\omega}_K = \sum_{M@K-1} \dot{\theta}_M \vec{q}_M$$

all trivial multiplications by zero called for by the matrix equation

$$\{\tilde{\omega}\} = [q] \{\dot{\theta}\}$$

are circumvented.

The relative angular-rate vectors stored in the array ROMC are used to compute the inertial angular-rate vectors; the components of which, relative to the computing frame, are stored in the array FOMC. Making use of equation 37 of Reference 1, the following parameters are stored in the array FOMC:

- Body K is a rigid or flexible body;  $K = 1, 2, \dots, \text{NBOD}$

$$\{\text{FOMC (I,K)}\} = \{\vec{\omega}_K\}_c = \sum_{i \in S_{0,K-1}} \{\vec{\omega}_i\}_c$$

- Body K is a point mass, since hinge point K-1 is at a node point on body JCON(K);  $K = 2, 3, \dots, \text{NBOD}$

$$\{\text{FOMC (I,K)}\} = \{\vec{\omega}_{J(K)}\}_c$$

- $K = \text{NB1}$

$$\{\text{FOMC (I,K)}\} = \{\text{ROMC (I,K)}\}$$

The elements of this array FOMC are used primarily to compute gyroscopic cross-coupling loads.

The user may choose the frame of computation to be either the inertially fixed-reference frame or the body 1 fixed-reference frame. In Reference 1, pages 34 and 35, the angular rate of the body K fixed-reference relative to the computing frame is defined. The components of these vectors, relative to the computing frame, are stored in the array COMC; that is,

- Body K is a rigid or flexible body;  $K = 1, 2, \dots, \text{NBOD}$

$$\{\text{COMC}(I,K)\} = \{ {}_c \vec{\omega}_K \}_c$$

- Body K is a point mass;  $K = 2, 3, \dots, \text{NBOD}$

$$\{\text{COMC}(I,K)\} = \{ {}_c \vec{\omega}_{J(K)} \}_c$$

- $K = \text{NB1}$

$$\{\text{COMC}(I,K)\} = \{\text{ROMC}(I,K)\}$$

The elements of this array COMC are used primarily in the evaluation of the direction cosine equations to be integrated.

In Reference 1, equation 136, differentiation of the relative angular-velocity vector yields

$$\dot{\vec{\omega}}_K = \sum_{M@K-1} [\ddot{\theta}_M \vec{q}_M + \dot{\theta}_M \dot{\vec{q}}_M]$$

The second term reflects the fact that free-coordinate vectors are not inertially fixed. This term carries through the development and appears in the final partitioned matrix form of the equations of motion: Reference 1, equation 144; or Reference 2, equation 136; as

$$[\dot{q}] \{\dot{\theta}\}$$

The elements of the column matrix are computed and stored in the array DOMC according to the following storage convention:

- Body K is a rigid or flexible body;  $K = 1, 2, \dots, \text{NBOD}$

$$\{\text{DOMC}(I,K)\} = \sum_{M@K-1} \dot{\theta}_M \{ \dot{\vec{q}}_M \}_c$$

- Body K is a point mass  $K = 2, 3, \dots, \text{NBOD}$  or  $K = \text{NB1}$ ;

$$\{\text{DOMC}(I,K)\} = \{0\}$$

The components of the vectors  $\dot{\vec{q}}_M$  are needed only for the above computation and are not stored in COMMON; furthermore, only those having nonzero components are evaluated.

For any particular problem, all computation carried out in RATE can be outputted on the line printer by defining

LRATE = .TRUE.

on card 1 of the data deck.

In the coding of subroutine RATE, it should be noted that by use of the EQUIVALENCE statement the single subscript arrays EFOMC, EROMC, ECOMC, and EDOMC, are used interchangeably with the double subscript arrays FOMC, ROMC, COMC, and DOMC, respectively. This is done to optimize computation speed.

### PROGRAMMER'S GUIDE TO SUBROUTINE XDY

The primary purpose of subroutine XDY is to compute all required hinge point to center-of-mass position vectors and all inertia and pseudo-inertia tensors. The user need never interface with this subroutine.

On the left-hand side of the system equations of motion given in Reference 1 by equation 144 and in Reference 2 by equation 136, the computation of numerous pseudo-inertia tensors is required. To compute these tensors, it is also required to compute the components of the position vectors which locate center-of-mass position relative to hinge points.

In TRANVD, the center-of-mass vector, hinge-point vector, and inertia tensor for each body of the system is computed. Once these vectors and tensors are determined, no distinction need be made between rigid and flexible bodies for the computation of inertia and pseudo-inertia tensors of nests of bodies. Consequently, equations 11 and 63 through 70 of Reference 1 may be applied to define the contents of the matrix of inertia tensors defined by the partition

[X]

in the equations of motion.

The following variables are defined in table 1 and computed in XDY:

{GAM (I,KL)}

[XDIC (I,J,KI)]

In Reference 1, equation 11, hinge point to center-of-mass position vectors are defined by the equation

$$\vec{\gamma}_{K-1,L} = \sum_{\substack{i \in S_{K-1,L-1} \\ i \neq K}} \vec{\beta}_i + \vec{\alpha}_L$$



where

$$K = 1, 2, \dots, \text{NBOD} \quad \text{and}$$

$$K \leq L$$

To conserve computer storage and eliminate a superfluous dimension, the integer function KTO, defined along with the other utility subroutines, is used to store the triangular matrix of vectors in a two-dimensional array. That is

$$\{ \text{GAM}(I, KL) \} = \{ \vec{\gamma}_{K-1, L} \}_c$$

where

$$KL = \text{KTO}(\text{NBI}, K-1, L)$$

In Reference 1, equations 63 through 70, the tensor in row K, column I,  $I \geq K$ , is defined by the equation that satisfies one of the following criteria:

1.  $K \in S_R, I \in S_{K-1}, I \in S_R$

$$x_{K,I} = \sum_{\substack{\lambda \in S_{I-1} \\ \lambda \notin S_L}} \Phi_\lambda + \sum_{\lambda \in S_{I-1}} G_{K-1, I-1}^\lambda$$

2.  $K \in S_R, I \in S_{K-1}, I \in S_L$

$$x_{K,I} = -m_I \Gamma_{K-1, I}$$

3.  $K \in S_R, I \notin S_{K-1}$

$$x_{K,I} = 0$$

4.  $K \in S_L, I = K$

$$x_{K,I} = m_I \mathbf{1}$$

$$5. \quad K \in S_L, I \notin S_{K-1}$$

$$X_{K,I} = 0$$

$$6. \quad K \in S_R, I = NB1$$

$$X_{K,I} = - \sum_{\lambda \in S_{K-1}} m_\lambda \Gamma_{K-1,\lambda}$$

$$7. \quad K \in S_L, I = NB1$$

$$X_{K,I} = m_K \mathbf{1}$$

$$8. \quad K = NB1, I = NB1$$

$$X_{K,I} = \sum_{\lambda \in S_0} m_\lambda \mathbf{1}$$

where

$$G_{K-1,I-1}^\lambda = m_\lambda \left[ (\vec{\gamma}_{I-1,\lambda} \cdot \vec{\gamma}_{K-1,\lambda}) \mathbf{1} - \vec{\gamma}_{I-1,\lambda} \vec{\gamma}_{K-1,\lambda} \right]$$

$$\Gamma_{K-1,\lambda} = \mathcal{P}(\vec{\gamma}_{K-1,\lambda})$$

and

$\mathbf{1}$  = unit dyad

To conserve computer storage and eliminate a superfluous dimension, the integer function KT1, defined along with the other utility subroutines, is used to store the symmetric matrix of tensors in a three-dimensional array. That is

$$[XDIC(I, J, KI)]_{i,j,c} = [X_{K,I}]_c$$

where

$$KI = KT1(NB1, K, I)$$

From symmetry

$$\left[ \mathbf{x}_{I,K} \right]_c = [\text{XDIC}(I, J, KI)]^T$$

where post superscript T implies "transpose."

On the first pass through XDY, all elements of the arrays XDIC and GAM are computed. Thereafter, only those elements of the arrays that have time-varying components are re-computed. To do this, the integer arrays, defined by the code words SPI, SVD, and SXT, are used.

For any particular problem, all computation carried out in XDY can be outputted on the line printer by defining

LXDY = .TRUE.

on card 1 of the data deck.

### PROGRAMMER'S GUIDE TO SUBROUTINE ETA

The primary purpose of subroutine ETA is to compute the forces and torques associated with the gyroscopic motion of each body and each nest of coupled bodies. The user need never interface with this subroutine.

On the right-hand side of equation 112 of Reference 2, the matrix

$$\{ \eta_1 \}$$

contains all gyroscopic forces and torques that enter into the coupled body equations of motion. The vector elements of this matrix are defined by the terms appearing in equations 107 and 108 of Reference 2. For the case of coupled rigid bodies, these terms reduce to those defined in Reference 1 by the partitions

$$\{ \eta^c \} \text{ and } \{ \eta^I \}$$

of equation 88.

Subroutine ETA is programmed to avoid redundant and trivial computations. The integer code words SCN, SCR, SIX, SMC, SFLX, SFXM, SSCN, and SSIX are used to pick out exactly which bodies contribute significantly in each step of the computation. Only non-zero time-varying contributions are computed.

The following variables are defined in table 1 and computed in ETA:

$\{ \text{ETIC}(I,K) \}$	$\{ \text{FLCRC}(I,K) \}$
$\{ \text{CNF}(I,K) \}$	$\{ \text{ETMC}(I,M) \}$
$\{ \text{FLIRC}(I,K) \}$	$\{ \text{ETC}(I,K) \}$

The gyroscopic torque acting on the nest K-1; which consists of at least one rigid body, is from Reference 2, equation 107

$$- \sum_{\lambda \in S_{K-1}} \left[ \dot{\Phi}_\lambda \cdot \vec{\omega}_\lambda + \vec{\omega}_\lambda \times \vec{L}_{\lambda,\lambda} + \vec{\gamma}_{K-1,\lambda} \times \vec{C}_\lambda \right]$$

where

$$\vec{L}_{\lambda,\lambda} = \Phi_\lambda \cdot \vec{\omega}_\lambda + \Delta \vec{L}_{\lambda,\lambda} + \sum_{\substack{M: \\ MO(M)=\lambda}} \vec{H}_M$$

$$\Delta \vec{L}_{\lambda,\lambda} = \sum_{N,\lambda} \vec{Q}_{N,\lambda} \dot{a}_{N,\lambda}(t)$$

$$\dot{\Phi}_\lambda = \sum_{N,\lambda} E_{N,\lambda} \dot{a}_{N,\lambda}(t)$$

$$\vec{C}_\lambda = m_\lambda \left[ \sum_{\substack{i \in S_{0,\lambda-1} \\ i \neq 1}} \left( \vec{\omega}_{J(i)} \times (\vec{\omega}_{J(i)} \times \vec{\beta}_i) + 2\vec{\omega}_{J(i)} \times \dot{\vec{\beta}}_i \right) + \vec{\omega}_\lambda \times (\vec{\omega}_\lambda \times \vec{\alpha}_\lambda) + 2\vec{\omega}_\lambda \times \dot{\vec{\alpha}}_\lambda \right]$$

(In N-BOD2 all  $\dot{\vec{\beta}}_i$  assumed zero)

The gyroscopic force acting on the nest L-1, which consists of the point mass defined as body L, is from Reference 2, equation 108

$$- \vec{C}_L$$

and from the same equation, the gyroscopic force acting on the composite system is

$$- \sum_{\lambda \in S_0} \vec{C}_\lambda$$

These resultant quantities are computed and stored in the array ETC, so that

$$\{ETC(I,K)\} = \text{components of the gyroscopic load vector, in computing frame coordinates, acting on the nest K-1; } K = 1, 2, \dots, \text{NBOD.}$$

$\{\text{ETC}(I, \text{NBI})\}$  = components of the gyroscopic force vector, in computing frame coordinates, acting on the composite system.

The following procedure is used to compute the elements of the array ETC:

1. For all bodies L contained in the set defined by the code word SSIX, compute

$$\{\text{ETIC}(I, L)\} = \{\vec{\omega}_L \times (\Phi_L \cdot \vec{\omega}_L)\}_c$$

2. For all bodies L contained in the set defined by the code word SSCN, compute

$$\{\text{CNF}(I, L)\} = \left\{ m_L \left[ \sum_{\substack{i \in S_{0,L-1} \\ i \neq 1}} \vec{\omega}_{J(i)} \times (\vec{\omega}_{J(i)} \times \vec{\beta}_i) + \vec{\omega}_L \times (\vec{\omega}_L \times \vec{\alpha}_L) \right] \right\}_c$$

3. For all flexible bodies L, that is, for those bodies contained in the set defined by the integer code word SFLX, compute

$$\begin{aligned} \{\text{FLIRC}(I, L)\} &= \left\{ \vec{\Phi}_L \cdot \vec{\omega}_L + \vec{\omega}_L \times \vec{\Delta}_{L,L} \right\}_c \\ &= \left\{ \left[ \sum_{N,L} (E_{N,L} + \mathcal{P}(\vec{Q}_{N,L})) \dot{a}_{N,L}(t) \right] \cdot \vec{\omega}_L \right\}_c \end{aligned}$$

$$\{\text{FLCRC}(I, L)\} = \left\{ 2m_L \vec{\omega}_L \times \dot{\vec{\alpha}}_L \right\}_c$$

$$= \left\{ 2m_L \vec{\omega}_L \times \sum \dot{a}_{N,L}(t) \vec{A}_{N,L} \right\}_c$$

4. Add gyroscopic loads associated with rigid body motion to that associated with relative elastic deformation to obtain

$$\{\text{ETIC}(I, L)\} = \{\text{ETIC}(I, L)\} + \{\text{FLIRC}(I, L)\}$$

$$\{\text{CNF}(I, L)\} = \{\text{CNF}(I, L)\} + \{\text{FLCRC}(I, L)\}$$

for each flexible body L.

5. For each symmetric wheel of the system, compute

$$\{\text{ETMC}(I, M)\} = \left\{ \vec{\omega}_{\text{MO}(M)} \times \vec{H}_M \right\}_c$$

6. Sum the gyroscopic loads associated with each body L contained in the nest K-1 to obtain the components of the gyroscopic load external to the nest K-1;  $K = 1, 2, \dots, \text{NBOD}+1$ , as follows:

a. Initialize the array ETC:

$$\{\text{ETC}(I, K)\} = \{0\}; \quad K = 1, 2, \dots, \text{NB1}$$

b. Add inertia cross-coupling terms for all bodies L contained in the set defined by the code word SIX(K-1):

$$\{\text{ETC}(I, K)\} = \{\text{ETC}(I, K)\} - \sum_{L \in \text{SIX}(K-1)} \{\text{ETIC}(I, L)\}$$

where

$$K = 1, 2, \dots, \text{NBOD}$$

c. Add centripetal cross-coupling terms for all bodies L contained in the set defined by the code word SCN(K-1):

- If body L is a rigid or flexible body

$$\{\text{CNF}(I, L)\} = \left\{ \vec{C}_L \right\}_c$$

- If body L is a point mass

$$\{\text{CNF}(I, L)\} = \left\{ \vec{C}_L - 2m_L \vec{\omega}_L \times \vec{\alpha}_L \right\}_c$$

Let

$$\{\text{CNF}(I, L)\} = \left\{ \vec{C}_L^* \right\}_c$$

then, for all nests K-1 containing one or more rigid bodies

$$\{\text{ETC}(I, K)\} = \{\text{ETC}(I, K)\} - \sum_{L \in \text{SCN}(K-1)} \left\{ \vec{\gamma}_{K-1, L} \times \vec{C}_L^* \right\}_c$$

and for all nests K-1 containing the single-point mass labeled body K

$$\{\text{ETC}(\text{I}, \text{K})\} = \{\text{ETC}(\text{I}, \text{K})\} - \left\{ \vec{\gamma}_{\text{K-1}, \text{K}} \times \vec{c}_{\text{K}}^* \right\}_c$$

and for K = NB1

$$\{\text{ETC}(\text{I}, \text{K})\} = \{\text{ETC}(\text{I}, \text{K})\} - \sum_{\text{L}=1}^{\text{NBOD}} \left\{ \vec{c}_{\text{L}}^* \right\}_c$$

- d. Add Coriolis cross-coupling terms for all point masses (bodies L) contained in the set defined by the code word SCR(K-1):

For all nests K-1 containing one or more rigid bodies

$$\{\text{ETC}(\text{I}, \text{K})\} = \{\text{ETC}(\text{I}, \text{K})\} - \sum_{\text{L} \in \text{SCR}(\text{K-1})} \left\{ \vec{\gamma}_{\text{K-1}, \text{L}} \times 2m_{\text{L}} \vec{\omega}_{\text{L}} \times \overset{\circ}{\vec{\alpha}}_{\text{L}} \right\}_c$$

For all nests K-1 containing the single-point mass labeled body K

$$\{\text{ETC}(\text{I}, \text{K})\} = \{\text{ETC}(\text{I}, \text{K})\} - \left\{ 2m_{\text{K}} \vec{\omega}_{\text{K}} \times \overset{\circ}{\vec{\alpha}}_{\text{K}} \right\}_c$$

and for K = NB1

$$\{\text{ETC}(\text{I}, \text{K})\} = \{\text{ETC}(\text{I}, \text{K})\} - \sum_{\text{L} \in \text{SL}} \left\{ 2m_{\text{L}} \vec{\omega}_{\text{L}} \times \overset{\circ}{\vec{\alpha}}_{\text{L}} \right\}_c$$

where SL is the code word defining all point-mass body labels.

- e. Add symmetric wheel cross-coupling terms for all wheels M contained in the set defined by the code word SMC(K-1):

$$\{\text{ETC}(\text{I}, \text{K})\} = \{\text{ETC}(\text{I}, \text{K})\} - \sum_{\text{M} \in \text{SMC}(\text{K-1})} \{\text{ETMC}(\text{I}, \text{M})\}$$

where

$$\text{K} = 1, 2, \dots, \text{NBOD}$$

For any particular problem, all computation carried out in ETA can be outputted on the line printer by defining

$$\text{LETA} = \text{.TRUE.}$$

on card 1 of the data deck.

## PROGRAMMER'S GUIDE TO SUBROUTINE TORQUE

The primary purpose of subroutine TORQUE is to provide a place for the user to define the nongyroscopic forces and torques acting on the particular problem under investigation. In Reference 1, pages 41 through 53, several nongyroscopic effects commonly encountered in the analysis of spacecraft attitude dynamics are described along with the methods that must be used to enter their resultant effects into the equations of motion. This section will define the coding (by example) that is required to interface with N-BOD2.

The following variables are defined in table 1 and computed by user defined algorithms in TORQUE:

$\{\text{PHI}(I,K)\}$   
 $\{\text{CLM}(M)\}$   
NTQ  
Y(N) N = NEQ+1, ..., NEQ+NTQ  
YD(N) N = NEQ+1, ..., NEQ+NTQ

In subroutine TORQUE, the user has access to all computed system state variables. These are stored in the various labeled common blocks. Any user-defined function of state variables may be computed and stored in array DUMMY in labeled common block /SATELL/.

The nested body approach has been used to derive the equations of motion presented in References 1 and 2. In Reference 1, equation 88, the forces and torques acting external to the nests are contained in the partitions

$$\left\{ \phi^{H_1} \right\} + \left\{ \phi^{E_1} \right\}$$

while torques acting external to symmetric wheels are contained in the partition

$$\left\{ \phi^{H_2} \right\}$$

In Reference 2, equation 111, the forces and torques acting external to the nests are contained in the partition

$$\left\{ \phi^{(e)} \right\}$$

while torques acting external to symmetric wheels are contained in the partition

$$\{\text{CL}\}$$



The user must define the algorithm that is to be used to compute

- $\{ \text{PHI}(I,K) \}$  — resultant torque (force\*) that is acting on and external to the nest K-1;  $K = 1, 2, \dots, \text{NBOD}$
- $\{ \text{PHI}(I,\text{NBI}) \}$  — resultant sum of all external forces acting on the composite system of coupled bodies
- $\text{CLM}(M)$  — scalar magnitude of the torque acting external to the symmetric wheel M

Frequently the algorithm involves the integration of differential equations such as in the description of an onboard control system. The user must define

- $\text{NTQ}$  — total number of differential equations that are defined by the user in TORQUE and which must be integrated.

The state variables and associated time derivatives for all first order differential equations defined in TORQUE are stored in the arrays Y and YD.

Let

$$N = \text{NEQ} + M$$

where

$$M = 1, 2, \dots, \text{NTQ}$$

then:

$Y(N)$  = storage location to the state variables associated with the *Mth* differential equation defined by the user in TORQUE.

$YD(N)$  = storage location of the time derivative of the state variable stored in  $Y(N)$ .

To code subroutine TORQUE, the user should first zero out the external torque matrix upon entry. The resultant torque matrix is then obtained by vector addition of the various components. Thus, the first executable statements of TORQUE should be

```
C  ZERO ALL ELEMENTS OF EXTERNAL TORQUE MATRIX
DO 1  K = 1, NB1
DO 1  I = 1, 3
1  PHI (I,K) = 0.D0
DO 2  M = 1, NMO
2  CLM(M) = 0.D0
```

---

\*Force if body K in nest K-1 is a point mass.

Motion about or along any free coordinate vector may be constrained by springs, dashpots, or motors. The computation of the associated reaction loads and their inclusion into the equations of motion are discussed in Reference 1, pages 42 through 45. The following coding is to be used to include these effects in N-BOD2:

```

C          REACTION TORQUE ACTING ACROSS OR ALONG GIMBAL AXIS M
C          AT HINGE POINT K-1 DUE TO :
C          LINEAR SPRINGS
C          LINEAR VISCOUS DAMPERS
C          MOTORS
C
C      LET:
C      SPR(M) = SPRING CONSTANT ABOUT OR ALONG GIMBAL AXIS M
C              (M*L**2/T**2 OR M/T**2)
C      DPC(M) = DAMPING CONSTANT ABOUT OR ALONG GIMBAL AXIS M
C              (M*L**2/T OR M/T)
C      CLT(M) = CONTROL TORQUE APPLIED BY MOTOR ABOUT OR ALONG GIMBAL
C              AXIS M M*L**2/T**2 OR M*L/T**2
C
C          DIMENSION TEM(3)
C          SPRING TORQUE
C          SPR(M) = USER INPUT
C          A = SPR(M)*THA(M)
C          CALL SCLV(A, QFC(1,M), TEM)
C          CALL VECSUB(PHI(1,K), TEM, PHI(1,K))
C          DAMPER TORQUE
C          DPC(M) = USER INPUT
C          A = DPC(M)*THAD(M)
C          CALL SCLV(A, QFC(1,M), TEM)
C          CALL VECSUB(PHI(1,K), TEM, PHI(1,K))
C          MOTOR TORQUE
C          CLT(M) = FUNCTION OF STATE VARIABLES, USER DEF.
C          CALL SCLV(CLT(M), QFC(1,M), TEM)
C          CALL VECADD(PHI(1,K), TEM, PHI(1,K))

```

Symmetric wheels are usually used to simulate controlled momentum wheels. Since wheels have only one degree of freedom, it is not necessary to compute wheel torque as a vector quantity. The following coding is to be used to include this effect in N-BOD2:

```

C
C
C          REACTION TORQUES ON SYSTEM DUE TO A CONTROL TORQUE
C          APPLIED TO MOMENTUM WHEEL MW
C
C      LET:
C      CLM(MW) = CONTROL TORQUE APPLIED TO WHEEL MW ABOUT ITS SPIN AXIS
C              USER DEFINED FUNCTION OF STATE VARIABLES (M*L**2/T**2)
C
C          CLM(MW) = USER DEFINED FUNCTION OF STATE VARB.

```

Locally applied forces may be applied to any body of the system; they may not be applied to symmetric wheels directly. The computation of the reaction loads and their inclusion into the equations of motion are discussed in Reference 1, page 45. The following coding is used to include this effect in N-BOD2:

```

C
C          REACTION TORQUES ON SYSTEM DUE TO A LOCALLY
C          APPLIED EXTERNAL FORCE (I.E. A GAS JET)
C
C      LET:
C          J = INTEGER LABEL ASSIGNED TO GAS JET
C          L = BODY TO WHICH EXTERNAL FORCE DIRECTLY APPLIED
C          RJ(I) = RADIUS VECTOR FROM CENTER OF MASS OF BODY L TO
C              GAS JET J, (COMPONENTS RELATIVE TO BODY L COORDINATES)
C          FJ(I) = COMPONENTS OF APPLIED FORCE DUE TO GAS JET J, (RELATIVE
C              TO BODY L COORDINATES) USER DEFINED FUNCTION OF STATE
C              VARIABLES (M*L/T**2)

```

```

C      DIMENSION RJC(3),FJC(3),TEM(3),TEM1(3),RJ(3),FJ(3)
C      INTEGER S1(10),NS1
C      LOGICAL CTAIN
C      C      RJ(1) = USER INPUT
C      CALL VECTRNR(RJ,XMC(1,1,L),RJC)
C      C      FJ(1) = USER DEFINED FUNCTION OF STATE VARIABLES
C      CALL VECTRNR(FJ,XMC(1,1,L),FJC)
C      CALL VECADD(PHI(1,NB1),FJC,PHI(1,NB1))
C      DO 3 K=1,NBOD
C      CALL UNPAC(S1,NS1,SK(K-1))
C      IF(.NOT.CTAIN(L,S1,NS1)) GO TO 3
C      IF(RBLO(K)) GO TO 4
C      CALL VECADD(PHI(1,K),FJC,PHI(1,K))
C      GO TO 3
C      4 KL = KTO(NB1,K-1,L)
C      CALL VECADD(GAM(1,KL),RJC,TEM)
C      CALL VECROS (TEM,FJC,TEM1)
C      CALL VECADD(PHI(1,K),TEM1,PHI(1,K))
C      3 CONTINUE
C
C
C

```

The system to be modeled may be in a gravitational field; i.e., an Earth-based system. The computation of the reaction loads and their inclusion into the equations of motion are discussed in Reference 1, pages 45 and 46. The following coding is used to include this effect in N-BOD2:

```

C
C
C      REACTION TORQUES ON SYSTEM DUE TO GRAVITATIONAL
C      EFFECTS ON AN EARTH BASED SYSTEM
C
C      LET:
C      GRAV = ACCELERATION OF GRAVITY (L/T**2)
C      RH(1) = COMPONENTS OF UNIT VECTOR FROM INERTIAL ORIGIN TO COMP.
C      SYSTEM CENTER OF MASS, (RELATIVE TO INERTIAL FRAME)
C      THAT IS, PARALLEL TO DIRECTION OF GRAVITY FORCE
C
C      INTEGER S1(10),NS1
C      DIMENSION TEM(3),BHC(3),RH(3)
C      C      RH(1) = USER INPUT
C      CALL VECTRNR(RH,XMC(1,1,0),BHC)
C      DO 4 K=1,NBOD
C      C      GRAV = USER INPUT
C      A = XMAS(K)*GRAV
C      CALL SCLV(A,BHC,TEM)
C      CALL VECSUB(PHI(1,NB1),TEM,PHI(1,NB1))
C      IF(RBLO(K)) GO TO 5
C      CALL VECSUB(PHI(1,K),TEM,PHI(1,K))
C      GO TO 4
C      5 CALL UNPAC(S1,NS1,SK(K-1))
C      DO 4 LL=1,NS1
C      L = S1(LL)
C      KL = KTO(NB1,K-1,L)
C      A = XMAS(L)*GRAV
C      CALL SCLV(A,BHC,TEM)
C      CALL VECROS (GAM(1,KL),TEM,TEM1)
C      CALL VECSUB(PHI(1,K),TEM1,PHI(1,K))
C      4 CONTINUE
C
C
C

```

The system to be modeled may be in orbit and subject to gravity gradient effects. The computation of the orbit and the gravity gradient torques are discussed in Reference 1, pages 46 through 52. The following coding is used to include these effects in N-BOD2:

```

C
C
C      KEPLERIAN ORBIT
C
C      LET:
C      CB(1,0) = COMPONENTS OF VECTOR FROM EARTH'S CENTER TO COMPOSITE
C      SYSTEM CENTER OF MASS, RELATIVE TO INERTIAL REFERENCE
C      FRAME, ORBIT ASSUMED TO BE IN INERTIAL 2-3 PLANE
C      ASM = SEMI-MAJOR AXIS OF ELLIPTIC ORBIT INERTIAL 2 DIRECTION
C      ECC = ORBIT ECCENTRICITY
C      TPP = TIME OF PERIHELION PASSAGE, (T)
C      GEV = EARTH'S GRAVITATIONAL CONSTANT, (L**3/T**2)
C      ETE = MEAN MOTION
C      AME = MEAN ANOMALY

```

```

C      ECE = ECCENTRIC ANOMALY
C      TVE = TRUE ANOMALY
C      BTO = MAGNITUDE OF CB(1,0), (L)
C
C
C      ASM = USER INPUT
C      GEV = USER INPUT
C      ETE = 1./SORT(ASM**3/GEV)
C      TPP = USER INPUT
C      AME = ETE*(T-TPP)
C      SM1 = SIN(AME) ; SM4 = SIN(4*AME)
C      SM2 = SIN(2*AME)
C      SM3 = SIN(3*AME)
C      ECC = USER INPUT
C      ECE = AME + ECC*SM1 + ECC**2*SM2/2
C      * +ECC**3*(9*SM3 - 3*SM1)/(24)
C      * +ECC**4*(64*SM4 - 32*SM2)/192
C      CE = COS(ECE)
C      SE = SIN(ECE)
C      BTO = ASM*(1 - ECC*CE)
C      CB(1,0) = 0
C      CB(2,0) = BTO*((CE - ECC)/(1 - ECC*CE))
C      CB(3,0) = BTO*(SORT(1-ECC**2)*SE/(1-ECC*CE))
C      CALL VECTRNCB(1,0),XMC(1,1,0),CBC(1,0)

```

REACTION TORQUES ON ORBITING SYSTEM DUE TO GRAVITY GRADIENT EFFECTS

```

C      LET:
C      CBC(I,1) = COMPONENTS OF VECTOR FROM COMPOSITE SYSTEM CENTER OF
C      MASS TO CENTER OF MASS OF BODY I, RELATIVE TO
C      COMPUTING FRAME
C      * NOTE FOR GRAVITY GRADIENT OPTION
C      * CB(I,1) AND ITS INERTIAL DERIVATIVE
C      * ARE REDEFINED TO CIRCUMVENT DIFFERENCE
C      * OF LARGE NUMBER PROBLEMS, THAT IS
C      * THEY ARE MEASURED FROM COMPOSITE CM TO
C      * CM OF BODY I RATHER THAN FROM INERTIAL
C      * ORIGIN TO CM OF BODY I
C      BH(I) = UNIT VECTOR FROM EARTH'S CENTER TO SYSTEM COMPOSITE
C      CENTER OF MASS, COMPONENTS RELATIVE TO INERTIAL FRAME
C      DEL(I,K) = COMPONENTS OF VECTOR FROM COMPOSITE SYSTEM CENTER OF
C      MASS TO CENTER OF MASS OF BODY K
C      DFG(I,K) = COMPONENTS OF GRAVITY GRADIENT FORCE ACTING ON BODY K
C      SGG(I) = COMPACTED INTEGER WORD, THOSE BODIES IN THE NEXT I
C      WHICH SIGNIFICANTLY CONTRIBUTE TO GRAVITY GRADIENT EFF.
C      BTO = DISTANCE FROM EARTH'S CENTER TO COMPOSITE SYSTEM CM
C
C      DIMENSION DEL(3,10),DFG(3,10),BHC(3),BH(3),TEM(3),TEM1(3)
C      INTEGER SGG(0,9), S1(10)
C      KEPLERIAN ORBIT MUST BE USED WITH GRAVITY GRADIENT OPTION
C      DO 10 I=1,3
C      10 BHC(I) = CBC(I,0)/BTO
C      DO 7 K=1,NBOD
C      KL = KTO(NB1,0,K)
C      CALL VECADD(CBC(1,1),GAM(1,KL),DEL(1,K))
C      CALL VECDDT(BHC,DEL(1,K),A)
C      A = 3*A
C      CALL SCLV(A,BHC,TEM)
C      CALL VECSUB(DEL(1,K),TEM,TEM)
C      A = -GEV*XMAS(K)/BTO**3
C      CALL SCLV(A,TEM,DFG(1,K))
C      7 CONTINUE
C      DO 8 K=1,NBOD
C      IF(RBLO(K)) GO TO 9
C      CALL VECADD(PHI(1,K),DFG(1,K),PHI(1,K))
C      GO TO 8
C      SGG(I) = SK(I) IF ALL BODIES CONTRIBUTE TO GRAVITY GRAD.
C      EFFECTS. IF NOT USE COMPAC TO CONSTRUCT
C      SGG(I) FORM USER INPUT OR DEFINE DIRECTLY
C      9 CALL UNPAC(S1,NS1,SGG(K-I))
C      DO 8 LL=1,NS1
C      L = S1(LL)
C      KL = KTO(NB1,K-1,L)
C      CALL VECROS(GAM(1,KL),DFG(1,L),TEM)
C      CALL VXDYOV(BHC,XIC(1,1,L),TEM1)
C      A = 3*GEV/BTO**3
C      CALL SCLV(A,TEM1,TEM1)
C      CALL VECADD(TEM,TEM1,TEM)
C      CALL VECADD(PHI(1,K),TEM,PHI(1,K))
C      8 CONTINUE

```

As previously mentioned, it is often necessary to integrate several differential equations to define control torques in TORQUE. The following coding is used to include additional differential equations for integration by N-BOD2 along with the system equations:

```

C
C
C          PARAMETERS DEFINED BY FIRST ORDER
C          DIFFERENTIAL EQUATIONS
C
C      LET:
C          NTO = TOTAL NUMBER OF FIRST ORDER DIFFERENTIAL EQUATIONS TO
C              BE SOLVED FOR USE IN SUBROUTINE TORQUE
C          TQ(N) = MAGNITUDE OF PARAMETER NUMBER N DEFINED WITHIN SUB.
C              TORQUE AT TIME T
C          TOD(N) = TIME DERIVATIVE OF PARAMETER TQ(N). A USER DEFINED
C              FUNCTION OF THE SYSTEM'S STATE VARIABLES
C
C              DIMENSION TQ(20),TOD(20)
C              FOR THE PARAMETER N
C              IF(CT4.NE.1) GO TO 11
C              Y(NEQ+N) = TQ(N) = INITIAL VALUE FOR TQ(N)
C          11 TQ(N) = Y(NEQ+N)
C              TOD(N) = USER DEFINED FUNCTION OF STATE VARB.
C
C          AFTER DEFINITION OF LAST DIFFERENTIAL EQUATION
C          NTO = TOTAL NUMBER OF FIRST ORDER DIFFERENTIAL
C              EQUATIONS TO BE SOLVED FOR USE IN TORQUE
C          DO 12 N=1,NTO
C          12 YD(NEQ+N) = TOD(N)

```

At times, appendages on spacecraft are subject to thermally induced motion. This is discussed in Reference 1, pages 52 and 53. The following coding is used to include this effect in N-BOD2:

```

C
C
C          THERMALLY INDUCED MOTION ABOUT GIMBAL AXIS M
C          AT HINGE POINT K-1
C
C      ASSUME:
C          ALL THERMALLY INDUCED DEFLECTION IS SMALL ANGLE
C          RELATIVE TO THE SYSTEM'S NOMINAL ZERO STRESS STATE
C          THERMALLY INDUCED DEFLECTION IS MODELLED AS A MOVEMENT
C          OF THE ZERO STRESS STATE
C          ACROSS ALL HINGE POINTS SUBJECT TO THERMAL DEFORMATION
C          SPRINGS AND DAMPERS ACT
C          A REASONABLE MODEL OF THE THERMAL INPUT CAN BE DEFINED
C          IN TERMS OF THE SYSTEM'S STATE VARIABLES
C          THERMAL EQUILIBRIUM POSITION ABOUT ANY GIMBAL AXIS IS DEF.
C          BY SOLUTION OF THE HEAT CONDUCTION EQUATION
C
C      LET:
C          SPR(M) = SPRING CONSTANT ACROSS GIMBAL AXIS M
C          DPC(M) = DAMPING CONSTANT ACROSS GIMBAL AXIS M
C          TAU(M) = THERMAL TIME CONSTANT FOR DEFORMATION ABOUT GIMBAL
C              AXIS M, (T)
C          TQ(N) = THERMAL EQUILIBRIUM POSITION FOR THERMAL DEFORMATION
C              ABOUT GIMBAL AXIS M, (RAD)
C          TOD(N) = RATE OF CHANGE OF THERMAL EQUILIBRIUM POSITION ABOUT
C              GIMBAL AXIS M, FIRST ORDER DIFF. EQ., (RAD/T)
C          TINP = THERMAL INPUT USER DEFINED FUNCTION OF STATE VARIABLES,
C              (RAD/T)
C
C          DIMENSION TEM(3)
C          N = USER DEFINED LABEL, DEPENDS UPON EQUATION NUMBERING
C              SEQUENCE DEFINED WITHIN SUBROUTINE TORQUE
C          IF(CT4.NE.1) GO TO 13
C          Y(NEQ+N) = TQ(N) = INITIAL VALUE FOR THERMAL DEFORMATION
C              ABOUT GIMBAL AXIS M, USER INPUT
C          13 TQ(N) = Y(NEQ+N)
C          TINP = USER DEFINED THERMAL INPUT FOR THERMAL DEFORMATION
C              ABOUT GIMBAL AXIS M
C          TAU(M) = USER INPUT
C          TOD(N) = -TQ(N)/TAU(M) + TINP
C          A = SPR(M)*(THA(M) - TQ(N))
C          CALL SCLV(A,OF(1,M),TEM)
C          CALL VECSUB(PHI(1,K),TEM,PHI(1,K))
C          A = DPC(M)*THAD(M)
C          CALL SCLV(A,OF(1,M),TEM)
C          CALL VECSUB(PHI(1,K),TEM,PHI(1,K))

```

For any particular problem, all computation carried out in TORQUE can be outputted on the line printer by defining

L TORQU = .TRUE.

on card 1 of the data deck, if the user codes the print statements.

### PROGRAMMER'S GUIDE TO SUBROUTINE QFDOT

The primary purpose of subroutine QFDOT is to eliminate the forces and torques associated with kinematic constraints. In so doing, the coupled vector-dyadic equations of motion will reduce to a set of simultaneous scalar differential equations. The user need never interface with this subroutine.

The equations of motion to be integrated are given in Reference 2 by equation 136. This equation, in the absence of body flexibility, reduces to equation 144 of Reference 1. Making use of the array names used in the coding of N-BOD2, the equation takes on the general form

$$[XMN] \{THADD\} = \{ETM\}$$

where, to be consistent with the coding of N-BOD2 and the notation of Reference 2,

$$\{THADD\} = \left\{ \begin{array}{c} \ddot{\theta} \\ \ddot{a} \\ \ddot{\theta}_w \end{array} \right\}$$

This matrix equation defines exactly

$$NFER + NMODS + NMV$$

scalar second-order ordinary differential equations.

The following variables are defined in table 1 and computed in QFDOT:

[XMN (M,N)]

{ETM (M)}

The elements of these arrays are obtained by carrying out the vector-dyadic operations defined by the partitioned matrix equation 136 of Reference 2.

The equations coded in N-BOD2 assume that all hinge points and symmetric wheel attachment points are either on rigid bodies or at node points of flexible bodies. The restriction in modeling capability implies that in equation 136 of Reference 2, the following is true:

$$\varphi_H^R = 0,$$

$$\varphi_W^R = 0, \text{ and}$$

$$\varphi_H^T = 0$$

Furthermore, it is assumed in N-BOD2 that the user will not need to compute forces of constraint; that is, the elements of the partition  $F^c$ .

Introduction of these two modeling limitations significantly simplifies the required coding. The coding is structured so that, if required at a later date, both limitations can be removed by simple insertions into QFDOT and other relevant subroutines. Making use of the above assumptions, the equations coded in N-BOD2 are from equation 136 of Reference 2.

$$\begin{bmatrix} q^T & 0 & 0 \\ 0 & h^T & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X & I^c & F \\ I^{cT} & I^s & 0 \\ F^T & 0 & M^1 \end{bmatrix} \cdot \begin{bmatrix} q & 0 & 0 \\ 0 & h & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{Bmatrix} \ddot{\theta} \\ \ddot{\theta}_w \\ \ddot{a} \end{Bmatrix} =$$

$$\begin{bmatrix} q^T & 0 & 0 \\ 0 & h^T & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \left\{ \begin{bmatrix} X & I^c & F \\ I^{cT} & I^s & 0 \\ F^T & 0 & M^1 \end{bmatrix} \cdot \begin{bmatrix} q & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{Bmatrix} \dot{\theta} \\ \dot{\theta}_w \\ \dot{a} \end{Bmatrix} + \begin{Bmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \end{Bmatrix} + \begin{Bmatrix} \phi^{(e)} \\ CL \\ 0 \end{Bmatrix} \right\}$$

Coding requirements force a rearrangement of rows and columns from that used in the theoretical development. The following storage allocation scheme is used:

- $[XMN (M, N)] = [q^T] \cdot [X] \cdot [q]$

where  $M = 1, \dots, \text{NFER}$

$N = 1, \dots, \text{NFER}$

- $[XMN (M, N)] = [q^T] \cdot [F]$

where

$$M = 1, \dots, NFER$$

$$N = NFER+1, \dots, NFER+NMODS$$

- $[XMN (M, N)] = [q^T] \cdot [I^c] \cdot [h]$

where

$$M = 1, \dots, NFER$$

$$N = NFER+NMODS+1, \dots, NFER+NMODS+NMV$$

- $[XMN (M, N)] = [M^1]$

where

$$M = NFER+1, \dots, NFER+NMODS$$

$$N = NFER+1, \dots, NFER+NMODS$$

- $[XMN (M, N)] = [0]$

where

$$M = NFER+1, \dots, NFER+NMODS$$

$$N = NFER+NMODS+1, \dots, NFER+NMODS+NMV$$

- $[XMN (M, N)] = [h^T] [I^s] [h]$

where

$$M = NFER+NMODS+1, \dots, NFER+NMODS+NMV$$

$$N = NFER+NMODS+1, \dots, NFER+NMODS+NMV$$

- $\{ETM (M)\} = [q^T] \cdot \left\{ \{\eta_1\} + \{\phi^{(e)}\} - [X] \cdot [\dot{q}] \{\dot{\theta}\} \right\}$

where

$$M = 1, 2, \dots, NFER$$



$$\bullet \quad \{\text{ETM (M)}\} = \{\eta_3\} - [F^T][\dot{q}]\{\dot{\theta}\}$$

where

$$M = \text{NFER}+1, \dots, \text{NFER} + \text{NMODS}$$

$$\bullet \quad \{\text{ETM (M)}\} = [h^T] \cdot \left\{ \{\eta_2\} + \{\text{CL}\} - [c^T][\dot{q}]\{\dot{\theta}\} \right\}$$

where

$$M = \text{NFER} + \text{NMODS} + 1, \dots, \text{NFER} + \text{NMODS} + \text{NMV}$$

In the computation of the ETM array, the contents of the ETC array defined in subroutine ETA are changed. In ETA, the components of the vector in row K of  $\{\eta_1\}$  are stored in  $\{\text{ETC (I,K)}\}$ . In QFDOT, the components of the vector in row K of

$$\{\eta_1\} + \{\phi^{(e)}\} - [X] \cdot [\dot{q}]\{\dot{\theta}\}$$

are stored in  $\{\text{ETC (I,K)}\}$ .

In the computation of the full XMN array only the nontrivial entries of XMN are computed, and the lower triangular portion of the array is obtained by making use of the fact that the array must be symmetric. To ensure that all redundant and trivial operations are circumvented, the labels stored in the integer code words

SI, SK, SR, SCXC, SMV, SOK, SQF,  
JCON, SFCC, SFLX, and SFXM

are used extensively.

For any particular problem, all computation carried out in QFDOT can be outputted on the line printer by defining

LQFDOT = .TRUE.

on card 1 of the data deck.

## PROGRAMMER'S GUIDE TO SUBROUTINE DCT

The primary purpose of subroutine DCT is to set up the differential equations required to compute the elements of the transformation matrices that are to be obtained by direction cosine techniques. The user need never interface with this subroutine.

In Reference 1, equation 119, the direction cosine equation is given in a form compatible with this coupled body analysis; namely

$$[{}^c\mathcal{T}_\lambda] = -\mathcal{P}(\{{}^c\vec{\omega}_\lambda\}_c) [{}^c\mathcal{T}_\lambda]$$

N-BOD2 is coded to integrate only six of the nine equations defined for each transformation matrix that is to be derived by direction cosine techniques. The other three parameters of the respective transformation matrices are obtained in TRAN from orthogonality conditions.

The following array is defined in table 1 and the elements of it are computed in DCT:

YMCD (I,J,K).

In DCT, a unique numbering sequence is used to facilitate the handling of the direction cosine data. The body labels stored in the integer code word SD are obtained first. Let

$\lambda_1, \lambda_2, \dots, \lambda_N =$  body labels associated with those body fixed-reference frames for which transformation matrices are to be obtained by direction cosine techniques

where

$$\lambda_1 > \lambda_2 > \dots > \lambda_N$$

If  $\lambda_N = 1$  and the body 1 fixed frame is the frame of computation, then to obtain the transformation matrix between inertial and body 1 coordinates,

$$\{\text{YMCD}(1, 1, 1)\} = [{}^c\mathcal{T}_0] \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix}$$

$$\{\text{YMCD}(1, 2, 1)\} = [{}^c\mathcal{T}_0] \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix}$$

and for the rest,  $M = 1, 2, \dots, N-1$

$$\{YMCD(I, 1, M+1)\} = \begin{bmatrix} \dot{\mathcal{F}}_{\lambda_M} \\ c \end{bmatrix} \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix}$$

$$\{YMCD(I, 2, M+1)\} = \begin{bmatrix} \dot{\mathcal{F}}_{\lambda_M} \\ c \end{bmatrix} \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix}$$

Otherwise, for  $M = 1, 2, \dots, N$ ,

$$\{YMCD(I, 1, M)\} = \begin{bmatrix} \dot{\mathcal{F}}_{\lambda_M} \\ c \end{bmatrix} \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix}$$

$$\{YMCD(I, 2, M)\} = \begin{bmatrix} \dot{\mathcal{F}}_{\lambda_M} \\ c \end{bmatrix} \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix}$$

For any particular problem, all computation carried in DCT can be outputted on the line printer by defining

`LDCT = .TRUE.`

on card 1 of the data deck.

### PROGRAMMER'S GUIDE TO SUBROUTINE ANGLE

The primary purpose of subroutine ANGLE is to set up the differential equations required to compute the displacement coordinates that are measured about or along the free coordinate vectors. The user need never interface with this subroutine.

The following array is defined in table 1 and the elements of it are computed in ANGLE:

ANGD(N) — time derivative of the displacement coordinate associated with the motion about or along the free coordinate vector SFR(N);  $N = 1, 2, \dots, NFRC$

In Reference 1, pages 37 and 38, the differential equations which define relative displacement about or along free coordinate vectors are defined. These equations are coded in

ANGLE. The user has the capability by subroutine INOPT to select which displacement coordinate should be evaluated. The integer array SFR defines the selected coordinates.

Several special cases are recognized in ANGLE:

- If relative translational motion is measured along free coordinate vector

$$M = \text{SFR}(N)$$

then

$$\text{ANGD}(N) = \text{THAD}(M) = \dot{\theta}_M$$

(Point mass motion or body 1 center-of-mass motion.)

- If free coordinate vector

$$M = \text{SFR}(N)$$

is aligned with the gimbal axis of a one-axis gimbal, then

$$\text{ANGD}(N) = \text{THAD}(M) = \dot{\theta}_M$$

- If the free coordinate vectors

$$M = \text{SFR}(N) \quad \text{and}$$

$$M+1 = \text{SFR}(N+1)$$

are aligned with the gimbal axes of a two-axis gimbal at hinge point K-1, then

$$C = \vec{q}_M \cdot \vec{q}_{M+1}$$

$$A_1 = \vec{q}_M \cdot \vec{\omega}_K$$

$$A_3 = \vec{q}_{M+1} \cdot \vec{\omega}_K$$

and

$$\text{ANGD}(N) = \frac{A_1 - C \cdot A_3}{1 - C^2}$$

$$\text{ANGD}(N+1) = \frac{A_3 - C \cdot A_1}{1 - C^2}$$

- If the free coordinate vectors

$$M + i = \text{SFR}(N + i) \quad i = 0, 1, 2$$

are aligned with the gimbal axes of a three-axis gimbal at hinge point K-1, then

$$C = \vec{q}_M \cdot \vec{q}_{M+2}$$

$$A_1 = \vec{q}_M \cdot \vec{\varepsilon}_K$$

$$A_2 = \vec{q}_{M+1} \cdot \vec{\varepsilon}_K$$

$$A_3 = \vec{q}_{M+2} \cdot \vec{\varepsilon}_K$$

and

$$\text{ANGD}(N) = \frac{A_1 - C * A_3}{1 - C^2}$$

$$\text{ANGD}(N + 1) = A_2$$

$$\text{ANGD}(N + 2) = \frac{A_3 - C * A_1}{1 - C^2}$$

For any particular problem, all computations carried out in ANGLE can be outputted on the line printer by defining

LANGLE = .TRUE.

on card 1 of the data deck.

### PROGRAMMER'S GUIDE TO SUBROUTINE SETUP

The primary purpose of SETUP is to provide the necessary interface between the symbolic array names used in the derivation of the equations of motion in N-BOD2 and the symbolic array names used in the integration subroutine RUNGE. The user need never interface with this subroutine.

The following sequential ordering scheme, used to define the locations of state variables and their time derivatives in the Y and YD arrays, is identical to that used in EQIV:

- Accelerations and rates relative to free coordinate vectors

$$YD(N) = THADD(N) \quad ; \quad THAD(N) = Y(N)$$

$$N = 1, 2, \dots, NFER$$

Sum equations: NEQ = NFER

- Accelerations and rates of generalized elastic coordinates

$$YD(N) = THADD(N) \quad ; \quad THAD(N) = Y(N)$$

$$N = NEQ+1, \dots, NEQ+NMODS$$

Sum equations: NEQ = NEQ+NMODS

- Accelerations and rates of symmetric wheels with variable rate. Let

$M_1, M_2, \dots, M_L$  = wheel labels in the set defined by integer code word SMV

$$YD(N) = THADD(N) \quad ; \quad THADW(M_I) = Y(N)$$

$$N = NEQ+1, \dots, NEQ+L$$

$$I = 1, 2, \dots, L$$

Sum equations: NEQ = NEQ+L

- Rate and displacement relative to free coordinate vectors

$$YD(N) = ANGD(MM) \quad ; \quad THA(M) = Y(N)$$

$$N = NEQ+1, \dots, NEQ+NFRC$$

$$MM = 1, 2, \dots, NFRC$$

$$M = SFR(MM)$$

Sum equations: NEQ = NEQ+NFRC

- Rate and displacement of generalized elastic coordinates

$$YD(N) = THAD(M) \quad ; \quad THA(M) = Y(N)$$

$$N = NEQ+1, \dots, NEQ+NMODS$$

$$M = 1, 2, \dots, NMODS$$

Sum equations: NEQ = NEQ+NMODS

- Rate and angle of symmetric wheels

$$YD(N) = THADW(M) \quad ; \quad THAW(M) = Y(N)$$

$$N = NEQ+1, \dots, NEQ+NMOA$$

$$MM = 1, 2, \dots, NMOA$$

$$M = SMA(MM)$$

Sum equations:  $NEQ = NEQ+NMODS$

- Rate and magnitude of direction cosine

$K_1, K_2, \dots, K_{ID}$  = body labels associated with the coordinate frames for which transformation matrix is to be obtained via direction cosine

$$YD(N) = YMCD(I, J, M)$$

$$M = 1, 2, \dots, ID$$

$$J = 1, 2$$

$$I = 1, 2, 3$$

$$N = NEQ + 6 * (M - 1) + 3 * (J - 1) + I$$

$$XMC(I, J, K_M) = Y(N)$$

$$I = 1, 2, 3$$

$$J = 1, 2$$

$$M = 1, 2, \dots, ID$$

The exact algorithm is obtainable in the discussion of subroutine DCT.

For any particular problem, all computations carried out in SETUP can be outputted on the line printer by defining

**LSETUP = .TRUE.**

on card 1 of the data deck.

## PROGRAMMER'S GUIDE TO SUBROUTINES OUTPUT AND OUTPSP

The primary purpose of subroutines OUTPUT and OUTPSP is to provide a convenient place to put all output data statements. OUTPUT is entered only at time zero while OUTPSP is entered at the end of every integration step unless otherwise directed by the user. The user is expected to refrain from modifying subroutine OUTPUT. It provides a useful echo of the initial state of the system for checkout purposes..

Nominally, subroutine OUTPSP is identical to OUTPUT. The user is expected to remove unwanted print statements from OUTPSP, add additional computation if desired, and insert any other set of desirable output statements.

The following parameters are outputted on the line printer by subroutine OUTPUT:

TIME — simulation time  $t$

CENTER OF MASS — components of the vector from hinge point 0 to the center-of-mass of the composite system relative to the frame of computation

TOTAL SYSTEM MASS — total mass of the system

SYSTEM INERTIA TENSOR — components of the total system inertia tensor relative to the composite system center-of-mass and relative to the frame of computation

ANGULAR MOMENTUM — magnitude of the total system inertial angular momentum vector

HBODY — components of the total system inertial angular momentum vector relative to body 1 fixed coordinates

\* HINERT — components of the total system inertial angular momentum vector relative to inertial coordinates

LINEAR MOMENTUM — magnitude of the total system inertial linear momentum vector

LBODY — components of the total system inertial linear momentum vector in computing frame coordinates

KINETIC ENERGY — kinetic energy of total system

After the above parameters are printed, a block of data for each body of the system is printed:

BODY K — body label for the body for which the adjacent block of data applies;  
 $K = 1, 2, \dots, \text{NBOD}$

$$\text{ROMC} = \{ \text{ROMC}(I,K) \} = \left\{ \vec{\mathcal{E}}_K \right\}_c$$

$$\text{FOMC} = \{ \text{FOMC}(I,K) \} = \left\{ \vec{\omega}_K \right\}_c$$



$$\text{ACC} = \text{not in common} = \left\{ \dot{\vec{\omega}}_K \right\}_c$$

$$\text{ETC} = \{ \text{ETC}(I,K) \} = \text{see QFDOT, also page 83}$$

$$\text{PHI} = \{ \text{PHI}(I,K) \} = \text{see TORQUE}$$

$$\text{CAC} = \{ \text{CAC}(I,K) \} = \left\{ \vec{a}_K \right\}_c$$

$$\text{CBC} = \{ \text{CBC}(I,K) \} = \left\{ \vec{\beta}_K \right\}_c$$

$$\text{POS} = \text{not in common} = \left\{ \vec{\gamma}_{I,K} \right\}_c$$

$$\text{VEL} = \text{not in common} = \left\{ \dot{\vec{\gamma}}_{I,K} \right\}_c$$

$$\text{XMC} = [ \text{XMC}(I, J, K) ] = [ {}_c \mathcal{T}_K ]$$

$$\text{XIC} = [ \text{XIC}(I, J, K) ] = [ \Phi_K ]_c$$

$$\text{HB} = \text{not in common} = \left\{ \vec{L}_{I,K} \right\}_c$$

$$\text{LM} = \text{not in common} = \left\{ \vec{G}_{I,K} \right\}_c$$

$$\text{TK} = \text{not in common}$$

$$= \frac{1}{2} \left[ \vec{\omega}_K \cdot (\Phi_\lambda \cdot \vec{\omega}_K + m_K \vec{\gamma}_{I,K} \times \dot{\vec{\gamma}}_{I,K}) \right.$$

$$\left. + m_K \dot{\vec{\gamma}}_{I,K} \cdot \dot{\vec{\gamma}}_{I,K} + \sum_{\substack{M: \\ \text{MO}(M)=K}} \vec{\omega}_{WM} \cdot \vec{H}_M \right]$$

The block of data pertinent to the translational motion of the center-of-mass of body 1 is given adjacent to the title, "ORIGIN."

$$\begin{aligned}
 \text{FOMC} &= \{\text{FOMC}(I, \text{NB1})\} = \{\vec{\beta}_1\}_c \\
 \text{ACC} &= \text{not in common} = \{\vec{\beta}_1\}_c \\
 \text{CBC} &= \{\text{CBC}(I, 0)\} = \{\vec{\beta}_0\}_c \\
 \text{ETC} &= \{\text{ETC}(I, \text{NB1})\} = \text{See QFDOT} \\
 \text{PHI} &= \{\text{PHI}(I, \text{NB1})\} = \text{See TORQUE} \\
 \text{XMC} &= \{\text{XMC}(I, J, 0)\} = [{}_c \mathcal{T}_0]
 \end{aligned}$$

For each symmetric wheel of the system, the following data are outputted:

$$\begin{aligned}
 \text{HMOM}(M) &= |\vec{H}_M| \\
 \text{CLM}(M) &= |\vec{CL}_M|
 \end{aligned}$$

For each flexible body of the system, a block of data associated with its flexible body characteristics is outputted:

**FLEXIBLE BODY K** — body label for the flexible body for which the adjacent block of data applies

$$\begin{aligned}
 \text{EP} &= \{\Delta\vec{\alpha}_K\}_K = \sum_{N,K} \{\vec{A}_{N,K}\}_K a_{N,K}(t) \\
 \text{EPD} &= \{\dot{\vec{\alpha}}_K\}_c = \sum_{N,K} \{\vec{A}_{N,K}\}_c \dot{a}_{N,K}(t) \\
 \text{QD} &= \sum_{N,K} \{\vec{Q}_{N,K}\}_K \dot{a}_{N,K}(t) \\
 \text{EI} &= \sum_{N,K} [E_{N,K}]_K a_{N,K}(t) = [\Delta\Phi]_K \\
 \text{EID} &= \sum_{N,K} [E_{N,K}]_K \dot{a}_{N,K}(t) = [\dot{\Phi}_K]_K \\
 \text{FIR} &= \{\text{FLIRC}(I, K)\} = \text{See ETA}
 \end{aligned}$$

$$\text{FCR} = \{\text{FLCRC}(\text{I}, \text{K})\} = \text{See ETA}$$

$$\dot{\text{DHM}} = \sum_{\text{N}, \text{K}} \{Q_{\text{N}, \text{K}}\}_c \dot{a}_{\text{N}, \text{K}}(t) = \{\vec{\Delta L}_{\lambda, \lambda}\}_c$$

Following the last block of flexible body data, the system state variables associated with relative body motion and flexible body motion are printed:

For  $M = 1, 2, \dots, \text{NFER}$

$$\text{THA}(M) = \theta_M(t)$$

$$\text{THAD}(M) = \dot{\theta}_M(t)$$

$$\text{THADD}(M) = \ddot{\theta}_M(t)$$

$$\text{QFC}(M) = \{\text{QFC}(\text{I}, M)\}_c = \{\vec{q}_M\}_c$$

For  $\text{MN} = 1, 2, \dots, \text{NMODS}$  and  $M = \text{NFER} + \text{MN}$

$$\text{THA}(M) = a_{\text{N}, \text{K}}(t)$$

$$\text{THAD}(M) = \dot{a}_{\text{N}, \text{K}}(t)$$

$$\text{THADD}(M) = \ddot{a}_{\text{N}, \text{K}}(t)$$

where

$$\text{MN} = \text{N} + \sum_{\text{I}=1}^{\text{K}-1} \text{SFXM}(\text{I})$$

$$\text{K} = 2, 3, \dots, \text{NBOD}$$

$$\text{N} = 1, 2, \dots, \text{SFXM}(\text{K}) \text{ [See INOPT, subsection 15]}$$

All of the above data will be printed at the end of each integration step. The user is expected to delete the undesired print statements from OUTPSP and to insert those more applicable to the problem at hand.

## PROGRAMMER'S GUIDE TO THE UTILITY ROUTINES

The utility routines required by N-BOD2 are:

- SIMQ — used to obtain the solution to a set of simultaneous linear equations of the form

$$[X] \{ \theta \} = \{ \eta \}$$

A standard reduction technique is used to solve for  $\{ \theta \}$  without ever obtaining the inverse of the coefficient matrix.

- RUNGE — used to numerically integrate the set of simultaneous nonlinear differential equations,

$$\{ \dot{y}(t) \} = \{ F(t) \}$$

Fourth order, fixed-step Runge-Kutta integration is used.

- COMPAC — used to create integer code words.

Integer code words are used primarily to conserve computer storage. This was a critical factor on the small XDS-9300 computer on which N-BOD2 was originally debugged.

Let

$$(I_1, I_2, \dots, I_N)$$

be a set of unique integer labels so that

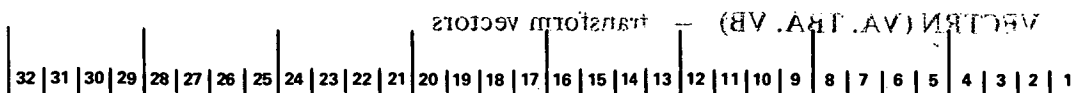
$$1 \leq I_i \leq 24 \quad ; \quad i = 1, 2, \dots, N$$

and

$$N \leq 24$$

If computer storage is limited, it is a waste of computer storage to reserve 24 words of memory to store this set.

All integer code words are 32 bits long.\* These bits may be numbered as shown below:



\*IBM 360 Computer, subroutine must be modified for computers having different word lengths.

COMPAC stores the number of labels ( $N$ ;  $N \leq 24$ ) in bits 25 through 32. The set of integers  $I_1, I_2, \dots, I_N$  are stored in bits 1 through 24. For  $J = 1, 2, \dots, 24$ , if bit  $J = 1$ ,  $J$  is a member of the set; if bit  $J = 0$ ,  $J$  is not a member of the set.

UNPAC — used to decode the integer code words created by COMPAC

KT0 and KT1 — integer function subprograms used to assign storage area locations in one-dimensional arrays for elements of triangular or symmetric matrices; another tool used to conserve storage requirements:

$$KT0(N, J, K) = \begin{cases} K(N-1) + J + 1 - K(K-1)/2 & K < J \\ J(N-1) + K + 1 - J(J-1)/2 & K \geq J \end{cases}$$

$$J = 0, 1, \dots, N-1$$

$$K = 0, 1, \dots, N-1$$

$$KT1(N, J, K) = \begin{cases} (K-1)(N-1) + J - (K-1)(K-2)/2 & K < J \\ (J-1)(N-1) + K - (J-1)(J-2)/2 & K \geq J \end{cases}$$

$$J = 1, 2, \dots, N$$

$$K = 1, 2, \dots, N$$

CTAIN — logical function subprogram used to determine if a particular integer is contained in a particular set of integers

CTAIN (I, S, N) = .TRUE. if the integer  $I$  is contained in the set of integers  $S(1), S(2), \dots, S(N)$

= .FALSE. if it is not

VECTRN (VA, TBA, VB) — transform vectors

$$\{\vec{v}\}_B = [{}_B\mathcal{T}_A] \{\vec{v}\}_A$$

TENTRN (XA, TBA, XB) — transform dyads

$$[\mathbf{x}]_B = [{}_B\mathcal{T}_A] [\mathbf{x}]_A [{}_B\mathcal{T}_A]^T$$

VECNRM (V) — normalize vector to unit magnitude

$$\vec{\omega} = \frac{\vec{v}}{|\vec{v} \cdot \vec{v}|^{1/2}}$$

$$\vec{v} = \vec{\omega}$$

Error message if  $\vec{v} \cdot \vec{v} = 0$  ; inputted vector destroyed

MATMUL (A, B, C, N) — multiply two N X N matrices of scalars

$$[C] = [A] [B]$$

TRNSPS (TBA) — transpose a 3 X 3 matrix of scalars

$$[TEM] = [TBA]^T$$

$$[TBA] = [TEM]$$

Inputted matrix destroyed

ROT (A, J, T) — forms transformation matrix for rotation about a coordinate axis

A — sine of the rotation angle

J — axis about which positive rotation is measured; J = +1, -1, +2, -2, +3 or -3

Define:

$$JJ = |J|$$

$$V_1 = 0$$

$$V_2 = 0$$

$$V_3 = 0$$

Redefine:

$$V_{JJ} = A * J / JJ$$

Then:

$$[T] = \begin{bmatrix} \sqrt{1 - V_2^2 - V_3^2} & -V_3 & V_2 \\ V_3 & \sqrt{1 - V_1^2 - V_3^2} & -V_1 \\ -V_2 & V_1 & \sqrt{1 - V_1^2 - V_2^2} \end{bmatrix}$$

VECADD (V1, V2, S) — add vectors

$$\vec{S} = \vec{v}_1 + \vec{v}_2$$

VECSUB (V1, V2, D) — subtract vectors

$$\vec{D} = \vec{v}_1 - \vec{v}_2$$

SCLV (SC, V, P) — multiply vector by a scalar

$$\vec{P} = SC * \vec{v}$$

VECDOT (V1, V2, D) — vector dot product

$$D = \vec{v}_1 \cdot \vec{v}_2$$

VECROS (V1, V2, C) — vector cross product

$$\vec{C} = \vec{v}_1 \times \vec{v}_2$$

TRIPVP (V1, V2, V) — special triple vector product

$$\vec{v} = \vec{v}_1 \times (\vec{v}_1 \times \vec{v}_2)$$

DYADD (D1, D2, D) — add dyads

$$D = D_1 + D_2$$

SCLD (A, D, T) — multiply dyad by a scalar

$$T = A * D$$

DYDOTV (A, V, D) — dyad dot vector

$$\vec{D} = \mathbf{A} \cdot \vec{v}$$

VXDYOV (V1, D, V) — special vector cross dyad dot vector

$$\vec{v} = \vec{v}_1 \times (\mathbf{D} \cdot \vec{v}_1)$$

DYTOV (D, V1, V) — dyad transpose dot vector

$$\vec{v} = \mathbf{D}^T \cdot \vec{v}_1$$

VODYOV (V1, D, V2, SC) — vector dot dyad dot vector equals scalar

$$SC = \vec{v}_1 \cdot (\mathbf{D} \cdot \vec{v}_2)$$

DYOP (V, D) — skew operator

$$\mathbf{D} = \mathcal{P}(\vec{v})$$

$$= \begin{bmatrix} 0 & v_3 & -v_2 \\ -v_3 & 0 & v_1 \\ v_2 & -v_1 & 0 \end{bmatrix}$$

SUEOP (V1, V2, XM, D) — construct pseudo-inertia tensors

$$\mathbf{D} = m [(\vec{v}_1 \cdot \vec{v}_2) \mathbf{1} - \vec{v}_2 \vec{v}_1]$$

QUTMUL (Q1, Q2, P) — multiply quaternions

$$\bar{p} = \bar{Q}_1 * \bar{Q}_2$$



QUATOP (QF, THA, ZT) — construct rotation quaternion

$$\theta = \text{THA}$$

$$\{\vec{QF}\} = \begin{Bmatrix} q_1 \\ q_2 \\ q_3 \end{Bmatrix}$$

$$\{\bar{ZT}\} = \begin{Bmatrix} \cos \theta/2 \\ q_1 \sin \theta/2 \\ q_2 \sin \theta/2 \\ q_3 \sin \theta/2 \end{Bmatrix}$$

TRANSO (ZT, TAU) — construct transformation matrix from quaternion component

$$\{\bar{ZT}\} = \begin{Bmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \end{Bmatrix}$$

$$[\text{TAU}] = \begin{bmatrix} e_0^2 + e_1^2 - e_2^2 - e_3^2 & 2(e_0 e_3 + e_1 e_2) & 2(e_1 e_3 - e_0 e_2) \\ 2(e_1 e_2 - e_0 e_3) & e_0^2 - e_1^2 + e_2^2 - e_3^2 & 2(e_2 e_3 + e_0 e_1) \\ 2(e_1 e_3 + e_0 e_2) & 2(e_2 e_3 - e_0 e_1) & e_0^2 - e_1^2 - e_2^2 + e_3^2 \end{bmatrix}$$

UNCAGE (SCG, SC, T, TUG) — check to see if it is time to uncage any of the caged degrees of freedom. If it is time, redefine SCG and SC.

COMPRS (XMN, THADD, N, SC, SCG, LG) — remove rows from XMN and THADD arrays and columns from XMN array associated with caged degrees of freedom, renumber and return compressed XMN and THADD arrays.

UNPRS (THADD, M, SCG, LG) — expand THADD array putting zeros in locations associated with caged degrees of freedom.

## NOTATIONAL CROSS-REFERENCE

The equations of motion are derived in References 1 and 2 using standard analytic notation. They are programmed in N-BOD2 using FORTRAN IV notation and are outputted on the line printer using an abbreviated FORTRAN notation. This section is intended to provide the user with a notation cross-reference between the three techniques used to define and evaluate the equations of motion.

### 1. Matrix Notation

$\{ \dots \}$  — column matrix

$[ \dots ]$  — square matrix

$[ \dots ]$  — row or rectangular matrix

### 2. Vector-Tensor-Matrix Notation

Vectors and tensors must be stored relative to particular reference frames. A post subscript outside of the brace or bracket is used to specify the reference frame in which the vector or tensor is defined. For example,

$\{\vec{V}_K\}_K$  —  $3 \times 1$  column matrix of the components of the vector  $\vec{V}_K$  relative to body K fixed coordinates.

$[T_K]_C$  —  $3 \times 3$  square matrix of the components of the tensor  $T_K$  relative to the frame of computation

If the elements of the vector  $\vec{V}_K$  are stored relative to the body K fixed-reference frame in the array VE and relative to the frame of computation in the array VEC,

where

DIMENSION VE (3, 10), VEC (3, 10),

the following notation is used:

Analytic Notation References 1 and 2	N-BOD2 FORTRAN IV Coding	N-BOD2 Users Manual	N-BOD2 Outputted Equations
$\{\vec{V}_K\}_K$	VE (1, K) VE (2, K) VE (3, K)	$\{VE (I,K)\}_K$	VE (K)
$\{\vec{V}_K\}_C$	VEC (1, K) VEC (2, K) VEC (3, K)	$\{VEC (I,K)\}_C$	VEC (K)

### 3. Mathematical Operational Notation

In the output of the equations of motion, the analytic symbology used to define several mathematical operations cannot be duplicated with a standard line printer. The following notation has been adopted:

Let

REAL S1, S2, S3,  
V1 (3), V2 (3), V3 (3),  
T1 (3, 3), T2 (3, 3), T3 (3, 3),  
Q1 (4), Q2 (4), Q3 (4),  
M1 (3, 3), M2 (3, 3), M3 (3, 3)

define the dimension of the following quantities:

S1, S2, S3 — storage locations for the scalars  $S_1, S_2, S_3$

V1, V2, V3 — storage locations for the components of the vectors

$\vec{V}_1, \vec{V}_2, \vec{V}_3$  — the elements of the vector  $\vec{V}_K$  are stored in the order  $V_1, V_2, V_3$

T1, T2, T3 — storage locations for the components of the tensors of rank two, dyads  $T_1, T_2, T_3$

Q1, Q2, Q3 — storage locations for the components of the quaternions  $\bar{q}_1, \bar{q}_2, \bar{q}_3$

M1, M2, M3 — storage locations for the components of the  $3 \times 3$  matrices of scalars  $[M_1], [M_2], [M_3]$

The following cross-reference for mathematical operations is used in the output of equations of motion:

- Addition

$$\begin{aligned}
 S_3 &= S_1 + S_2 & \approx S3 &= S1 + S2 \\
 \vec{v}_3 &= \vec{v}_1 + \vec{v}_2 & \approx V3 &= V1 + V2 \\
 T_3 &= T_1 + T_2 & \approx T3 &= T1 + T2 \\
 [M_3] &= [M_1] + [M_2] & \approx M3 &= M1 + M2
 \end{aligned}$$

- Subtraction

(Same as addition)

- Division

$$\begin{aligned}
 S_3 &= \frac{S_1}{S_2} & \approx S3 &= S1/S2 \\
 \vec{v}_3 &= \frac{1}{S_1} \vec{v}_1 & \approx V3 &= V1/S1 \\
 T_3 &= \frac{1}{S_1} T_1 & \approx T3 &= T1/S1
 \end{aligned}$$

- Scalar Multiplication

$$\begin{aligned}
 S_3 &= S_1 S_2 & \approx S3 &= S1*S2 \\
 \vec{v}_3 &= S_1 \vec{v}_1 & \approx V3 &= S1*V1 \\
 T_3 &= S_1 T_1 & \approx T3 &= S1*T1 \\
 [M_3] &= S_1 [M_1] & \approx M3 &= S1*M1
 \end{aligned}$$

- Quaternion Multiplication

$$\bar{Q}_3 = \bar{Q}_1 \bar{Q}_2 \approx Q3 = Q1*Q2$$

- Scalar Product

$$S_1 = \vec{v}_1 \cdot \vec{v}_2 \approx S1 = V1 \cdot V2$$

$$\vec{v}_1 = \mathbf{T}_1 \cdot \vec{v}_2 \approx V1 = T1 \cdot V2$$

$$\vec{v}_3 = \vec{v}_2 \cdot \mathbf{T}_1 \approx V3 = V2 \cdot T1$$

- Vector Product

$$\vec{v}_3 = \vec{v}_1 \times \vec{v}_2 \approx V3 = V1 \times V2$$

$$\mathbf{T}_1 = \vec{v}_1 \times \mathbf{T}_2 \approx T1 = V1 \times T2$$

$$\mathbf{T}_3 = \mathbf{T}_2 \times \vec{v}_1 \approx T3 = T2 \times V1$$

- Tensor Product

$$\mathbf{T}_1 = \vec{v}_1 \vec{v}_2 \approx T1 = V1 V2$$

- Transpose Matrix

$$[M_2] = [M_1]^T \approx M_2 = M1 ** T$$

- Transform Coordinates of Vectors and Tensors

If

$[M_1]$  — transformation matrix, reference frame A to reference frame B

$V1, T1$  — storage area for components of  $\vec{v}_1$  and  $\mathbf{T}_1$  relative to reference frame A respectively

$V2, T2$  — storage area for components of  $\vec{v}_1$  and  $\mathbf{T}_1$  relative to reference frame B respectively,

$$M1 * M2 = M \approx [M] \approx [M]$$

then

$$\{\vec{v}_1\}_B = [M_1] \{\vec{v}_1\}_A \approx V2 = M1 * V1$$

$$[T_1]_B = [M_1] [T_1]_A [M_1]^T \approx T2 = M1 * T1 * M1 ** T$$

- Rotation Quarternion

Using the notation of Reference 1 (see equations 99, 100, 101, and A-21),

$$\bar{Q}_1 = \cos \frac{S_1}{2} + \mathcal{Q} \left( \{ \vec{v}_1 \} \right) \sin \frac{S_1}{2}$$

$$\approx Q1 = \text{QUATOP}(V1, S1)$$

Subroutine QUATOP makes use of the scalar S1 and the three elements stored in the array V1 to compute the four components of the quarternion Q1 (see Utility Routine/QUATOP).

- Transformation Matrix from Rotation Quarternion

Using the notation of Reference 1 (see equations 96, 97, 98, and A-18),

$$[M_1] = \mathcal{F}(\bar{Q}_1)^T \approx M1 = \text{TRANSO}(Q1)$$

Subroutine TRANSO makes use of the four elements stored in the array Q1 to compute the components of the associated transformation matrix (see Utility Routine/TRANSO).

- Transform Vectors to Tensors

Using the notation of Reference 1 (see equation 51),

$$T_1 = \mathcal{P}(\vec{v}_1) \approx T1 = \text{SKEW}(V1)$$

- Pseudo-Inertia Tensors

Using the notation of Reference 1 (see equation 53 or Reference 2, equation 98),

$$T_1 = S_1 [ \vec{v}_1 \cdot \vec{v}_2 \mathbf{1} - \vec{v}_1 \vec{v}_2 ]$$

$$\approx T1 = S1 * (V1 \cdot V2 * \mathbf{1} - V1 V2)$$

$$= \text{SUEOP}(V2, V1, S1)$$

Subroutine SUEOP makes use of the scalar S1 and the elements stored in two arrays V1 and V2 to compute the components of the tensor T1 (see Utility Routines/SUEOP).

- **Normalize Vectors**

$$\vec{v}_2 = \frac{\vec{v}_3 \times \vec{v}_1}{|\vec{v}_3 \times \vec{v}_1|} \approx \text{NORM}(\text{V3} \times \text{V1})$$

Goddard Space Flight Center  
 National Aeronautics and Space Administration  
 Greenbelt, Maryland      December 1977

$$(\text{SV IV} - 1) * (\text{V3} + \text{IV}) * \text{I2} = \text{IT} *$$

## REFERENCES

1. Frisch, Harold P., "A Vector - Dyadic Development of the Equations of Motion for N-coupled Rigid Bodies and Point Masses," NASA Document TN D-7767, October 1974.
2. Frisch, Harold P., "A Vector - Dyadic Development of the Equations of Motion for N-coupled Flexible Bodies and Point Masses," NASA Document TN D-8047, August 1975.



**APPENDIX A**

**PROGRAM LISTING FOR N-BOD2**





C	IDEM5	I		LOCAL	SIZE OF /INTG/ (RSTART)	00112000
C	IDEM6	I		LOCAL	SIZE OF /INTGZ/ (RSTART)	00112100
C	IDEM7	I		LOCAL	SIZE OF /REAL/ (RSTART)	00112200
C	IDEM8	I		LOCAL	SIZE OF /REALZ/ (RSTART)	00112300
C	IDEM9	I		LOCAL	SIZE OF /SATELL/ (RSTART)	00112400
C	IINIT	I	IDEM5	EQIV(AWORK)	ZERO OUT /INTG/ (RSTART)	00112500
C	INEFF	L		/LOGIC/	FLAG,BFC OR IFC FOR CFC (INOPT)	00112600
C	IZINIT	I	IDEM6	EQIV(SCNDUM)	ZERO OUT /INTGZ/ (RSTART)	00112700
C	JCCN	I	N	/INTG/	BODY CONNECTION MATRIX (INBS)	00112800
C	LCON	I	2(N+1)	/INTG/	CODE,LOCKED VECTORS (INBS)	00112900
C	LANGLE	L		/LDEBUG/	PRINT EQUATIONS IN ANGLE? (MAIN)	00113000
C	LDCT	L		/LDEBUG/	PRINT EQUATIONS IN DCT? (MAIN)	00113100
C	LEGL	L		EQIV( )	EQIV IN EACH SUB TO PRINT FLAG	00113200
C	LEQUIV	L		/LDEBUG/	PRINT EQUATIONS IN EQIV? (MAIN)	00113300
C	LETA	L		/LDEBUG/	PRINT EQUATIONS IN ETA? (MAIN)	00113400
C	LINIT	L	IDEM4	EQIV(FG1)	ZERO OUT /LOGIC/ (RSTART)	00113500
C	LQFCOT	L		/LDEEUG/	PRINT EQUATIONS IN QFCOT? (MAIN)	00113600
C	LRATE	L		/LDEBUG/	PRINT EQUATIONS IN RATE? (MAIN)	00113700
C	LRTAPE	L		/CHECKS/	CREATE RESTART TAPE? (MAIN)	00113800
C	LRUNGE	L		/LDEBUG/	PRINT EQUATIONS IN RUNGE? (MAIN)	00113900
C	LSETUP	L		/LDEBUG/	PRINT EQUATIONS IN SETUP? (MAIN)	00114000
C	LSIMQ	L		/LDEBUG/	PRINT EQUATIONS IN SIMQ? (MAIN)	00114100
C	LTCFQL	L		/LDEBUG/	PRINT EQUATIONS IN TORQUE? (MAIN)	00114200
C	LTRAN	L		/LDEBUG/	PRINT EQUATIONS IN TRAN? (MAIN)	00114300
C	LTRANV	L		/LDEBUG/	PRINT EQUATIONS IN TRANVD? (MAIN)	00114400
C	LTRANSI	L		/LDEBUG/	PRINT EQUATIONS IN TRNSIV? (MAIN)	00114500
C	LVDIV	L		/LDEBUG/	PRINT EQUATIONS IN VDIV? (MAIN)	00114600
C	LXDY	L		/LDEBUG/	PRINT EQUATIONS IN XDY? (MAIN)	00114700
C	MC	I	N	/INTG/	BODY IN WHICH WHEEL IS IN (INBS)	00114800
C	NEOD	I		/INTG/	NUMBER OF BODIES (INBS)	00114900
C	NE1	I		/INTG/	NUMBER OF BODIES + 1 (INBS)	00115000
C	NCTC	I		/INTG/	(INOPT,NA) CONSTRAINT TORQUES	00115100
C	NEQ	I		LOCAL	# EQUATIONS SETUP BY N-BOD2 (EQIV)	00115200
C	NFER	I		/INTG/	# FREE COORD VECTORS (INBS)	00115300
C	NFKC	I		/INTG/	(INOPT,NA) CONSTRAINT FORCES	00115400
C	NFLXB	I		/INTG/	# FLEXIBLE BODIES (INOPT)	00115500
C	NFRC	I		/INTG/	# RELATIVE ANGLES COMPUTED (INOPT)	00115600
C	NLOR	I		/INTG/	# LOCKED COORD VECTORS (INBS)	00115700
C	NMC	I		/INTG/	TOTAL NUMBER OF WHEELS (INBS)	00115800
C	NMOA	I		/INTG/	# WHEELS TO COMP REL ANGLE (INOPT)	00115900
C	NWCDS	I		/INTG/	TOTAL # MODES FOR SYSTEM (INOPT)	00116000
C	NV	I		/INTG/	# VARIABLE SPEED WHEELS (INOPT)	00116100
C	NSTART	L		/CHECKS/	NEW CR RESTART RUN? (MAIN)	00116200
C	NSVP	I		/INTG/	# LOCKED VECTORS TRANSFORM(VDIV)	00116300
C	NSYC	I		/INTG/	# FREE VECTORS TRANSFORM (VDIV)	00116400
C	NTQ	I		/INTG/	# DIFF EQS. IN SUB TORQUE (TORQUE)	00116500
C	PCGN	I	N+1	/INTG/	# CONSTRAINED AXES AT HINGES (INBS)	00116600
C	PFI	R	3,N+1	/REAL/	EXTERNAL TORQUE ON NEST (TORQUE)	00116700
C	PLM	R	N	/REAL/	WHEEL SPIN INERTIA (INBS)	00116800
C	CF	R	3,3(N+1)	/REAL/	FREE VECTOR BFC (INBS)	00116900
C	CFC	R	3,3(N+1)	/REAL/	FREE VECTOR CFC (VDIV,TRANVD)	00117000
C	CL	R	3,2(N+1)	/REAL/	LOCKED VECTOR BFC (INBS)	00117100
C	QLC	R	3,2(N+1)	/REAL/	LOCKED VECTOR CFC (VDIV,TRANVD)	00117200
C	RELC	L	N	/LOGIC/	RIGID BODY OR POINT MASS? (INBS)	00117300
C	RINIT	R	IDEM7	EQIV(CA)	ZERO OUT /REAL/ (RSTART)	00117400
C	RCMC	R	3,N+1	/REAL/	RELATIVE RATE VECTOR (RATE)	00117500
C	RZINIT	R	IDEM8	EQIV(CBDUM)	ZERO OUT /REALZ/ (RSTART)	00117600
C	SC	I	3(N+1)	/INTG/	FREE VECTORS CAGED (INOPT,UNCAGE)	00117700
C	SCC	I	N	/INTG/	UNUSED	00117800
C	SCG	I		/INTG/	# CAGED DEGREES (INOPT,UNCAGE)	00117900

C	SCNDCUM,SCN	I	0:N-1	/INTGZ/	CODE,CENTRIPETAL EFFECTS (INOPT)	C0118000
C	SCRCUM,SCR	I	0:N-1	/INTGZ/	CODE,CORIOLIS EFFECTS (INOPT)	C0118100
C	SCXC	I	2N	EGIV(TCRQ)	CODE,X-COUP MODES (INOPT,QFCOT)	C0118200
C	SD	I		/INTG/	CODE,DIRECTION COSINES (INOPT)	C0118300
C	SEU	I		/INTG/	CODE,EULER ANGLES (INOPT)	C0118400
C	SFCC	I		/INTG/	CODE,BODIES FLEX X-COUPLEING (INOPT)	C0118500
C	SFKDCUM,SFK	I	0:N-1	/INTGZ/	CODE,CONSTRAINT FORCE(INOPT,NA)	C0118600
C	SFLX	I		/INTG/	CODE,ALL FLEXIBLE BODIES (INOPT)	C0118700
C	SFR	I	3(N+1)	/INTG/	CODE,COMPUTE FREE VEC ANGLE(INOPT)	C0118800
C	SFXM	I	N	/INTG/	# MODES EACH BODY (INOPT)	C0118900
C	SG	I		/INTG/	CODE,ALL GYROSTATS(SETS)	C0119000
C	SI	I	IDEM3	/INTG/	CODE,BODIES HINGE TO CM (SETS)	C0119100
C	SIG	I		/INTG/	UNUSED	C0119200
C	SIXDCUM,SIX	I	0:N-1	/INTGZ/	CODE,INERTIA EFFECTS (INOPT)	C0119300
C	SKDCUM,SK	I	0:N-1	/INTGZ/	CODE,BODIES IN EACH NEST (SETS)	C0119400
C	SL	I		/INTG/	CODE,ALL POINT MASSES (SETS)	C0119500
C	SLK	I	3(N+1)	/INTG/	CODE,CONSTRAINT TORQUE(INOPT,NA)	C0119600
C	SMA	I	N	/INTG/	CODE,WHEEL ANGLE COMPUTE (INOPT)	C0119700
C	SMAL	I		/INTG/	CODE,SMALL ANGLES (INOPT)	C0119800
C	SMDCUM,SMC	I	0:N-1	/INTGZ/	CODE,ALL WHEELS IN NEST (INOPT)	C0119900
C	SMV	I		/INTG/	CODE,VARIABLE SPEED WHEELS (INOPT)	C0120000
C	SCK	I	N+1	/INTG/	CODE,BODIES HINGE 0 - CM (VDIV)	C0120100
C	SPDCUM,SPI	I	0:N-1	/INTGZ/	CODE,PSUEDO INERTIA TENSORS(INOPT)	C0120200
C	SGF	I	N+1	/INTG/	CODE,FREE VECTOR AT HINGE (SETS)	C0120300
C	SQL	I	N+1	/INTG/	CODE,LOCKED VECTOR AT HINGE (SETS)	C0120400
C	SR	I		/INTG/	CODE,ALL RIGID BODIES (SETS)	C0120500
C	SECK	I		/INTG/	CODE,UNION OF ALL SCN (VDIV)	C0120600
C	SSIX	I		/INTG/	CODE,UNION OF ALL SIX (VDIV)	C0120700
C	SVA	I		/INTG/	CODE,CM VECTORS TRANSFORM (VDIV)	C0120800
C	SVB	I		/INTG/	CODE,HINGE VECTORS TRANSFORM(VDIV)	C0120900
C	SVD	I		/INTG/	CODE,DCN'T TRANSFORM (INOPT)	C0121000
C	SVI	I		/INTG/	CODE,INERTIA DYAD TRANSFORM (VDIV)	C0121100
C	SVM	I		/INTG/	CODE,SPIN VECTORS TRANSFORM(VDIV)	C0121200
C	SVP	I	2(N+1)	/INTG/	CODE,LOCKED VECTORS TRANSFORM(VDIV)	C0121300
C	SVQ	I	3(N+1)	/INTG/	CODE,FREE VECTORS TRANSFORM(VDIV)	C0121400
C	SXM	I	3,N	/INTG/	CODE,SMALL ANGLE KINEMATICS(INOPT)	C0121500
C	EXT	I		/INTG/	CODE,TIME VARY CUL INER MAT(INOPT)	C0121600
C	T	R		/REAL/	TIME (MAIN)	C0121700
C	TEM	R	2,160	LCCAL	TEMP STORAGE AREA (RUNGE)	C0121800
C	TFA	R	33	/REAL/	GENERALIZED COORDINATES(INBS,SETUP)	C0121900
C	TFAD	R	33	/REAL/	GENERALIZE COORD RATE (INBS,SETUP)	C0122000
C	TFACD	R	33	EQIV(ETM)	GENERALIZE COORD ACC (SETUP,SIMQ)	C0122100
C	TFACW	R	N	/REAL/	WHEEL RATE (INBS,SETUP)	C0122200
C	TFAW	R	N	/REAL/	WHEEL ANGLE (INBS,SETUP)	C0122300
C	TIMEND	R		/REAL/	TIME TO END RUN (INBS,DYN)	C0122400
C	TCRC	I	97	/INTG/	UNUSED STORAGE AREA FOR USER	C0122500
C	TLG	R	3(N+1)	/REAL/	TIME TO UNCAGE (INOPT,UNCAGE)	C0122600
C	XCIC	R	3,3,IDEM2	/REAL/	MATRIX OF INERTIA TENSORS(VDIV,XYC)	C0122700
C	XI	R	3,3,N	/REAL/	INERTIA DYAD BFC (INBS)	C0122800
C	XIC	R	3,3,N	/REAL/	INERTIA DYAD CFC (VDIV,TRANVC)	C0122900
C	XID	R	3,3,N	/REAL/	ZERO INERTIA DYAD BFC(VDIV)	C0123000
C	XMAS	R	N	/REAL/	BODY MASS (INBS)	C0123100
C	XMDCUM,XMC	R	3,3,C:N	/REALZ/	TRANSFORM BFC TO CFC (TRANSIV,TRAN)	C0123200
C	XMN	R	33,33	/REAL/	SCALAR INERTIA MATRIX (VDIV,QFCOT)	C0123300
C	Y	R	160	LOCAL	SYSTEM STATE (EQIV,SETUP,TORQUE)	C0123400
C	YD	R	160	LOCAL	SYSTEM STATE DERIV (SETUP,TORQUE)	C0123500
C	YMCD	R	3,2,N+1	EGIV(XMN)	DIRECTION CCSINE RATES (DCT)	C0123600
C	XMT	R	3,3,N	/REAL/	ZERO STATE TRANSFORMATION MAT(INES)	C0123700
C	ZETA	R	2N	/REAL/	MODAL DAMPING RATIO (INOPT)	C0123800
C						C0123900

	ROUTINE LOCATION	
C		C0124000
C		C0124100
C		C0124200
C		C0124300
C	MAIN	00000100
C	DYN	00200000
C	RESTART	00300000
C	INBS	00400000
C	INERCK	00500000
C	SETS	00600000
C	INJFT	00700000
C	INTCK	00800000
C	TRANSIV	00900000
C	VDIV	01000000
C	EQIV	01100000
C	TRAN	01200000
C	TRANVD	01300000
C	FATE	01400000
C	XDY	01500000
C	ETA	01600000
C	TCRQUE	01700000
C	GFOGT	01800000
C	DCT	01900000
C	ANGLE	02000000
C	SETUP	02100000
C	CUTPLT	02200000
C	CUTPSP	02300000
C	SIMG	02400000
C	RUNGE	02500000
C	LNCAGE	02600000
C	CCMFRS	02602600
C	LNPHS	02605100
C	CCMFAC	02606800
C	LNPAC	02609100
C	KTO	02612200
C	KT1	02613000
C	CTAIN	02613800
C	VECTRN	02700000
C	TENTRN	02702000
C	VECNRM	02704600
C	MATMLL	02706100
C	TRANSPS	02707400
C	RDT	02708600
C	VECADD	02800000
C	VECSLB	02801000
C	SCLV	02801500
C	VECDCT	02802800
C	VECRCS	02803500
C	TRIPVP	02804400
C	DYADD	02806000
C	SCLC	02807000
C	DYDCTV	02808500
C	VXDYCV	02809600
C	DYTCV	02811000
C	VBDYCV	02812400
C	DYDP	02813500
C	SLECP	02815400
C	GUTMLL	02900000
C	GLATCP	02902200
C	TRANSU	02903300
		C0129400
		C0129500
		C0129600
		C0129700
		C0129800
		C0129900

```

C
C
IMPLICIT REAL*8(A-F,O-Z,*)
LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLO, LEQU, LINIT(1)
LOGICAL LNSTART, LRTAPE
LOGICAL LRLNGE, LTRNSI, LVDIV, LEQIV, LTRAN,
* LTRANV, LRATE, LXDY, LETA, LTRQU,
* LGFDDT, LDCT, LANGLE, LSETUP, LSIMQ
C
C
INTEGER
* ANCRK, CT1, CT2, CT3, CT4, CT5, FCON, PCON,
* SCNDUM, SCN, SCRUM, SCR, SFKUM, SFK, SFR, SG,
* SI, SIG, SIXDUM, SIX, SKUM, SK, SL, SLK,
* SMA, SMCUM, SMC, SMV, SOK, SPIDUM, SPI, SQF,
* SGL, SR, SSCN, SEIX, SVA, SVB, SVD, SVI,
* SVM, SVP, SVQ, SXM, SXT, TORQ, SMAL, SEU,
* SC, SCG, NFLXB, SFLX, SFXM, NMODS, SFCC, SCC,
* IINIT(1), IZINIT(1), SD, SCXC(20)
C
C
REAL*8
* ANGC (33), CNF (3,10), ETIC (3,10), ETMC (3,10),
* FLQ (3,20), FLE (3,3,20), FLH (3,3,20),
* TRADD (33), YMCD (3,2,11), RINIT (1), RZINIT(1)
C
C
COMMON /CHECKS/ LNSTART, LRTAPE
C
COMMON /LDEBUG/ LRLNGE, LTRNSI, LVDIV, LEQIV, LTRAN,
* LTRANV, LRATE, LXDY, LETA, LTRQU,
* LGFDDT, LDCT, LANGLE, LSETUP, LSIMQ
C
COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RBLO(10)
C
COMMON /INTG/ ANCRK (200),
* CT1, CT2, CT3, CT4,
* CT5, FCON (33), JCON (10), LCUN (22),
* MC (10), NB1, NOD, NCTC,
* NFER, NFKC, NFR, NLR,
* NV, NM, NMOA, NSVP,
* NSVC, FCON (11), SD, SFR (33),
* SG, SI (55), SIG, SL,
* SLK (33), SMA (10), SOK (11), SQF (11),
* SGL (11), SMV, SR, SSCN,
* SEIX, SVA, SVB, SVD,
* SVI, SVM, SVP (22), SVQ (33),
* SXM (3,10), SXT, TORQ (97), SMAL,
* SEU, NTQ, SC (33), SCG,
* NFLXB, SFLX, SFXM (10), NMODS,
* SFCC, SCC (10)
C
C
COMMON /INTGZ/
* SCNDUM, SCN (5), SCRUM, SCR (9),
* SFKUM, SFK (5), SIXDUM, SIX (9),
* SKUM, SK (5), SPIDUM, SPI (9),
* SMCUM, SMC (5)
C

```





C	INPUT DESCRIPTION OF BASIC SYSTEM	C0142000
C	TOPOLOGY	C0142100
C	INERTIA	C0142200
C	MOMENTUM WHEELS	C0142300
C	GEOMETRY	C0142400
C	NOMINAL STATE	C0142500
C	KINEMATICAL CONSTRAINTS	C0142600
C	INPUT INITIAL CONDITIONS	C0142700
C	RATES	C0142800
C	DISPLACEMENT	C0142900
C	FREE COORDINATES	C0143000
C	MOMENTUM WHEELS	C0143100
C	INTEGRATION TIME STEP	C0143200
C	CALL INDS	C0143300
C	CHECK FOR PHYSICALLY REALIZABLE SYSTEM	C0143400
C	CALL INERDR	C0143500
C	IF(.NOT.FG2) GO TO 4	C0143600
C	FG2 RESET FALSE IN INERDR IF PHYSICALLY UNREALIZABLE	C0143700
C		C0143800
C	COMPUTE BODY LABEL SETS NEEDED FOR SUMMATION CHAINS	C0143900
C	CALL SETS	C0144000
C		C0144100
C	INPUT COMPUTATION OPTIONS	C0144200
C	FRAME OF COMPUTATION	C0144300
C	AUGMENTED SETS FOR SUMMATION TRUNCATION	C0144400
C	DIRECTION COSINE DELETION	C0144500
C	COLUMNS OF INERTIA TENSOR DELETION	C0144600
C	TRANSFORMATION SUPPRESSION	C0144700
C	EULER ANGLE TECHNIQUES	C0144800
C	SMALL ANGLE ASSUMPTIONS	C0144900
C	ANGULAR DISPLACEMENT	C0145000
C	MOMENTUM WHEEL RATE	C0145100
C	MOMENTUM WHEEL ANGLE	C0145200
C	FLEXIBLE BODIES	C0145300
C	MODAL COUPLING	C0145400
C	LAGGED DEGREES OF FREEDOM	C0145500
C	CALL INOPT	C0145600
C	IF(.NOT.FG3) GO TO 4	C0145700
C	FG3 RESET FALSE IN INOPT IF OPTION CARD NOT RECOGNIZED	C0145800
C		C0145900
C	INPUT PARAMETERS NEEDED TO DEFINE EXTERNAL DISTURBANCES	C0146000
C	GRAVITY	C0146100
C	GRAVITY GRADIENT	C0146200
C	CRBIT	C0146300
C	LOCALLY APPLIED FORCES	C0146400
C	SPRINGS	C0146500
C	DAMPERS	C0146600
C	MOTORS	C0146700
C	MOMENTUM WHEEL CONTROL	C0146800
C	CONTROL SYSTEMS	C0146900
C	THERMAL DEFORMATION	C0147000
C	OTHER	C0147100
C	CALL INTCR	C0147200
C		C0147300
C	COMPUTE INITIAL VALUES FOR ALL SYSTEM PARAMETERS	C0147400
C	TRANSFORMATION MATRICES	C0147500
C	CENTER OF MASS VECTORS	C0147600
C	HINGE POINT VECTORS	C0147700
C	INERTIA TENSOR	C0147800
C	FREE VECTORS	C0147900

C	LCKED VECTORS	C0148000
C	RATE VECTORS	C0148100
C	COMPOSITE VECTORS AND DYADS	C0148200
C	CROSS COUPLING	C0148300
C	EXTERNAL DISTURBANCES	C0148400
C	SYSTEM DYNAMICS	C0148500
C	MOMENTUM WHEEL DYNAMICS	C0148600
C	ACCELERATION ABOUT-ALONG FREE VECTORS	C0148700
C	DIRECTION COSINE RATES	C0148800
	T = C	C0148900
	CALL DYN(Y,YD,NEQ)	C0149000
C	OUTPUT TOTAL SYSTEM STATE AT T=0	C0149100
	CALL OUTPUT	C0149200
	PRINT 101	C0149300
	CALL OUTPSP	C0149400
		C0149500
		C0149600
C	START BASIC INTEGRATION OF SYSTEM EQUATIONS OF MOTION	C0149700
C	USE FIXED STEP FOURTH ORDER RUNGE KUTTA	C0149800
	CONTINUE	C0149900
C	DEGREES OF FREEDOM MAY BE UNCAGED ONLY AT THE START OF	C0150000
C	AN INTEGRATION STEP, CHECK IN UNCAGE IF IT IS TIME TO	C0150100
C	UNCAGE, SCG.EQ.0 IMPLIES NO MORE UNCAGING TO BE DONE	C0150200
C	NOTE: INITIAL UNCAGING VELOCITY = 0, IMPULSE EFFECTS	C0150300
C	HAVE NOT BEEN CODED IN PROGRAM	C0150400
	IF(SCG.EQ.0) GO TO 3	C0150500
	CALL UNCAGE(SCG,SC,T,TUG)	C0150600
	CONTINUE	C0150700
	CALL RUNGE(T,H,Y,YC,NEG,NTG,TEM)	C0150800
		C0150900
C	SUBROUTINE RUNGE CALLS DYN IN WHICH ALL SYSTEM DIFFERENTIAL	C0151000
C	EQUATIONS ARE SETUP AND PUT IN THE YD ARRAY	C0151100
		C0151200
C	CHECK END OF RUN FLAG	C0151300
	IF(.NOT.FG1) GO TO 5	C0151400
C	OUTPUT COMPUTED PARAMETERS	C0151500
	IF FGS TRUE PRINT	C0151600
	IF FGS FALSE SKIP PRINT AND GO TO RUNGE	C0151700
C	DEFAULT FOR FGS IS TRUE BUT MAY BE OVERRIDDEN IN TORQUE	C0151800
	IF(.NOT.FGS) GO TO 2	C0151900
	CALL OUTPSP	C0152000
C	HAS END OF RUN FLAG BEEN SET IN OUTPSP BY USER?	C0152100
	IF(FG1) GO TO 2	C0152200
	CONTINUE	C0152300
		C0152400
C	SHOULD A RESTART TAPE BE MADE?	C0152500
	IF(LRTAPE) GO TO 6	C0152600
	CALL RSTART (2,Y,YC,NEG,TEM,62)	C0152700
	CALL OUTPSP	C0152800
	CONTINUE	C0152900
C	USER SHOULD WRITE OUTPUT DATA ON FILE 1 OF TAPE 11, RESTART DATA	C0153000
C	IS PUT INTO FILE 2 OF TAPE 11	C0153100
	REWIND 11	C0153200
		C0153300
C	GO TO 1 TO SEE IF ANOTHER N-BOD2 RUN FOLLOWS	C0153400
	GO TO 1	C0153500
	STOP	C0153600
	100 FORMAT (A4)	C0153700
	101 FORMAT ('1')	C0153800
	102 FORMAT (4X,17L1)	C0153900

C	SLERCUTINE DYN(Y,YD,NEQ)	C0200000
C		C0200100
C	DEFINES THE LOGICAL PATHS THROUGH THE SUBROUTINES USED TO	C0200200
C	SET UP THE SIMULTANEOUS DIFFERENTIAL EQUATIONS	C0200300
C	OF MOTION FOR THE COUPLED N-BODY SYSTEM	C0200400
C		C0200500
C		C0200600
C		C0200700
C		C0200800
C	IMPLICIT REAL*8(A-H,O-Z,*)	C0200900
C	LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLO, LEQU, LINIT(1)	C0201000
C	LCCICAL NSTART, LRTAPE	C0201100
C		C0201200
C		C0201300
C	INTEGER	C0201400
C	* ANGRK, CT1, CT2, CT3, CT4, CT5, FCON, PCON,	C0201500
C	* SCNDLM, SCN, SCRDM, SCR, SFKDM, SFK, SFR, SG,	C0201600
C	* SI, SIG, SIXDM, SIX, SKDM, SK, SL, SLK,	C0201700
C	* SMA, SMCDLM, SMC, SMV, SOK, SPIDUM, SPI, SUI,	C0201800
C	* SGL, SR, SSCN, SSIX, SVA, SVB, SVD, SVI,	C0201900
C	* SVM, SVP, SVQ, SXM, SXT, TORQ, SMAL, SEU,	C0202000
C	* SC, SCG, NFLXB, SFLX, SFXM, NMDS, SFCC, SCC,	C0202100
C	* IINIT(1), IZINIT(1), SD	C0202200
C		C0202300
C		C0202400
C		C0202500
C	REAL*8	C0202600
C	* ANGD (33), CNF (3,10), ETIC (3,10), ETMC (3,10),	C0202700
C	* FLD (3,20), FLE (3,3,20), FLH (3,3,20),	C0202800
C	* THAD (33), YMCD (3,2,11), RINIT (1), RZINIT(1)	C0202900
C		C0203000
C		C0203100
C	COMMON /CHEKS/ NSTART, LRTAPE	C0203200
C		C0203300
C		C0203400
C	COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RBLO(10)	C0203500
C		C0203600
C		C0203700
C	COMMON /INTG/ AWORK (200)	C0203800
C	* CT1, CT2, CT3, CT4	C0203900
C	* CT5, FCON (33), JCCN (10), LCON (22)	C0204000
C	* MC (10), NBI, NBD, NCTC	C0204100
C	* NFER, NFKC, NFRC, NLOR	C0204200
C	* NMV, NMD, NMDA, NSVP	C0204300
C	* NSVG, PCGN (11), SD, SFR (33)	C0204400
C	* SG, SI (55), SIG, SL	C0204500
C	* SLK (33), SMA (10), SOK (11), SQF (11)	C0204600
C	* SGL (11), SMV, SR, SSCN	C0204700
C	* SSIX, SVA, SVB, SVD	C0204800
C	* SVI, SVM, SVP (22), SVQ (33)	C0204900
C	* SXM (3,10), SXT, TORQ (97), SMAL	C0205000
C	* SEU, NTQ, SC (33), SCG	C0205100
C	* NFLXB, SFLX, SFXM (10), NMDS	C0205200
C	* SFCC, SCC (10)	C0205300

```

C
C
C          COMMEN /INTGZ/
* SCNDUM      , SCN   (5)      , SCRNUM      , SCR   (9)      ,
* SFKCLM      , SFK   (5)      , SIXDUM      , SIX   (9)      ,
* SKDLM       , SK    (5)      , SPIDUM      , SPI   (9)      ,
* SMCCLM      , SMC   (5)      ,
C
C          COMMEN /REAL/
* CA   (3,10) , CAC   (3,10) , CLM   (10) , CGMC (3,11) ,
* DCMC (3,11) , ETC   (3,11) , ETM   (33) , FUMC (3,11) ,
* GAM  (3,60) , F     (3,10) , HM    (3,10) , HMC  (3,10) ,
* HMCN (10)   , PHI   (3,11) , PLM   (10) , QF   (3,33) ,
* QFC  (3,33) , GL    (3,22) , QLC   (3,22) , RCMC (3,11) ,
* T     (3,33) , THA   (3,33) , THAD  (33) ,
* THADW (10) , THAW  (10) , XDIC  (3,3,60) , XI   (3,3,10) ,
* XIC   (3,3,10) , XMAS (10) , XMN   (33,33) , XMT  (3,3,10) ,
* TUG   (33) , FLA   (3,20) , FLB   (3,20) , FLC  (3,20) ,
* FLD   (3,3,20) , FLJ  (3,3,20) , CAD   (3,10) , XIU  (3,3,10) ,
C
* FLIRC (3,10) , FLCRC (3,10) , FLAC  (3,20) , FLQC  (3,20) ,
* FLCM  (20) , ZETA  (20) , FCF   (3,3,40) , FCK  (3,40) ,
* TIMEND
C
C          COMMEN /REALZ/
* CBDLM (1,3) , CB   (3,10) , CBCDUM(1,3) , CBC   (3,10) ,
* XMCCLM(1,1,9) , XMC (3,3,10) , CBN(3)
C
C          EQLVALENCE (ETM(1),THAD(1)) , (XMN(1,1),ANGD(1)) ,
* (XMN(1,3),YMCD(1,1,1)) , (XMN(1,6),CNF(1,1)) ,
* (XMN(1,8),ETIC(1,1)) , (XMN(1,10),ETMC(1,1)) ,
* (FLB(1,1),FLG(1,1)) , (FLE(1,1,1),FLD(1,1,1)) ,
* (FLH(1,1,1),FLJ(1,1,1)) ,
* (FG1,LINIT(1)) , (CA(1,1),RINIT(1)) ,
* (CBDUM(1,1),RZINIT(1)) , (AWGRK(1),IINIT(1)) ,
* (SCNDUM,IZINIT(1))
C
C          DIMENSION Y(NEQ),YC(NEQ)
LOGICAL LG(33)
C
C          COLNT TIMES THRU DYN
CT4 = CT4 + 1
CHECK FOR FIRST PASS
IF(CT4.GT.1) GO TO 1
C
C          MAKE USE OF
C          XMT = NOMINAL STATE TRANSFORMATION MATRICES
C          QF = FREE COORDINATE VECTORS
C          TFA = INITIAL ROTATION ABOUT QF VECTOR
C          TO COMPUTE THE INITIAL VALUES FOR THE COMPONENTS OF THE
C          TRANSFORMATION MATRICES.
C          XMC = TRANSFORMATION MATRIX WHICH TAKES VECTORS FROM
C          BODY FIXED COORDINATES TO COMPUTING FRAME COORDINATES
CALL TRANSIV(XMT,QF,THA,JCCN,PCGN,NBGD,RBLO,INERF,XMCDUM,XMC)
GO TO 2
C

```

C	1 CONTINUE	C0211400
C		C0211500
C	RETURN HERE AFTER INITIALIZATION PASS THROUGH DYN	C0211600
C	.. SORT GLT QUANTITIES IN ARRAY Y	C0211700
C	(CALL SETLP(Y, YC, NEQ, .TRUE.))	C0211800
C		C0211900
C	COMPLETE TRANSFORMATION MATRICES ARE NOT OBTAINED VIA INTEGRATION	C0212000
C	SIX OF NINE ELEMENTS IN EACH MATRIX HAVING LABEL IN SD OBTAINED	C0212100
C	BY INTEGRATION, REMAINING CNES BY ORTHOGONALITY	C0212200
C	MATRICES WITH LABELS NOT IN SD ASSUMED OBTAINABLE ALGEBRAICALLY	C0212300
C	BY SMALL ANGLE OR EULER ANGLE TECHNIQUES	C0212400
C	CALL TRAN	C0212500
C		C0212600
C	2 IF (CT4.NE.1) GO TO 3	C0212700
C	FIRST PASS THROUGH	C0212800
C	TAKE INTO ACCOUNT SET SVD, ONLY THOSE VECTORS AND DYADS WITH	C0212900
C	BODY LABELS IN SET SVD ARE TO BE TRANSFORMED	C0213000
C	COMPUTE OF ANG QL VECTORS NOT DEFINED BY INPUT USING	C0213100
C	VECTOR CROSS PRODUCT DEFINITION	C0213200
C	GET ALL VECTORS AND DYADS INTO COMPUTING REFERENCE FRAME	C0213300
C	SET UP SETS WHICH SPECIFY EXACTLY WHICH VECTORS TO BE	C0213400
C	TRANSFORMED FOR DC LOOPS IN TRANVD	C0213500
C	CALL VDIV	C0213600
C	SET UP INITIAL VALUES FOR EQUATIONS AS THEY GO INTO RUNGE	C0213700
C	(CALL EQIV(Y, NEQ))	C0213800
C	GO TO 4	C0213900
C		C0214000
C	NOT FIRST PASS THROUGH	C0214100
C	TRANSFORM ONLY SELECTED VECTORS AND DYADS	C0214200
C	MAKE USE OF XMC TO TRANSFORM ALL BODY 'FIXED' VECTORS AND DYAD INTO	C0214300
C	COMPUTING FRAME COORDINATES	C0214400
C	CAC = XMC*CA                  - CENTER OF MASS IN SET SVA	C0214500
C	CBC = XMC*CB                  - HINGE POINT IN SET SVB	C0214600
C	QFC = XMC*QF                  - FREE VECTOR IN SET SVQ	C0214700
C	QLC = XMC*QL                  - LOCKED VECTORS IN SET SVP	C0214800
C	XIC = XMC*XI*(XMC)**(-1)      - INERTIA DYAD IN SET SVI	C0214900
C	HMC = XMC*HM                  - WHEEL AXIS IN SET SVM	C0215000
C	3 CALL TRANVD	C0215100
C	4 CONTINUE	C0215200
C		C0215300
C	MAKE USE OF FREE VECTORS IN COMPUTING FRAME COORDINATES AND	C0215400
C	RATE ABOUT OR ALONG THEM TO DEFINE ALL RATE DEPENDENT TERMS	C0215500
C	CALL RATE	C0215600
C		C0215700
C	MAKE USE OF TRANSFORMED BODY FIXED VECTORS AND DYADS TO CONSTRUCT	C0215800
C	THE MATRIX OF INERTIA AND PSEUDO INERTIA TENSORS	C0215900
C	PASS ONE - COMPUTE ELEMENTS,	C0216000
C	SKIP ALL ZERO ELEMENTS	C0216100
C	TRUNCATE SUMMATIONS USE SET SPI	C0216200
C	THERE-AFTER - COMPUTE ELEMENTS,	C0216300
C	SKIP ALL ZERO ELEMENTS	C0216400
C	SKIP TIME CONSTANT ELEMENTS USE SET SXT	C0216500
C	TRUNCATE SUMMATIONS USE SET SPI	C0216600
C	CALL XDY	C0216700
C		C0216800
C	MAKE USE OF VELOCITY AND BODY FIXED VECTORS TO COMPUTE GYROSCOPIC	C0216900
C	CROSS COUPLING TERMS	C0217000
C	INERTIA CROSS COUPLING TRUNCATE ACCORDING TO SIX(I)	C0217100
C	CENTRIPITAL CROSS COUPLING TRUNCATE ACCORDING TO SCN(I)	C0217200
C	CORIOLIS CROSS COUPLING TRUNCATE ACCORDING TO SCR(I)	C0217300

C	MOMENTUM WHEEL COUPLING NOT TRUNCATED	C0217400
C	FLEXIBLE BODY EFFECTS NOT TRUNCATED SET SFLX	C0217500
C	CALL ETA	C0217600
C		C0217700
C	MAKE USE OF POSITION AND RATE INFORMATION TO COMPUTE	C0217800
C	ALL NON-GYROSCOPIC TORQUES	C0217900
C	NOTE - SUBROUTINE TORQUE IS USER DEFINED (EMPTY IF NOT)	C0218000
C	CALL TORQUE(Y,YD,NEG)	C0218100
C		C0218200
C	MAKE USE OF FREE COORDINATE VECTOR TO DOT VECTOR-CYADIC EQUATION	C0218300
C	OF MOTION TO GET ACCELERATIONS ABOUT FREE COORDINATE AXES	C0218400
C	ALSO SET UP AND EXPAND EQUATIONS TO ACCOUNT FOR VARIABLE SPEED	C0218500
C	MOMENTUM WHEELS AND FLEXIBLE BODY EFFECTS	C0218600
C	CALL GFDDT	C0218700
C	EXIT FROM QFDDT WITH EQUATIONS OF MOTION IN SCALAR FORM	C0218800
C		C0218900
C	REDUCE THE GFDDT EQUATIONS TO OBTAIN THADD	C0219000
C	ENTER SIMQ WITH ELEMENTS XMN AND ETM OBTAINED IN QFDDT.	C0219100
C	EXIT WITH ACCELERATIONS THADD, XMN DESTROYED IN SIMQ	C0219200
C	EQUIVALENCE PUTS THADD AND ETM IN SAME STORAGE LOCATION	C0219300
C		C0219400
C	CHECK TO SEE IF ANY DEGREES OF FREEDOM CAGED	C0219500
C	IF(SCG.NE.0) GO TO 5	C0219600
C	NONE CAGED.	C0219700
C	N = NFER+NMV+NMDS	C0219800
C	CALL SIMQ(XMN,THADD,N,33)	C0219900
C	GO TO 6	C0220000
C	5 CONTINUE	C0220100
C	ONE OR MORE CAGED DEGREES OF FREEDOM	C0220200
C	DELETE AND RENUMBER ROWS AND COLS OF XMN,ETM IN COMPRS	C0220300
C	SOLVE REDUCED SET OF EQUATIONS IN SIMQ	C0220400
C	RESTRUCTURE THADD ARRAY PLUGGING IN ZEROS IN UMPRS	C0220500
C	N = NFER+NMV+NMDS-SCG	C0220600
C	CALL COMPRS(XMN,THADD,N,SC,SCG,LG)	C0220700
C	CALL SIMQ(XMN,THADD,N,33)	C0220800
C	CALL UNPRS(THADD,N,SCG,LG)	C0220900
C	6 CONTINUE	C0221000
C		C0221100
C	DEFINE DIRECTION COSINE EQUATIONS IN ACCORDANCE WITH SET SD	C0221200
C	CALL DCT	C0221300
C		C0221400
C	DEFINE ANGULAR POSITION EQUATIONS ACCORDING TO SFR(I) AND SMA(I)	C0221500
C	CALL ANGLE	C0221600
C		C0221700
C	PUT ALL FIRST ORDER EQUATIONS IN ONE DIMENSIONAL ARRAY ACCEPTABLE	C0221800
C	TO INTEGRATION ROUTINE FUNGE	C0221900
C	CALL SETLP(Y,YD,NEG,.FALSE.)	C0222000
C	IF(T.GE.TIMEND) GO TO 500	C0222100
C	RETURN	C0222200
C		C0222300
C	500 CONTINUE	C0222400
C	FG1 = .FALSE.	C0222500
C	RETURN	C0222600
C	END	C0222700
C		C0300000

```

SUBROUTINE RSTART (J,Y,YC,NEC,TEM,*)                                C0300100
C                                                                    C0300200
C                                                                    C0300300
ZERC OUT ALL ARRAYS OR FILL THEM IN FROM THE RESTART TAPE         C0300400
C   AT THE END OF RUN CREATE A RESTART TAPE. TO DO THIS           C0300500
C   PUT THE RESTART DATA IN FILE 2 OF TAPE 11                     C0300600
C                                                                    C0300700
NOTE: APPROPRIATE JOB CONTROL LANGUAGE MUST ACCOMPANY             C0300800
C                                                                    C0300900
SUGGESTION: FILE 1 ON TAPE 10 PREVIOUS OUTPUT DATA               C0301000
C   FILE 2 ON TAPE 10 RESTART DATA                               C0301100
C   WITH A UTILITY ROUTINE COPY FILE 1 OF TAPE 10                 C0301200
C   ONTO FILE 1 OF TAPE 11, THEN PUT NEW OUTPUT DATA            C0301300
C   RIGHT AFTER IT WITHOUT AN EOF MARK, TAPE 11 WILL THEN        C0301400
C   HAVE IN FILE 1 A CONTINUOUS RECORD OF THE OUTPUT              C0301500
C   DATA FROM T=0                                                C0301600
C   AT END OF RUN NEW RESTART DATA PUT AFTER AN EOF MARK        C0301700
C   ON TAPE 11; IT WILL THEN BE IN FILE 2 OF TAPE 11            C0301800
C   OLD TAPE 10 WILL BE UNDISTURBED                               C0301900
C                                                                    C0302000
IMPLICIT REAL*8(A-H,O-Z,*)                                       C0302100
C                                                                    C0302200
LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLO, LEQU, LINIT(1)     C0302300
LOGICAL                               NSTART, LRTAPE                C0302400
C                                                                    C0302500
INTEGER                                                                C0302600
* ANCRK, CT1, CT2, CT3, CT4, CT5, FCON, PCON,                    C0302700
* SCNDUM, SCN, SCR, SCDUM, SFR, SFC, SG,                         C0302800
* SI, SIG, SIXDUM, SIX, SKDUM, SK, SL, SLK,                       C0302900
* SMA, SMCDUM, SMC, SMV, SCK, SPIDUM, SPI, SQF,                  C0303000
* SGL, SR, SSCN, SSIX, SVA, SVE, SVD, SVI,                       C0303100
* SVM, SVP, SVQ, SXM, SXT, TURQ, SMAL, SEU,                      C0303200
* SC, SCG, NFLXB, SFLX, SFXM, NMUDS, SFCC, SCC,                  C0303300
* IINIT(1), IZINIT(1), SD                                        C0303400
C                                                                    C0303500
C                                                                    C0303600
REAL*8                                                                C0303700
* ANGC (33), CNF (3,10), ETIC (3,10), ETMC (3,10),              C0303800
* FLO (3,20), FLE (3,3,20), FLH (3,3,20),                        C0303900
* THADD (33), YMCD (3,2,11), RINIT (1), RZINIT(1)                C0304000
C                                                                    C0304100
C                                                                    C0304200
COMMON /CHEKS/                               NSTART, LRTAPE       C0304300
C                                                                    C0304400
C                                                                    C0304500
COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RBLO(10)         C0304600
C                                                                    C0304700
C                                                                    C0304800
COMMON /INTG/          ANCRK (200),                               C0304900
* CT1, CT2, CT3, CT4,                                           C0305000
* CT5, FCON (33), JCCN (10), LCON (22),                          C0305100
* MC (10), NB1, NBD, NCTC,                                       C0305200
* NFER, NFKC, NFRC, NLOR,                                       C0305300
* NMV, NMO, NMOA, NSVP,                                         C0305400
* NSVG, PGCN (11), SD (11), SFR (33),                            C0305500
* SG, SI (55), SIG, SL,                                         C0305600
* SLK (33), SMA (10), SOK (11), SQF (11),                         C0305700
* SGL (11), SMV, SR, SSCN,                                       C0305800
* SSIX, SVA, SVB, SVD,                                           C0305900
* SVI, SVM, SVP (22), SVG (33),                                  C0306000

```

```

* SXM (3,10) , SXT , TORQ (97) , SMAL , C0306100
* SEU , NTQ , SC (33) , SCG , C0306200
* NFLXB , SFLX , SFXM (10) , NMUDS , C0306300
* SFCC , SCC (10) , C0306400
C
C
COMMON /INTGZ/
* SCNCUM , SCN (9) , SCRUM , SCR (9) , C0306800
* SFKUM , SFK (9) , SIXUM , SIX (9) , C0306900
* SKDUM , SK (9) , SPIDUM , SPI (9) , C0307000
* SMCCUM , SMC (9) , C0307100
C
C
COMMON /REAL/
* CA (3,10) , CAC (3,10) , CLM (10) , GCMC (3,11) , C0307500
* DCMC (3,11) , ETC (3,11) , ETM (33) , FCMC (3,11) , C0307600
* GAM (3,66) , F , HM (3,10) , HMC (3,10) , C0307700
* HMQM (10) , PHI (3,11) , PLM (10) , QF (3,33) , C0307800
* QFC (3,33) , GL (3,22) , QLC (3,22) , RCMC (3,11) , C0307900
* T , THA (33) , THAD (33) , C0308000
* THACW (10) , THAW (10) , XDIC (3,3,66) , XI (3,3,10) , C0308100
* XIC (3,3,10) , XMAS (10) , XMN (33,33) , XMT (3,3,10) , C0308200
* TUG (33) , FLA (3,20) , FLB (3,20) , FLC (3,20) , C0308300
* FLD (3,3,20) , FLJ (3,3,20) , CAD (3,10) , XID (3,3,10) , C0308400
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLQC (3,20) , C0308500
* FLOW (20) , ZETA (20) , FCF (3,3,40) , FCK (3,40) , C0308600
* TIMEND , C0308700
C
C
COMMON /REALZ/
* CEDLM (1,3) , CE (3,10) , CBDUM(1,3) , CBC (3,10) , C0308800
* XMCLM(1,1,9) , XMC (3,3,10) , CBN(3) , C0308900
C
COMMON /SATELL/ DUMMY(1000)
C
EQUIVALENCE (ETM(1),THAD(1)) , (XMN(1,1),ANGD(1)) , C0309600
* (XMN(1,3),YMCD(1,1,1)) , (XMN(1,6),CNF(1,1)) , C0309700
* (XMN(1,8),ETIC(1,1)) , (XMN(1,10),ETMC(1,1)) , C0309800
* (FLB(1,1),FLQ(1,1)) , (FLE(1,1,1),FLD(1,1,1)) , C0309900
* (FLH(1,1,1),FLJ(1,1,1)) , C0310000
* (FG1,LINIT(1)) , (CA(1,1),RINIT(1)) , C0310100
* (CBDUM(1,1),RZINIT(1)) , (AWORK(1),IINIT(1)) , C0310200
* (SCNDUM,IZINIT(1)) , C0310300
C
C
DIMENSION Y(160),YD(160),TEM(2,160)
C
COMMON BLOCK SIZES
IDEM4 = 16 C0310400
IDEM5 = 724 C0310500
IDEM6 = 70 C0310600
IDEM7 = 4354 C0310700
IDEM8 = 168 C0310800
IDEM9 = 1000 C0310900
C
GO TO (1,2),J C0311000
C
INITIALIZE ALL STORAGE LOCATIONS TO ZERO C0311100
1 DO 3 I=1,IDEM7 C0311200
3 FINIT(I) = 0.00 C0311300
C0311400
C0311500
C0311600
C0311700
C0311800
C0311900
C0312000

```



DO 4 I=1, IDEM5	C0312100
4 DUMMY(I) = 0.D0	C0312200
DO 10 I=1, IDEM8	C0312300
10 RZINIT(I) = 0.D0	C0312400
DO 11 I=1, IDEM5	C0312500
11 IINIT(I) = 0	C0312600
DO 12 I=1, IDEM6	C0312700
12 IZINIT(I) = 0	C0312800
DO 13 I=1, IDEM4	C0312900
13 LINIT(I) = .TRUE.	C0313000
DO 14 I=1, 160	C0313100
Y(I) = 0.D0	C0313200
YD(I) = 0.D0	C0313300
TEM(1,I) = 0.D0	C0313400
14 TEM(2,I) = 0.D0	C0313500
IF(.NOT.NSTART) RETURN	C0313600
C	C0313700
C RESTART RUN LOAD ALL COMMON BLOCKS AND LOCAL ARRAYS FROM	C0313800
C THE RESTART TAPE, TAPE 10 FILE 2	C0313900
READ(10,102) Y	C0314000
READ(10,102) YD	C0314100
READ(10,101) NEG	C0314200
READ(10,102) (DUMMY(I), I=1, IDEM9)	C0314300
READ(10,102) (RINIT(I), I=1, IDEM7)	C0314400
READ(10,102) (RZINIT(I), I=1, IDEM8)	C0314500
READ(10,101) (IINIT(I), I=1, IDEM5)	C0314600
READ(10,101) (IZINIT(I), I=1, IDEM6)	C0314700
READ(10,101) (LINIT(I), I=1, IDEM4)	C0314800
NSTART = .FALSE.	C0314900
FG1 = .TRUE.	C0315000
C UPDATE TERMINATION TIME	C0315100
READ 103, TIMEND	C0315200
C ALL DATA REQUIRED TO RESUME COMPUTATION HAS BEEN INPUTTED	C0315300
C GO TO THE START OF THE INTEGRATION LOOP IN MAIN	C0315400
RETURN	C0315500
C	C0315600
C	C0315700
C PUT EOF MARK ON TAPE 11 TO SEPARATE RESTART DATA FROM OUTPUT DATA	C0315800
C IN FILE 2 OF TAPE 11 PUT RESTART DATA	C0315900
2 ENC FILE 11	C0316000
WRITE(11,102) Y	C0316100
WRITE(11,102) YD	C0316200
WRITE(11,101) NEG	C0316300
WRITE(11,102) (DUMMY(I), I=1, IDEM9)	C0316400
WRITE(11,102) (RINIT(I), I=1, IDEM7)	C0316500
WRITE(11,102) (RZINIT(I), I=1, IDEM8)	C0316600
WRITE(11,101) (IINIT(I), I=1, IDEM5)	C0316700
WRITE(11,101) (IZINIT(I), I=1, IDEM6)	C0316800
WRITE(11,101) (LINIT(I), I=1, IDEM4)	C0316900
C REST IN PEACE ALL DATA NEEDED TO RESTART JOB IS ON FILE 2, TAPE 11	C0317000
C FURTHERMORE OLD RESTART TAPE, TAPE 10 HAS NOT BEEN DESTROYED	C0317100
C	C0317200
101 FORMAT (16Z8)	C0317300
102 FORMAT (E216)	C0317400
103 FORMAT(D15.5)	C0317500
RETURN	C0317600
ENC	C0317700

```

C                                     CC400000
C SUBROUTINE INBS                                     CC400100
C                                     CC400200
C                                     CC400300
C ACCEPTS ALL INFORMATION NEEDED TO DEFINE BASIC N-BODY SYSTEM CC400400
C   1) TOPOLOGY                                     CC400500
C   2) INERTIA CHARACTERISTICS                       CC400600
C   3) GEOMETRIC CHARACTERISTICS                     CC400700
C   4) KINEMATIC TRANSFORMATION                      CC400800
C   5) MOMENTUM WHEELS, CYCSTATS                     CC400900
C   6) INITIAL CONDITIONS                             CC401000
C   7) FREE AND LOCKED COORDINATE AXES               CC401100
C   8) INTEGRATION STEP SIZE                         CC401200
C                                     CC401300
C ***** INBS INPUT DATA SETUP *****           CC401400
C                                     FORMATS CC401500
C                                     +CCDES CC401600
C                                     ***** CC401700
C                                     CC401800
C NBOD                                         A 100 CC401900
C                                     CC402000
C ***** NBOD SETS OF THE FOLLOWING CARDS. ONE PER BODY ***** CC402100
C *          N          MESS(J)          B 105* CC402200
C *   FBLC(N)          JCCN(N)          PCCN(N)          XMAS(N)      C 101* CC402300
C *   XI(1,1,N)        XI(1,2,N)        XI(1,3,N)          D 102* CC402400
C *   XI(2,1,N)        XI(2,2,N)        XI(2,3,N)          D 102* CC402500
C *   XI(3,1,N)        XI(3,2,N)        XI(3,3,N)          D 102* CC402600
C *   XMT(1,1,N)       XMT(1,2,M)       XMT(1,3,N)       E 102* CC402700
C *   XMT(2,1,N)       XMT(2,2,N)       XMT(2,3,N)       E 102* CC402800
C *   XMT(3,1,N)       XMT(3,2,N)       XMT(3,3,N)       E 102* CC402900
C *   CA(1,N)          CA(2,N)          CA(3,N)          F 102* CC403000
C *   CB(1,N)          CB(2,N)          CB(3,N)          G 102* CC403100
C *   CF(1,M)          CF(2,M)          CF(3,M)          H 102* CC403200
C *   QF(1,M+2)        QF(2,M+2)        QF(3,M+2)        H 102* CC403300
C *   THA(M)           THA(M+1)         THA(M+2)        H 102* CC403400
C *   THAD(M)          THAD(M+1)        THAD(M+2)        H 102* CC403500
C *   QF(1,M)          QF(2,M)          QF(3,M)          I 102* CC403600
C *   QF(1,M+1)        QF(2,M+1)        QF(3,M+1)        I 102* CC403700
C *   THA(M)           THA(M+1)         I 102* CC403800
C *   THAD(M)          THAD(M+1)        I 102* CC403900
C *   CF(1,M)          CF(2,M)          QF(3,M)          J 102* CC404000
C *   CL(1,L)          CL(2,L)          QL(3,L)          J 102* CC404100
C *   THA(M)           J 102* CC404200
C *   THAD(M)          J 102* CC404300
C *   QL(1,L)          CL(2,L)          QL(3,L)          K 102* CC404400
C *   CL(1,L+1)        QL(2,L+1)        QL(3,L+1)        K 102* CC404500
C ***** END OF SET OF BODY N DESCRIPTIVE CARDS *****           CC404600
C                                     CC404700
C PCCN(NBOD+1)                                     L 100 CC404800
C THAD(M) THAD(M+1) THAD(M+2) M 102 CC404900
C THAD(M) THAD(M+1) N 102 CC405000
C THAD(M) O 102 CC405100
C NMC P 100 CC405200
C CC405300
C ***** NMO SETS OF THE FOLLOWING CARDS. ONE PER WHEEL ***** CC405400
C *   NO(I)            HM(1,I)            HM(2,I)            HM(3,I)      Q 104* CC405500
C *   PLM(I)           MESS(J)           R 106* CC405600
C *   THAW(I)          THADW(I)          S 102* CC405700
C ***** END OF SET OF WHEEL I DESCRIPTIVE CARDS *****           CC405800
C                                     CC405900

```

```

C                               TIMEND                               T 102 C0406000
C                               C0406100
C ***** END OF DATA CALLED FOR BY INBS ***** C0406200
C                               C0406300
C                               CCDE LIST                           C0406400
C                               C0406500
C A - ALWAYS READ, FIRST CARD READ BY INBS                       C0406600
C                               NBOD=TOTAL NUMBER OF BODIES (RIGID+FLEXIBLE+POINT MASSES) C0406700
C                               C0406800
C E - FIRST CARD FOR EACH SET OF BODY DESCRIPTION CARDS          C0406900
C                               N=BODY NUMBER (SETS TO BE READ SEQUENTIALLY N=1,2,... C0407000
C                               MESS(J)=ANY ALPHANUMERIC MESSAGE WILL BE PRINTED AS A LEADING C0407100
C                               FCR BODY N IN THE INPUT DATA ECHO, PRINTED BY INBS C0407200
C                               C0407300
C C - SECOND CARD FOR EACH SET OF BODY DESCRIPTION CARDS         C0407400
C                               JCCN(N)=BODY LABEL OF BODY CONTIGUOUS TO AND INBOARD OF C0407500
C                               BODY N, HINGE POINT BETWEEN BODIES JCCN(N) AND N C0407600
C                               IS DEFINED AS HINGE POINT N-1, IF N=1 THEN JCCN(1)=0 C0407700
C                               AND HINGE POINT 0 IS THE CENTER OF MASS OF BODY 1 C0407800
C                               RBLC(N)=.TRUE. IF BODY N A RIGID OR FLEXIBLE BODY C0407900
C                               .FALSE. IF BODY N A POINT MASS C0408000
C                               PCCN(N)=TOTAL NUMBER OF RIGIDLY CONSTRAINED ROTATIONAL C0408100
C                               DEGREES OF FREEDOM AT HINGE POINT N-1 OF BODY N C0408200
C                               XMAS(N)=TOTAL MASS OF BODY N AND ALL IMBEDDED WHEELS(IF ANY) C0408300
C                               C0408400
C D - DELETE THESE 3 CARDS IF BODY N A POINT MASS                C0408500
C                               XII(I,J,N)=INERTIA TENSOR OF BODY N WITH ALL DESPUN WHEELS, C0408600
C                               INCLUDED(IF ANY) ABOUT ITS OWN CENTER OF MASS AND C0408700
C                               RELATIVE TO THE BODY N FIXED REFERENCE FRAME, IF BODY C0408800
C                               N IS FLEXIBLE INPUT INERTIA TENSOR IN UNDEFORMED C0408900
C                               ZERO INTERNAL STRESS STATE C0409000
C                               C0409100
C E - DELETE THESE 3 CARDS IF BODY N A POINT MASS                C0409200
C                               XMT(I,J,N)=TRANSFORMATION MATRIX WHICH TAKES VECTORS FROM THE C0409300
C                               BODY N TO THE BODY JCCN(N) FIXED REFERENCE FRAMES C0409400
C                               FOR ZERO RELATIVE ANGULAR ROTATION BETWEEN THE ECCIES C0409500
C                               IF BODY N IS A POINT MASS THE BODY N FIXED REFERENCE C0409600
C                               FRAME AXES ARE RESPECTIVELY PARALLEL TO THOSE OF THE C0409700
C                               BODY JCCN(N) FIXED REFERENCE FRAME C0409800
C                               IF BODY N IS A FLEXIBLE BODY, IT IS ASSUMED TO BE C0409900
C                               CLAMPED IN TRANSLATION AND ROTATION AT THE ORIGIN C0410000
C                               OF THE BODY N FIXED REFERENCE FRAME. C0410100
C                               C0410200
C F - ALWAYS READ                                                C0410300
C                               CA(I,N)=CENTER OF MASS VECTOR COMPONENTS OF THE VECTOR FROM C0410400
C                               THE HINGE POINT N-1(ORIGIN OF BODY N FIXED REFERENCE) C0410500
C                               TO THE CENTER OF MASS OF BODY N(UNDEFORMED POSITION C0410600
C                               IF BODY N FLEXIBLE) RELATIVE TO BODY N FIXED C0410700
C                               REFERENCE FRAME, FOR N=1 CA(I,N)=0 I=1,2,3 SINCE BY C0410800
C                               DEFINITION THE HINGE POINT OF BODY 1 IS THE CENTER C0410900
C                               OF MASS OF BODY 1 C0411000
C                               C0411100
C                               C0411200
C                               CB(I,N)=HINGE VECTOR, COMPONENTS OF THE VECTOR FROM HINGE C0411300
C                               POINT JCCN(N)-1 TO HINGE POINT N-1 RELATIVE TO THE C0411400
C                               BODY JCCN(N) FIXED REFERENCE. FOR N=1 IT IS THE C0411500
C                               VECTOR FROM THE INERTIAL ORIGIN TO THE CENTER OF C0411600
C                               MASS OF BODY 1, WHICH IS THE HINGE POINT OF BODY 1, C0411700
C                               RELATIVE TO THE INERTIAL REFERENCE FRAME, C0411800
C                               C0411900

```



C									CC418000
C									CC418100
	INTEGER								CC418200
	* AWRK	, CT1	, CT2	, CT3	, CT4	, CT5	, FCCN	, FCON	, CC418300
	* SCNCLM	, SCN	, SCR	, SFCR	, SFK	, SFR	, SG		, CC418400
	* SI	, SIG	, SIX	, SKDUM	, SK	, SL	, SLK		, CC418500
	* SMA	, SMC	, SMV	, SOK	, SPIDUM	, SPI	, SQF		, CC418600
	* SGL	, SR	, SSCN	, SSIX	, SVA	, SVB	, SVD	, SVI	, CC418700
	* SVM	, SVP	, SVQ	, SXM	, SXT	, TORQ	, SMAL	, SEU	, CC418800
	* SC	, SCG	, NFLXB	, SFLX	, SFXM	, NMQDS	, SFCC	, SCC	, CC418900
	* IINIT(1)		, IZINIT(1)		, SD				, CC419000
C									CC419100
C									CC419200
	REAL*8								CC419300
	* ANGE	(33)	, CNF	(3,10)	, ETIC	(3,10)	, ETMC	(3,10)	, CC419400
	* FLQ	(3,20)	, FLE	(3,3,20)	, FLH	(3,3,20)			, CC419500
	* THADD	(33)	, YMCD	(3,2,11)	, RINIT	(1)	, RZINIT(1)		, CC419600
C									CC419700
C									CC419800
	COMMON /CHECKS/								CC419900
									CC420000
C									CC420100
	COMMON /LOGIC/								CC420200
									CC420300
C									CC420400
	COMMON /INTG/								CC420500
	* CT1		, CT2		, CT3		, CT4		, CC420600
	* CT5		, FCCN	(33)	, JCGN	(10)	, LCGN	(22)	, CC420700
	* MC	(10)	, NEI		, NBD		, NCTC		, CC420800
	* NFER		, NFKC		, NFRC		, NLUR		, CC420900
	* NMV		, NMD		, NMOA		, NSVP		, CC421000
	* NSVG		, PCGN	(11)	, SD		, SFR	(33)	, CC421100
	* SG		, SI	(55)	, SIG		, SL		, CC421200
	* SLK	(33)	, SMA	(10)	, SCK	(11)	, SQF	(11)	, CC421300
	* SGL	(11)	, SMV		, SR		, SSCN		, CC421400
	* SSIX		, SVA		, SVB		, SVD		, CC421500
	* SVI		, SVM		, SVP	(22)	, SVQ	(33)	, CC421600
	* SXM	(3,10)	, SXT		, TORQ	(97)	, SMAL		, CC421700
	* SEU		, NTQ		, SC	(33)	, SCG		, CC421800
	* NFLXB		, SFLX		, SFXM	(10)	, NMQDS		, CC421900
	* SFCC		, SCC	(10)					, CC422000
C									CC422100
C									CC422200
	COMMON /INTGZ/								CC422300
	* SCNCLM		, SCN	(5)	, SCR	(9)	, SFCR	(9)	, CC422400
	* SFCR		, SFK	(5)	, SIX	(9)	, SPI	(9)	, CC422500
	* SKDUM		, SK	(5)	, SPIDUM				, CC422600
	* SMC		, SMC	(5)					, CC422700
C									CC422800
C									CC422900
	COMMON /REAL/								CC423000
	* CA	(3,10)	, CAC	(3,10)	, CLM	(10)	, CCMC	(3,11)	, CC423100
	* DCMC	(3,11)	, ETC	(3,11)	, ETM	(33)	, FCMC	(3,11)	, CC423200
	* GAM	(3,10)	, HMC	(3,10)	, HMC	(3,10)	, HMC	(3,10)	, CC423300
	* HVMC	(10)	, PHI	(3,11)	, PLM	(10)	, QFC	(3,33)	, CC423400
	* QFC	(3,33)	, GL	(3,22)	, QLC	(3,22)	, RCMC	(3,11)	, CC423500
	* T				, THA	(33)	, THAD	(33)	, CC423600
	* THADW	(10)	, THAW	(10)	, XDIC	(3,3,66)	, XI	(3,3,10)	, CC423700
	* XIC	(3,3,10)	, XMAS	(10)	, XMN	(33,33)	, XMT	(3,3,10)	, CC423800
	* TLG	(33)	, FLA	(3,20)	, FLB	(3,20)	, FLC	(3,20)	, CC423900

```

* FLD (3,3,20), FLJ (3,3,20), CAU (3,10) , XID (3,3,10). C0424000
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLQC (3,20) . C0424100
* FLCM (20) , ZETA (20) , FCF (3,3,40), FCK (3,40) . C0424200
* TIMEND C0424300
C C0424400
C C0424500
COMMON /REALZ/ C0424600
* CBDUM (1,3) , CP (3,10) , CBCDUM(1,3) , CBC (3,10) , C0424700
* XMCCLM(1,1,9) , XMC (3,3,10), CBN(3) C0424800
C C0424900
C C0425000
EQUIVALENCE (ETM(1),THACC(1)) ,(XMN(1,1),ANGD(1)) , C0425100
* (XMN(1,3),YPCD(1,1,1)) ,(XMN(1,6),CNF(1,1)) , C0425200
* (XMN(1,8),ETIC(1,1)) ,(XMN(1,10),ETMC(1,1)) , C0425300
* (FLB(1,1),FLC(1,1)) ,(FLE(1,1,1),FLD(1,1,1)) . C0425400
* (FLH(1,1,1),FLJ(1,1,1)) , C0425500
* (FG1,LINIT(1)) ,(CA(1,1),RINIT(1)) , C0425600
* (CBDUM(1,1),RZINIT(1)) ,(AWORK(1),IINIT(1)) , C0425700
* (SCNDLM,IZINIT(1)) C0425800
C C0425900
C C0426000
C C0426100
C C0426200
DIMENSION TEM1(3,3),TEM2(3,3) C0426300
INTEGER MESS(18) C0426400
C C0426500
C C0426600
C C0426700
L = 1 C0426800
NBCD = NUMBER OF EDDIES C0426900
READ 100, NBOD C0427000
NB1 = NBOD+1 C0427100
PRINT 199 C0427200
PRINT 200, NBOD C0427300
PRINT 254 C0427400
PRINT 255 C0427500
PRINT 256 C0427600
PRINT 257 C0427700
PRINT 258 C0427800
PRINT 259 C0427900
PRINT 226 C0428000
C READ INPLT FOR EACH BODY C0428100
DO 1 K=1,NBOD C0428200
K1 = K-1 C0428300
C MESS(J) = ALPHANUMERIC DESCRIPTION OF BODY N TO BE PRINTED C0428400
READ 105, N, (MESS(J),J=1,18) C0428500
IF(N.EQ.K) GO TO 24 C0428600
PRINT 247 C0428700
C TYPE 247 C0428800
RETURN C0428900
24 CONTINUE C0429000
C RBLO(K) = TRUE IF BODY K RIGID BODY C0429100
C = FALSE IF BODY K POINT MASS C0429200
C JCCN(K) = BODY LABEL TO WHICH BODY K IS ATTACHED AT HINGE K=1 C0429300
C FCCN(K) = NUMBER OF CONSTRAINED AXES AT HINGE POINT K-1 C0429400
C XMAS(K) = TOTAL MASS OF EDDY K PLUS MOMENTUM WHEELS C0429500
READ 101, RBLO(K),JCCN(K),PCON(K),XMAS(K) C0429600
IF(RELD(K)) GO TO 2 C0429700
PRINT 201, K, (MESS(J),J=1,18) C0429800
PRINT 206,K,XMAS(K) C0429900

```

	GO TO 3	CC430000
2	FRINT 2C2, K, (MESS(J),J=1,18)	CC430100
	FRINT 2C6, K,XMAS(K)	CC430200
C	XI(I,J,K),I,J=1,3 = COMPONENTS OF INERTIA TENSOR OF BODY K WITH	CC430300
C	ALL DESPUN MOMENTUM WHEELS IN BODY K	CC430400
C	MASS RELATIVE TO BODY K FIXED FRAME	CC430500
C	COORDINATES (ABOUT THE COMPOSITE CENTER OF	CC430600
	READ 102, ((XI(I,J,K),J=1,3),I=1,3)	CC430700
	FRINT 216, (XI(1,J,K),J=1,3), K	CC430800
	FRINT 217, (XI(2,J,K),J=1,3)	CC430900
	FRINT 218, (XI(3,J,K),J=1,3)	CC431000
3	CONTINUE	CC431100
	IF(K.EQ.1) GO TO 4	CC431200
	FRINT 2C3, K,JCON(K),K1	CC431300
	GO TO 5	CC431400
4	FRINT 2C4	CC431500
	FRINT 2E2, NBDG	CC431600
5	CONTINUE	CC431700
	IF(.NOT.FBLO(K)) GO TO 8	CC431800
C	XMT(I,J,K),I,J=1,3 = COMPONENT OF TRANSFORMATION MATRIX BODY	CC431900
C	K TO BODY JCON(K) COORDINATES	CC432000
	READ 102, ((XMT(I,J,K),J=1,3),I=1,3)	CC432100
	GO TO 9	CC432200
6	XMT(1,1,K) = 1	CC432300
	XMT(1,2,K) = 0	CC432400
	XMT(1,3,K) = 0	CC432500
	XMT(2,1,K) = 0	CC432600
	XMT(2,2,K) = 1	CC432700
	XMT(2,3,K) = 0	CC432800
	XMT(3,1,K) = 0	CC432900
	XMT(3,2,K) = 0	CC433000
	XMT(3,3,K) = 1	CC433100
9	FRINT 219, (XMT(1,J,K),J=1,3)	CC433200
	FRINT 220, K,(XMT(2,J,K),J=1,3)	CC433300
	IF(K.EQ.1) GO TO 10	CC433400
	FRINT 222, JCON(K),(XMT(3,J,K),J=1,3)	CC433500
	GO TO 11	CC433600
10	FRINT 221, (XMT(3,J,K),J=1,3)	CC433700
11	CONTINUE	CC433800
C	CA(J,K),J=1,3 = COMPONENTS OF CENTER OF MASS FROM HINGE POINT K-1	CC433900
	READ 102, (CA(J,K),J=1,3)	CC434000
	FRINT 2C7, CA(1,K), K	CC434100
	FRINT 2C8, K1,CA(2,K)	CC434200
	FRINT 2C5, K,CA(3,K)	CC434300
C	CB(J,K),J=1,3 = COMPONENTS OF VECTOR FROM HINGE POINT (JCON(K)-1)	CC434400
C	HINGE POINT K-1 IN BODY JCON(K) COORDINATES	CC434500
	READ 102, (CB(J,K),J=1,3)	CC434600
	IF(K.NE.1) GO TO 6	CC434700
	FRINT 210, CB(1,K)	CC434800
	FRINT 211, CB(2,K)	CC434900
	FRINT 212, CB(3,K)	CC435000
	GO TO 7	CC435100
6	K2 = JCON(K)-1	CC435200
	FRINT 213, CB(1,K),JCON(K)	CC435300
	FRINT 214, K2,CB(2,K)	CC435400
	FRINT 215, K1,CB(3,K)	CC435500
7	CONTINUE	CC435600
C		CC435700
C	READ IN FREE AND LOCKED COORDINATE VECTORS ALONG WITH INITIAL CCND	CC435800
	IF(K.EQ.1) GO TO 12	CC435900

M = M + 3 - FCCN(K-1)	C0436000
L = L + FCCN(K-1)	C0436100
12 M1 = M+1	C0436200
M2 = M+2	C0436300
L1 = L+1	C0436400
L2 = L+2	C0436500
IF(RELC(K)) GO TO 20	C0436600
IASIGN = 1	C0436700
GO TO 21	C0436800
20 IASIGN = 0	C0436900
21 CONTINUE	C0437000
IGCTC = FCCN(K) + 1	C0437100
GO TO (13,14,15,17),IGCTC	C0437200
C	C0437300
C FCCN(K) = 0, THREE DEGREES OF FREEDOM	C0437400
13 READ 102, ((QF(J,I),J=1,3),I=M,M2,2)	C0437500
READ 102, (THA(I),I=M,M2)	C0437600
READ 102, (THAD(I),I=M,M2)	C0437700
PRINT 246, K1	C0437800
IF(RELC(K)) GO TO 31	C0437900
FCCN(M) = K	C0438000
FCCN(M1) = K	C0438100
GO TO 32	C0438200
31 FCCN(M) = JCCN(K)	C0438300
C CF(I,M1) COMPUTED FROM QF(1,M) AND QF(1,M2)	C0438400
C FCCN(M1) = -M HELPS LOGIC IN SUBROUTINE TRANVD	C0438500
FCCN(M1) = -M	C0438600
32 FCCN(M2) = K	C0438700
PRINT 228, QF(1,M),QF(1,M2)	C0438800
PRINT 229, M,QF(2,M),M2,QF(2,M2)	C0438900
IF(K.EQ.1) GO TO 29	C0439000
PRINT 230, FCCN(M),QF(3,M),FCCN(M2),QF(3,M2)	C0439100
GO TO 30	C0439200
29 PRINT 223, QF(3,M),FCCN(M2),QF(3,M2)	C0439300
30 CONTINUE	C0439400
PRINT 263, M1	C0439500
IF(IASIGN .EQ. 0 ) PRINT 224	C0439600
IF(IASIGN .EQ. 1 ) PRINT 243	C0439700
PRINT 225, (I,THA(I),I,THAD(I),I=M,M2)	C0439800
PRINT 226	C0439900
GO TO 1	C0440000
C	C0440100
C FCCN(K) = 1, TWO DEGREES OF FREEDOM	C0440200
14 READ 102, ((QF(J,I),J=1,3),I=M,M1)	C0440300
READ 102, THA(M),THA(M1)	C0440400
READ 102, THAD(M),THAD(M1)	C0440500
PRINT 227, K1	C0440600
IF(RELC(K)) GO TO 33	C0440700
FCCN(M) = K	C0440800
FCCN(M1) = K	C0440900
LCCN(L) = K	C0441000
GO TO 34	C0441100
33 FCCN(M) = JCON(K)	C0441200
C LCCN(L) = -M HELPS LOGIC IN SUBROUTINE TRANVD	C0441300
C LCCN(L) = -M	C0441400
34 PRINT 228, QF(1,M), QF(1,M1)	C0441500
PRINT 229, M,QF(2,M),M1,QF(2,M1)	C0441600
IF(K.EQ.1) GO TO 35	C0441700
	C0441800
	C0441900



PRINT 230, FCCN(M), QF(3,M), FCUN(M1), QF(3,M1)	CC442000
GO TO 36	CC442100
35 PRINT 223, QF(3,M), FCUN(M1), QF(3,M1)	CC442200
36 CONTINUE	CC442300
PRINT 231, L	CC442400
IF(IASIGN.EQ.0) PRINT 224	CC442500
IF(IASIGN.EQ.1) PRINT 243	CC442600
PRINT 225, (I, THA(I), I, THAD(I), I=M, M1)	CC442700
PRINT 226	CC442800
GO TO 1	CC442900
C	CC443000
C FCCN(K) = 2 ONE DEGREE OF FREEDOM	CC443100
15 READ 102, (QF(J,M), J=1,3)	CC443200
READ 102, (QL(J,L), J=1,3)	CC443300
READ 102, THA(M)	CC443400
READ 102, THAD(M)	CC443500
PRINT 232, K1	CC443600
IF(RELO(K)) GO TO 37	CC443700
FCCN(M) = K	CC443800
LCCN(L) = K	CC443900
LCCN(L1) = K	CC444000
GO TO 38	CC444100
37 FCCN(M) = JCCN(K)	CC444200
LCCN(L) = K	CC444300
LCCN(L1) = K	CC444400
38 PRINT 233, QF(1,M)	CC444500
PRINT 234, M, QF(2,M)	CC444600
IF(K.EQ.1) GO TO 39	CC444700
PRINT 235, FCUN(M), QF(3,M)	CC444800
GO TO 40	CC444900
39 PRINT 268, QF(3,M)	CC445000
40 PRINT 269, QL(1,L)	CC445100
PRINT 270, L, QL(2,L)	CC445200
PRINT 271, LCCN(L), QL(3,L)	CC445300
PRINT 231, L1	CC445400
IF(IASIGN.EQ.0) PRINT 224	CC445500
IF(IASIGN.EQ.1) PRINT 243	CC445600
PRINT 225, M, THA(M), M, THAD(M)	CC445700
PRINT 226	CC445800
GO TO 1	CC445900
C	CC446000
C FCCN(K) = 3, ZERO DEGREES OF FREEDOM	CC446100
17 READ 102, ((QL(J,I), J=1,3), I=L, L1)	CC446200
PRINT 272, K1	CC446300
LCCN(L) = K	CC446400
LCCN(L1) = K	CC446500
LCCN(L2) = K	CC446600
PRINT 236, QL(1,L), QL(1,L1)	CC446700
PRINT 237, L, QL(2,L), L1, QL(2,L1)	CC446800
PRINT 238, LCON(L), QL(3,L), LCON(L1), QL(3,L1)	CC446900
PRINT 231, L2	CC447000
PRINT 226	CC447100
1, CONTINUE	CC447200
C	CC447300
C READ IN TRANSLATIONAL VELOCITY CONDITIONS FOR HINGE POINT ZERO	CC447400
C RELATIVE TO INERTIAL ORIGIN	CC447500
READ 100, PCON(NBCD+1)	CC447600
NFR = M+3-PCON(NBCD)	CC447700
NLR = L+FCUN(NBCD) - 1	CC447800
PRINT 244	CC447900

PRINT 260,PCUN(NBCD*1)	CO448000
I3 = 3-PCUN(NBOD+1)	CO448100
IF(I3.EQ.0) GO TO 25	CO448200
C COMPUTE INITIAL DISPLACEMENT OF HINGE POINT ZERO FROM CB(1)	CO448300
C INFUTING THIS WOULD BE REDUNDANT INFORMATION	CO448400
C DO 16, I=NFR,NFR+I3-1	CO448500
ITEST=NFR+I3-1	CO448600
IF(ITEST.LT.NFR)GO TO 5000	CO448700
DO 16 I=NFR,ITEST	CO448800
18 THA(I) = CB(I+1-NFR,1)	CO448900
5000 CONTINUE	CO449000
READ 102,(THAD(I),I=NFR,ITEST)	CO449100
PRINT 261	CO449200
PRINT 243	CO449300
PRINT 225,(I,THA(I),I,THAD(I),I=NFR,ITEST)	CO449400
PRINT 253	CO449500
GO TO 26	CO449600
25 PRINT 248	CO449700
26 PRINT 264	CO449800
DO 22 I=1,3	CO449900
IF(I.GT.I3) GO TO 23	CO450000
CBN(I) = 0.00+00	CO450100
GO TO 22	CO450200
23 CBN(I) = CB(I,1)	CO450300
22 CONTINUE	CO450400
PRINT 265	CO450500
PRINT 266	CO450600
PRINT 267	CO450700
NFR = NFR+I3-1	CO450800
NLOR = NLOR+PCUN(NBCD+1)	CO450900
PRINT 253	CO451000
PRINT 249, NFR	CO451100
PRINT 250,NLOR	CO451200
PRINT 226	CO451300
C	CO451400
C READ IN MOMENTUM WHEEL DESCRIPTION	CO451500
C	CO451600
READ 103, NMD	CO451700
IF(NMD.EQ.0) GO TO 50	CO451800
PRINT 239, NMD	CO451900
DO 16 I=1,NMD	CO452000
READ 104, MO(I),(HM(J,I),J=1,3)	CO452100
READ 106, PLM(I),(MESS(J),J=1,16)	CO452200
READ 102, THAW(I),THADW(I)	CO452300
88 PRINT 285, I,MO(I),(MESS(J),J=1,16)	CO452400
PRINT 274, HM(1,I),MO(I)	CO452500
PRINT 275, HM(2,I)	CO452600
PRINT 276, I,HM(3,I)	CO452700
PRINT 286, PLM(I)	CO452800
PRINT 280, I	CO452900
PRINT 281, I,THAW(I)	CO453000
PRINT 282, I,THADW(I)	CO453100
HMCN(I) = PLM(I)*THADW(I)	CO453200
PRINT 284, HMCN(I)	CO453300
IF((I/2)*2.EQ.I) GO TO 15	CO453400
PRINT 253	CO453500
PRINT 253	CO453600
GO TO 16	CO453700
15 PRINT 226	CO453800
16 CONTINUE	CO453900

```

SC CONTINUE
C
C READ IN INTEGRATION STEP SIZE
READ 102, H, TIMEND
PRINT 253
PRINT 273, H
PRINT 253
PRINT 255, TIMEND
100 FORMAT (I5)
101 FORMAT (L5,2I5,D15.5)
102 FORMAT (3D15.5)
103 FORMAT (3L5)
104 FORMAT (I5,3D15.5)
105 FORMAT (I5,18A4)
106 FORMAT (D15.5,18A4)
199 FORMAT ('1',20(/))
200 FORMAT (22X,'INPUT DATA FOR ',I3,' BODIES (A CONSISTANT SET OF UNIC
*IS MUST BE USED) ',/)
201 FORMAT (5X,'BODY NUMBER',I3,' (POINT MASS)',18A4,/)
202 FORMAT (5X,'BODY NUMBER',I3,' (RIGID BODY)',18A4,/)
203 FORMAT (10X,'BODY',I3,' CONNECTED TO BODY',I3,' AT HINGE POINT',I3C0456000
1,/)
204 FORMAT (10X,'CENTER OF MASS OF BODY 1 CAN TRANSLATE AND ROTATE IN C0456200
*INERTIAL SPACE ')
205 FORMAT (10X,'MOTION CONSTRAINED ABOUT',I3,' AXES AT HINGE POINT',IC0456400
*3,' OF BODY',I3,/)
206 FORMAT (10X,'TOTAL MASS OF BODY ',I2,' =',D11.5,' (M) ',//)
207 FORMAT(10X,'COMPONENTS OF VECTOR FROM ',D15.5,' (BODY',I3,C0456700
*' FIXED ')
208 FORMAT (10X,' HINGE POINT',I3,' TO CENTER ',D15.5,' COORDC0456900
*INATES)')
209 FORMAT (10X,' OF MASS OF BODY',I3,11X,D15.5,10X,'(L)',//)
210 FORMAT (10X,'COMPONENTS OF VECTOR FROM ',D15.5,' (INERTIAL C0457200
*')
211 FORMAT (10X,' INERTIAL ORIGIN TO CENTER ',D15.5,' CGORDC0457400
*INATES)')
212 FORMAT (10X,' OF MASS OF BODY 1 ',D15.5,10X,'(L)',//)
213 FORMAT (10X,'COMPONENTS OF VECTOR FROM ',D15.5,' (BODY',I3,C0457700
*' FIXED ')
214 FORMAT (10X,' HINGE POINT',I3,' TO HINGE ',D15.5,' COCRDC0457900
*INATES) ')
215 FORMAT (10X,' POINT',I3,21X,D15.5,10X,'(L)',//)
216 FORMAT (26X,3D12.5,' BODY',I3,' COORDINATES ')
217 FORMAT (10X,'INERTIA TENSOR =',3D12.5,' UNITS ')
218 FORMAT (26X,3D12.5,5X,'(M*L**2) ',//)
219 FORMAT (10X,'TRANSFORMATION MATRIX ',3D12.5,5X,'DEFINES NOMINAL RC0458500
RELATIVE ')
220 FORMAT (10X,' BODY',I3,' TO =',3D12.5,12X,'ORIENTATION')C0458700
221 FORMAT (10X,' INERTIAL ',3D12.5,7X,'OF BODY FIXED AXEC0458800
*5',//)
222 FORMAT (10X,' BODY',I3,13X,3D12.5,7X,'OF BODY FIXED AXES',//)
223 FORMAT (12X,'INERTIALLY',9X,D12.5,6X,' IN BODY',I3,10X,D12.5,//)
224 FORMAT (10X,' INITIAL CONDITIONS ABOUT FREE AXES (RAD,RAD/SEC0459200
*C)',//)
225 FORMAT (15X,'THA(',I2,') =',D12.5,' THAD(',I2,') =',D12.5)
226 FORMAT ('1')
227 FORMAT (10X,' TWO DEGREES OF FREEDOM AT HINGE POINT ',I3,/)
228 FORMAT (10X,'COMPONENTS OF FREE ',D12.5,6X,'COMPONENTS OF FREE C0459700
*' ,D12.5)
229 FORMAT (10X,' VECTOR',I3,' FIXED =',D12.5,6X,' VECTOR',I3,' FIXC0459900

```

```

*ED      =',D12.5)                                C0460000
230 FORMAT (10X,' IN BODY',I3,10X,D12.5,6X,' IN BODY',I3,10X,D12.5,/) C0460100
231 FORMAT (10X,'COMPONENTS OF LOCKED VECTOR',I3,' DEFINED INTERNALLY C0460200
*EY VECTOR CROSS PRODUCT ',/) C0460300
232 FORMAT (10X,'ONE DEGREE OF FREEDOM AT HINGE POINT',I3,/) C0460400
233 FORMAT (10X,'COMPONENTS OF FREE ',D12.5) C0460500
234 FORMAT (10X,' VECTOR',I3,' FIXED      =',D12.5) C0460600
235 FORMAT (10X,' IN BODY',I3,10X,D12.5,/) C0460700
236 FORMAT (10X,'COMPONENTS OF LOCKED ',D12.5,6X,'COMPONENTS OF LOCKED C0460800
* ',D12.5) C0460900
237 FORMAT (10X,' VECTOR',I3,' FIXED      =',D12.5,6X,' VECTOR',I3,' FIX C0461000
*ED      =',D12.5) C0461100
238 FORMAT (10X,' IN BODY',I3,10X,D12.5,6X,' IN BODY',I3,10X,D12.5,/) C0461200
239 FORMAT (20X,I3,' MOMENTUM WHEELS IN SYSTEM',/) C0461300
240 FORMAT (10X,'MOMENTUM WHEEL',I3,' EMBEDDED ',D12.5,' UNITS') C0461400
241 FORMAT (10X,' IN BODY',I3,' COMPONENTS      =',D12.5,' L*F*T') C0461500
242 FORMAT (10X,' FIXED IN BODY',I3,' ',D12.5,/) C0461600
243 FORMAT (10X,' INITIAL CONDITIONS ALONG FREE AXES (L,L/T) C0461700
* ',/) C0461800
244 FORMAT (10X,' TRANSLATIONAL CONDITIONS ON CENTER OF MASS OF BODY 1 C0461900
* RELATIVE TO INERTIAL ORIGIN ') C0462000
245 FORMAT (10X,5X,I3,' FREE DIRECTIONS OF TRANSLATION, PARALLEL TO I, J C0462100
* K INERTIAL AXES RESP ',/) C0462200
246 FORMAT (10X,'THREE DEGREES OF FREEDOM AT HINGE POINT',I3,/) C0462300
247 FORMAT (' INPUT ERROR IN DATA CARDS, BODIES OUT OF SEQUENCE OR CAC C0462400
* RDS MISSING FROM PRECEDING BODY CARDS ') C0462500
248 FORMAT (10X,' TRANSLATIONAL MOTION OF BODY 1 C.M. CONSTRAINED ALONG C0462600
* G ALL THREE INERTIALLY FIXED AXES ',/) C0462700
249 FORMAT (10X,' ENTIRE SYSTEM HAS',I3,' UNCONSTRAINED DEGREES OF FRE C0462800
* EDCM ',/) C0462900
250 FORMAT (10X,' ENTIRE SYSTEM HAS',I3,' LOCKED OR CONSTRAINED DEGREE C0463000
* S OF FREEDOM ',/) C0463100
251 FORMAT (10X,' DISPLACEMENT ABOUT OR ALONG FREE VECTOR ',I2,' COMPU C0463200
* TEC ') C0463300
252 FORMAT (10X,' CONSTRAINT TORQUE ABOUT OR ALONG LOCKED VECTOR ',I2 C0463400
* , 'COMPUTED ') C0463500
253 FORMAT (3(/)) C0463600
254 FORMAT (25X,'(QUANTITY)',6X,'(ENGLISH)',9X,' (SI)',9X,'(SYMECL)') C0463700
255 FORMAT (25X,' LENGTH ',6X,' FOOT ',8X,' METER ',10X,'L') C0463800
256 FORMAT (25X,' FORCE ',6X,' POUND ',8X,' NEWTON ',10X,'F') C0463900
257 FORMAT (25X,' TIME ',6X,' SECOND ',8X,' SECOND ',10X,'T') C0464000
258 FORMAT (25X,' MASS ',6X,' SLUG ',8X,' KILOGRAM',10X,'M') C0464100
259 FORMAT (25X,' ANGLE ',6X,' RADIAN ',8X,' RADIAN ',10X,'R',/) C0464200
260 FORMAT (20X,'(MOTION CONSTRAINED ALONG ',I3,' AXES)',/) C0464300
261 FORMAT (15X,'FREE VECTORS ALIGNED WITH INERTIALLY FIXED AXES RESPEC C0464400
* TIVELY ',/) C0464500
262 FORMAT (10X,' (TRANSLATIONAL CONDITIONS GIVEN AFTER BODY',I3,' C0464600
* DATA)',/) C0464700
263 FORMAT (10X,' COMPONENTS OF FREE VECTOR',I3,' DEFINED INTERNALLY BC0464800
* Y VECTOR CROSS PRODUCT ',/) C0464900
264 FORMAT (10X,'INITIAL CONDITIONS OF ALL DIRECTION COSINE TRANSFORMAC C0465000
* TION MATRICES COMPUTED') C0465100
265 FORMAT (12X,' INTERNALLY FROM THE TRANSFORMATION MATRICES WHICH DEF C0465200
* INE NOMINAL RELATIVE') C0465300
266 FORMAT (12X,'ORIENTATION (ZERO INTERNAL STRESS STATE) AND INITIAL C0465400
* CONDITIONS FOR') C0465500
267 FORMAT (12X,' ROTATION ABOUT THE SPECIFIED FREE VECTORS') C0465600
268 FORMAT (11X,' INERTIALLY',10X,D12.5,/) C0465700
269 FORMAT (10X,'COMPONENTS OF LOCKED ',D12.5) C0465800
270 FORMAT (10X,' VECTOR',I3,' FIXED      =',D12.5) C0465900

```

```

271 FORMAT (10X, ' IN BODY', I3, 10X, D12.5, //)          C0466000
272 FORMAT (10X, 'ZERO DEGREES OF FREEDOM AT HINGE POINT', I3, //) C0466100
273 FORMAT (10X, 'INTEGRATION STEP SIZE FOR FIXED STEP RUNGE KUTTA INTECO466200
*GRATION (FOURTH ORDER) =', D12.5)                      C0466300
274 FORMAT (15X, 'COMPONENTS OF UNIT VECTOR      ', D12.5, ' (BODY', I3, ' FIXC0466400
*ED')                                                    C0466500
275 FORMAT (15X, ' ALONG SPIN AXIS OF           =', D12.5, ' COORDINATES )C0466600
**')                                                    C0466700
276 FORMAT (15X, ' WHEEL', I3, 16X, D12.5, //)          C0466800
280 FORMAT (25X, ' INITIAL CONDITIONS FOR WHEEL', I3, //) C0466900
281 FORMAT (30X, 'THAW(', I2, ') = ', D12.5, ' RAD')      C0467000
282 FORMAT (29X, 'THADN(', I2, ') = ', D12.5, ' RAD/SEC ', //) C0467100
284 FORMAT (10X, 'ANGULAR MOMENTUM ABOUT SPIN AXIS =', D12.5, ' M*L**2/SEC0467200
*C ', //)                                               C0467300
285 FORMAT (10X, 'MOMENTUM WHEEL', I3, ' EMBEDDED IN BODY', I3, 5X, 16A4, //) C0467400
286 FORMAT (10X, 'ROTATIONAL INERTIA ABOUT SPIN AXIS  ', D12.5, ' M*LCC0467500
***2 ', //)                                           C0467600
288 FORMAT (10X, 'RIGID BODY OR GYROSTAT NUMBER', I3)    C0467700
289 FORMAT(10X, 'PROGRAM TIME TERMINATOR=', D12.5)      C0467800
RETURN                                                  C0467900
END                                                    C0468000

```

```

C                                                    C0500000
SUBROUTINE INEROR                                     C0500100
C                                                    C0500200
C                                                    C0500300
C INEROR PERFORMS VARIOUS CHECKS ON INPUT DATA TO CHECK C0500400
C FOR BASIC TYPING ERRORS IN INPUT DATA AND VIOLATIONS C0500500
C OF BASIC FORMALISM RULES                          C0500600
C                                                    C0500700
C THE DEFINED SYSTEM MUST BE PHYSICALLY REALIZABLE. C0500800
C                                                    C0500900
C                                                    C0501000
C IMPLICIT REAL*8(A-F,G-Z,*)                        C0501100
C LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLG, LEQU, LINIT(1) C0501200
C LOGICAL NSTART, LRTAPE                            C0501300
C                                                    C0501400
C                                                    C0501500
C INTEGER                                           C0501600
* AWORK, CT1, CT2, CT3, CT4, CT5, FCCN, PCOM, C0501700
* SCNDUM, SCN, SCRDM, SCR, SFKDUM, SFK, SFR, SG, C0501800
* SI, SIG, SIXDUM, SIX, SKDUM, SK, SL, SLK, C0501900
* SMA, SMCDUM, SMC, SMV, SGK, SPIDUM, SPI, SQF, C0502000
* SCL, SR, SSCN, SSIX, SVA, SVB, SVD, SVI, C0502100
* SVM, SVP, SVQ, SXM, SXT, TGRQ, SMAL, SEU, C0502200
**SC, SCG, NFLXB, SFLX, SFXM, NMCDS, SFCC, SCC, C0502300
* IINIT(1), IZINIT(1), SD C0502400
C                                                    C0502500
C0502600
C REAL*8
* ANGE (3,3), CNF (3,10), ETIC (3,10), ETMC (3,10), C0502800
* FLU (3,20), FLE (3,3,20), FLH (3,3,20), C0502900
* THAD (3,3), YMCD (3,2,11), RINIT (1), RZINIT(1) C0503000
C                                                    C0503100
C                                                    C0503200
C                                                    C0503300

```

```

COMMEN /CHEKS/          NSTART, LRTAPE          C0E03400
C          C0E03500
C          C0E03600
COMMEN /LOGIC/ FG1, FG2, FG3, FG4, FG5, INEKF, RBLQ(10) C0E03700
C          C0E03800
C          C0E03900
COMMEN /INTG/          AWORK (200)          C0E04000
* CT1          , CT2          , CT3          , CT4          , C0E04100
* CT5          , FCON (33)    , JCGN (10)    , LCON (22)    , C0E04200
* NC (10)      , NEI          , NBD          , NCTC         , C0E04300
* NFER        , NFKC         , NERC         , NLUR         , C0E04400
* NMV         , NMD          , NMDA         , NSVP         , C0E04500
* NEVC        , FCON (11)    , SD           , SFR (33)    , C0E04600
* SG          , SI (55)      , SIG          , SL           , C0E04700
* SLK (33)    , SMA (10)    , SCK (11)    , SQF (11)    , C0E04800
* SGL (11)    , SMV         , SR           , SSCN         , C0E04900
* SEIX        , SVA          , SVB         , SVC          , C0E05000
* SVI         , SVM          , SVP (22)    , SVQ (33)    , C0E05100
* SXM (3,10)  , SXT          , TGRQ (97)   , SMAL        , C0E05200
* SEU         , NTQ          , SC (33)     , SCG         , C0E05300
* NFL>B      , SFLX         , SFXM (10)   , NMDS        , C0E05400
* SFCC        , SCC (10)     ,              ,              , C0E05500
C          C0E05600
C          C0E05700
COMMEN /INTGZ/          C0E05800
* SCNDUM      , SCN (5)      , SCRDM        , SCR (9)      , C0E05900
* SFKDM       , SFK (5)     , SIXDM        , SIX (9)     , C0E06000
* SKDLM       , SK (5)      , SPIDUM       , SPI (9)     , C0E06100
* SNCELM      , SMC (5)     ,              ,              , C0E06200
C          C0E06300
C          C0E06400
COMMEN /REAL/          C0E06500
* CA (3,10)   , CAC (3,10)  , CLM (10)     , CCMC (3,11) , C0E06600
* DCMC (3,11) , ETC (3,11)  , ETM (33)     , FCMC (3,11) , C0E06700
* GAM (3,66)  , F           , HM (3,10)    , HMC (3,10)  , C0E06800
* HMCN (10)   , PHI (3,11)  , PLM (10)     , QF (3,32)   , C0E06900
* QFC (3,33)  , GL (3,22)   , QLC (3,22)   , RCMC (3,11) , C0E07000
* T           , THA (33)    , THAD (33)    ,              , C0E07100
* THADW (10)  , THAW (10)   , XDIC (3,3,66), XI (3,3,10) , C0E07200
* XIC (3,3,10), XMAS (10)   , XMN (33,33)  , XMT (3,3,10), C0E07300
* TLG (33)    , FLA (3,20)  , FLB (3,20)   , FLC (3,20)  , C0E07400
* FLD (3,3,20), FLJ (3,3,20), CAC (3,10)   , XIU (3,3,10), C0E07500
* FLIRC (3,10), FLCRC (3,10), FLAC (3,20)  , FLQC (3,20) , C0E07600
* FLDN (20)   , ZETA (20)   , FCF (3,3,40) , FCK (3,40)  , C0E07700
* TIMEND      ,              ,              ,              , C0E07800
C          C0E07900
C          C0E08000
COMMEN /REALZ/          C0E08100
* CEDUM (1,3) , CE (3,10)   , CHCDUM(1,3) , CBC (3,10)  , C0E08200
* XNCELM(1,1,9), XMC (3,3,10), Cn(3)       ,              , C0E08300
C          C0E08400
C          C0E08500
EQUIVALENCE (ETM(1),THAD(1))          ,(XMN(1,1),ANGD(1))          , C0E08600
*          (XMN(1,3),YMCD(1,1,1))      ,(XMN(1,6),CNF(1,1))      , C0E08700
*          (XMN(1,6),ETIC(1,1))        ,(XMN(1,10),ETMC(1,1))   , C0E08800
*          (FLB(1,1),FLG(1,1))         ,(FLE(1,1,1),FLD(1,1,1)) , C0E08900
*          (FLH(1,1,1),FLJ(1,1,1))     ,                          , C0E09000
*          (FG1,LINIT(1))              ,(CA(1,1),RINIT(1))      , C0E09100
*          (CBDUM(1,1),RZINIT(1))      ,(AWORK(1),IINIT(1))    , C0E09200
*          (SCNDUM,IZINIT(1))          ,                          , C0E09300

```

```

C
C
C
C
C      DIMENSION QD(3)
C
C      N = 1
C      L = 1
C
C      CYCLE THRU DATA FOR ALL BODIES
C      DO 1 K=1,NBCD
C
C      CHECK TO SEE IF BODY MASS IS POSITIVE
C      IF(XMAS(K).GT.0) GO TO 10
C      PRINT 200, K,XMAS(K)
C      FG2 = .FALSE.
C
C      CHECK TO SEE IF TOPOLOGICAL TREE PROPERLY LABELED
C      10 IF(JCCN(K).LT.K) GO TO 11
C      PRINT 201, JCCN(K),K
C      FG2 = .FALSE.
C
C      CHECK NUMBER OF CONSTRAINED AXES, AT MOST 3
C      11 IF(PCCN(K).GE.0.AND.PCCN(K).LE.3) GO TO 12
C      K1 = K-1
C      PRINT 202, K,PCCN(K),K1
C      FG2 = .FALSE.
C      12 IF(.NOT.FBLG(K)) GO TO 13
C      DO 2 I=1,3
C      DO 2 J=1,3
C
C      CHECK SYMMETRY OF INERTIA TENSOR
C      IF(XI(I,J,K).EG.XI(J,I,K)) GO TO 2
C      PRINT 203, K
C      FG2 = .FALSE.
C      2 CONTINUE
C
C      DETERMINANT OF INERTIA TENSOR MUST BE POSITIVE FOR IT
C      TO HAVE REAL PRINCIPAL MOMENTS OF INERTIA
C      DET = XI(1,1,K)*(XI(2,2,K)*XI(3,3,K) - XI(2,3,K)*XI(3,2,K))
C      * -XI(1,2,K)*(XI(2,1,K)*XI(3,3,K) - XI(3,1,K)*XI(2,3,K))
C      * +XI(1,3,K)*(XI(2,1,K)*XI(3,2,K) - XI(2,2,K)*XI(3,1,K))
C      IF(DET.GT.0) GO TO 13
C      PRINT 204, K,DET
C      FG2 = .FALSE.
C
C      CHECK ORTHOGONALITY OF TRANSFORMATION MATRIX
C      13 DO 3 I=1,3
C      DO 3 J=1,3
C      TEST = 0
C      DO 4 L=1,3
C      4 TEST = TEST + XMT(L,I,K)*XMT(L,J,K)
C      IF(I.EQ.J) GO TO 14
C      IF(DABS(TEST).LT.,.1D-03) GO TO 3
C      PRINT 205, I,J,TEST,K
C      FG2 = .FALSE.
C
C      GO TO 3
C      14 IF(CABS(DABS(TEST)-1.D0).LT.,.1D-03) GO TO 3
C      PRINT 205, I,J,TEST,K
C      FG2 = .FALSE.
C      3 CONTINUE
C      1 CONTINUE
C
C      200 FORMAT (10X,'MASS OF BODY',I3,' = ',D15.5,' ZERO OR NEGATIVE MASS ELEMENTS UNACCEPTABLE ',/)
C      201 FORMAT (10X,'JCON(K) MUST BE LESS THAN K BODY',I3,' CANNOT LIE BETWEEN BODY 1 AND BODY',I3,' IN TOPOLOGICAL TREE',/)

```

```

202 FORMAT (10X,'BODY',I3,' CANNOT HAVE',I3,' CONSTRAINED AXES AT HING(C0E15400
  *E FCINT',I3,/) C0E15500
203 FORMAT (10X,'INERTIA TENSOR OF BODY',I3,' NON-SYMMETRIC ') C0E15600
204 FORMAT (10X,'DETERMINANT OF THE INERTIA TENSOR OF BODY',I3,' =',D1(C0E15700
  *5.5,' IT MUST BE POSITIVE FOR REAL PRINCIPAL MOMENTS OF INERTIA ',C0E15800
  */) C0E15900
205 FORMAT (10X,'EIGENVECTOR',I3,' DOT',I3,' =',D15.5,' FOR THE TRANSF C0E16000
  *FORMATION MATRIX OF BODY',I3,/) C0E16100
RETURN C0E16200
END C0E16300

```

```

C SUBROUTINE SETS C0E00000
C C0E00100
C C0E00200
C C0E00300
IMPLICIT REAL*8(A-H,O-Z,I) C0E00400
LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLC, LEQU, LINIT(1) C0E00500
LOGICAL NSTART, LRTAPE C0E00600
C C0E00700
C C0E00800
INTEGER S1(10),S2(10),S2NM1 C0E00900
INTEGER C0E01000
* AWORK , CT1 , CT2 , CT3 , CT4 , CT5 , FCCN , PCCN , C0E01100
* SCNDUM, SCN , SCRUM, SCR , SKDUM, SK , SFR , SG , C0E01200
* SI , SIG , SIXDUM, SIX , SKDUM, SK , SL , SLK , C0E01300
* SMA , SMCUM, SMC , SMV , SOK , SPIDUM, SPI , SQF , C0E01400
* SGL , SR , SSCN , SSIX , SVA , SVB , SVD , SVI , C0E01500
* SVM , SVP , SVQ , SXP , SXT , TORQ , SMAL , SEU , C0E01600
* SC , SCG , NFLXB , SFLX , SFXM , NMODS , SFCC , SCC , C0E01700
* IINIT(1) , IZINIT(1) , SD C0E01800
C C0E01900
C C0E02000
REAL*8 C0E02100
* ANGE (33) , CNF (3,10) , ETIC (3,10) , ETMC (3,10) , C0E02200
* FLO (3,20) , FLE (3,3,20) , FLH (3,3,20) , C0E02300
* THADD (33) , YMCD (3,2,11) , RINIT (1) , RZINIT(1) C0E02400
C C0E02500
C C0E02600
C C0E02700
COMMON /CHECKS/ NSTART, LRTAPE C0E02800
C C0E02900
C C0E03000
COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RBLC(10) C0E03100
C C0E03200
C C0E03300
COMMON /INTG/ AWORK (200) , C0E03400
* CT1 , CT2 , CT3 , CT4 , C0E03500
* CT5 , FCCN (33) , JCCN (10) , LCON (22) , C0E03600
* MC (10) , NB1 , NBUD , NCTC , C0E03700
* NFER , NFKC , NFRC , NLOR , C0E03800
* NMV , NMG , NMDA , NSVP , C0E03900
* NSVC , FCCN (11) , SD , SFR (33) , C0E04000
* SG , SI (55) , SIG , SL , C0E04100
* SLK (33) , SMA (10) , SOK (11) , SQF (11) , C0E04200
* SGL (11) , SMV , SR , SSCN , C0E04300
* SSIX , SVA , SVB , SVD , C0E04400

```



```

* SVI          , SVM          , SVP (22)    , SVQ (33)    , C0604500
* SXM (3,10)   , SXT          , TORQ (97)   , SMAL        , C0604600
* SEU          , NTQ          , SC (33)     , SCG         , C0604700
* NFLXB       , SFLX        , SFXM (10)   , NMODS       , C0604800
* SFCC        , SCC (10)    ,              ,              , C0604900
C              ,              ,              ,              , C0605000
C              ,              ,              ,              , C0605100
COMMON /INTGZ/
* SCNDUM      , SCN (5)     , SCRDM       , SCR (9)     , C0605300
* SFKCLM     , SFK (5)    , SIXDUM     , SIX (9)    , C0605400
* SKDLM      , SK (5)     , SPIDUM     , SPI (9)    , C0605500
* SPCDUM     , SMC (5)    ,              ,              , C0605600
C              ,              ,              ,              , C0605700
C              ,              ,              ,              , C0605800
C              ,              ,              ,              , C0605900
COMMON /REAL/
* CA (3,10)   , CAC (3,10) , CLM (10)    , CGMC (3,11) , C0606000
* DCMC (3,11) , ETC (3,11) , ETM (33)   , FCMC (3,11) , C0606100
* GAM (3,66)  , H          , HM (3,10)  , HMC (3,10)  , C0606200
* HMC (10)    , PHI (3,11) , PLM (10)   , QF (3,33)   , C0606300
* OFC (3,33)  , QL (3,22) , QLC (3,22) , RQMC (3,11) , C0606400
* T          ,          , THA (33)   , THAD (33)   , C0606500
* THADW (10) , THAW (10) , XDIC (3,3,66), XI (3,3,10) , C0606600
* XIC (3,3,10), XNAS (10) , XMN (33,33), XMT (3,3,10), C0606700
* TUG (33)    , FLA (3,20) , FLB (3,20) , FLC (3,20)  , C0606800
* FLD (3,3,20), FLJ (3,3,20), CAU (3,10) , XIU (3,3,10), C0606900
* FLIRC (3,10), FLCRC (3,10), FLAC (3,20) , FLUC (3,20) , C0607000
* FLCM (20)   , ZETA (20) , FCF (3,3,40), FCK (3,40) , C0607100
* TIMEND     ,              ,              ,              , C0607200
C              ,              ,              ,              , C0607300
C              ,              ,              ,              , C0607400
COMMON /REALZ/
* CEDUM (1,3) , CB (3,10) , CBCDUM(1,3) , CBC (3,10) , C0607600
* XCCUM(1,1,9) , XMC (3,3,10), CBN(3) ,              , C0607700
C              ,              ,              ,              , C0607800
C              ,              ,              ,              , C0607900
EQUIVALENCE (ETM(1),THADD(1)) , (XMN(1,1),ANGD(1)) , C0608000
* (XMN(1,3),YMCD(1,1,1)) , (XMN(1,6),CNF(1,1)) , C0608100
* (XMN(1,8),ETIC(1,1)) , (XMN(1,10),ETMC(1,1)) , C0608200
* (FLB(1,1),FLC(1,1)) , (FLE(1,1,1),FLD(1,1,1)) , C0608300
* (FLH(1,1,1),FLJ(1,1,1)) , , C0608400
* (FG1,LINIT(1)) , (CA(1,1),RINIT(1)) , C0608500
* (CBDUM(1,1),RZINIT(1)) , (AORR(1),LINIT(1)) , C0608600
* (SCNDUM,IZINIT(1)) , , C0608700
C              ,              , C0608800
C              ,              , C0608900
C              ,              , C0609000
C              ,              , C0609100
C              ,              , C0609200
C              ,              , C0609300
C              ,              , C0609400
C              ,              , C0609500
C              ,              , C0609600
C              ,              , C0609700
C              ,              , C0609800
C              ,              , C0609900
C              ,              , C0610000
C              ,              , C0610100
C              ,              , C0610200
C              ,              , C0610300
C              ,              , C0610400

```

	S2(NL) = I	C0610500
	GO TC 1	C0610600
14	NR = NR + 1	C0610700
	S1(NR) = I	C0610800
1	CONTINUE	C0610900
C	COMPACT S1(NR) INTO SR AND	C0611000
C	S2(NL) INTO SL	C0611100
	CALL CUMFAC(S1,NR,SR)	C0611200
	CALL CUMFAC(S2,NL,SL)	C0611300
C		C0611400
C	GET SET SG OF GYROSTAT EDDY LABELS	C0611500
	IF(NMG.EC.0) GO TC 19	C0611600
	DO 16 I=1,10	C0611700
	S1(I) = 0	C0611800
16	S2(I) = 0	C0611900
	DO 17 M=1,NMG	C0612000
	L = MC(M)	C0612100
17	S1(L) = 1	C0612200
	K = 0	C0612300
	DO 18 I=1,10	C0612400
	IF(S1(I).EQ.0) GO TC 18	C0612500
	K = K+1	C0612600
	S2(K) = I	C0612700
18	CONTINUE	C0612800
	CALL CCMPAC(S2,K,SG)	C0612900
	GO TC 19	C0613000
19	SG = 0	C0613100
	GO TC 19	C0613200
C		C0613300
C	GET THE SETS OF BODIES OUTBOARD OF HINGE POINT K	C0613400
C	(COMPACTED FORMS IN SK(K))	C0613500
15	NM1 = NECD - 1	C0613600
	DO 2 L=1,NBOD	C0613700
	K = L-1	C0613800
	DO 4 I=1,NBOD	C0613900
4	S1(I) = 0	C0614000
	NC = 0	C0614100
C	CYCLE THRU BODIES WHICH CAN BE OUTBOARD OF K	C0614200
	ITEST = K+1	C0614300
	DO 3 J=ITEST,NBOD	C0614400
	IN = J	C0614500
5	IF(IN-1-K) 6,7,8	C0614600
C	EDDY J NOT ON PATH FROM K	C0614700
6	GO TC 3	C0614800
C	EDDY J OUTBOARD OF HINGE K ON CHAIN FROM K	C0614900
7	NC = NC + 1	C0615000
	S1(NC) = J	C0615100
	GO TC 3	C0615200
C	EDDY J CAN BE ON CHAIN FROM K	C0615300
8	IN = JCCN(IN)	C0615400
900	GO TC 5	C0615500
3	CONTINUE	C0615600
C	COMPACT SET SK(K)	C0615700
900	CALL CCMPAC(S1,NC,SK(K))	C0615800
2	CONTINUE	C0615900
C	GET SETS OF BODIES ON PATH FROM HINGE POINT I	C0616000
C	TO C.M. OF BODY N+1; S1(KTO(NBOD,I,N))	C0616100
C	KTC = FUNCTION TO PUT TRIANGULAR ARRAYS IN ONE DIMENSIONAL ARRAY	C0616200
	DO 9 L=1,NBOD	C0616300
	I = L-1	C0616400

	CALL UNPAC(S2,NSET,SK(1))	00616500
	DO 10 N=1,NM1	00616600
10	SI(KT0(NBCD,I,N)) = 0	00616700
	DO 11 NN=1,NSET	00616800
	DO 13 K=1,NBCD	00616900
13	SI(K) = 0	00617000
	NC = 0	00617100
	IN = S2(NN)	00617200
12	IF(IN-1.LT.1) GO TO 20	00617300
	NC = NC + 1	00617400
	SI(NC) = IN	00617500
	IN = JCCN(IN)	00617600
	GO TO 12	00617700
20	CONTINUE	00617800
C		00617900
	CALL COMPAC(S1,NC,SI(KT0(NBCD,I,S2(NN)-1)))	00618000
11	CONTINUE	00618100
9	CONTINUE	00618200
C		00618300
C	COMPLETE REQUIRED SUMMATION SETS	00618400
	MM = 1	00618500
	LL = 1	00618600
	DO 22 K=1,NB1	00618700
	IF(K.EQ.1) GO TO 23	00618800
	MM = MM + PCUN(K-1)	00618900
	LL = LL + FCCN(K-1)	00619000
23	IF(PCUN(K).NE.3) GO TO 24	00619100
	SQF(K) = 0	00619200
	SQL(K) = LL	00619300
	GO TO 22	00619400
24	IF(PCUN(K).NE.0) GO TO 25	00619500
	SQF(K) = MM	00619600
	SQL(K) = 0	00619700
	GO TO 22	00619800
25	SQF(K) = MM	00619900
	SQL(K) = LL	00620000
22	CONTINUE	00620100
	RETURN	00620200
	END	00620300

C	SUERCUTINE INOPT	00700000
C	ACCEPTS ALL INFORMATION NEEDED TO DEFINE THE COMPUTATIONAL	00700100
C	OPTIONS WHICH CAN BE EXECUTED BY THE USER	00700200
C	THE FOLLOWING AUGMENTED SET DEFINITIONS ARE MADE	00700300
C	'NEST I' = ALL ECDIES OUTCARD OF HINGE POINT IAGE BODY I+1	00700400
C	SPI(I) = ALL ECDIES OF NEST I CONTRIBUTING SIGNIFICANTLY	00700500
C	TO PSEUDO-INERTIA TENSORS OF NEST I	00700600
C	SIX(I) = ALL ECDIES OF NEST I CONTRIBUTING SIGNIFICANTLY	00700700
C	TO INERTIA CROSS COUPLING EFFECTS IN NEST I MOTION	00700800
C	EQUATION	00700900
C	SCN(I) = ALL ECDIES OF NEST I CONTRIBUTING SIGNIFICANTLY	00701000
C	TO CENTRIPITAL CROSS COUPLING EFFECTS IN NEST I	00701100
C	MOTION EQUATION	00701200
C	SCR(I) = ALL ECDIES OF NEST I CONTRIBUTING SIGNIFICANTLY	00701300
C	EQUATION	00701400
C		00701500

```

C      TO COFOLIS CROSS COUPLING EFFECTS IN NEST I MOTION      C0701600
C      SMC(I) = ALL CONSTANT AND VARIABLE SPEED MOMENTUM WHEELS      C0701700
C      IN NEST I      C0701800
C      SD = ALL CONTIGUOUS PAIRS OF BODIES (JCON(K),K) FOR WHICH      C0701900
C      DIRECTION COSINE TECHNIQUES ARE TO BE USED TO COMPUTE      C0702000
C      RELATIVE TRANSFORMATION MATRICES      C0702100
C      SD SHOULD CONTAIN ONLY LABELS FOR RIGID OR FLEXIBLE      C0702200
C      BODIES      C0702300
C      SMAL = ALL CONTIGUOUS PAIRS OF BODIES (JCON(K),K) FOR WHICH      C0702400
C      SMALL ANGLE TECHNIQUES ARE TO BE USED TO COMPUTE      C0702500
C      RELATIVE TRANSFORMATION MATRICES      C0702600
C      SEU = ALL CONTIGUOUS PAIRS OF BODIES (JCON(K),K) FOR WHICH      C0702700
C      EULER ANGLE TECHNIQUES ARE TO BE USED TO COMPUTE      C0702800
C      RELATIVE TRANSFORMATION MATRICES      C0702900
C      SXM(I,K) = COMPILED CODE WORD ARRAY      C0703000
C      THE COORDINATE AXIS IN BODY K ALONG WHICH THE I-TH      C0703100
C      FREE OR LOCKED GIMBAL AXIS IS ALIGNED, SIGN IMPLIES      C0703200
C      DIRECTION      C0703300
C      'NOTE' INTERNAL LOGIC ASSUMES THAT THE FREE      C0703400
C      COORDINATE VECTORS AT HINGE POINTS AT WHICH SMALL      C0703500
C      ANGLE OR EULER ANGLE TECHNIQUES ARE APPLIED ARE      C0703600
C      PARALLEL TO THE BODY FIXED AXES DEFINED      C0703700
C      AT THAT POINT      C0703800
C      SXT = ALL COLUMNS OF THE SYSTEM INERTIA MATRIX OF PSUEDO-      C0703900
C      INERTIA TENSORS WHICH HAVE TIME VARYING ELEMENTS.      C0704000
C      (SINCE INERTIA MATRIX SYMMETRIC COLUMNS DESIGNATED      C0704100
C      EXTEND DOWN ONLY TO DIAGONAL ELEMENT, COLUMN N+1 IS      C0704200
C      ALWAYS ASSUMED TIME VARYING)      C0704300
C      ELEMENTS IN COLUMN K, K=1,2,...,N GIVE THE INERTIA      C0704400
C      CONTRIBUTIONS OF THE NEST K-1 TO THE SYSTEM EQUATIONS      C0704500
C      OF MOTION      C0704600
C      SMV = ALL VARIABLE SPEED MOMENTUM WHEELS      C0704700
C      SVD = ALL BODIES IN WHICH BODY FIXED VECTORS AND DYADS ARE      C0704800
C      TIME VARYING IN THE COMPUTATIONAL REFERENCE FRAME      C0704900
C      SFR(I) = SET OF FREE COORDINATE VECTOR LABELS OF THOSE VECTORS      C0705000
C      ABOUT OR ALONG WHICH DISPLACEMENT IS COMPUTED      C0705100
C      SLK(I) = SET OF LOCKED COORDINATE VECTOR LABELS OF THOSE VECTORS      C0705200
C      ABOUT OR ALONG WHICH CONSTRAINT TORQUE IS COMPUTED.      C0705300
C      THIS CAPABILITY NOT AVAILABLE IN N-BOD2      C0705400
C      SMA(I) = SET OF MOMENTUM WHEEL LABELS OF THOSE WHEELS FOR      C0705500
C      WHICH ANGULAR DISPLACEMENT IS TO BE COMPUTED      C0705600
C      INERF = .TRUE. COMPUTATION FRAME IS INERTIAL REFERENCE      C0705800
C      .FALSE. COMPUTATION FRAME FIXED IN BODY 1      C0705900
C      C0706000
C      SC(I) = ALL CAGED DEGREES OF FREEDOM      C0706100
C      SCG = TOTAL NUMBER OF CAGED DEGREES OF FREEDOM      C0706200
C      TUG(J) = TIME AT WHICH DEGREE OF FREEDOM J UNCAGED      C0706300
C      C0706400
C      C0706500
C      NFLXB = TOTAL NUMBER OF FLEXIBLE BODIES      C0706600
C      SFLX = SET OF ALL FLEXIBLE BODIES      C0706700
C      SFXM(I) = TOTAL NUMBER OF FLEXIBLE BODY MODES USED FOR BODY I      C0706800
C      (ZERO IF BODY I RIGID)      C0706900
C      NMDS = TOTAL NUMBER OF FLEXIBLE BODY MODES USED FOR ENTIRE      C0707000
C      SIMULATION      C0707100
C      C0707200
C      RESULTANT FLEXIBLE BODY MODAL PARAMETERS      C0707300
C      FLA(I,MN) = MODE MN CENTER OF MASS      C0707400
C      FLB(I,MN) = MASS MODAL MOMENT ABOUT HINGE POINT FOR      C0707500

```

```

C          MODE MN DUE TO TRANSLATION OF MASS POINTS          C0707600
C      FLC(I,MN) = INERTIA MODAL MOMENT FOR MODE MN DUE TO     C0707700
C          ROTATION OF MASS POINTS                             C0707800
C      FLD(I,J,MN) = PSUEDO-INERTIA COUPLING TENSOR FOR MODE MN C0707900
C      FLJ(I,J,MN) = CENTRIPITAL COUPLING TENSOR FOR MODE MN   C0708000
C                                                                C0708100
C          FOR MODE M OF BODY N, MODAL NUMBER MN IS           C0708200
C          MN = SFXM(1)+...+SFXM(N-1)+M                       C0708300
C                                                                C0708400
C          FOR ELASTIC COORDINATE MN                           C0708500
C      THA(NFER+MN) = INITIAL GENERALIZED DISPLACEMENT, ELASTIC COORD. MN C0708600
C      THAC(NFER+MN) = INITIAL GENERALIZED RATE, ELASTIC COORD. MN C0708700
C          NFER = TOTAL NUMBER OF FREE COORDINATE VECTORS     C0708800
C                                                                C0708900
C      SFCC = SET OF ALL FLEXIBLE BODIES FOR WHICH CENTRIPETAL AND C0709000
C          CORIOLIS EFFECTS SIGNIFICANT IN DEFORMATION EQUATION C0709100
C      SCXC(MN) = SET OF ALL MODES N WHICH SIGNIFICANTLY CONTRIBUTE C0709200
C          TO THE CENTRIFETAL OR CORIOLIS CROSS COUPLING EFFECTS C0709300
C          IN THE EQUATION FOR MODE WITH MODE NUMBER MN       C0709400
C                                                                C0709500
C          RESULTANT MODE COUPLING COEFFICIENTS FOR MODE M AND N C0709600
C      FCF(M,N) = MODE CROSS COUPLING TENSOR CENTRIPETAL ACC. EFFECTS C0709700
C          (IN MOST CASES FCF(M,N); M=N CAN HAVE SIGNIFICANT C0709800
C          NON-ZERO TERMS)                                     C0709900
C      FCK(M,N) = MODE CROSS COUPLING VECTOR FOR CORIOLIS ACC. EFFECTS C0710000
C                                                                C0710100
C                                                                C0710200
C                                                                C0710300
C      IMPLICIT REAL*8(A-F,O-Z,I)                             C0710400
C      LOGICAL FG1, FG2, FG3, FG4, FGS, INERF, RBLG, LEQU, LINIT(1) C0710500
C      LOGICAL          NSTART, LRTAPE                         C0710600
C                                                                C0710700
C                                                                C0710800
C                                                                C0710900
C      INTEGER
C      * AWORK , CT1 , CT2 , CT3 , CT4 , CT5 , FCON , FCGN , C0711000
C      * SCNDUM, SCN , SCRDUM, SCR , SFKDUM, SFK , SFR , SG , C0711100
C      * SI , SIG , SIXDUM, SIX , SKDUM , SK , SL , SLK , C0711200
C      * SMA , SMCUM, SMC , SMV , SOK , SPIDUM, SPI , SQF , C0711300
C      * SGL , SR , SSCN , SSIX , SVA , SVB , SVC , SVI , C0711400
C      * SVM , SVP , SVQ , SXM , SXT , TORQ , SMAL , SEU , C0711500
C      * SC , SCG , NFLXB , SFLX , SFXM , NMCDS , SFCC , SCC , C0711600
C      * IINIT(1) , IZINIT(1) , SD , SCXC(20)                 C0711700
C                                                                C0711800
C                                                                C0711900
C                                                                C0712000
C      REAL*8
C      * ANGC (33) , CNF (3,10) , ETIC (3,10) , ETMC (3,10) , C0712100
C      * FLO (3,20) , FLE (3,3,20) , FLH (3,3,20) , C0712200
C      * THACD (33) , YMCD (3,2,11) , RINIT (1) , RZINIT(1) C0712300
C                                                                C0712400
C                                                                C0712500
C      COMMON /CHEKS/          NSTART, LRTAPE                  C0712600
C                                                                C0712700
C                                                                C0712800
C      COMMON /LOGIC/ FG1, FG2, FG3, FG4, FGS, INERF, RBLG(10) C0712900
C                                                                C0713000
C                                                                C0713100
C      COMMON /INTG/          AWORK (200) , C0713200
C      * CT1 , CT2 , CT3 , CT4 , C0713300
C      * CT5 , FCON (33) , JCGN (10) , LCON (22) , C0713400
C      * MC (10) , NE1 , NBD , NCTC , C0713500

```

* NFER		* NFKC		* NFRC		* NLOR		C0713600
* NMV		* NMO		* NMOA		* NSVP		C0713700
* NEVC		* PCGN (11)		* SD		* SFR (33)		C0713800
* SG		* SI (55)		* SIG		* SL		C0713900
* SLK (33)		* SMA (10)		* SCK (11)		* SQF (11)		C0714000
* SGL (11)		* SMV		* SR		* SSCN		C0714100
* SSIX		* SVA		* SVB		* SVD		C0714200
* SVI		* SVM		* SVP (22)		* SVQ (33)		C0714300
* SXM (3,10)		* SXT		* TORQ (97)		* SMAL		C0714400
* SEU		* NTQ		* SC (33)		* SCG		C0714500
* NFLXB		* SFLX		* SFXM (10)		* NMDS		C0714600
* SFCC		* SCC (10)						C0714700
								C0714800
								C0714900
								C0715000
CCMMCN /INTGZ/								
* SCNCLM		* SCN (5)		* SCRDUM		* SCR (9)		C0715100
* SFKCLM		* SFK (5)		* SIXDUM		* SIX (9)		C0715200
* SKDUM		* SK (5)		* SPIDUM		* SPI (9)		C0715300
* SMCCUM		* SMC (5)						C0715400
								C0715500
								C0715600
								C0715700
CCMMCN /REAL/								
* CA (3,10)		* CAC (3,10)		* CLM (10)		* CCMC (3,11)		C0715800
* DCMC (3,11)		* ETC (3,11)		* ETM (33)		* FOMC (3,11)		C0715900
* GAM (3,66)		* H		* HM (3,10)		* HMC (3,10)		C0716000
* HMDM (10)		* PHI (3,11)		* PLM (10)		* QF (3,33)		C0716100
* QFC (3,33)		* QL (3,22)		* QLC (3,22)		* RCMC (3,11)		C0716200
* T				* THA (33)		* THAD (33)		C0716300
* THADW (10)		* THAW (10)		* XDIC (3,3,66)		* XI (3,3,10)		C0716400
* XIC (3,3,10)		* XMAS (10)		* XMN (33,33)		* XMT (3,3,10)		C0716500
* TLG (33)		* FLA (3,20)		* FLB (3,20)		* FLC (3,20)		C0716600
* FLJ (3,3,20)		* FLJ (3,3,20)		* CAD (3,10)		* XIO (3,3,10)		C0716700
* FLIRC (3,10)		* FLCRC (3,10)		* FLAC (3,20)		* FLQC (3,20)		C0716800
* FLOW (20)		* ZETA (20)		* FCF (3,3,40)		* FCK (3,40)		C0716900
* TIMEND								C0717000
								C0717100
								C0717200
								C0717300
CCMMCN /REALZ/								
* CBDUM (1,3)		* CB (3,10)		* CBCDUM(1,3)		* CBC (3,10)		C0717400
* XMCCUM(1,1,9)		* XMC (3,3,10)		* GBN(3)				C0717500
								C0717600
								C0717700
								C0717800
EQUIVALENCE (ETM(1),THADD(1))								
								C0717900
								C0718000
								C0718100
								C0718200
								C0718300
								C0718400
								C0718500
								C0718600
								C0718700
								C0718800
								C0718900
								C0719000
								C0719100
								C0719200
								C0719300
								C0719400
								C0719500

```

* IXM,IIC,ICE,ICD/'X',,'IXC',,'GEN',,'CCR'//, C0719600
* IEC,ICL,IVT,IFR,ILO/'EULE',,'COLX',,'VTIN',,'FREE',,'LOCK'//, C0719700
* ICF,IMA/'CFGR',,'MCMA'//, C0719800
* IVA/'VARW'//, C0719900
* ISM/'SMAL'//, C0720000
* IEN/'END'//, C0720100
* NU,IBL,ICG/'NO',,'',,'CAGE'//, C0720200
* IFL,IMD/'FLEX',,'MCDE'// C0720300
C C0720400
C DEFAULT CONDITIONS C0720500
C C0720600
C SET UP OPTION DATA SETS MAKING NO ENGINEERING ASSUMPTIONS AND THAT C0720700
C COMPUTATION IS RELATIVE TO INERTIAL FRAME C0720800
C C0720900
INERF = .FALSE. C0721000
DO 1 L=1,NBOD C0721100
I = L-1 C0721200
SFK(I) = 0 C0721300
SPI(I) = SK(I) C0721400
SIX(I) = SK(I) C0721500
SCN(I) = SK(I) C0721600
1 SCR(I) = SK(I) C0721700
EXT = SK(0) C0721800
EVC = SK(0) C0721900
SMAL = 0 C0722000
SEL = 0 C0722100
NFRC = NFER C0722200
NCTC = 0 C0722300
NMCA = 0 C0722400
NFKC = 0 C0722500
SCG = 0 C0722600
NB2 = 3*NB1 C0722700
DO E4 I=1,NB3 C0722800
SLK(I) = 0 C0722900
E4 SC(I) = 0 C0723000
SFLX = 0 C0723100
NMCDS = 0 C0723200
DO 79 I=1,NBOD C0723300
79 SFXM(I) = 0 C0723400
NFLXE = 0 C0723500
SFCC = 0 C0723600
NB2 = 2*NBOD C0723700
DO 85 I=1,NB2 C0723800
85 SCXC(I) = 0 C0723900
DO 2 I=1,NFER C0724000
2 SFR(I) = I C0724100
DO 60 I=1,NMO C0724200
SMA(I) = 0 C0724300
EO S1(I) = I C0724400
CALL CCMFAC(S1,NMC,SMV) C0724500
NMV = NMC C0724600
C C0724700
C NOTE C0724800
C CORICLIS EFFECTS ONLY FOR LINEAR OSCILLATORS C0724900
C INERTIA CROSS COUPLING ONLY FOR RIGID BODIES C0725000
C FOR LINEAR OSCILLATOR TRANSFORMATION MATRICES FIXED C0725100
C SUBTRACT THESE FROM NON-AUGMENTED SETS CONSTRUCTED C0725200
DO 5 L=1,NBOD C0725300
I = L-1 C0725400
CALL LNPAC(S1,N1,SCR(I)) C0725500

```

.CALL UNPAC(S2,N2,SL)	C0725600
N3= C	C0725700
IF(N1.EQ.0) GO TO 666	C0725800
DO 6 J=1,N1	C0725900
IF(N2.EQ.0) GO TO 667	C0726000
DO 668 K=1,N2	C0726100
IF(S1(J).NE.S2(K)) GO TO 668	C0726200
N3 = N3 + 1	C0726300
S3(N3) = S1(J)	C0726400
668 CONTINUE	C0726500
667 CONTINUE	C0726600
6 CONTINUE	C0726700
666 CONTINUE	C0726800
CALL COMFAC(S3,N3,SCH(1))	C0726900
6 CONTINUE	C0727000
C	C0727100
DO 10 L=1,NBOD	C0727200
I = L-1	C0727300
CALL UNPAC(S1,N1,SIX(I))	C0727400
CALL UNPAC(S2,N2,SR)	C0727500
N3 = C	C0727600
IF(N1.EQ.C) GO TO 1111	C0727700
DO 11 J=1,N1	C0727800
IF(N2.EQ.0) GO TO 1112	C0727900
DO 1113 K=1,N2	C0728000
IF(S1(J).NE.S2(K)) GO TO 1113	C0728100
N3 = N3 + 1	C0728200
S3(N3) = S1(J)	C0728300
1113 CONTINUE	C0728400
1112 CONTINUE	C0728500
11 CONTINUE	C0728600
1111 CONTINUE	C0728700
CALL COMFAC(S3,N3,SIX(I))	C0728800
10 CONTINUE	C0728900
C	C0729000
C	C0729100
12 READ 100, ICD1,ICD2,ICD3,I,NSET,(S1(J),J=1,10)	C0729200
CT1 = CT1 + 1	C0729300
IF(ICD1.EQ.IBL) GO TO 20	C0729400
IF(ICD1.EQ.NU) GO TO 21	C0729500
IF(ICD1.EQ.IEN) GO TO 21	C0729600
20 IF(ICD2.EQ.IAS) GO TO 22	C0729700
IF(ICD2.EQ.IEU) GO TO 23	C0729800
IF(ICD2.EQ.ICL) GO TO 24	C0729900
IF(ICD2.EQ.IVT) GO TO 25	C0730000
IF(ICD2.EQ.IFR) GO TO 26	C0730100
IF(ICD2.EQ.ILG) GO TO 27	C0730200
IF(ICD2.EQ.IMA) GO TO 28	C0730300
IF(ICD2.EQ.ICF) GO TO 29	C0730400
IF(ICD2.EQ.INR) GO TO 30	C0730500
IF(ICD2.EQ.IBD) GO TO 31	C0730600
IF(ICD2.EQ.IVA) GO TO 62	C0730700
IF(ICD2.EQ.ISM) GO TO 69	C0730800
IF(ICD2.EQ.ICG) GO TO 77	C0730900
IF(ICD2.EQ.IFL) GO TO 80	C0731000
IF(ICD2.EQ.IMD) GO TO 90	C0731100
37 PRINT 200, CT1,ICD1,ICD2,ICD3	C0731200
FG3 = .FALSE.	C0731300
RETURN	C0731400
30 INERF = .TRUE.	C0731500



```

GO TO 12
31 INERF = .FALSE.
GO TO 12
22 IF(ICD3.EQ.IXM) GO TO 32
   IF(ICD3.EQ.IIC) GO TO 33
   IF(ICD3.EQ.ICE) GO TO 34
   IF(ICD3.EQ.ICD) GO TO 35
GO TO 37
23 CALL COMPAC(S1,NSET,SEL)
GO TO 12
C
BC NFLXE = NSET
CO 81 I=1,NBUD
   SFXM(I) = S1(I)
81 NMDDS = NMDDS + SFXM(I)
C CREATE THE SET SFLX
   II = 0
   CO 82 I=1,NBUD
     S1(I) = 0
     IF(SFXM(I).EQ.0) GO TO 88
     II = II + 1
     S1(II) = I
88 CONTINUE
   CALL COMPAC(S1,NFLXB,SFLX)
C READ IN ALL FLEXIBLE BODY MODAL DATA
   CALL UNPAC(S1,NS1,SFLX)
   MN = 0
   MM = 0
   PRINT 201
   PRINT 242, (S1(I),I=1,NS1)
   CO 85 II=1,NS1
     I = S1(NS1+1-II)
     READ 103, N, (MESS(J),J=1,18)
     PRINT 241, N, (MESS(J),J=1,18)
     IF(I.EQ.N) GO TO 86
     PRINT 232
     FGB = .FALSE.
     RETURN
86 SFXMI = SFXM(I)
   CO 87 N=1,SFXMI
     MM = MM + 1
     MN = MN + 1
     READ 104, FLOM(MN),ZETA(MN)
     READ 104, THA(NFER+MN),THAD(NFER+MN)
     READ 104, (FLA(K,MN),K=1,3)
     READ 104, (FLB(K,MN),K=1,3)
     READ 104, (FLC(K,MN),K=1,3)
     READ 104, ((FLD(K,L,MN),L=1,3),K=1,3)
     READ 104, ((FLJ(K,L,MN),L=1,3),K=1,3)
     PRINT 233, MN, MN,FLCM(MN),MN,ZETA(MN)
     NFMN = NFER + MN
     PRINT 235, NFMN,THA(NFMN),NFMN,THAD(NFMN)
     PRINT 234, MN, (FLA(K,MN),K=1,3),I
     PRINT 243
     PRINT 235, MN, (FLB(K,MN),K=1,3),I
     PRINT 243
     PRINT 236, MN, (FLC(K,MN),K=1,3),I
     PRINT 243
     PRINT 236, (FLD(1,L,MN),L=1,3)
     PRINT 237, MN, (FLC(2,L,MN),L=1,3),I

```

```

CO731600
CO731700
CO731800
CO731900
CO732000
CO732100
CO732200
CO732300
CO732400
CO732500
CO732600
CO732700
CO732800
CO732900
CO733000
CO733100
CO733200
CO733300
CO733400
CO733500
CO733600
CO733700
CO733800
CO733900
CO734000
CO734100
CO734200
CO734300
CO734400
CO734500
CO734600
CO734700
CO734800
CO734900
CO735000
CO735100
CO735200
CO735300
CO735400
CO735500
CO735600
CO735700
CO735800
CO735900
CO736000
CO736100
CO736200
CO736300
CO736400
CO736500
CO736600
CO736700
CO736800
CO736900
CO737000
CO737100
CO737200
CO737300
CO737400
CO737500

```

	PRINT 23E, (FLC(3,L,MN),L=1,3)	C0737600
	PRINT 243	C0737700
	PRINT 23E, (FLJ(1,L,MN),L=1,3)	C0737800
	PRINT 240, MN,(FLJ(2,L,MN),L=1,3),I	C0737900
	PRINT 23E, (FLJ(3,L,MN),L=1,3)	C0738000
	PRINT 21E	C0738100
	IF(MMN.NE.2) GO TO 87	C0738200
	PRINT 2C1	C0738300
	MMN = 0	C0738400
	87 CONTINUE	C0738500
	6E CONTINUE	C0738600
	GO TO 12	C0738700
C		C0738800
C	CONSTRUCT CENTRIPETAL AND CORIOLIS MODE COUPLING COEFC WORD	C0738900
	90 CONTINUE	C0739000
C	CHECK IF 'FLEXIBLE' READ YET	C0739100
	IF(NMODS.NE.0) GO TO 91	C0739200
	154 PRINT 232	C0739300
	FG3=.FALSE.	C0739400
	RETURN	C0739500
	91 DO 92 I=1,NMODS	C0739600
	92 SCXC(I) = 0	C0739700
	DO 93 I=1,NBOD	C0739800
	93 S2(I) = 0	C0739900
	NS2 = 0	C0740000
	DO 94 J=1,NBOD	C0740100
	IF(S1(J).EQ.0) GO TO 94	C0740200
	NS2 = NS2+1	C0740300
	S2(NS2) = J	C0740400
	94 CONTINUE	C0740500
	CALL COMFAC(S2,NS2,SFCC)	C0740600
	PRINT 2C1	C0740700
	PRINT 244, (S2(I),I=1,NS2)	C0740800
C	READ IN MODE COUPLING DATA, SET UP COUNTERS	C0740900
	MN = 0	C0741000
	KF = 0	C0741100
	DO 150 K=1,NBOD	C0741200
	IF(SFXM(K).EQ.0) GO TO 150	C0741300
	IF(S1(K).NE.0) GO TO 151	C0741400
C	NO CROSS COUPLING TERMS, UPDATE MODE NUMBER COUNTER MN	C0741500
	MN = MN+SFXM(K)	C0741600
	GO TO 15C	C0741700
	151 S1K = S1(K)	C0741800
	DO 152 I=1,S1K	C0741900
	READ 10E, MB,NB,KE	C0742000
	IF(K.NE.KB) GO TO 154	C0742100
	MNN = MN + NB	C0742200
	CALL UNPAC(S2,NS2,SCXC(MNN))	C0742300
	NS2 = NS2+1	C0742400
	S2(NS2) = MB	C0742500
	CALL COMFAC(S2,NS2,SCXC(MNN))	C0742600
	KF = KF+1	C0742700
	READ 104, ((FCF(I1,JJ,KF),JJ=1,3),I1=1,3)	C0742800
	PRINT 243	C0742900
	PRINT 24E, (FCF(1,JJ,KF),JJ=1,3)	C0743000
	PRINT 247, MB,NB,KE,(FCF(2,JJ,KF),JJ=1,3),K,KF	C0743100
	PRINT 24E, (FCF(3,JJ,KF),JJ=1,3)	C0743200
	PRINT 243	C0743300
	READ 104, (FCK(I1,KF),I1=1,3)	C0743400
	PRINT 24E, MB,NB,KE,(FCK(JJ,KF),JJ=1,3),K,KF	C0743500

FRINT 243	C0743600
152 CONTINUE	C0743700
C CHECK AND OUTPUT INTEGER ARRAYS FOR BODY K	C0743800
KKF = KF-S1(K)	C0743900
SFXMI = SFXM(K)	C0744000
DO 153 I=1,SFXMI	C0744100
MN = MN+1	C0744200
CALL UNPAC(S2,NS2,SCXC(MN))	C0744300
IF(NS2.NE.0) GO TO 155	C0744400
FRINT 243	C0744500
FRINT 245, MN	C0744600
GO TO 153	C0744700
155 DO 156 J=1,NS2	C0744800
MB = S2(NS2+1-J)	C0744900
KKF = KKF+1	C0745000
FRINT 250, K,MB,MN,KKF	C0745100
156 CONTINUE	C0745200
153 CONTINUE	C0745300
150 CONTINUE	C0745400
GO TO 12	C0745500
C	C0745600
24 CALL COMFAC(S1,NSET,SXT)	C0745700
GO TO 12	C0745800
25 CALL COMFAC(S1,NSET,SVD)	C0745900
GO TO 12	C0746000
62 CALL COMFAC(S1,NSET,SMV)	C0746100
NMV = NSET	C0746200
GO TO 12	C0746300
65 CALL COMFAC(S1,NSET,SMAL)	C0746400
GO TO 12	C0746500
77 SCG = NSET	C0746600
DO 95 I=1,SCG	C0746700
READ 101, SC(I),TEM	C0746800
99 TUC(SC(I)) = TEM	C0746900
C CHECK TO ON THAD(SC(I)); MUST BE ZERO, RESET IF NOT.	C0747000
C CODING FOR IMPULSE EFFECT DUE TO NON-ZERO THAD NOT INCLUDED	C0747100
DO 78 I=1,SCG	C0747200
IF(THAD(SC(I)).EQ.0.0) GO TO 78	C0747300
THAD(SC(I)) = 0.0	C0747400
FRINT 230, SC(I)	C0747500
78 CONTINUE	C0747600
GO TO 12	C0747700
26 CT2 = CT2+1	C0747800
JDCNE = CT2+NSET-1	C0747900
IF(CT2.GT.JDCNE) GO TO 5001	C0748000
DO 38 J=CT2,JDCNE	C0748100
38 SFR(J) = S1(J+1-CT2)	C0748200
5001 CONTINUE	C0748300
CT2 = CT2+NSET-1	C0748400
GO TO 12	C0748500
27 CT3 = CT3+1	C0748600
JDCNE = CT3+NSET-1	C0748700
IF(CT3.GT.JDCNE) GO TO 5002	C0748800
DO 39 J=CT3,JDCNE	C0748900
39 SLK(J) = S1(J+1-CT3)	C0749000
5002 CONTINUE	C0749100
CT3 = CT3+NSET-1	C0749200
GO TO 12	C0749300
28 CT4 = CT4+1	C0749400
JDCNE = CT4+NSET-1	C0749500

IF (C14.GT.JDCNE) GO TO 5003	C0749600
DO 40 J=CT4,JDONE	C0749700
40 SMA(J) = S1(J+1-CT4)	C0749800
5003 CONTINUE	C0749900
CT4 = CT4+NSET-1	C0750000
GO TO 12	C0750100
29 CTE = CTS+1	C0750200
JDCNE = CTS+NSET-1	C0750300
IF (C15.GT.JDONE) GO TO 5004	C0750400
DO 41 J=CTS,JDONE	C0750500
41 SFK(J) = S1(J+1-CTE)	C0750600
5004 CONTINUE	C0750700
CTE = CTE+NSET-1	C0750800
GO TO 12	C0750900
32 CALL COMFAC(S1,NSET,SPI(I))	C0751000
GO TO 12	C0751100
33 CALL COMFAC(S1,NSET,SIX(I))	C0751200
GO TO 12	C0751300
34 CALL COMFAC(S1,NSET,SCN(I))	C0751400
GO TO 12	C0751500
35 CALL COMFAC(S1,NSET,SCR(I))	C0751600
GO TO 12	C0751700
21 CT1 = 0	C0751800
IF (C12.EG.0) GO TO 42	C0751900
NFAC = CT2	C0752000
CT2 = 0	C0752100
42 IF (C13.EG.0) GO TO 43	C0752200
NCTC = CT3	C0752300
CT3 = 0	C0752400
43 IF (C14.EG.0) GO TO 44	C0752500
NMCA = CT4	C0752600
CT4 = 0	C0752700
44 IF (C15.EG.0) GO TO 45	C0752800
NFKC = CT5	C0752900
CT5 = 0	C0753000
45 CONTINUE	C0753100
C	C0753200
C	C0753300
C OBTAIN ELEMENTS OF ARRAY SXM	C0753400
C INITIALIZE BOTH SXM AND TH ARRAYS	C0753500
DO 3 K=1,NBOD	C0753600
DO 3 I=1,3	C0753700
SXM(I,K) = 0	C0753800
3 CONTINUE	C0753900
C DEFINE CONTIGUOUS PAIRS FOR WHICH DIRECTION COSINES ARE TO BE USED	C0754000
SD = SR-SMAL-SEU	C0754100
C DEFINE CONTIGUOUS PAIRS FOR WHICH SMALL ANGLE OR EULER ANGLE	C0754200
C TECHNIQUES TO BE USED	C0754300
SML = SR - SD	C0754400
CALL UNFAC(S1,NS1,SML)	C0754500
C GO THROUGH ELEMENTS OF SPL TO DEFINE SXM ARRAY	C0754600
IF (NS1.EG.0) GO TO 5080	C0754700
DO 4 KK=1,NS1	C0754800
K = S1(KK)	C0754900
C DEFINE NOMINAL STATE TRANSFORMATION FROM BODY JCCN(K) TO BODY K	C0755000
CALL TRNPS (XMT(1.1,K))	C0755100
C PICK OUT GIMBAL CONFIGURATION AT HINGE POINT K-1	C0755200
IGCTC = FCON(K) + 1	C0755300
GO TO (13,14,15,16),IGLTC	C0755400
16 CALL TRNPS (XMT(1.1,K))	C0755500

GO TC 4	C075560C
15 NGA = 3	C0755700
M = SQF(K)	C0755800
L = SQL(K)	C0755900
CALL VECTR N (QF(1,M),XMT(1,1,K),QFK(1,1))	C0756000
DO 75 I=1,3	C0756100
75 QFK(1,2) = QL(I,L)	C0756200
CALL VECROS (QFK(1,1),QFK(1,2),QFK(1,3))	C0756300
GO TC 7	C0756400
14 NGA = 3	C0756500
M = SQF(K)	C0756600
M1 = M+1	C0756700
CALL VECTR N (QF(1,M),XMT(1,1,K),QFK(1,1))	C0756800
DO 8 I=1,3	C0756900
8 QFK(1,2) = QF(I,M1)	C0757000
CALL VECROS (QFK(1,1),QFK(1,2),QFK(1,3))	C0757100
GO TC 7	C0757200
13 NGA = 3	C0757300
M = SQF(K)	C0757400
M1 = M+1	C0757500
M2 = M+2	C0757600
CALL VECTR N (QF(1,M),XMT(1,1,K),QFK(1,1))	C0757700
DO 9 I=1,3	C0757800
9 QFK(1,3) = QF(I,M2)	C0757900
CALL VECROS (QFK(1,3),QFK(1,1),QFK(1,2))	C0758000
GO TC 7	C0758100
7 IF(NGA.EC.0) GO TC 16	C0758200
DO 17 N=1,NGA	C0758300
DO 17 I=1,3	C0758400
IF(QFK(I,N).NE.0) GO TC 18	C0758500
GO TC 17	C0758600
18 IF(QFK(I,N).NE.1) GO TC 19	C0758700
SXM(N,K) = 1	C0758800
GO TC 17	C0758900
19 IF(QFK(I,N).NE.-1) GO TC 58	C0759000
SXM(N,K) = -1	C0759100
GO TC 17	C0759200
58 CONTINUE	C0759300
C CME HERE ONLY IF FREE COORDINATE AXES NOT ALIGNED WITH	C0759400
C BODY K FIXED AXES IN NOMINAL STATE.	C0759500
C SMALL ANGLE OR EULER ANGLE METHODS CANNOT BE USED, INCLUSION	C0759600
C OF THIS CAPABILITY OF LIMITED VALUE SINCE IT WOULD SACRIFICE	C0759700
C COMPUTATION SPEED AND MEMORY STORAGE.	C0759800
C PUT K BACK IN SD AND DELETE IT FROM SEU OR SMAL	C0759900
S2(1) = K	C0760000
CALL COMFAC(S2,1,NK)	C0760100
SD = SD + NK	C0760200
CALL UNPAC(S3,NS3,SEU)	C0760300
IF(NS3.EC.0) GO TO 5020	C0760400
DO 71 JJ=1,NS3	C0760500
J = S3(JJ)	C0760600
IF(J.NE.K) GO TO 71	C0760700
SEL = SEC - NK	C0760800
GO TC 16	C0760900
71 CONTINUE	C0761000
5020 CONTINUE	C0761100
CALL UNPAC(S3,NS3,SMAL)	C0761200
IF(NS3.EC.0) GO TC 16	C0761300
DO 72 JJ=1,NS3	C0761400
J = S3(JJ)	C0761500

IF(J.NE.K) GO TO 72	C0761600
SMAL = SMAL - NK	C0761700
GO TC 16	C0761800
72 CONTINUE	C0761900
GO TC 16	C0762000
17 CONTINUE	C0762100
GO TC 16	C0762200
4 CONTINUE	C0762300
5080 CONTINUE	C0762400
C	C0762500
C MAKE SURE ALL ANGLES CALLED FOR BY SXM WILL BE COMPUTED	C0762600
DO 36 K=1,NB0D	C0762700
IF(SXM(1,K).EQ.0) GO TC 36	C0762800
MDCNE = 3-PCDN(K)	C0762900
DO 45 LL=1,MDCNE	C0763000
L = LL-1	C0763100
M = SQF(K) + L	C0763200
IF(NFRC.EQ.0) GO TC 5021	C0763300
DO 57 N=1,NFRC	C0763400
IF(M.EQ.SFR(N)) GO TO 49	C0763500
57 CONTINUE	C0763600
5021 CONTINUE	C0763700
C COME HERE IF M NOT IN SFR	C0763800
NFRC = NFRC+1	C0763900
SFR(NFRC) = M	C0764000
49 CONTINUE	C0764100
36 CONTINUE	C0764200
C	C0764300
C MAKE SURE TRANSLATION COMPONENTS OF C.M. OF BODY 1 (HINGE POINT 0)	C0764400
C ARE COMPUTED SO AS TO BE ABLE TO DEFINE CB(1), NEEDED FOR INERTIA	C0764500
C ANGULAR MOMENTUM AND ENERGY CALCULATION	C0764600
MDCN1 = SQF(NB1)	C0764700
MDCNE = SQF(NB1)+2-PCCN(NB1)	C0764800
IF(MDCN1.EQ.0) GO TO 5033	C0764900
DO 73 M=MDCN1,MDCNE	C0765000
IF(NFRC.EQ.0) GO TC 5022	C0765100
DO 74 N=1,NFRC	C0765200
IF(M.EQ.SFR(N)) GO TO 73	C0765300
74 CONTINUE	C0765400
5022 CONTINUE	C0765500
NFRC = NFRC+1	C0765600
SFR(NFRC) = M	C0765700
73 CONTINUE	C0765800
5033 CONTINUE	C0765900
C	C0766000
C MAKE SURE ALL POINT MASS COORDINATES IN SQF ARRAY	C0766100
CALL UNPAC(S1,N1,SL)	C0766200
IF(N1.EQ.0) GO TO 5018	C0766300
DO 96 II=1,N1	C0766400
COO(I) = S1(II)	C0766500
MDCN1 = SQF(I)	C0766600
MDCNE = SQF(I) + 2 - PCCN(I)	C0766700
IF(MDCN1.EQ.0) GO TO 5016	C0766800
DO 97 M=MDCN1,MDCNE	C0766900
IF(NFRC.EQ.0) GO TC 5017	C0767000
DO 98 N=1,NFRC	C0767100
IF(M.EQ.SFR(N)) GO TO 97	C0767200
98 CONTINUE	C0767300
5017 CONTINUE	C0767400
NFRC = NFRC + 1	C0767500

SFR(NFRC) = M	C0767600
57 CONTINUE	C0767700
5016 CONTINUE	C0767800
56 CONTINUE	CC767900
5018 CONTINUE	CC768000
C	C0768100
C FIND ALL MOMENTUM WHEELS IN THE NEXT K-1; K=1,2,...,NBOD	C0768200
DO 66 II=1,NBOD	C0768300
I=II-1	C0768400
CALL UNFAC(S1,N1,SK(I))	C0768500
DO 67 J=1,10	C0768600
67 S2(J) = C	C0768700
N2 = C	C0768800
IF(NM0.EQ.0) GO TO 5023	C0768900
DO 68 J=1,NM0	C0769000
IF(.NOT.CTAIN(MU(J),S1,N1)) GO TO 68	C0769100
N2 = N2 + 1	C0769200
S2(N2) = J	C0769300
68 CONTINUE	C0769400
5023 CONTINUE	C0769500
CALL COMFAC(S2,N2,SMC(I))	C0769600
66 CONTINUE	C0769700
C	C0769800
C IF COMPLETING FRAME BODY I DELETE LABEL I FROM SVD	C0769900
CALL UNPAC(S1,NSET,SVD)	CC770000
IF(INEFF.OR.S1(NSET).EQ.C) GO TO 76	C0770100
NSET = NSET - 1	C0770200
CALL CCMPAC(S1,NSET,SVD)	C0770300
76 CONTINUE	CC770400
C	CC770500
C ALL TRUNCATED SUMMATIONS DEFINED	C0770600
C PRINT THEM OUT	C0770700
C	C0770800
PRINT 201	C0770900
IF(INERF) GO TO 46	CC771000
PRINT 203	C0771100
GO TO 47	C0771200
46 PRINT 202	C0771300
47 PRINT 204	C0771400
PRINT 205	C0771500
PRINT 206	C0771600
DO 48 II=1,NBOD	CC771700
I=II-1	C0771800
CALL UNPAC(S1,NSET,SK(I))	C0771900
PRINT 207, I,(S1(J),J=1,NSET)	C0772000
CALL UNPAC(S1,NSET,SMC(I))	C0772100
PRINT 222, I,(S1(J),J=1,NSET)	CC772200
PRINT 224	C0772300
CALL UNPAC(S1,NSET,SPI(I))	C0772400
PRINT 209, (S1(J),J=1,NSET)	CC772500
CALL UNPAC(S1,NSET,SIX(I))	C0772600
PRINT 210, (S1(J),J=1,NSET)	C0772700
CALL UNPAC(S1,NSET,SCN(I))	CC772800
PRINT 211, (S1(J),J=1,NSET)	C0772900
CALL UNPAC(S1,NSET,SCR(I))	CC773000
PRINT 212, (S1(J),J=1,NSET)	CC773100
48 PRINT 225, I	C0773200
PRINT 201	C0773300
CALL UNPAC(S1,NSET,SD)	C0773400
IF(NSET.EQ.0) GO TO 5005	C0773500

CC 50 I=1,NSET	C0773600
50 PRINT 214, S1(I),JCON(S1(I))	C0773700
FRINT 215	C0773800
5005 CONTINUE	C0773900
(CALL UNPAC(S1,NSET,SMAL)	C0774000
IF(NSET.EQ.0) GO TO 5006	C0774100
DO 55 I=1,NSET	C0774200
55 FRINT 22E, S1(I),JCON(S1(I))	C0774300
PRINT 215	C0774400
5006 CONTINUE	C0774500
(CALL UNPAC(S1,NSET,SEU)	C0774600
IF(NSET.EQ.0) GO TO 5007	C0774700
DO 70 I=1,NSET	C0774800
70 PRINT 225, S1(I),JCON(S1(I))	C0774900
FRINT 215	C0775000
5007 CONTINUE	C0775100
(CALL UNPAC(S1,NSET,SXT)	C0775200
IF(NSET.EQ.0) GO TO 5008	C0775300
DO 51 I=1,NSET	C0775400
51 FRINT 216,S1(I)	C0775500
FRINT 215	C0775600
5008 CONTINUE	C0775700
(CALL UNPAC(S1,NSET,SVD)	C0775800
IF(NSET.EQ.0) GO TO 5009	C0775900
DO 52 I=1,NSET	C0776000
52 FRINT 217,S1(I)	C0776100
FRINT 215	C0776200
5009 CONTINUE	C0776300
C PUT ARRAY SFR IN SEQUENTIAL ORDER	C0776400
DO 61 I=1,NB3	C0776500
61 S4(I) = C	C0776600
IF(NFRC.EQ.0) GO TO 5010	C0776700
DO 63 I=1,NFRC	C0776800
63 S4(SFR(I)) = I	C0776900
5010 CONTINUE	C0777000
K = C	C0777100
DO 64 I=1,NB3	C0777200
IF(S4(I).EQ.C) GO TO 64	C0777300
K = K+1	C0777400
SFR(K) = I	C0777500
64 CONTINUE	C0777600
IF(NFRC.EQ.0) GO TO 5011	C0777700
DO 53 I=1,NFRC	C0777800
53 PRINT 218, SFR(I)	C0777900
PRINT 215	C0778000
5011 CONTINUE	C0778100
IF(NCTC.EQ.0) GO TO 5012	C0778200
DO 54 I=1,NCTC	C0778300
54 PRINT 219, SLK(I)	C0778400
PRINT 215	C0778500
5012 CONTINUE	C0778600
IF(NFKC.EQ.0) GO TO 5013	C0778700
DO 55 I=1,NFKC	C0778800
55 PRINT 220, SFK(I)	C0778900
PRINT 215	C0779000
5013 CONTINUE	C0779100
(CALL UNPAC(S1,NSET,SMV)	C0779200
IF(NSET.EQ.0) GO TO 5014	C0779300
DO 65 I=1,NSET	C0779400
65 PRINT 213, S1(I)	C0779500



```

FRINT 215
5014 CCNTINUE
IF(NMOA.EQ.0) GO TO 5015
DO 56 I=1,NMOA
56 PRINT 221, SMA(I)
FRINT 215
5015 CCNTINUE
IF(SCG.EQ.0) GO TO 83
DO 82 I=1,SCG
82 PRINT 231, SC(I),TUG(SC(I))
83 CONTINUE
CT1 = 0
100 FORMAT (3A4,I3,11I5)
101 FORMAT (15,D15.5)
102 FORMAT (15,20I5.5)
103 FORMAT (15,18A4)
104 FORMAT (3D15.5)
105 FORMAT (5I5)
200 FORMAT (' IDENTIFICATION CODE NOT RECOGNIZED IN SUBROUTINE INCPT.
*INFUT OPTION CARD',I4,' (CODE READ IS ',J4)
201 FORMAT ('')
202 FORMAT (10X,'COMPLTING FRAME TAKEN TO BE THAT FIXED INERTIALLY',//
*)
203 FORMAT (10X,'COMPLTING FRAME TAKEN TO BE THAT FIXED IN BODY 1',//
*)
204 FORMAT (30X,'TU, SPEED UP COMPUTATION VARIOUS TRUNCATED SETS OF BOD
*Y LABELS HAVE BEEN DEFINED')
205 FORMAT (34X,'THE SPECIFICATION OF THESE SETS PERMITS ENGINEERING J
*JUDGMENT TO')
206 FORMAT (38X,'BE INTRODUCED INTO THE FORMALISM AND MODELLING ',//)
207 FORMAT (30X,'BODY LABELS OF BODIES IN NEST',I3,3X,10I5)
208 FORMAT (20X,10I5,/)
209 FORMAT (30X,'PSUEDO INERTIA TENSORS',I3X,10I5)
210 FORMAT (30X,'INERTIA CROSS COUPLING',I3X,10I5)
211 FORMAT (30X,'CENTRIPITAL CRSS COUPLING',9X,10I5)
212 FORMAT (30X,'CORICLIS CRSS COUPLING',12X,10I5)
213 FORMAT (10X,'MCMENTUM WHEEL',I3,' IS ASSUMED TO BE VARIABLE SPEED
*)
214 FORMAT (10X,'RELATIVE ANGULAR DISPLACEMENT BETWEEN BODIES',I3,' AN
*D',I3,' IS COMPUTED VIA INTEGRATION OF DIRECTION COSINE EQUATIONS'
*)
215 FORMAT (////)
216 FORMAT (10X,'THE ELEMENTS OF COLUMN',I3,' DOWN TO THE DIAGONAL IN
*THE SYSTEM INERTIA MATRIX OF DYADS ARE ASSUMED TIME VARYING')
217 FORMAT (10X,'VECTCRS AND CYADS FIXED IN BODY',I3,' ARE ASSUMEC TIM
*E VARYING IN THE FRAME OF COMPUTATION ')
218 FORMAT (10X'DISPLACEMENT ABOUT UR ALONG FREE VECTOR',I3,' CCMPUTED
*)
219 FORMAT (10X,'CCNSTRANT 1CRQUE ABOUT UR ALONG LOCKED VECTOR',I3,'
*COMPUTED ')
220 FORMAT (10X,'CCNSTRANT FCRCF AT HINGE POINT',I3,' COMPUTED ')
221 FORMAT (10X,'ANGULAR PCSITICN OF MCMENTUM WHEEL',I3,' COMPUTEC ')
222 FORMAT (30X,'MOMENTUM WHEEL LABELS IN NEST',I3,3X,10I5)
224 FORMAT (20X,'PRIME CONTRIBUTORS TO COMPUTATION OF ')
225 FORMAT (20X,'FOR THE EQUATION OF MOTION OF NEST',I3,///)
228 FORMAT (10X,'RELATIVE ANGULAR DISPLACEMENT BETWEEN BODIES',I3,' AN
*D',I3,' IS COMPUTED VIA SMALL ANGLE ASSUMPTIONS ')
229 FORMAT (10X,'RELATIVE ANGULAR DISPLACEMENT BETWEEN BODIES',I3,' AN
*D',I3,' IS COMPUTED VIA EULER ANGLE TECHNIQUES ')
230 FORMAT (10X,'NOTE: THAC(',I2,') = 0 INITIAL RATE CONDITION RESET

```

```

* TC ZERO, SEE SUB INOPT AND MAIN *)
231 FORMAT (10X, 'MOTION ABOUT FREE VECTOR', I3, ' UNCAGED AT T=', D15.5)
232 FORMAT (' MODAL DATA CUT OF SEQUENCE ')
233 FORMAT (' MODE', I3, 2X, ' FLCM(', I2, ') =', D12.5, 5X, ' ZETA(', I2, ') =', D12.5)
234 FORMAT (' FLA(', I2, ') =', D12.5, 3X, ' (BODY', I2, ' FIXED COORDINATE', I2)
235 FORMAT (' FLB(', I2, ') =', D12.5, 3X, ' (BODY', I2, ' FIXED COORDINATE', I2)
236 FORMAT (' FLC(', I2, ') =', D12.5, 3X, ' (BODY', I2, ' FIXED COORDINATE', I2)
237 FORMAT (' FLD(', I2, ') =', D12.5, 3X, ' (BODY', I2, ' FIXED COORDINATE', I2)
238 FORMAT (11X, 3D12.5)
239 FORMAT (11X, ' THA(', I2, ') =', D12.5, 5X, ' THAD(', I2, ') =', D12.5, /)
240 FORMAT (' FLJ(', I2, ') =', D12.5, 3X, ' (BODY', I2, ' FIXED COORDINATE', I2)
241 FORMAT (///, 10X, ' MODAL DATA FOR BODY', I5, 18A4, /)
242 FORMAT (10X, ' THE FOLLOWING BODIES HAVE BEEN REDEFINED TO BE FLEXIBLE', I5)
243 FORMAT (' ')
244 FORMAT (10X, ' THE FOLLOWING FLEXIBLE BODIES HAVE SIGNIFICANT MODE COUPLING IN THEIR DEFORMATION EQUATIONS', I5)
245 FORMAT (17X, 3D12.5)
246 FORMAT (' FCF(', I2, ', ', I2, ', ', I2, ') =', D12.5, ' (BODY', I2, ' FIXED COORDINATES) LOCATED AT FCF(1,1,', I2, ')')
247 FORMAT (' FCK(', I2, ', ', I2, ', ', I2, ') =', D12.5, ' (BODY', I2, ' FIXED COORDINATES) LOCATED AT FCK(1,', I2, ')')
248 FORMAT (' SCXC(', I2, ') = 0')
249 FORMAT (' BODY', I2, ' MODE', I3, ' CROSS COUPLES IN COORDINATE EQUATION', I2)
250 FORMAT (' TICN', I2, ' COEFFICIENTS AT KF =', I3)
RETURN
END

```

```

C SUERCUTINE INTER
C
C IMPLICIT REAL*8(A-F, D-Z, I)
C LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLU, LEQU, LIMIT(1)
C LOGICAL NSTART, LRTAPE
C
C INTEGER
C * AMCRK, CT1, CT2, CT3, CT4, CT5, FCUN, FCON,
C * SCNDUM, SCN, SCRUM, SCF, SFKDUM, SFK, SFR, SG,
C * SI, SIG, SIXDUM, SIX, SKDUM, SK, SL, SLK,
C * SMA, SMCDUM, SMC, SMV, SCK, SPIDUM, SPI, SQF,
C * SGL, SR, SSCN, SSIX, SVA, SVB, SVD, SVI,
C * SVM, SVP, SVQ, SXM, SXT, TORQ, SMAL, SEU,
C * SC, SCG, NFLXR, SFLX, SFXM, NMCD, SFCC, SCC,
C * IINIT(1), IZINIT(1), SD
C
C REAL*8
C * ANGC (3,3), CNF (3,10), ETIC (3,10), ETMC (3,10)

```

```

* FLQ (3,20) , FLE (3,3,20), FLH (3,3,20), CC02200
* THAD (33) , YMCD (3,2,11), RINIT (1) , RZINIT(1) CC02300
C CC02400
COMMON /CHEKS/ NSTART, LRTAPE CC02500
C CC02600
C CC02700
C CC02800
COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INEFF, RBLU(10) CC02900
C CC03000
C CC03100
COMMON /INTG/ AWORK (200) , CC03200
* CT1 , CT2 , CT3 , CT4 , CC03300
* CTS , FCON (33) , JCON (10) , LCON (22) , CC03400
* MC (10) , NBI , NBUD , NCTC , CC03500
* NFER , NFKC , NFRG , NLGR , CC03600
* NMV , NMD , NMCA , NSVP , CC03700
* NSVG , FCCN (11) , SO , SFR (33) , CC03800
* SG , SI (55) , SIG , SL , CC03900
* SLK (33) , SMA (10) , SCK (11) , SQF (11) , CC04000
* SGL (11) , SMV , SR , SSCN , CC04100
* SEI , SVA , SVB , SVC , CC04200
* SVI , SVM , SVP (22) , SVQ (33) , CC04300
* SXM (3,10) , SXT , TURQ (97) , SMAL , CC04400
* SEU , NTQ , SC (33) , SCG , CC04500
* NFLXB , SFLX , SFXM (10) , NMGDS , CC04600
* SFCC , SCC (10) CC04700
C CC04800
C CC04900
COMMON /INTGZ/ CC05000
* SCNDLM , SCN (5) , SCRDM , SCR (9) , CC05100
* SFKDLM , SFK (5) , SIXDM , SIX (9) , CC05200
* SKDLM , SK (5) , SPIDUM , SPI (9) , CC05300
* SPCDLM , SPC (5) CC05400
C CC05500
C CC05600
COMMON /REAL/ CC05700
* CA (3,10) , CAC (3,10) , CLM (10) , CGMC (3,11) , CC05800
* DCMC (3,11) , ETC (3,11) , ETM (33) , FOMC (3,11) , CC05900
* GAM (3,66) , H , HM (3,10) , HMC (3,10) , CC06000
* HMOM (10) , PHI (3,11) , PLM (10) , QF (3,33) , CC06100
* OFC (3,33) , QL (3,22) , QLC (3,22) , RCMC (3,11) , CC06200
* T , THA (33) , THAD (33) , CC06300
* THADW (10) , THAW (10) , XDIC (3,3,66) , XI (3,3,10) , CC06400
* XIC (3,3,10) , XMAS (10) , XMN (33,33) , XMT (3,3,10) , CC06500
* TLG (33) , FLA (3,20) , FLB (3,20) , FLC (3,20) , CC06600
* FLD (3,3,20) , FLJ (3,3,20) , CAO (3,10) , XIU (3,3,10) , CC06700
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLQC (3,20) , CC06800
* FLCW (20) , ZETA (20) , FCF (3,40) , FCK (3,3,40) , CC06900
* TIMEND CC07000
C CC07100
C CC07200
COMMON /REALZ/ CC07300
* CEDLM (1,3) , CB (3,10) , CBCDUM(1,3) , CBC (3,10) , CC07400
* XMCDLM(1,1,9) , XMC (3,3,10) , CBN(3) , CC07500
C CC07600
COMMON /SATELL/ DUMMY(1000) CC07700
C CC07800
EQUIVALENCE (ETM(1),THAD(1)) ,(XMN(1,1),ANGD(1)) , CC07900
* (XMN(1,3),YMCD(1,1,1)) ,(XMN(1,6),CNF(1,1)) , CC08000
* (XMN(1,8),ETIC(1,1)) ,(XMN(1,10),ETMC(1,1)) , CC08100

```

```

*          (FLB(1,1),FLG(1,1))          ,(FLE(1,1,1),FLD(1,1,1)),      CCE08200
*          (FLH(1,1,1),FLJ(1,1,1))      ,                               C0808300
*          (FGI,LINIT(1))                ,(CA(1,1),RINIT(1))          ,      CCEC8400
*          (CBDUM(1,1),RZINIT(1))        ,(AWORK(1),IINIT(1))          ,      C0808500
*          (SCNDLM,IZINIT(1))            ,                               CCEC8600
C                                               C08C8700
C                                               C0808800
RETURN                                          C0808900
ENC                                             CCE09000

```

```

C                                               C0500000
SUBROUTINE TRNSIV(XMT,GF,THA,JCON,PCCN,NBOD,RBLJ,INERF,XMCDUM,XMC) C0500100
C  COMPUTE INITIAL TRANSFORMATION MATRICES                                C0500200
C  BODY K TO COMPUTING FRAME COORDINATES                                C0500300
C                                               C0500400
C  ENTER SUBROUTINE WITH                                                C0500500
C  XMT = NOMINAL STATE TRANSFORMATION MATRICES BODY K TO BODY JCC0500600
C  GF = FREE COORDINATE VECTOR, EIGENVECTORS                            C0500700
C  THA = ROTATION ABOUT RESPECTIVE EIGENVECTORS                          C0500800
C  RETURN FROM SUBROUTINE WITH                                           C0500900
C  XMC = INITIAL TRANSFORMATION MATRICES BODY K TO COMPUTING FRAME C0501000
C                                               C0501100
IMPLICIT REAL*8(A-H,C-Z,I)                                               C0501200
LOGICAL      RBLD( 1),INERF,LEQU                                         C0501300
LOGICAL      LRUNGE , LTRNSI , LVDIV , LEQUIV , LTRAN ,                 C0501400
*            LTRANV , LRATE , LXDY , LETA , LTORQU ,                     C0501500
*            LQFDOT , LDCT , LANGLE , LSETUP , LSIMO                      C0501600
INTEGER      JCON( 1),PCCN( 1)                                           C0501700
DIMENSION   XMT(3,3,1),GF(3,1),THA( 1)                                   C0501800
DIMENSION   XMCDUM(1,1,9),XMC(3,3,1)                                     C0501900
DIMENSION   XTMP(3,3),XTM1(3,3),XTM2(3,3)                               C0502000
DIMENSION   QT1(3),ZT1(4),ZT2(4),ZT3(4),ZT4(4),QT2(3)                  C0502100
COMMON /LDEBUG/ LRUNGE , LTRNSI , LVDIV , LEQUIV , LTRAN ,             C0502200
*            LTRANV , LRATE , LXDY , LETA , LTORQU ,                     C0502300
*            LQFDOT , LDCT , LANGLE , LSETUP , LSIMO                      C0502400
EQUIVALENCE (LTRNSI,LEQU)                                                C0502500
C                                               C0502600
C  COMPUTE TRANSFORMATION MATRICES WHICH TAKE VECTORS FROM BODY JCCN(C0502700
C  TO BODY K COORDINATES                                                C0502800
C                                               C0502900
C  NOTE XMT TAKES VECTORS BODY K TO JCCN(K) IN NOMINAL STATE           C0503000
* = 1                                                                    C0503100
IF(.NOT. LEQU) GO TO 1001                                                C0503200
PRINT 200                                                                C0503300
PRINT 230                                                                C0503400
DO 19 K=1,NBOD                                                            C0503500
PRINT 201                                                                C0503600
PRINT 202, (XMT(I,J,K),J=1,3)                                           C0503700
PRINT 203, K,(XMT(2,J,K),J=1,3)                                         C0503800
--19 PRINT 202, (XMT(3,J,K),J=1,3) --                                     C0503900
C01001 DO 1 K=1,NBOD                                                       C0504000
IF(LEQU) K1 = K-1                                                         C0504100
C  XMT(I,J,K) - 3X3 TRANSFORMATION MATRIX BODY K TO JCCN(K)           C0504200
C  XTMP(I,J) - 3X3 TRANSFORMATION MATRIX BODY JCCN(K) TO K           C0504300
DO 2 I=1,3                                                                C0504400
DO 2 J=1,3                                                                C0504500

```

2	XTMP(I,J) = XMT(J,I,K)	C0904600
	IF(.NOT. LEQU) GO TO 1002	C0904700
	PRINT 204	C0904800
	PRINT 231, K1	C0904900
	PRINT 205, K	C0905000
C		C0905100
C	COMPLT FREE COORDINATE LABEL	C0905200
1002	IF(K.EQ.1) GO TO 3	C0905300
	M = N + 3 - PCON(K-1)	C0905400
3	M1 = M + 1	C0905500
	M2 = M + 2	C0905600
C		C0905700
C	CHECK RIGID BODY OF LINEAR OSCILLATOR	C0905800
	IF(REQ(K).AND.PCCN(K).NE.3) GO TO 4	C0905900
C	LINEAR OSCILLATOR OR THREE CONSTRAINED AXES. JCON(K) TO K	C0906000
	DO 5 I=1,3	C0906100
	DO 5 J=1,3	C0906200
5	XMC(I,J,K) = XTMP(I,J)	C0906300
	IF(LEQU) PRINT 206, K	C0906400
	GO TO 1	C0906500
4	CONTINUE	C0906600
C	BODY K IS A RIGID BODY WHICH IS CONNECTED TO BODY JCON(K)	C0906700
C	BY EITHER A ONE, TWO OR THREE AXIS GIMBAL	C0906800
C		C0906900
C	PUT FREE VECTOR M IN COORDINATE FRAME I SUB N	C0907000
	CALL VECTR(N(QF(I,M),XTMP,QT1))	C0907100
	IF(LEQU) PRINT 207, M	C0907200
C	FORM ROTATION QUATERNION EQUATION FRAME I SUB N INTO I SUB 1	C0907300
	CALL QUATOP(QT1,THA(M),ZT1)	C0907400
	IF(LEQU) PRINT 208,M,(ZT1(I),I=1,4)	C0907500
C		C0907600
	IF(PCCN(K).NE.2) GO TO 6	C0907700
C	BODY K CONNECTED TO BODY JCON(K) BY A ONE AXIS GIMBAL	C0907800
C	COORDINATE FRAME I SUB 1 IS BODY K FIXED FRAME	C0907900
C		C0908000
	CALL TRANSO(ZT1,XTM1)	C0908100
	IF(LEQU) PRINT 209	C0908200
C	FORM INITIAL TRANSFORMATION MATRIX JCON(K) TO K	C0908300
	CALL MATMUL(XTM1,XTMP,XMC(1,1,K),3)	C0908400
	IF(LEQU) PRINT 210, K	C0908500
	GO TO 1	C0908600
C		C0908700
6	IF(PCCN(K).NE.1) GO TO 7	C0908800
C	BODY K CONNECTED TO BODY JCON(K) BY A TWO AXIS GIMBAL	C0908900
C	FREE VECTOR M1 GIVEN IN BODY K COORDINATES. FORM ROTATION QUATERNION	C0909000
	CALL QUATOP(QF(1,M1),THA(M1),ZT2)	C0909100
	IF(LEQU) PRINT 211, M1,M1,(ZT2(I),I=1,4)	C0909200
C	FORM RESULTANT ROTATION QUATERNION	C0909300
	CALL QUTMUL(ZT1,ZT2,ZT4)	C0909400
	IF(LEQU) PRINT 212, (ZT4(I),I=1,4)	C0909500
	CALL TRANSO(ZT4,XTM1)	C0909600
	IF(LEQU) PRINT 213	C0909700
C	FORM INITIAL TRANSFORMATION MATRIX JCON(K) TO K	C0909800
	CALL MATMUL(XTM1,XTMP,XMC(1,1,K),3)	C0909900
	IF(LEQU) PRINT 214, K	C0910000
	GO TO 1	C0910100
C		C0910200
C	BODY K CONNECTED TO BODY JCON(K) BY A THREE AXIS GIMBAL	C0910300
7	CALL VECROS(QF(1,M2),QT1,QT2)	C0910400
	CALL VECARM(QT2)	C0910500

```

IF(LEQU) PRINT 215,M2                                C0910600
C THE COMPONENTS OF FREE VECTOR M1 FOR A THREE AXIS GIMBAL IN THE C0910700
C INTERMEDIATE FRAME I SUB I ARE IDENTICAL TO ITS COMPONENTS COMPUT C0910800
C WHEN SYSTEM IN NOMINAL STATE                        C0910900
CALL QJATOP(QT2,THA(M1),ZT2)                          C0911000
IF(LEQU) PRINT 216, M1,(ZT2(I),I=1,4)                 C0911100
CALL QJATOP(QF(1,M2),THA(M2),ZT3)                    C0911200
IF(LEQU) PRINT 217, M2,M2,(ZT3(I),I=1,4)             C0911300
C FORM RESULTANT QUATERNION BY SUCCESSIVE QUATERNION MULTIPLICATION C0911400
CALL QJTMUL (ZT1,ZT2,ZT4)                             C0911500
IF(LEQU) PRINT 218                                    C0911600
CALL QJTMUL (ZT4,ZT3,ZT1)                             C0911700
IF(LEQU) PRINT 219, (ZT1(I),I=1,4)                   C0911800
CALL TRANSQ(ZT1,XTM1)                                 C0911900
IF(LEQU) PRINT 220                                    C0912000
C FORM INITIAL TRANSFORMATION MATRIX JCON(K) TO K    C0912100
CALL MATMUL(XTM1,XTMP,XMC(1,1,K),3)                  C0912200
IF(LEQU) PRINT 214, K                                 C0912300
1 CONTINUE                                           C0912400
IF(LEQU) PRINT 204                                    C0912500
C                                                     C0912600
C ALL CONTIGUOUS BODY TRANSFORMATION MATRICES COMPUTED C0912700
C XMC(I,J,K) - 3X3 TRANSFORMATION MATRIX BODY JCON(K) TO BODY K AT C0912800
IF(.NOT. LEQU) GO TO 100C                             C0912900
PRINT 232                                             C0913000
CC 20 K=1,NBOD                                       C0913100
PRINT 201                                             C0913200
PRINT 221, (XMC(1,J,K),J=1,3)                       C0913300
PRINT 222, K,(XMC(2,J,K),J=1,3)                     C0913400
20 PRINT 221, (XMC(3,J,K),J=1,3)                     C0913500
PRINT 204                                             C0913600
C                                                     C0913700
C COMPUTE TRANSFORMATION MATRICES, COMPUTING FRAME TO BODY K C0913800
C                                                     C0913900
100C IF(INERF) GO TO 8                                C0914000
IF(LEQU) PRINT 223                                    C0914100
IF(LEQU) PRINT 201                                    C0914200
IC = 1                                                C0914300
GO TO 9                                               C0914400
8 IC = 0                                              C0914500
IF(LEQU) PRINT 224                                    C0914600
IF(LEQU) PRINT 201                                    C0914700
9 ICB1 = IC + 1                                       C0914800
DO 10 KKK=ICB1,NBOD                                  C0914900
K = NBOD - (KKK-ICB1)                                C0915000
IF(LEQU) PRINT 201                                    C0915100
KK = K                                                C0915200
12 JK = JCON(KK)                                       C0915300
IF(JK.EC.IC) GO TO 10                                C0915400
DO 11 I=1,3                                           C0915500
DO 11 J=1,3                                           C0915600
XTM1(I,J) = XMC(1,J,K)                               C0915700
11 XTM2(I,J) = XMC(1,J,JK)                           C0915800
CALL MATMUL(XTM1,XTM2,XMC(1,1,K),3)                  C0915900
IF(LEQU) PRINT 225, K,K,JK                           C0916000
KK = JK                                               C0916100
GO TO 12                                              C0916200
10 CONTINUE                                           C0916300
IF(LEQU) PRINT 204                                    C0916400
C                                                     C0916500

```

```

C     GET COMPUTING FRAME TO BODY 1 AND TO INERTIAL REFERENCE      C0916600
      IF(IC.EQ.1) GO TO 14                                         C0916700
      DO 15 I=1,3                                                  C0916800
      DO 15 J=1,3                                                  C0916900
15    XMC(I,J,C) = 0                                             C0917000
      XMC(1,1,C) = 1                                             C0917100
      XMC(2,2,C) = 1                                             C0917200
      XMC(3,3,C) = 1                                             C0917300
      IF(LEQU) PRINT 226                                         C0917400
      GO TO 16                                                    C0917500
14    DO 17 I=1,3                                                C0917600
      DO 17 J=1,3                                                C0917700
17    XMC(J,I,C) = XMC(I,J,1)                                    C0917800
      IF(LEQU) PRINT 227                                         C0917900
      DO 18 I=1,3                                                C0918000
      DO 18 J=1,3                                                C0918100
18    XMC(I,J,1) = 0                                             C0918200
      XMC(1,1,1) = 1                                             C0918300
      XMC(2,2,1) = 1                                             C0918400
      XMC(3,3,1) = 1                                             C0918500
      IF(LEQU) PRINT 228                                         C0918600
16    CONTINUE                                                  C0918700
C     TRANSPOSE TO GET TRANSFORMATION MATRICES BODY K TO COMPUTING FRAME C0918800
C     NBCD1=NECD + 1                                             C0918900
      DO 13 KKK=1,NBCD1                                           C0919000
      K=KKK - 1                                                  C0919100
      CALL TRNPS (XMC(1,1,K))                                     C0919200
      IF(LEQU) PRINT 229, K,K                                     C0919300
13    CONTINUE                                                  C0919400
      IF(.NOT. LEQU) RETURN                                       C0919500
      PRINT 204                                                  C0919600
      PRINT 233                                                  C0919700
      DO 21 KKK=1,NBCD1                                           C0919800
      K = KKK-1                                                  C0919900
      PRINT 201                                                  C0920000
      PRINT 221, (XMC(1,J,K),J=1,3)                             C0920100
      PRINT 222, K,(XMC(2,J,K),J=1,3)                           C0920200
21    PRINT 221, (XMC(3,J,K),J=1,3)                             C0920300
      PRINT 204                                                  C0920400
200  FORMAT (' SUBROUTINE TRNSIV ENTERED ',//)                   C0920500
201  FORMAT (' ')                                               C0920600
202  FORMAT (12X,3D15.5)                                         C0920700
203  FORMAT (' XMT(',I2,') = ',3D15.5)                          C0920800
204  FORMAT (3(/))                                              C0920900
205  FORMAT (' XTMP = XMT(',I2,')*T ')                          C0921000
206  FORMAT (' XMC(',I2,') = XTMP ')                             C0921100
207  FORMAT (' QT1 = XTMP * CF(',I2,') ')                       C0921200
208  FORMAT (' ZT1 = CLATOP(QT1,THA(',I2,'))',8X,' = ',4D15.5) C0921300
209  FORMAT (' XTM1 = TRANSU(ZT1) ')                             C0921400
210  FORMAT (' XMC(',I2,') = XTM1 * XTMP ')                     C0921500
211  FORMAT (' ZT2 = CLATOP(CF(',I2,'),THA(',I2,'))',5X,' = ',4D15.5) C0921600
212  FORMAT (' ZT4 = ZT1 * ZT2 ',15X,' = ',4D15.5)             C0921700
213  FORMAT (' XTM1 = TRANSU(ZT4) ')                             C0921800
214  FORMAT (' XMC(',I2,') = XTM1 * XTMP ')                     C0921900
215  FORMAT (' QT2 = NGRM(CF(',I2,') X QT1) ')                  C0922000
216  FORMAT (' ZT2 = QUATOP(QT2,THA(',I2,'))',8X,' = ',4D15.5) C0922100
217  FORMAT (' ZT3 = CLATOP(CF(',I2,'),THA(',I2,'))',5X,' = ',4D15.5) C0922200
218  FORMAT (' ZT4 = ZT1 * ZT2 ')                               C0922300
219  FORMAT (' ZT1 = ZT4 * ZT3 ',15X,' = ',4D15.5)             C0922400

```

```

220 FORMAT (' XTM1 = TRANSG(ZT1) ') C0922600
221 FORMAT (12X,3D15.5) C0922700
222 FORMAT (' XMC(' ,I2,' ) = ' ,3D15.5) C0922800
223 FORMAT (' COMPUTING FRAME FIXED IN BODY 1 ') C0922900
224 FORMAT (' COMPUTING FRAME FIXED INERTIALLY') C0923000
225 FORMAT (' XMC(' ,I2,' ) = XMC(' ,I2,' ) * XMC(' ,I2,' ) ') C0923100
226 FORMAT (' XMC( 0 ) = 1 ') C0923200
227 FORMAT (' XMC( 0 ) = XMC( 1)**T ') C0923300
228 FORMAT (' XMC( 1 ) = 1 ') C0923400
229 FORMAT (' XMC(' ,I2,' ) = XMC(' ,I2,' )**T ') C0923500
230 FORMAT (10X,' TRANSFORMATION MATRICES, NOMINAL STATE BODY K TO EOD C0923600
      *Y JCON(K) ',//) C0923700
231 FORMAT (10X,' HINGE POINT ',I2,/) C0923800
232 FORMAT (10X,' TRANSFORMATION MATRICES, TIME ZERO BODY JCON(K) TO B C0923900
      *CDY K ') C0924000
233 FORMAT (10X,' TRANSFORMATION MATRICES, TIME ZERO BODY K TO COMPUTI C0924100
      *NG FRAME ' ,//) C0924200
      RETURN C0924300
      END C0924400

```

```

C C100000
C SUERCUTINE VDIV C1000100
C USED TO TRANSFORM ALL VECTORS AND DYADS TO COMPUTING FRAME C1000200
C DEFINES FREE AND LOCKED VECTORS NOT INPUTED C1000300
C SETS UP DU LOOP SETS FOR TRANVD C1000400
C C1000500
C C1000600
C IMPLICIT REAL*8(A-F,O-Z,*) C1000700
C LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, R3BLD, LEGU, LINIT(1) C1000800
C LOGICAL LRUNGE, LTRNSI, LVDIV, LEQUIV, LTRAN, C1000900
C * LTRANV, LRATE, LXDY, LETA, LTORQU, C1001000
C * LCFDOT, LDCT, LANGLE, LSETUP, LSIMQ C1001100
C C1001200
C C1001300
C INTEGER C1001400
C * AMCRK, CT1, CT2, CT3, CT4, CT5, FCCN, PCCN, C1001500
C * SCNDLM, SCN, SCR DUM, SCR, SFKDUM, SFK, SFR, SG, C1001600
C * SI, SIG, SIXDUM, SIX, SKDUM, SK, SL, SLK, C1001700
C * SMA, SMC DUM, SMC, SMV, SJK, SPIDUM, SPI, SQF, C1001800
C * SCL, SK, SSCN, SSIX, SVA, SVB, SVD, SVI, C1001900
C * SVM, SVP, SVQ, SX#, SXT, TORQ, SMAL, SEU, C1002000
C * SC, SCG, NFLXB, SFLX, SFXM, NMDCS, SFCC, SCC, C1002100
C * IINIT(1), IZINIT(1), SD C1002200
C C1002300
C C1002400
C C1002500
C REAL*8 C1002600
C * ANGC (33), CNF (3,10), ETIC (3,10), ETMC (3,10), C1002700
C * FLQ (3,20), FLE (3,3,20), FLH (3,3,20), C1002800
C * TPAED (33), YMCD (3,2,11), RINIT (1), RZINIT(1) C1002900
C C1003000
C C1003100
C C1003200
C C1003300
C CCMCN /LDEBUG/ LRUNGE, LTRNSI, LVDIV, LEQUIV, LTRAN, C1003400
C * LTRANV, LRATE, LXDY, LETA, LTORQU, C1003500

```



```

      *          LQFDOT , LDCT , LANGLE , LSETUP , LSIMQ          C1C03600
C          C1C03700
C          C1C03800
      *          COMMON /LUGIC/ FG1, FG2, FG3, FG4, FG5, INE#F, RBLU(10) C1C03900
C          C1C04000
C          C1C04100
      *          COMMON /INTG/      A#ORK (200) ,                C1C04200
      *          CT1 , CT2 , CT3 , CT4 ,                C1C04300
      *          CT5 , FCUN (33) , JCCN (10) , LCCN (22) , C1C04400
      *          MC (10) , NE1 , NBUD , NCTC ,                C1C04500
      *          NFER , NFKC , NFRG , NLUR ,                C1C04600
      *          NMV , NMO , NMOA , NSVP ,                C1C04700
      *          NSVC , FCON (11) , SD , SFR (33) , C1C04800
      *          SC , SI (55) , SIG , SL ,                C1C04900
      *          SLK (33) , SMA (10) , SCK (11) , SQF (11) , C1C05000
      *          SGL (11) , SMV , SR , SSCN ,                C1C05100
      *          SSIX , SVA , SVB , SVD ,                C1C05200
      *          SVI , SVM , SVP (22) , SVU (33) , C1C05300
      *          SXM (3,10) , SXT , TURQ (97) , SMAL , C1C05400
      *          SEU , NTQ , SC (33) , SCG ,                C1C05500
      *          NFLXB , SFLX , SFXM (10) , NMGDS , C1C05600
      *          SFCC , ECC (10) ,                C1C05700
C          C1C05800
C          C1C05900
      *          COMMON /INTGZ/                C1C06000
      *          SCNDLM , SCN (5) , SCRDM , SCR (9) , C1C06100
      *          SFKDLM , SFK (5) , SIXDUM , SIX (9) , C1C06200
      *          SKDLM , SK (5) , SPIDUM , SPI (9) , C1C06300
      *          SMCDLM , SMC (5) ,                C1C06400
C          C1C06500
C          C1C06600
      *          COMMON /REAL/                C1C06700
      *          CA (3,10) , CAC (3,10) , CLM (10) , CCMC (3,11) , C1C06800
      *          DCMC (3,11) , ETC (3,11) , ETM (33) , FOMC (3,11) , C1C06900
      *          GAM (3,66) , H , HM (3,10) , HMC (3,10) , C1C07000
      *          HMDM (10) , PHI (3,11) , PLM (10) , QF (3,33) , C1C07100
      *          QFC (3,33) , GL (3,22) , QLC (3,22) , RCMC (3,11) , C1C07200
      *          T , THA (33) , THAD (33) , C1C07300
      *          THADW (10) , THAW (10) , XDIC (3,3,66) , XI (3,3,10) , C1C07400
      *          XIC (3,3,10) , XMAS (10) , XMN (33,33) , XMT (3,3,10) , C1C07500
      *          TLG (33) , FLA (3,20) , FLB (3,20) , FLC (3,20) , C1C07600
      *          FLD (3,3,20) , FLJ (3,3,20) , CAD (3,10) , XID (3,3,10) , C1C07700
      *          FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLCC (3,20) , C1C07800
      *          FLQ (20) , ZETA (20) , FCF (3,3,40) , FCK (3,40) , C1C07900
      *          TIMEND ,                C1C08000
C          C1C08100
C          C1C08200
      *          COMMON /REALZ/                C1C08300
      *          CBDLM (1,3) , CE (3,10) , CBCDUM(1,3) , CBC (3,10) , C1C08400
      *          XMCDLM(1,1,9) , XMC (3,3,10) , CBN(3) , C1C08500
C          C1C08600
C          C1C08700
      *          EQUIVALENCE (ETM(1),THADD(1)) , (XMN(1,1),ANGD(1)) , C1C08800
      *          (XMN(1,3),YMCD(1,1,1)) , (XMN(1,6),CNE(1,1)) , C1C08900
      *          (XMN(1,8),ETIC(1,1)) , (XMN(1,10),ETMC(1,1,1)) , C1C09000
      *          (FLB(1,1),FLQ(1,1)) , (FLE(1,1,1),FLD(1,1,1)) , C1C09100
      *          (FLH(1,1,1),FLJ(1,1,1)) , C1C09200
      *          (FG1,LINIT(1)) , (CA(1,1),RINIT(1)) , C1C09300
      *          (CBDUM(1,1),RZINIT(1)) , (AWURK(1),IINIT(1)) , C1C09400
      *          (SCNDLM,IZINIT(1)) , C1C09500

```

C		C1C09600
	LOGICAL CTAIN	C1C09700
	INTEGER SET(18),S(18),SS(18)	C1C09800
	INTEGER ST1(10),ST2(10),ST3(10),ST4(10),SFXMN	C1C09900
	REAL*8 TEM(3), TEM1(3,3), TEM2(3,3), TEM3(3,3)	C1C10000
	EQUIVALENCE (LVDIV,LEQU)	C1C10100
C		C1C10200
C		C1C10300
C		C1C10400
	IF(LEQU)FRINT 200	C1C10500
C		C1C10600
	IF(INERF) GO TO 1	C1C10700
	IC = 1	C1C10800
	IF(LEQU)FRINT 201	C1C10900
	GO TO 7	C1C11000
1	IC = 0	C1C11100
	IF(LEQU)PRINT 202	C1C11200
7	CONTINUE	C1C11300
C		C1C11400
C	INCLUDE FLEXIBILITY EFFECTS	C1C11500
	IF(NFLXB,EQ,C) GO TO 7E	C1C11600
	CALL UNPAC(SET,NSET,SFLX)	C1C11700
	IF(LEQU) PRINT 231, SFLX,(SET(I),I=1,NSET)	C1C11800
C	SAVE UNDEFORMED CM VECTOR AND INERTIA TENSOR DATA	C1C11900
	MN = C	C1C12000
	DO 76 NN=1,NSET	C1C12100
	N = SET(NSET+1-NN)	C1C12200
	IF(LEQU) PRINT 257, N	C1C12300
	DO 77 I=1,3	C1C12400
	CAC(I,N) = CA(I,N)	C1C12500
	DO 77 J=1,3	C1C12600
77	>IC(I,J,N) = XI(I,J,N)	C1C12700
	IF(.NOT.LEQU) GO TO 1001	C1C12800
	FRINT 232, N,N	C1C12900
	FRINT 234, N,N	C1C13000
	FRINT 233	C1C13100
1001	CONTINUE	C1C13200
	SFXMN = SFXM(N)	C1C13300
	DO 78 M=1,SFXMN	C1C13400
	MN = MN+1	C1C13500
	(CALL VECTR(FLA(1,MN),XMC(1,1,N),FLAC(1,MN))	C1C13600
	IF(.NOT.LEQU) GO TO 1002	C1C13700
	FRINT 256, N,MN	C1C13800
	FRINT 246, MN,N,MN,(FLAC(I,MN),I=1,3)	C1C13900
	FRINT 233	C1C14000
1002	CONTINUE	C1C14100
C	USE TEMPORARY LOCATIONS SO THAT	C1C14200
C	EQUIVALENCE MAY BE USED TO SAVE STORAGE	C1C14300
	DO 79 I=1,3	C1C14400
	DO 79 J=1,3	C1C14500
	TEM1(I,J) = FLD(I,J,MN)	C1C14600
	TEM2(I,J) = FLD(J,I,MN)	C1C14700
79	TEM3(I,J) = FLJ(I,J,MN)	C1C14800
	(CALL DYADD(TEM1,TEM2,FLE(1,1,MN))	C1C14900
00000	CALL DYADD(TEM1,TEM3,FLH(1,1,MN))	C1C15000
	IF(.NOT.LEQU) GO TO 1003	C1C15100
	FRINT 244, (FLE(1,1,MN),I=1,3)	C1C15200
	FRINT 242,MN,MN,MN,(FLE(2,1,MN),I=1,3)	C1C15300
	FRINT 244, (FLE(3,1,MN),I=1,3)	C1C15400
	FRINT 233	C1C15500

FRINT 244,	(FLH(1,I,MN),I=1,3)	C1C15600
FRINT 243,MN,MN,MN,(FLH(2,I,MN),I=1,3)		C1C15700
FRINT 244,	(FLH(3,I,MN),I=1,3)	C1C15800
FRINT 233		C1015900
1003 CONTINUE		C1C16000
C		C1C16100
C	COMPLTE G VECTOR IN BODY N FIXED FRAME	C1C16200
	CALL VECROS(CAO(1,N),FLA(1,MN),TEM)	C1C16300
	CALL VEC SUB(FLB(1,MN),TEM,TEM)	C1C16400
	CALL SOLV(XMAS(N),TEM,TEM)	C1C16500
	CALL VECADD(FLC(1,MN),TEM,FLQ(1,MN))	C1C16600
	CALL VECTR N(FLQ(1,MN),XMC(1,1,N),FLOC(1,MN))	C1C16700
	IF(.NOT.LEQU) GO TO 1004	C1C16800
	PRINT 245, MN,MN,N,MN,N,MN,(FLQ(1,MN),I=1,3)	C1C16900
	FRINT 233	C1C17000
	FRINT 249, MN,N,MN,(FLOC(1,MN),I=1,3)	C1C17100
	FRINT 233	C1C17200
1004 CONTINUE		C1C17300
	CALL SOLV(THA(NFER+MN),FLA(1,MN),TEM)	C1C17400
	CALL VECADD(CA(1,N),TEM,CA(1,N))	C1C17500
	CALL SOLC(THA(NFER+MN),FLE(1,1,MN),TEM1)	C1C17600
	CALL DYADD(XI(1,1,N),TEM1,XI(1,1,N))	C1C17700
	IF(.NOT.LEQU) GO TO 1006	C1C17800
	NFCM = NFER+MN	C1C17900
	PRINT 246, N,N,NFCM,MN	C1C18000
	PRINT 247, N,N,NFCM,MN	C1C18100
	FRINT 233	C1C18200
1006 CONTINUE		C1C18300
78 CONTINUE		C1C18400
	IF(.NOT.LEQU) GO TO 1007	C1C18500
	FRINT 250, N	C1C18600
	PRINT 252, N,(CA(I,N),I=1,3)	C1C18700
	FRINT 233	C1018800
	FRINT 254, (XI(1,I,N),I=1,3)	C1018900
	PRINT 255, N,(XI(2,I,N),I=1,3)	C1C19000
	FRINT 254, (XI(3,I,N),I=1,3)	C1019100
	FRINT 233	C1C19200
1007 CONTINUE		C1C19300
76 CONTINUE		C1C19400
75 CONTINUE		C1C19500
C		C1C19600
C		C1019700
C	CENTER OF MASS VECTORS	C1C19800
	IF(LEQU)FRINT 204	C1C19900
	DO 2 J=1,NBOD	C1C20000
	2 CALL VECTR N(CA(1,J),XMC(1,1,J),CAC(1,J))	C1C20100
	IF(.NOT.LEQU) GO TO 1020	C1C20200
	DO 8 J=1,NBOD	C1C20300
	8 PRINT 203, J,J,J,(CAC(I,J),I=1,3)	C1C20400
C		C1020500
C	INERTIA TENSORS	C1020600
	FRINT 205	C1C20700
1020 CALL UNPAC(SET,NSET,SR)		C1C20800
	DO 46 J=1,NSET	C1020900
	K = SET(J)	C1C21000
	46 CALL TENR N(XI(1,1,K),XMC(1,1,K),XIC(1,1,K))	C1C21100
	IF(.NOT.LEQU) GO TO 1000	C1021200
	DO 49 J=1,NSET	C1021300
	FRINT 233	C1C21400
	K=SET(J)	C1C21500

FRINT 206,	(XIC(1,L,K),L=1,3)	C1C21600
FRINT 207, K,K,K,K,	(XIC(2,L,K),L=1,3)	C1O21700
49 FRINT 206,	(XIC(3,L,K),L=1,3)	C1C21800
C		C1C21900
1000 CONTINUE		C1C22000
C		C1O22100
C	GET ELEMENTS OF SVA; THAT IS, SVD MINUS ZERO CA VECTORS	C1C22200
C	IF BODY FLEXIBLE BOTH CM VECTOR AND INERTIA TENSOR MUST BE	C1C22300
C	TRANSFORMED EVERY STEP	C1C22400
	CALL UNPAC(SET,NSET,SVD)	C1C22500
	CALL UNPAC(ST1,NS1,SFLX)	C1C22600
	IF(LEQU)FRINT 208, SVD,((SET(I),I=1,NSET)	C1C22700
	NS = 0	C1C22800
	DO 51 I=1,18	C1C22900
51	S(I) = 0	C1C23000
	IF(NSET.EQ.0) GO TO 5024	C1C23100
	DO 50 J=1,NSET	C1O23200
	K = SET(J)	C1C23300
	IF(CTAIN(K,ST1,NS1)) GO TO 50	C1C23400
	IF(CAC(1,K).EQ.0.AND.CAC(2,K).EQ.0.AND.CAC(3,K).EQ.0) GO TO 50	C1C23500
80	CONTINUE	C1C23600
	NS = NS+1	C1C23700
	S(NS) = K	C1C23800
50	CONTINUE	C1C23900
5024	CONTINUE	C1C24000
	CALL CUMFAC(S,NS,SVA)	C1C24100
	IF(LEQU)FRINT 241, SVA,((S(I),I=1,NS)	C1C24200
	DO 47 I=1,18	C1C24300
47	S(I) = 0	C1C24400
	CALL UNPAC(SS,NS,SVD)	C1C24500
	CALL UNPAC(SET,NSET,SR)	C1C24600
	K = 0	C1C24700
	DO 48 J=1,NBOD	C1C24800
	IF(.NOT.(CTAIN(J,SET,NSET).AND.CTAIN(J,SS,NS))) GO TO 48	C1C24900
81	CONTINUE	C1O25000
	K = K+1	C1O25100
	S(K) = J	C1O25200
48	CONTINUE	C1C25300
	CALL CCMFAC(S,K,SVI)	C1O25400
	IF(LEQU)FRINT 209, SVI,((S(I),I=1,K)	C1O25500
C	DO ON SVI ELEMENTS IN TRANVC TO GET XIC	C1C25600
C		C1C25700
C	HINGE VECTORS	C1C25800
	IF(LEQU)FRINT 210	C1C25900
	DO 4 J=1,18	C1O26000
4	S(J) = 0	O1O26100
	CALL UNPAC(SET,NSET,SVD)	C1O26200
	K = 0	C1C26300
	DO 5 J=1,NBOD	C1O26400
	JJ = JCCN(J)	C1C26500
	CALL VECTRN (CB(1,J),XMC(1,1,JJ),CBC(1,J))	C1C26600
	IF(LEQU)FRINT 211, J,JJ,J,(CBC(I,J),I=1,3)	C1C26700
	IF(CEC(1,J).EQ.0.AND.CBC(2,J).EQ.0.AND.CBC(3,J).EQ.0) GO TO 5	C1C26800
-2002	IF(.NOT.CTAIN(JJ,SET,NSET)) GO TO 5	C1C26900
	K = K+1	C1C27000
	S(K) = J	C1O27100
5	CONTINUE	C1C27200
C	CB(1) ALONG WITH CBC(1) MUST BE COMPUTED AT EACH INTEGRATION STEP.	C1C27300
C	PLT 1 IN SVB	C1C27400
	K = K+1	C1C27500

```

S(K) = 1
6 CALL COMPAC(S,K,SVE)
IF(LEQU)PRINT 212, SVB,((S(1),I=1,K)
DO CN SVB ELEMENTS IN TRANVC
C
C NOTE CBC(I,0) IS COMPOSITE CENTER OF MASS, MUST BE CALCULATED
C
C FREE AND LOCKED COORDINATE VECTORS
IF(LEQU)PRINT 213
M = 1
L = 1
DO 9 K=1,NBOD
IF(K.EQ.1) GO TO 10
M = M+3-FCON(K-1)
L = L+FCCN(K-1)
10 M1 = M+1
M2 = M+2
L1 = L+1
L2 = L+2
IGCTC = FCON(K) + 1
GO TC(11,12,13,14),IGOTD
C
C THREE DEGREES OF FREEDOM
11 CALL VECTR (QF(1,M),XMC(1,1,FCON(M)),QFC(1,M))
CALL VECTR (QF(1,M2),XMC(1,1,FCON(M2)),QFC(1,M2))
IF(LEQU)PRINT 214, M, FCCN(M), M
IF(LEQU)PRINT 214, M2, FCCN(M2), M2
IF(FCON(M1).LT.0) GO TC 15
CALL VECRUS (QF(1,M2),QF(1,M),QF(1,M1))
IF(LEQU)PRINT 236, M1,M2,M
CALL VECTR (QF(1,M1),XMC(1,1,FCON(M1)),QFC(1,M1))
IF(LEQU)PRINT 214, M1, FCCN(M1), M1
GO TC 9
15 CALL VECRUS (QFC(1,M2),QFC(1,M),QFC(1,M1))
CALL VECRUS (QFC(1,M1))
DO 45 I=1,3
45 QF(1,M1) = 0
IF(LEQU)PRINT 216, M1, M2, M
GO TC 9
C
C TWO DEGREES OF FREEDOM
12 CALL VECTR (QF(1,M),XMC(1,1,FCON(M)),QFC(1,M))
CALL VECTR (QF(1,M1),XMC(1,1,FCON(M1)),QFC(1,M1))
IF(LEQU)PRINT 214, M, FCCN(M), M
IF(LEQU)PRINT 214, M1, FCCN(M1), M1
IF(LCON(L).LT.0) GO TO 16
CALL VECRUS (QF(1,M),QF(1,M1),QL(1,L))
IF(LEQU)PRINT 237, L,M,M1
CALL VECTR (QL(1,L),XMC(1,1,LCON(L)),QLC(1,L))
IF(LEQU)PRINT 217, L, LCON(L), L
GO TC 9
16 CALL VECRUS (QFC(1,M),QFC(1,M1),QLC(1,L))
CALL VECRUS (QLC(1,L))
DO 25 I=1,3
25 QL(1,L) = 0
IF(LEQU)PRINT 235, L,M,M1
GO TC 9
C
C ONE DEGREE OF FREEDOM
13 CALL VECTR (QF(1,M),XMC(1,1,FCON(M)),QFC(1,M))

```

```

C1027600
C1027700
C1027800
C1027900
C1028000
C1028100
C1028200
C1028300
C1028400
C1028500
C1028600
C1028700
C1028800
C1028900
C1029000
C1029100
C1029200
C1029300
C1029400
C1029500
C1029600
C1029700
C1029800
C1029900
C1030000
C1030100
C1030200
C1030300
C1030400
C1030500
C1030600
C1030700
C1030800
C1030900
C1031000
C1031100
C1031200
C1031300
C1031400
C1031500
C1031600
C1031700
C1031800
C1031900
C1032000
C1032100
C1032200
C1032300
C1032400
C1032500
C1032600
C1032700
C1032800
C1032900
C1033000
C1033100
C1033200
C1033300
C1033400
C1033500

```

I = (SM, E)

	CALL VECTRAN (QL(1,L),XMC(1,1,LCON(L)),QLC(1,L))	C1C33600
	IF(LEQU)PRINT 214, M,FCCN(M),M	C1C33700
	IF(LEQU)PRINT 217, L,LCCN(L),L	C1033800
	IF(RBLD(K)) GO TO 17	01C33900
	CALL VECFOS (QF(1,M),QL(1,L),QL(1,L1))	01C34000
	IF(LEQU)PRINT 238, L1,M,L	C1C341CC
	CALL VECTRAN (QL(1,L1),XMC(1,1,LCON(L1)),QLC(1,L1))	C1034200
	IF(LEQU)PRINT 217, L1,LCCN(L1),L1	C1034300
	GO TO 9	C1C34400
17	CALL TRNSPS (XMT(1,1,K))	C1C34500
	CALL VECTRAN (QF(1,M),XMT(1,1,K),TEM)	01C34600
	CALL TRNEPS (XMT(1,1,K))	C1C34700
	CALL VECFOS (TEM,QL(1,L),QL(1,L1))	C1C34800
	IF(LEQU)PRINT 239, L1,K,M,L	C1C34900
	CALL VECTRAN (QL(1,L1),XMC(1,1,LCON(L1)),QLC(1,L1))	C1C35000
	IF(LEQU)PRINT 217, L1,LCCN(L1),L1	C1C35100
	GO TO 9	C1C35200
C		C1035300
C	PCCN(K) = 3 ZERO DEGREES OF FREEDOM	C1C35400
14	CALL VECFOS (QL(1,L),QL(1,L1),QL(1,L2))	C1C35500
	IF(LEQU)PRINT 217, L,LCCN(L),L	C1C35600
	IF(LEQU)PRINT 217, L1,LCCN(L1),L1	C1C35700
	IF(LEQU)PRINT 240, L2,L,L1	C1C35800
	IF(LEQU)PRINT 217, L2,LCCN(L2),L2	C1035900
	DO 18 II=1,3	C1036000
	I=II-1	C1C36100
18	CALL VECTRAN (QL(1,L+I),XMC(1,1,LCON(L+I)),QLC(1,L+I))	C1C36200
C		C1C36300
	9 CONTINUE	C1C36400
C		01036500
C	AT INERTIAL ORIGIN	C1C36600
	M = M+3-FCCN(NBOD)	C1C36700
	L = L+PCCN(NBOD)	C1C36800
	DO 19 I=1,3	01036900
	MB2 = M+2	C1037000
	DO 20 J=M,MB2	01037100
20	QF(I,J) = 0	C1037200
	LB2 = L+2	C1C37300
	DO 21 J=L,LB2	C1037400
21	QL(I,J) = 0	C1C37500
19	CONTINUE	C1C37600
	IF(LEQU)M0 = 0	C1C37700
C		C1C37800
	IGCTC = PCCN(NBOD+1)+1	C1C37900
	GO TO (22,23,24,25),IGCTC	C1C38000
C		C1C38100
C	THREE DEGREES OF FREEDOM	C1C38200
22	M1 = M+1	01038300
	M2 = M+2	C1C38400
	FCCN(M) = 0	C1C38500
	FCCN(M1) = 0	C1C38600
	FCCN(M2) = 0	C1C38700
	QF(1,M) = 1	C1C38800
	QF(2,M1) = 1	C1C38900
	QF(3,M2) = 1	C1039000
	IF(IC.EQ.0) GO TO 26	C1039100
	DO 27 II=1,3	C1C39200
	I = II-1	C1C39300
	IF(LEQU)JM = M+I	C1C39400
	IF(LEQU)PRINT 214, JM,M0,JM	01C39500

27	CALL VECTR N (QF(1,M+1),XMC(1,1,0),QFC(1,M+1))	C1C39600
	GO TO 26	C1C39700
C		C1C39800
C	TMC DEGREES CF FREEDCM	C1C39900
23	M1 = M+1	C1C40000
	FCCN(M) = 0	C1C40100
	FCCN(M1) = 0	C1040200
	LCCN(L) = 0	C1C40300
	GF(1,M) = 1	C1C40400
	GF(2,M1) = 1	C1C40500
	GL(3,L) = 1	C1C40600
	IF(IC.EQ.0) GO TO 26	C1C40700
	IF(LEQU)FRINT 214, M,MC,M	C1C40800
	IF(LEQU)FRINT 214, M1,MO,M1	C1C40900
	IF(LEQU)FRINT 217, L,MO,L	C1C41000
	CALL VECTR N (QF(1,M),XMC(1,1,0),QFC(1,M))	C1C41100
	CALL VECTR N (QF(1,M1),XMC(1,1,0),QFC(1,M1))	C1041200
	CALL VECTR N (QL(1,L),XMC(1,1,0),QLC(1,L))	C1C41300
	GO TO 26	C1C41400
C		C1C41500
C	CNE DEGREE CF FREEDCM	C1C41600
24	L1 = L+1	C1C41700
	FCCN(M) = 0	C1C41800
	LCCN(L) = 0	C1C41900
	LCCN(L1) = 0	C1C42000
	GF(1,M) = 1	C1042100
	GL(2,L) = 1	C1C42200
	GL(3,L1) = 1	C1042300
	IF(IC.EQ.0) GO TO 26	C1C42400
	IF(LEQU)FRINT 214, M,MC,M	C1C42500
	IF(LEQU)FRINT 217, L,MC,L	C1C42600
	IF(LEQU)FRINT 217, L1,MO,L1	C1C42700
	CALL VECTR N (QF(1,M),XMC(1,1,0),QFC(1,M))	C1C42800
	CALL VECTR N (QL(1,L),XMC(1,1,0),QLC(1,L))	C1C42900
	CALL VECTR N (QL(1,L1),XMC(1,1,0),QLC(1,L1))	C1C43000
	GO TO 26	C1C43100
C		C1C43200
C	ZERC DEGREES OF FREEDCM	C1C43300
25	L1 = L+1	C1C43400
	L2 = L+2	C1C43500
	LCCN(L) = 0	C1C43600
	LCCN(L1) = 0	C1C43700
	LCCN(L2) = 0	C1043800
	GL(1,L) = 1	C1C43900
	GL(2,L1) = 1	C1044000
	GL(3,L2) = 1	C1044100
	IF(IC.EQ.0) GO TO 26	C1044200
	DO 26 I=1,3	C1044300
	I = I-1	C1C44400
	IF(LEQU)JL = L+1	C1C44500
	IF(LEQU)FRINT 217, JL,MC,JL	C1C44600
2E	CALL VECTR N (QL(1,L+1),XMC(1,1,0),QLC(1,L+1))	C1C44700
	GO TO 26	C1C44800
C		C1C44900
2E	CONTINUE	C1045000
	IF(IC.NE.0) GO TO 1025	C1C45100
	DO 30 I=1,3	C1C45200
	MB2=I+2	C1045300
	DO 32 J=M,MB2	C1C45400
31	QFC(1,J) = QF(1,J)	C1C45500

LB2=L+2	C1C45600
CO 32 J=L, LB2	C1C45700
32 GLC(I,J) = QL(I,J)	C1C45800
30 CONTINUE	C1C45900
1025 CONTINUE	C1C46000
IF(.NOT. LEQU) GO TO 1015	C1C46100
FRINT 221	C1C46200
CO 60 J=1, NFER	C1C46300
60 FRINT 222, J, (QF(I,J), I=1,3), J, (QFC(I,J), I=1,3)	C1C46400
FRINT 221	C1C46500
IF(NLGR.EQ.0)GC TC 1015	C1C46600
CO 61 J=1, NLOR	C1C46700
61 FRINT 223, J, (QL(I,J), I=1,3), J, (QLC(I,J), I=1,3)	C1C46800
FRINT 221	C1C46900
C	C1C47000
C CYCLE THROUGH FREE VECTORS PICK OUT ONES TO BE TRANSFORMED IN TRAN	C1C47100
1015 CO 33 J=1, NFER	C1C47200
33 SVG(J) = 0	C1C47300
K = C	C1C47400
CC 34 M=1, NFER	C1C47500
C IS FREE VECTOR M FIXED INERTIALLY	C1C47600
IF(FCCN(M)) 35,36,37	C1C47700
36 IF(IC.EQ.0) GO TO 34	C1C47800
35 K = K+1	C1C47900
SVG(K) = M	C1C48000
CO TC 34	C1C48100
C ELEMENTS OF SVD IN SET(J)	C1C48200
37 IF(CTAIN(FCCN(M), SET, NSET1)) GO TO 35	C1C48300
34 CONTINUE	C1C48400
NSVG = K	C1C48500
IF(.NOT. LEQU) GO TO 1005	C1C48600
IF(NSVG.EQ.0)GC TC 5016	C1C48700
DO 64 I=1, NSVG	C1C48800
64 FRINT 224, I, SVG(I)	C1C48900
FRINT 221	C1C49000
C	C1C49100
C CYCLE THROUGH LOCKED VECTORS PICK OUT ONES TO BE TRANSFORMED IN TR	C1C49200
5016 IF(NLOR.EQ.0)GO TO 5017	C1C49300
1005 CO 38 J=1, NLOR	C1C49400
38 SVF(J) = 0	C1C49500
5017 CONTINUE	C1C49600
K = C	C1C49700
IF(NLOR.EQ.0) GO TC 5025	C1C49800
CO 39 L=1, NLOR	C1C49900
C IS CONSTRAINT TORQUE ABOUT CL(I,L) COMPUTED	C1C50000
IF(.NOT. CTAIN(L, SLK, NCTC)) GO TO 39	C1C50100
IF(LCCN(L)) 40,41,42	C1C50200
41 IF(IC.EQ.0) GO TO 39	C1C50300
40 K = K+1	C1C50400
SVF(K) = L	C1C50500
GO TC 39	C1C50600
C ELEMENTS OF SVD IN SET(J)	C1C50700
42 IF(CTAIN(LCCN(L), SET, NSET)) GO TO 40	C1C50800
39 CONTINUE	C1C50900
5005-5025 CONTINUE	C1C51000
NSVP = K	C1C51100
IF(.NOT. LEQU) GO TO 1010	C1C51200
IF(NSVP.EQ.0)GO TC 1010	C1C51300
CO 65 I=1, NSVP	C1C51400
65 FRINT 225, I, SVP(I)	C1C51500



C		C1051600
C	ZERO ALL ARRAY ELEMENTS OF GAM STORED UPPER TRIANGULAR	C1051700
1010	CO 52 K=1,NB0D	C1051800
	IF(LEQU)PRINT 233	C1051900
	CO 52 L=K,NB0D	C1052000
	KL = KTC(NB1,K-1,L)	C1052100
C	PUT IN LOGIC TO AVOID CLCDBERING CODE WORDS STORED BY EQUIV.	C1052200
	IF(KL.EQ.1) GO TO 52	C1052300
	IF(LEQU)K1 = K-1	C1052400
	IF(LEQU)PRINT 218,K1,L,KL	C1052500
	CO 52 I=1,3	C1052600
	GAM(I,KL) = 0.D0	C1052700
	52 CONTINUE	C1052800
C		C1052900
C	ZERO ALL ARRAY ELEMENTS OF XDIC STORED LOWER TRIANGULAR	C1053000
	CO 53 K=1,NB1	C1053100
	IF(LEQU)PRINT 233	C1053200
	CO 53 I=K,NB1	C1053300
	IK = KI(NB1,I,K)	C1053400
	IF(LEQU)PRINT 215, I,K,IK	C1053500
	CO 53 M=1,3	C1053600
	CO 53 N=1,3	C1053700
	53 XDIC(M,N,IK) = 0.D0	C1053800
	IF(LEQU)PRINT 221	C1053900
C		C1054000
C		C1054100
C		C1054200
C	FIRST PASS THROUGH, FIND THE UNION OF ALL LABELS IN THE	C1054300
C	SETS SIX(K),K=C.....NECC-1 AND THE SETS SCN(K),K=0.....NBUC-1	C1054400
	CO 55 I=1,NB0D	C1054500
	ST3(I) = 0	C1054600
55	ST4(I) = 0	C1054700
	CO 56 JJJ=1,NBUD	C1054800
	J=JJJ-1	C1054900
	CALL UNPAC(ST1,NST1,SIX(J))	C1055000
	CALL UNPAC(ST2,NST2,SCN(J))	C1055100
	IF(NST1.EQ.0)GO TO 5018	C1055200
	CO 57 I=1,NST1	C1055300
57	ST3(ST1(I)) = 1	C1055400
5018	IF(NST2.EQ.0)GO TO 5019	C1055500
	CO 58 I=1,NST2	C1055600
58	ST4(ST2(I)) = 1	C1055700
5019	CONTINUE	C1055800
56	CONTINUE	C1055900
	NST1 = 0	C1056000
	NST2 = 0	C1056100
	CO 55 J=1,NB0D	C1056200
	IF(ST3(J).EQ.0) GO TO 62	C1056300
	NST1 = NST1 + 1	C1056400
	ST1(NST1) = J	C1056500
62	IF(ST4(J).EQ.0) GO TO 59	C1056600
	NST2 = NST2 + 1	C1056700
	ST2(NST2) = J	C1056800
59	CONTINUE	C1056900
	CALL COMPAC(ST1,NST1,SSIX)	C1057000
	CALL COMPAC(ST2,NST2,SSCN)	C1057100
	IF(LEQU)PRINT 219, SSIX, (ST1(I), I=1,NST1)	C1057200
	IF(LEQU)PRINT 220, SSCN, (ST2(I), I=1,NST2)	C1057300
C		C1057400
C		C1057500

C	DO 63 K=1,NBOD	C1057600
	63 SOK(K) = SI(K)	01057700
	CALL UNPAC(ST1,NST1,SKDUM)	C1057800
	NST1 = NST1 + 1	C1057900
	STJ(NST1) = NB1	C1058000
	CALL COMPAC(ST1,NST1,SCK(NB1))	C1058100
C		C1058200
C	COMPUTE ACTUAL POSITION OF FCINT MASSES	C1058300
C	NOMINAL CAC(K) + DISPLACED THA(M)*QFC(M)	C1058400
	CALL UNPAC(SET,NSET,SL)	C1058500
	IF(NSET.EQ.0)GO TO 5026	C1058600
	DO 70 KK=1,NSET	C1058700
	K = SET(KK)	C1058800
	MM=SCF(K)	C1058900
	MMM=MM+2-PCUN(K)	C1059000
	DO 72 M=MM,MMM	C1059100
	DO 71 I=1,3	C1059200
	71 CAC(I,K) = CAC(I,K) + THA(M)*QFC(I,M)	C1059300
	IF(LEQU)PRINT 228, K,K,M,M	C1059400
	72 CONTINUE	C1059500
	70 CONTINUE	C1059600
	5026 CONTINUE	C1059700
C		C1059800
C		C1059900
C	ANGULAR MOMENTUM WHEEL	C1060000
	IF(LEQU)PRINT 226	C1060100
	IF(NMC.EQ.0)GO TO 5020	C1060200
	DO 43 J=1,NMC	C1060300
	43 S(J) = 0	C1060400
	5020 CONTINUE	C1060500
	K = 0	C1060600
	IF(NMC.EQ.0) GO TO 5027	C1060700
	DO 44 J=1,NMC	C1060800
C	HM(I,J) = COMPONENTS OF SPIN AXIS	C1060900
C	HMC(J) = RELATIVE MOMENTUM I.V. ABOUT SPIN AXIS	C1061000
	CALL VECTRN (HM(I,J),XMC(I,1,MC(J)),HMC(I,J))	C1061100
C	PMC(I,J) = SPIN AXIS COMPONENTS IN COMPUTING FRAME	C1061200
	IF(LEQU)PRINT 227, J,MC(J),J	C1061300
	JJ = MC(J)	C1061400
C	ELEMENTS OF SVD IN SET(J)	C1061500
	IF(.NOT.CTAIN(JJ,SET,NSET)) GO TO 44	C1061600
	K = K+1	C1061700
	S(K) = J	C1061800
	44 CONTINUE	C1061900
	5027 CONTINUE	C1062000
	CALL COMPAC(S,K,SVM)	C1062100
	IF(LEQU)PRINT 229, SVM, (S(I),I=1,K)	01062200
C	USE SVM ELEMENTS IN TRANVD TO GET HMC	C1062300
	IF(LEQU)PRINT 233	C1062400
	IF(.NOT.LEQU) RETURN	C1062500
	IF(NMC.EQ.0) RETURN	C1062600
	DO 66 J=1,NMC	C1062700
	66 PRINT 230, J,(HMC(I,J),I=1,3)	C1062800
	PRINT 233	C1062900
	200 FORMAT ('1 SUBROUTINE VCIV ENTERED ',2('/))	C1063000
	201 FORMAT (' COMPUTING FRAME IS BODY-1 ',/)	C1063100
	202 FORMAT (' COMPUTING FRAME INERTIALLY FIXED ',/)	C1063200
	203 FORMAT (' CAC(' ,I2,') = XMC(' ,I2,') * CA(' ,I2,') = ',3D15.5)	C1063300
	204 FORMAT (2(/),	C1063400
		C1063500

```

*
*           CENTER OF MASS VECTORS (/,/)
205 FORMAT (2(/),
*           INERTIA TENSORS (/,/)
206 FORMAT (44X,3D15.5)
207 FORMAT (' XIC(',I2,') = XMC(',I2,') * XI(',I2,') * XMC(',I2,')**T
* = ',3D15.5)
208 FORMAT (/, ' SVD = ',Z8,' ELEMENTS OF SET ARE ',10I5,/)
209 FORMAT (/, ' SVI = ',Z8,' ELEMENTS OF SET ARE ',10I5,/)
210 FORMAT (2(/),
*           HINGE VECTORS (/,/)
211 FORMAT (' CBC(',I2,') = XMC(',I2,') * CB(',I2,') = ',3D15.5)
212 FORMAT (/, ' SVE = ',Z8,' ELEMENTS OF SET ARE ',10I5,/)
213 FORMAT (2(/),
*           FREE AND LOCKED COORDINATE VECTORS (/,/)
214 FORMAT (' QFC(',I2,') = XMC(',I2,') * QF(',I2,') ')
215 FORMAT (' XDIC(',I2,',',I2,') = XDIC(',I2,') = 0 ')
216 FORMAT (' QFC(',I2,') = NJRM(QFC(',I2,') X QFC(',I2,') ')
217 FORMAT (' QFC(',I2,') = NJRM(QFC(',I2,') X QFC(',I2,') ')
218 FORMAT (' GAM(',I2,',',I2,') = GAM(',I2,') = 0 ')
219 FORMAT (5X, ' SSIX = ',Z8,' UNION OF ALL LABELS IN THE SETS SIX(K)
* ARE ',10I5)
220 FORMAT (5X, ' SSCN = ',Z8,' UNION OF ALL LABELS IN THE SETS SCN(K)
* ARE ',10I5)
221 FORMAT (3(/))
222 FORMAT (' QF(',I2,') = ',3D15.5,5X, ' QFC(',I2,') = ',3D15.5)
223 FORMAT (' QL(',I2,') = ',3D15.5,5X, ' QLC(',I2,') = ',3D15.5)
224 FORMAT (' SVQ(',I2,') = ',15)
225 FORMAT (' SVP(',I2,') = ',15)
226 FORMAT (3(/),25X, ' MOMENTUM WHEELS (/,/)
227 FORMAT (' HMC(',I2,') = XMC(',I2,') * HM(',I2,') ')
228 FORMAT (' CAC(',I2,') = CAC(',I2,') + THA(',I2,')*QFC(',I2,') = '
* ',3D15.5)
229 FORMAT (2(/),
*           SVM = ',Z8,' ELEMENTS OF SET ARE ',10I5)
230 FORMAT (' HMC(',I2,') = ',3D15.5)
231 FORMAT (/, ' SFLX = ',Z8,' ELEMENTS OF SET ARE ',10I5,/)
232 FORMAT (39X, ' CAO(',I2,') = CA(',I2,') ')
233 FORMAT (' ')
234 FORMAT (39X, ' XIC(',I2,') = XI(',I2,') ')
235 FORMAT (65X, ' QLC(',I2,') = NJRM(QFC(',I2,') X QFC(',I2,') ')
236 FORMAT (35X, ' QF(',I2,') = QF(',I2,') X QF(',I2,') ')
237 FORMAT (35X, ' QL(',I2,') = QF(',I2,') X QF(',I2,') ')
238 FORMAT (35X, ' CL(',I2,') = QF(',I2,') X CL(',I2,') ')
239 FORMAT (35X, ' QL(',I2,') = XMT(',I2,')**T * QF(',I2,') X QL(',I2,')
* ',2,') ')
240 FORMAT (35X, ' CL(',I2,') = QL(',I2,') X QL(',I2,') ')
241 FORMAT (/, ' SVA = ',Z8,' ELEMENTS OF SET ARE ',10I5,/)
242 FORMAT (' FLE(',I2,') = FLD(',I2,') + FLD(',I2,')**T = ',27X,3D12.
* E)
243 FORMAT (' FLH(',I2,') = FLD(',I2,') + FLJ(',I2,') = ',30X,3D12.5)
244 FORMAT (61X,3D12.5)
245 FORMAT (' FLU(',I2,') = FLC(',I2,') + XMAS(',I2,')*(FLB(',I2,') -
* CAO(',I2,') X FLA(',I2,') = ',3D12.5)
246 FORMAT (30X, ' CA(',I2,') = CA(',I2,') + THA(',I2,')*FLA(',I2,')
247 FORMAT (30X, ' XI(',I2,') = XI(',I2,') + THA(',I2,')*FLE(',I2,')
248 FORMAT (' FLAC(',I2,') = XMC(',I2,')*FLA(',I2,') = ',31X,3D12.5)
249 FORMAT (' FLQC(',I2,') = XMC(',I2,')*FLU(',I2,') = ',31X,3D12.5)
252 FORMAT (30X, ' CA(',I2,') = ',3D12.5)
254 FORMAT (41X,3D12.5)
255 FORMAT (30X, ' XI(',I2,') = ',3D12.5)

```

```

256 FORMAT (/, ' SVA = ',Z8,' ELEMENTS OF SET ARE ',10I5,/) C1C69600
257 FORMAT (/////,40X,'FLEXIELE BODY ',I3,/) C1C69700
258 FORMAT (30X,' BODY ',I3,' EFFECTS OF ELASTIC MODE ',I3,/) C1C69800
259 FORMAT (15X,' ELASTICALLY DEFORMED CENTER OF MASS VECTOR AND INERC C1C69900
* TIA TENSOR FOR BODY ',I3,/) C1C70000
RETURN C1C70100
ENC 01C70200

```

```

C SUBROUTINE EGIV(Y,NEQ) C1100000
C USED TO SET UP INITIAL VALUES FOR RUNGE C1100100
C C1100200
C C1100300
C C1100400

```

```

IMPLICIT REAL*8(A-H,O-Z,3) C1100500
LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLO, LEQU, LINIT(1) C1100600
LOGICAL LRUNGE, LTRNSI, LVDIV, LEQIV, LTRAN, LTRANV, LRATE, LXDY, LETA, LTORQU, C1100700
* LQFDOT, LDCT, LANGLE, LSETUP, LSIMQ C1100800
* C1100900
C1101000

```

```

C C1101100
C INTEGER C1101200
* ADRK, CT1, CT2, CT3, CT4, CT5, FCON, FCCN, C1101300
* SCNCLM, SCN, SCRDM, SCR, SFKDM, SFK, SFR, SG, C1101400
* SI, SIG, SIXDM, SIX, SKDM, SK, SL, SLK, C1101500
* SMA, SMCDUM, SMC, SMV, SGK, SPIDUM, SPI, SQF, C1101600
* SGL, SR, SSCN, SSIX, SVA, SVB, SVD, SVI, C1101700
* SVM, SVP, SVQ, SXM, SXT, TORQ, SMAL, SEU, C1101800
* SC, SCG, NFLXB, SFLX, SFXM, NMUDS, SFCC, SCC, C1101900
* IINIT(1), IZINIT(1), SO C1102000

```

```

C C1102100
C REAL*8 C1102200
* ANGD (33), CNF (2,10), ETIC (3,10), ETMC (3,10), C1102400
* FLO (3,20), FLE (3,3,20), FLH (3,3,20), C1102500
* THACD (33), YMCD (3,2,11), RINIT (1), RZINIT(1) C1102600
C1102700

```

```

C C1102800
C C1102900
C C1103000
COMMON /LDEBUG/ LRUNGE, LTRNSI, LVDIV, LEQIV, LTRAN, C1103100
* LTRANV, LRATE, LXDY, LETA, LTORQU, C1103200
* LQFDOT, LDCT, LANGLE, LSETUP, LSIMQ C1103300
C1103400
C1103500

```

```

C COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RBLO(10) C1103600
C C1103700
C C1103800

```

```

C COMMON /INTG/ A*ORX (200) C1103900
C C1104000
* CT5, SI, FCON, (33), JCON (10), LCON (22), C1104100
* MC (10), NE1, N800, NCTC, C1104200
* NFER, NFKC, NFRG, NLOR, C1104300
* NVV, NMO, NMOA, NSVP, C1104400
* NSVC, FCON (11), SD, SFR (33), C1104500
* SC, SI (55), SIG, SL, C1104600
* SLK (33), SMA (10), SUK (11), SQF (11), C1104700

```

```

* SGL (11) , SMV , SR , SSCN , C1104800
* SSIX , SVA , SVB , SVD , C1104900
* SVI , SVM , SVP (22) , SVC (33) , C1105000
* SXM (3,10) , SXT , TURQ (97) , SMAL , C1105100
* SEU , NTQ , SC (33) , SCG , C1105200
* NFLXB , SFLX , SFXM (10) , NMDS , C1105300
* SFCC , SCC (10) , C1105400
C C1105500
C C1105600
COMMON /INTGZ/ C1105700
* SCNCLM , SCN (5) , SCRDM , SCR (9) , C1105800
* SFKCLM , SFK (5) , SIXDM , SIX (9) , C1105900
* SKDLM , SK (5) , SPIDUM , SPI (9) , C1106000
* SMCCDM , SMC (5) , C1106100
C C1106200
C C1106300
COMMON /REAL/ C1106400
* CA (3,10) , CAC (3,10) , CLM (10) , CCMC (3,11) , C1106500
* DCMC (3,11) , ETC (3,11) , ETM (33) , FUMC (3,11) , C1106600
* GAM (3,66) , H , HM (3,10) , HMC (3,10) , C1106700
* HMCM (10) , FHI (3,11) , PLM (10) , QF (3,33) , C1106800
* QFC (3,33) , GL (3,22) , QLC (3,22) , RCMC (3,11) , C1106900
* T , THA (33) , THAD (33) , C1107000
* THADW (10) , TFAW (10) , XDIC (3,3,66) , XI (3,3,10) , C1107100
* XIC (3,3,10) , XMAS (10) , XMN (3,3,33) , XMT (3,3,10) , C1107200
* TUG (33) , FLA (3,20) , FLB (3,20) , FLC (3,20) , C1107300
* FLD (3,3,20) , FLJ (3,3,20) , CAO (3,10) , XID (3,3,10) , C1107400
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLQC (3,20) , C1107500
* FLCM (20) , ZETA (20) , FCF (3,3,40) , FCK (3,40) , C1107600
* TIMEND , C1107700
C 01107800
C 01107900
COMMON /REALZ/ C1108000
* CEDLM (1,3) , CB (3,10) , CBCDM (1,3) , CBC (3,10) , C1108100
* XCCDM (1,1,9) , XMC (3,3,10) , CBN (3) , C1108200
C C1108300
C C1108400
EQUIVALENCE (ETM(1),THADD(1)) ,(XMN(1,1),ANGD(1)) , C1108500
* (XMN(1,3),YMCD(1,1,1)) ,(XMN(1,6),CNF(1,1)) , C1108600
* (XMN(1,8),ETIC(1,1)) ,(XMN(1,10),ETMC(1,1)) , C1108700
* (FLB(1,1),FLQ(1,1)) ,(FLE(1,1,1),FLD(1,1,1)) , C1108800
* (FLH(1,1,1),FLJ(1,1,1)) , C1108900
* (FG1,LINIT(1)) ,(CA(1,1),RINIT(1)) , C1109000
* (CBDUM(1,1),RZINIT(1)) ,(A*ORK(1),IINIT(1)) , C1109100
* (SCNDUM,IZINIT(1)) , C1109200
C C1109300
C C1109400
INTEGER ST1(11) C1109500
EQUIVALENCE (LEQUIV,LEQU) C1109600
C C1109600
C DIMENSION Y(NEG) C1109700
C C1109800
C C1109900
C ANGULAR OR LINEAR RATE WITH RESPECT TO FREE AXES C1110000
IF(LEQU)PRINT 100 C1110100
IF(LEQU)PRINT 101, NFER ADMN: 4 C1110200
DO 1 N=1,NFER C1110300
Y(N) = THAD(N) C1110400
IF(LEQU)PRINT 102, N,N,Y(N) C1110500
1 CONTINUE C1110600
AN = NFER C1110700

```

C		C1110800
C		C1110900
C	GENERALIZED ELASTIC COORDINATE RATE EQUATIONS	C1111000
	IF(LEQU) PRINT 112, NMODS	C1111100
	IF(NMODS.EQ.0) GO TO 7	C1111200
	DO 7 N=1,NMODS	C1111300
	NA = NN + N	C1111400
	Y(NA) = THAD(NA)	C1111500
	IF(LEQU) PRINT 102, NA, NA, Y(NA)	C1111600
	7 CONTINUE	C1111700
	NN = NN + NMODS	C1111800
C		C1111900
C		C1112000
C	RELATIVE ANGULAR MOMENTUM OF WHEEL	C1112100
	(CALL UNPAC(ST1,NST1,SMV)	C1112200
	IF(LEQU)PRINT 103, NST1	C1112300
	IF(NST1.EQ.0) GO TO 5028	C1112400
	DO 2 NNN=1,NST1	C1112500
	N=NST1-(NNN-1)	C1112600
	NA = NN + NST1 + 1 - N	C1112700
	Y(NA) = THAD*(ST1(N))	C1112800
	IF(LEQU)PRINT 104, NA,ST1(N),Y(NA)	C1112900
	2 CONTINUE	C1113000
5028	CONTINUE	C1113100
	NN = NN + NMV	C1113200
C		C1113300
C	DISPLACEMENT ABOUT OR ALONG FREE AXES	C1113400
	IF(LEQU)PRINT 105, NFRC	C1113500
	IF(NFRC.EQ.0) GO TO 5029	C1113600
	DO 3 N=1,NFRC	C1113700
	M = SFR(N)	C1113800
	NA = NN+N	C1113900
	Y(NA) = THA(M)	C1114000
	IF(LEQU)PRINT 106, NA,M,Y(NA)	C1114100
	3 CONTINUE	C1114200
5029	CONTINUE	C1114300
	NN = NN+NFRC	C1114400
C		C1114500
C		C1114600
C	GENERALIZED ELASTIC COORDINATE DISPLACEMENT EQUATIONS	C1114700
	IF(LEQU) PRINT 112, NMODS	C1114800
	IF(NMODS.EQ.0) GO TO 9	C1114900
	DO 9 N=1,NMODS	C1115000
	NA = NN + N	C1115100
	NST1 = NFER + N	C1115200
	Y(NA) = THA(NST1)	C1115300
	IF(LEQU) PRINT 106, NA,NST1,Y(NA)	C1115400
	9 CONTINUE	C1115500
	NN = NN + NMODS	C1115600
C		C1115700
C		C1115800
C	DISPLACEMENT ABOUT WHEEL SPIN AXIS	C1115900
	IF(LEQU)PRINT 107, NMOA	C1116000
	IF(NMOA.EQ.0) GO TO 5030	C1116100
5030	DO 4 N=1,NMOA	C1116200
	M = SMA(N)	C1116300
	NA = NN+N	C1116400
	Y(NA) = THAW(M)	C1116500
	IF(LEQU)PRINT 108, NA,M,Y(NA)	C1116600
	4 CONTINUE	C1116700

5030 CONTINUE	C1116800
NN = NN + NMUA	C1116900
C	C1117000
C    DIRECTICA COSINES	C1117100
(CALL UNPAC(ST1,NST1,SD)	C1117200
IF(LEQU)PRINT 109, SD,(ST1(I),I=1,NST1)	C1117300
IF(LEQU)PRINT 110, INERF	C1117400
IF(NST1.EQ.0) GO TO 5031	C1117500
N = NN	C1117600
M = C	C1117700
IF(INERF.OR.ST1(NST1).NE.1) GO TO 5	C1117800
NST1 = NST1 - 1	C1117900
DO 8 J=1,2	C1118000
DO 8 I=1,3	C1118100
NN = NN + 1	C1118200
Y(NN) = XMC(I,J,M)	C1118300
IF(LEQU)PRINT 111, NN,I,J,M,Y(NN)	C1118400
E CONTINUE	C1118500
5 IF(NST1.EQ.0) GO TO 5031	C1118600
DO 6 N=1,NST1	C1118700
M = ST1(N)	C1118800
DO 6 J=1,2	C1118900
DO 6 I=1,3	C1119000
NN = NN + 1	C1119100
Y(NN) = XMC(I,J,M)	C1119200
IF(LEQU)PRINT 111, NN,I,J,M,Y(NN)	C1119300
6 CONTINUE	C1119400
5031 CONTINUE	C1119500
NEG = NN	C1119600
C	C1119700
100 FORMAT ('1 ENTER SUBROUTINE EQIV ')	C1119800
101 FORMAT ('    NFER =',I5)	C1119900
102 FORMAT ('    Y(',I2,') = THAC(',I2,') = ',D15.5)	C1120000
103 FORMAT ('    NST1 =',I5)	C1120100
104 FCRMAT ('    Y(',I2,') = THADW(',I2,') = ',D15.5)	C1120200
105 FORMAT ('    NFERC =',I5)	C1120300
106 FORMAT ('    Y(',I2,') = THA(',I2,') = ',D15.5)	C1120400
107 FORMAT ('    NMGA =',I5)	C1120500
108 FORMAT ('    Y(',I2,') = THAW(',I2,') = ',D15.5)	C1120600
109 FORMAT ('    SD =',Z8,' ELEMENTS IN ARRAY ST1 ARE',I0I5)	C1120700
110 FCRMAT ('    INERF =',L10)	C1120800
111 FORMAT ('    Y(',I2,') = XMC(',I2,',',I2, ',',',I2,') = ',D15.5)	C1120900
112 FORMAT ('    NMODS =',I5)	C1121000
C	C1121100
RETURN	C1121200
END	C1121300
C	C1200000
SUBROUTINE TRAN	C1200100
C    COMPUTE ALL TRANSFORMATION MATRICES	C1200200
C        1) USE ORTHOGONALITY TO GET MISSING ELEMENTS	C1200300
C        2) CONSTRUCT OTHERS NOT IN SD	C1200400
C	C1200500
IMPLICIT REAL*8(A-H,O-Z,4)	C1200600
C	C1200700
C	C1200800

	LOGICAL	FG1, FG2, FG3, FG4, FG5, INERF, RBLU, LEQU, LINIT(1)	C120C900
	LOGICAL	LRLNGE, LTRNSI, LVDIV, LEQJIV, LTRAN,	C1201000
	*	LTRANV, LRATE, LXDY, LETA, LTRQQU,	C1201100
	*	LCFDDOT, LDCT, LANGLE, LSETUP, LSIMG	C1201200
C			C1201300
C			C1201400
	INTEGER		C1201500
	* AWGFK, CT1	, CT2, CT3, CT4, CT5, FCCN, PCON,	C1201600
	* SCNCUM, SCN	, SCRUM, SCR, SFKDUM, SFK, SFR, SG,	C1201700
	* SI, SIG	, SIXDUM, SIX, SKDUM, SK, SL, SLK,	C1201800
	* SMA, SMCDUM, SMC	, SMV, SOK, SPIDUM, SPI, SQF,	C1201900
	* SCL, SR	, SSCN, SEIX, SVA, SVB, SVD, SVI,	C1202000
	* SVM, SVP	, SVQ, SXM, SXT, TORQ, SMAL, SEU,	C1202100
	* SC, SCG	, SFLXB, SFLX, SFXM, NMCDS, SFCC, SCC,	C1202200
	* IINIT(1)	, IZINIT(1), SD	C1202300
C			C1202400
C			C1202500
	REAL*8		C1202600
	* ANGE (33)	, CNF (3,10), ETIC (3,10), ETMC (3,10),	C1202700
	* FLQ (3,20)	, FLE (3,3,20), FLH (3,3,20),	C1202800
	* THAED (33)	, YMCD (3,2,11), RINIT (1), RZINIT(1)	C1202900
C			C1203000
C			C1203100
C			C1203200
C			C1203300
	COMMON /LDEBUG/	LRUNGE, LTRNSI, LVDIV, LEQUIV, LTRAN,	C1203400
	*	LTRANV, LRATE, LXDY, LETA, LTRQQU,	C1203500
	*	LQFDDOT, LDCT, LANGLE, LSETUP, LSIMG	C1203600
C			C1203700
C			C1203800
	COMMON /LOGIC/	FG1, FG2, FG3, FG4, FG5, INERF, RBLJ(10)	C1203900
C			C1204000
C			C1204100
	COMMON /INTG/	A*ORK(200),	C1204200
	* CT1	, CT2, CT3, CT4,	C1204300
	* CT5	, FCON (33), JCON (10), LCON (22),	C1204400
	* MC (10)	, NB1, NBD, NCTC,	C1204500
	* NFEF	, NFKC, NFR, NLOR,	C1204600
	* NMV	, NMD, NMCA, NSVP,	C1204700
	* NSVC	, FCON (11), SD, SFR (33),	C1204800
	* SC	, SI (55), SIG, SL,	C1204900
	* SLK (33)	, SMA (10), SOK (11), SQF (11),	C1205000
	* SCL (11)	, SMV, SR, SSCN,	C1205100
	* SEIX	, SVA, SVB, SVD,	C1205200
	* SVI	, SVM, SVP (22), SVQ (33),	C1205300
	* SXM (3,10)	, SXT, TORQ (97), SMAL,	C1205400
	* SEU	, NTQ, SC (33), SCG,	C1205500
	* NFLXB	, SFLXB, SFLX, SFXM (10), NMODS,	C1205600
	* SFCC	, SCC (10)	C1205700
C			C1205800
C			C1205900
	COMMON /INTGZ/		C1206000
	* SCNCUM	, SCN (5), SCRUM, SCR (9),	C1206100
	* SFKDUM	, SFK (5), SIXDUM, SIX (9),	C1206200
00500	* SKDUM	, SK (5), SPIDUM, SPI (9),	C1206300
	* SMCDUM	, SMC (5)	C1206400
C			C1206500
C			C1206600
	COMMON /REAL/		C1206700
	* CA (3,10)	, CAC (3,10), CLM (10), CCMC (3,11),	C1206800



```

* DCMC (3,11) , ETC (3,11) , ETM (33) , FCMC (3,11) , C1206900
* GAM (3,66) , F (3,10) , HM (3,10) , HMC (3,10) , C1207000
* HNCM (10) , PHI (3,11) , PLM (10) , QF (3,33) , C1207100
* QFC (3,33) , GL (3,22) , QLC (3,22) , RQMC (3,11) , C1207200
* T , THA (33) , THAD (33) , C1207300
* THAD* (10) , THAW (10) , XCIC (3,3,66) , XI (3,3,10) , C1207400
* XIC (3,3,10) , XMAS (10) , XMN (33,33) , XMT (3,3,10) , C1207500
* TLG (33) , FLA (3,20) , FLB (3,20) , FLC (3,20) , C1207600
* FLD (3,3,20) , FLJ (3,3,20) , CAO (3,10) , XIG (3,3,10) , C1207700
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLQC (3,20) , C1207800
* FLOM (20) , ZETA (20) , FCF (3,3,40) , FCK (3,40) , C1207900
* TIMEND , C1208000
C , C1208100
C , C1208200
COMMON /REALZ/ , C1208300
* CEDLM (1,3) , CB (3,10) , CBCDUM(1,3) , CBC (3,10) , C1208400
* XMCDLM(1,1,9) , XMC (3,3,10) , CBN(3) , C1208500
C , C1208600
C , C1208700
EQUIVALENCE (ETM(1),THAD(1)) , (XMN(1,1),ANGD(1)) , C1208800
* (XMN(1,3),YMCD(1,1,1)) , (XMN(1,6),CNF(1,1)) , C1208900
* (XMN(1,8),ETIC(1,1)) , (XMN(1,10),ETMC(1,1)) , C1209000
* (FLB(1,1),FLC(1,1)) , (FLE(1,1,1),FLD(1,1,1)) , C1209100
* (FLH(1,1,1),FLJ(1,1,1)) , C1209200
* (FG1,LINIT(1)) , (CA(1,1),RINIT(1)) , C1209300
* (CBDUM(1,1),RZINIT(1)) , (AWORK(1),IINIT(1)) , C1209400
* (SCNDLM,IZINIT(1)) , C1209500
C , C1209600
LOGICAL CTAIN , C1209700
DIMENSION TEMP(3,3) , C1209800
DIMENSION XMS(3,3),Q(2),CS(2),TEM(3,2),TEMP(3,3) , C1209900
INTEGER STI(10),SET(10) , C1210000
EQUIVALENCE (LTRAN,LEQU) , C1210100
C , C1210200
C , C1210300
C , C1210400
IF(.NOT. LEQU) GO TO 100C , C1210500
PRINT 100 , C1210600
J0 = 0 , C1210700
J1 = 1 , C1210800
J2 = 2 , C1210900
J3 = 3 , C1211000
PRINT 101, INERF , C1211100
100C CALL UNPAC (SET,NSET,SD) , C1211200
IF(LEQU)PRINT 102, SD,(SET(I),I=1,NSET) , C1211300
IF(NSET.EQ.0) GO TO 5022 , C1211400
KST = 1 , C1211500
IF(INERF.OR.SET(NSET).NE.1) GO TO 1 , C1211600
NSET = NSET - 1 , C1211700
KST = 0 , C1211800
K = 0 , C1211900
KK=KST , C1212000
IF(KST.GT.NSET)GO TO 5022 , C1212100
5021 IF(KK.EQ.0) GO TO 19 , C1212200
K = SET(KK) , C1212300
19 CALL VECFOS (XMC(1,1,K),XMC(1,2,K),XMC(1,3,K)) , C1212400
IF(.NOT. LEQU) GO TO 2 , C1212500
PRINT 104 , C1212600
PRINT 115, (XMC(1,1,K),I=1,3) , C1212700
PRINT 116,K, (XMC(2,1,K),I=1,3) , C1212800

```

	FRINT 115, (XMC(3,I,K),I=1,3)	C1212900
2	CONTINUE	C1213000
	KK=KK+1	C1213100
	IF(KK.LE.NSET) GO TO 5021	C1213200
5022	CONTINUE	C1213300
C		C1213400
C	COMPUTE ELEMENTS OF TRANSFORMATION MATRICES OBTAINABLE	C1213500
C	VIA SMALL ANGLE ASSUMPTIONS OR EULER ANGLES	C1213600
C		C1213700
	IS = SK - SD	C1213800
	CALL UNFAC(SET,NSET,IS)	C1213900
	CALL UNFAC(ST1,NST1,SMAL)	C1214000
	IF(.NCT.LEQU) GO TO 3	C1214100
	FRINT 107, IS,(SET(I),I=1,NSET)	C1214200
	FRINT 104	C1214300
3	IF(NSET.EQ.0) GO TO 5022	C1214400
	DO 4 III=1,NSET	C1214500
	JJJ=NSET-(III-1)	C1214600
	JJ = SET(JJJ)	C1214700
	J = JCCN(JJ)	C1214800
	DO 20 I=1,3	C1214900
	DO 20 L=1,3	C1215000
20	TEMP(I,L) = XMT(I,L,JJ)	C1215100
C	CHECK FOR THREE CONSTRAINED AXES	C1215200
	IF(PCCN(JJ).NE.3) GO TO 5026	C1215300
	DO 27 I=1,3	C1215400
	DO 27 L=1,3	C1215500
	XMS(I,L) = TEMP(I,L)	C1215600
27	CONTINUE	C1215700
5026	CONTINUE	C1215800
	MDCNE = 3-PCCN(JJ)	C1215900
	DO 5 II=1,MDCNE	C1216000
	I = II-1	C1216100
C	CYCLE THROUGH FREE COORDINATE ROTATIONS AT HINGE POINT JJ-I	C1216200
	M = SQF(JJ) + I	C1216300
	KB = EXM(I+1,JJ)	C1216400
C	CHECK SMALL ANGLE ASSUMPTIONS	C1216500
	E = THA(M)	C1216600
	C = 1.0DC	C1216700
	IF(CTAIN(JJ,ST1,NST1)) GO TO 22	C1216800
	B = CSIN(THA(M))	C1216900
	C = CCOS(THA(M))	C1217000
22	CONTINUE	C1217100
	IF(.NCT.LEQU) GO TO 5027	C1217200
	FRINT 120, JJ, J,I,M,KB,E,C	C1217300
120	FORMAT (' JJ = ',I3,' J = ',I3,' I = ',I3,' M = ',I3,' KB = ',I3,' B = ',	C1217400
	*,D17.8,' C = ',D17.8)	C1217500
5027	CONTINUE	C1217600
	CALL ROT(B,C,KE,TEMP1)	C1217700
	CALL MATMUL(TEMP,TEMP1,XMS,3)	C1217800
	DO 25 L=1,3	C1217900
	DO 25 LL=1,3	C1218000
	TEMP(L,LL) = XMS(L,LL)	C1218100
25	CONTINUE	C1218200
5028	CONTINUE	C1218300
	IF(LEQU) FRINT 118, ((TEMP(I,L),L=1,3),I=1,3)	C1218400
	CALL MATMUL(XMT(1,1,JJ),TEMP,XMS,3)	C1218500
	IF(.NCT.LEQU) GO TO 1002	C1218600
	FRINT 104	C1218700
	FRINT 109, (XMS(1,I),I=1,3)	C1218800

FRINT 110, JJ, (XMS(2,1), I=1,3)	C121890C
FRINT 109, (XMS(3,1), I=1,3)	C1219000
FRINT 104	C1219100
1002 IF (J.NE.C) GO TO 16	C121920C
IF (INERF) GO TO 17	C1219300
DO 15 I=1,3	01219400
DO 15 L=1,3	C1219500
15 XMC(I,L,C) = XMS(L,I)	C1219600
IF (.NOT. LEQU) GO TO 4	C1219700
FRINT 112, (XMC(1,1,0), I=1,3)	C121980C
FRINT 112, J, JJ, (XMC(2,1,0), I=1,3)	C121990C
FRINT 112, (XMC(3,1,0), I=1,3)	C122000C
FRINT 104	0122010C
GO TO 4	C1220200
17 DO 16 I=1,3	C122030C
DO 16 L=1,3	C122040C
18 XMC(I,L,1) = XMS(I,L)	C122050C
IF (.NOT. LEQU) GO TO 4	C1220600
FRINT 112, (XMC(1,1,1), I=1,3)	C1220700
FRINT 114, JJ, JJ, (XMC(2,1,1), I=1,3)	C122080C
FRINT 112, (XMC(3,1,1), I=1,3)	C1220900
GO TO 4	C1221000
16 CALL MATMUL(XMC(1,1,J), XMS, XMC(1,1, JJ), 3)	C1221100
IF (.NOT. LEQU) GO TO 4	C122120C
FRINT 105, (XMC(1,1, JJ), I=1,3)	C1221300
FRINT 106, JJ, J, JJ, (XMC(2,1, JJ), I=1,3)	C122140C
FRINT 105, (XMC(3,1, JJ), I=1,3)	C122150C
FRINT 104	C1221600
4 CONTINUE	C122170C
5032 CONTINUE	C1221800
C	C1221900
C UPDATE TRANSFORMATION MATRICES ASSOCIATED WITH LINEAR OSCILLATORS	C1222000
CALL UNPAC(SET, NSET, SL)	C122210C
IF (NSET.EQ.0) GO TO 5033	C1222200
DO 13 JJJ=1, NSET	C1222300
JJ = SET(JJJ)	C1222400
J = JCCN(JJ)	C1222500
DO 14 I=1,3	C1222600
DO 14 L=1,3	C1222700
14 XMC(I,L, JJ) = XMC(I,L, J)	C1222800
IF (LEQU) PRINT 111, JJ, J	C122290C
13 CONTINUE	C1223000
5033 CONTINUE	C1223100
IF (LEQU) PRINT 104	C1223200
100 FORMAT ('1 SUBROUTINE TRAN ENTERED ', //)	C1223300
101 FORMAT (' INERF = ', L5, //)	C1223400
102 FORMAT (' SD = ', Z8, ' SET ELEMENTS ARE ', I015, //)	C1223500
104 FORMAT (' ')	C1223600
105 FORMAT (30X, 3D17.8)	C1223700
106 FORMAT (' XMC(' , I2, ') = XMC(' , I2, ') * XMS(' , I2, ') = ', 3D17.8)	C1223800
107 FORMAT (' SR-SD = ', Z8, ' SET ELEMENTS ARE ', I015, //)	C1223900
109 FORMAT (12X, 3D17.8)	C1224000
110 FORMAT (' XMS(' , I2, ') = ', 3D17.8)	C1224100
111 FORMAT (' , ' XMC(' , I2, ') = XMC(' , I2, ') ')	C1224200
112 FORMAT (25X, 3D17.8)	C1224300
113 FORMAT (' XMC(' , I2, ') = XMS(' , I2, ') **T = ', 3D17.8)	C1224400
114 FORMAT (' XMC(' , I2, ') = XMS(' , I2, ') = ', 3D17.8)	C1224500
115 FORMAT (12X, 3D17.8)	C1224600
116 FORMAT (' XMC(' , I2, ') = ', 3D17.8)	C122470C
118 FORMAT (' TEMP = ', 3D17.8)	C1224800

RETURN  
END

C1224900  
C1225000

SUBROUTINE TRANVD  
TRANSFORMS ONLY THOSE VECTORS AND DYADS TIME VARYING IN  
COMPUTING FRAME

IMPLICIT REAL\*8(A-H,O-Z,I)  
LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLO, LEQU, LINIT(1)  
LOGICAL LRNGE, LTRNSI, LVDIV, LEQUIV, LTRAN,  
\* LTRANV, LRATE, LXDY, LETA, LTORQU,  
\* LGFDOT, LDCT, LANGLE, LSETUP, LSIMQ

INTEGER  
\* AWCCK, CT1, CT2, CT3, CT4, CT5, FCON, PCCN,  
\* SCNDUM, SCN, SCR, SFR, SFCUM, SFC, SFR, SG,  
\* SI, SIG, SIXDUM, SIX, SKDUM, SK, SL, SLK,  
\* SMA, SMCDUM, SMC, SMV, SOK, SPIDUM, SPI, SQF,  
\* SCL, SR, SSCN, SSIX, SVA, SVB, SVD, SVI,  
\* SVM, SVP, SVQ, SXM, SXT, TORQ, SMAL, SEU,  
\* SC, SCG, NFLXB, SFLX, SFXM, NMGDS, SFCC, SCC,  
\* IINIT(1), IZINIT(1), SD

REAL\*8  
\* ANGE (33), CNF (3,10), ETIC (3,10), ETMC (3,10),  
\* FLQ (3,20), FLE (3,3,20), FLH (3,3,20),  
\* THACD (33), YMCD (3,2,11), RINIT (1), RZINIT(1)

COMMON /LDEBUG/ LRNGE, LTRNSI, LVDIV, LEQUIV, LTRAN,  
\* LTRANV, LRATE, LXDY, LETA, LTORQU,  
\* LGFDOT, LDCT, LANGLE, LSETUP, LSIMQ

COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RBLO(10)

COMMON /INTG/ AWCCK(200),

\* CT1, CT2, CT3, CT4  
\* CT5, FCON (33), JCON (10), LCCN (22)  
\* MC (10), NBI, NBOO, NCTC  
\* NFER, NFKC, NFRG, NLDR  
\* NMV, NMO, NMDA, NSVP  
\* NEVC, FCON (11), SD, SFR (33)  
\* SC, SI (55), SIG, SL  
\* SLK (33), SMA (10), SOK (11), SQF (11)  
\* SCL (11), SMV, SR, SSCN  
\* SSIX, SVA, SVB, SVD  
\* SVI, SVM, SVP (22), SVQ (33)  
\* SXM (3,10), SXT, TORQ (97), SMAL

```

* SEU          , NTQ          , SC   (33)   , SCG          , C1305300
* NFLXB       , SFLX          , SFXM  (10)   , AMCDS       , C1305400
* SFCC        , SCC   (10)   ,          ,          , C1305500
C
C
COMMON /INTGZ/
* SCNDLM      , SCN   (9)    , SCRDM      , SCR   (9)    , C1305900
* SFKCDM      , SFK   (9)    , SIXDM      , SIX   (9)    , C1306000
* SKDLM       , SK    (9)    , SPIDUM     , SPI   (9)    , C1306100
* SMCDLM      , SMC   (9)    ,          ,          , C1306200
C
C
COMMON /REAL/
* CA   (3,10) , CAC   (3,10) , CLM   (10) , CGMC  (3,11) , C1306500
* DCMC (3,11) , ETC   (3,11) , ETM   (33) , FDMC  (3,11) , C1306600
* GAM  (3,66) , H     , HM    (3,10) , HMC   (3,10) , C1306800
* HMOP (10)   , PHI  (3,11) , PLM   (10) , QF    (3,33) , C1306900
* QFC  (3,33) , QL   (3,22) , QLC   (3,22) , RCMC  (3,11) , C1307000
* T     ,          , THA   (33) , THAD  (33) , C1307100
* THACW (10) , THAW (10) , XDIC  (3,3,66) , XI   (3,3,10) , C1307200
* XIC   (3,3,10) , XMAS (10) , XMN   (33,33) , XMT  (3,3,10) , C1307300
* TUG   (33) , FLA  (3,20) , FLB   (3,20) , FLC   (3,20) , C1307400
* FLD   (3,3,20) , FLJ  (3,3,20) , CAU   (3,10) , XIU  (3,3,10) , C1307500
* FLIRC (3,10) , FLCRC (3,10) , FLAC  (3,20) , FLOC  (3,20) , C1307600
* FLCW  (20) , ZETA (20) , FCF   (3,3,40) , FCK   (3,40) , C1307700
* TIMEND
C
C
COMMON /REALZ/
* CEDLM (1,3) , CE   (3,10) , CbCDUM(1,3) , CBC   (3,10) , C1308100
* XMCDUM(1,1,9) , XMC  (3,3,10) , CbN(3) ,          , C1308200
C
C
EQUIVALENCE (ETM(1),THAD(1)) , (XMN(1,1),ANGD(1)) , C1308500
* (XMN(1,3),YMCD(1,1,1)) , (XMN(1,6),CNF(1,1)) , C1308700
* (XMN(1,8),ETIC(1,1)) , (XMN(1,10),ETMC(1,1)) , C1308800
* (FLB(1,1),FLQ(1,1)) , (FLE(1,1,1),FLD(1,1,1)) , C1308900
* (FLH(1,1,1),FLJ(1,1,1)) ,          , C1309000
* (FG1,LINIT(1)) , (CA(1,1),RINIT(1)) , C1309100
* (CBDUM(1,1),RZINIT(1)) , (AWORK(1),IINIT(1)) , C1309200
* (SCNDLM,IZINIT(1)) ,          , C1309300
C
LOGICAL CTAIN C1309400
DIMENSION TEM(3) C1309500
DIMENSION TEM1(3,3) C1309600
INTEGER ST1(10),SET(18),SFXMN C1309700
EQUIVALENCE (LTRANV,LECU) C1309800
C1309900
C1310000
IF(LEQU)PRINT 200 C1310100
C1310200
C1310300
ELASTIC DEFORMATION PARAMETERS C1310400
IF(NFLXB.EQ.0) GO TO 15 C1310500
MN = C C1310600
CALL UNPAC(SET,NSET,SFLX) C1310700
CALL UNPAC(ST1,NST1,SVD) C1310800
IF(NSET.EQ.0) GO TO 5000 C1310900
DO 16 NN=1,NSET C1311000
N = SET(NSET+1-NN) C1311100
IFLX = C C1311200

```

IF(CTAIN(N,ST1,NST1)) GO TO 19	C1311300
IFLX = 1	C1311400
15 CONTINUE	C1311500
C IF IFLX=0 MODAL VECTORS AND TENSORS FIXED IN	C1311600
C BODY N MUST BE TRANSFORMED	C1311700
C COMPLETE DEFORMED CM VECTOR AND INERTIA TENSOR BODY N FIXED FRAME	C1311800
IF(.NOT.LEQU) GO TO 1000	C1311900
PRINT 226,N	C1312000
PRINT 205	C1312100
1000 CONTINUE	C1312200
DO 17 I=1,3	C1312300
CA(I,N) = CAC(I,N)	C1312400
DO 17 J=1,3	C1312500
17 XI(I,J,N) = X10(I,J,N)	C1312600
IF(.NOT.LEQU) GO TO 1001	C1312700
PRINT 220, N,N	C1312800
PRINT 221, N,N	C1312900
PRINT 205	C1313000
1001 CONTINUE	C1313100
SFXM = SFXM(N)	C1313200
DO 18 M=1,SFXM	C1313300
MN = MN+1	C1313400
CALL SCLV(THA(NFER+MN),FLA(1,MN),TEM)	C1313500
CALL VECADD(CA(1,N),TEM,CA(1,N))	C1313600
CALL SCLV(THA(NFER+MN),FLE(1,1,MN),TEM1)	C1313700
CALL CYADD(XI(1,1,N),TEM1,XI(1,1,N))	C1313800
IF(.NOT.LEQU) GO TO 1002	C1313900
NFXM = NFER+MN	C1314000
PRINT 216, N,N,NFXM,MN	C1314100
PRINT 217, N,N,NFXM,MN	C1314200
PRINT 205	C1314300
1002 CONTINUE	C1314400
IF(IFLX.EQ.1) GO TO 20	C1314500
CALL VECTR(FLA(1,MN),XMC(1,1,N),FLAC(1,MN))	C1314600
CALL VECTR(FLO(1,MN),XMC(1,1,N),FLOC(1,MN))	C1314700
IF(.NOT.LEQU) GO TO 1003	C1314800
PRINT 218, MN,N,MN,(FLAC(I,MN),I=1,3)	C1314900
PRINT 219, MN,N,MN,(FLOC(I,MN),I=1,3)	C1315000
PRINT 205	C1315100
1003 CONTINUE	C1315200
20 CONTINUE	C1315300
18 CONTINUE	C1315400
IF(.NOT.LEQU) GO TO 1004	C1315500
PRINT 223, N,(CA(I,N),I=1,3)	C1315600
PRINT 205	C1315700
PRINT 224, (XI(1,I,N),I=1,3)	C1315800
PRINT 225, N,(XI(2,I,N),I=1,3)	C1315900
PRINT 224, (XI(3,I,N),I=1,3)	C1316000
1004 CONTINUE	C1316100
16 CONTINUE	C1316200
5000 CONTINUE	C1316300
15 CONTINUE	C1316400
C	C1316500
C	C1316600
C CENTER OF MASS VECTORS	C1316700
CALL UNPAC(SET,NSET,SVA)	C1316800
IF(NSET.EQ.0) GO TO 5034	C1316900
DO 1 J=1,NSET	C1317000
K = SET(J)	C1317100
IF(.NOT.RBLO(K)) GO TO 1	C1317200

CALL VECTR (CA(1,K),XMC(1,1,K),CAC(1,K))	C1317300
IF(LEQU)PRINT 201, K,K,K,(CAC(1,K),I=1,3)	C1317400
1 CONTINUE	C1317500
5034 CONTINUE	C1317600
IF(LEQU)PRINT 202	C1317700
C	C1317800
C INERTIA TENSORS	C1317900
CALL UNPAC(SET,NSET,SVI)	C1318000
IF(NSET.EQ.0) GO TO 5035	C1318100
DO 10 J=1,NSET	C1318200
K = SET(J)	C1318300
CALL TENR (XI(1,1,K),XMC(1,1,K),XIC(1,1,K))	C1318400
IF(LEQU)PRINT 203, (XIC(1,1,K),I=1,3)	C1318500
IF(LEQU)PRINT 204, K,K,K,K,(XIC(2,1,K),I=1,3)	C1318600
IF(LEQU)PRINT 203, (XIC(3,1,K),I=1,3)	C1318700
IF(LEQU)PRINT 205	C1318800
10 CONTINUE	C1318900
5035 CONTINUE	C1319000
IF(LEQU)PRINT 202	C1319100
C	C1319200
C FINGE VECTORS	C1319300
C COMPT INERTIAL POSITION OF CENTER OF MASS OF BODY 1	C1319400
DO 14 I=1,3	C1319500
14 CB(I,1) = CBN(I)	C1319600
IF(LEQU)PRINT 214,(CB(I,1),I=1,3)	C1319700
IF(SCF(NB1).EQ.0) GO TO 5042	C1319800
ITERM1=SCF(NB1)	C1319900
ITERM2=ITERM1+2-PCCN(NB1)	C1320000
DO 13 M=ITERM1,ITERM2	C1320100
CALL SCLV(THA(M),QF(1,M),TEM)	C1320200
CALL VECADD(CB(1,1),TEM,CB(1,1))	C1320300
IF(LEQU)PRINT 215,M,M	C1320400
13 CONTINUE	C1320500
5042 CONTINUE	C1320600
CALL UNPAC(SET,NSET,SVB)	C1320700
IF(NSET.EQ.0) GO TO 5036	C1320800
DO 2 J=1,NSET	C1320900
K = SET(J)	C1321000
JJ = JCN(K)	C1321100
CALL VECTR (CB(1,K),XMC(1,1,JJ),CBC(1,K))	C1321200
IF(LEQU)PRINT 206, K,JJ,K,(CBC(1,K),I=1,3)	C1321300
2 CONTINUE	C1321400
5036 CONTINUE	C1321500
IF(LEQU)PRINT 202	C1321600
C	C1321700
C FREE COORDINATE VECTORS	C1321800
IF(NSVQ.EQ.0) GO TO 5037	C1321900
DO 3 J=1,NSVQ	C1322000
M = SVQ(J)	C1322100
IF(FCCN(M).GE.0) GO TO 4	C1322200
CALL VECTR (QF(1,M+1),XMC(1,1,FCCN(M+1)),QFC(1,M+1))	C1322300
CALL VECFOS (QFC(1,M+1),QFC(1,M-1),QFC(1,M))	C1322400
CALL VECR (QFC(1,M))	C1322500
IF(LEQU)M1 = M-1	C1322600
IF(LEQU)M2 = M+1	C1322700
IF(LEQU)PRINT 208, M,M2,M1,(QFC(1,M),I=1,3)	C1322800
GO TO 3	C1322900
4 CALL VECTR (QF(1,M),XMC(1,1,FCCN(M)),QFC(1,M))	C1323000
IF(LEQU)PRINT 207, M,FCCN(M),M,(QFC(1,M),I=1,3)	C1323100
3 CONTINUE	C1323200

5037	CONTINUE	C1323300
	IF(LEQU)PRINT 202	C1323400
C		C1323500
C	LOCKED COORDINATE VECTORS	C1323600
	IF(NSVP.EQ.0) GO TO 5038	C1323700
	DO 5 J=1,NSVP	C1323800
	L = SVP(J)	C1323900
	IF(LCCN(L).GE.0) GO TO 6	C1324000
	M = -LCCN(L)	C1324100
	CALL VECROS (GFC(1,M),GFC(1,M+1),QLC(1,L))	C1324200
	CALL VECNRM (QLC(1,L))	C1324300
	IF(LEQU)M1 = M+1	C1324400
	IF(LEQU)PRINT 209, L,M,M1,(QLC(I,L),I=1,3)	C1324500
	GO TO 5	C1324600
E	CALL VECTR (QL(1,L),XMC(1,1,LCCN(L)),QLC(1,L))	C1324700
	IF(LEQU)PRINT 210, L,LCCN(L),L,(QLC(I,L),I=1,3)	C1324800
	CONTINUE	C1324900
5038	CONTINUE	C1325000
	IF(LEQU)PRINT 202	C1325100
C		C1325200
C	PCINT MASS POSITION	C1325300
	CALL UNPAC(SET,NSET,SL)	C1325400
	IF(NSET.EQ.0) GO TO 5039	C1325500
	DO 9 KK=1,NSET	C1325600
	K = SET(KK)	C1325700
	DO 11 I=1,3	C1325800
11	TEM(I) = CA(I,K)	C1325900
	INI=SGF(K)	C1326000
	IT1=INI+2-PCUN(K)	C1326100
	DO 12 M=INI,IT1	C1326200
	IF(LEQU)PRINT 212, K,K,K,M	C1326300
	DO 12 I=1,3	C1326400
12	TEM(I) = TEM(I) + THA(M)*QF(I,M)	C1326500
	CALL VECTR (TEM,XMC(1,1,K),CAC(1,K))	C1326600
	IF(LEQU)PRINT 213, K,K,K,(CAC(I,K),I=1,3)	C1326700
	CONTINUE	C1326800
5039	CONTINUE	C1326900
C		C1327000
C	ANGULAR MOMENTUM WHEEL	C1327100
	CALL UNPAC(SET,NSET,SVM)	C1327200
	IF(NSET.EQ.0) GO TO 5040	C1327300
	DO 7 JJ=1,NSET	C1327400
	J = SET(JJ)	C1327500
	CALL VECTR (HM(1,J),XMC(1,1,MO(J)),HMC(1,J))	C1327600
7	CONTINUE	C1327700
5040	CONTINUE	C1327800
	IF(.NOT. LEQU) RETURN	C1327900
	IF(NSET.EQ.0) GO TO 5041	C1328000
	DO 8 JJ=1,NSET	C1328100
	J = SET(JJ)	C1328200
	PRINT 211, J,MO(J),J,(HMC(I,J),I=1,3)	C1328300
	CONTINUE	C1328400
5041	CONTINUE	C1328500
	PRINT 202	C1328600
C		C1328700
C		C1328800
	200 FORMAT (' SUBROUTINE TRANVD ENTERED ',2('/))	C1328900
	201 FORMAT (' CAC(' ,I2,') = XMC(' ,I2,') * CA(' ,I2,') = ',3D17.8)	C1329000
	202 FORMAT(3('/))	C1329100
	203 FORMAT (4X,3D17.8)	C1329200



```

204 FORMAT (' XIC(' ,I2,' ) = XMC(' ,I2,' ) * XI(' ,I2,' ) * XMC(' ,I2,' )**TC1329300
      * = ' ,3D17.8) C1329400
205 FORMAT (' ') C1329500
206 FORMAT (' CBC(' ,I2,' ) = XMC(' ,I2,' ) * CB(' ,I2,' ) = ' ,3D17.8) C1329600
207 FORMAT (' QFC(' ,I2,' ) = XMC(' ,I2,' ) * QF(' ,I2,' ) = ' ,3D17.8) C1329700
208 FORMAT (' QFC(' ,I2,' ) = NORM(QFC(' ,I2,' ) X QFC(' ,I2,' )) = ' ,3D17. C1329800
      *E) C1329900
209 FORMAT (' QLC(' ,I2,' ) = NORM(QFC(' ,I2,' ) X QFC(' ,I2,' )) = ' ,3D17. C1330000
      *E) C1330100
210 FORMAT (' QLC(' ,I2,' ) = XMC(' ,I2,' ) * QL(' ,I2,' ) = ' ,3D17.8) C1330200
211 FORMAT (' HMC(' ,I2,' ) = XMC(' ,I2,' ) * HM(' ,I2,' ) = ' ,12X,3D17.8) C1330300
212 FORMAT (' CA(' ,I2,' ) = CA(' ,I2,' ) + THA(' ,I2,' ) * QF(' ,I2,' ) ' ) C1330400
213 FORMAT (35X, ' CAC(' ,I2,' ) = XMC(' ,I2,' ) * CA(' ,I2,' ) = ' ,3D17.8) C1330500
214 FORMAT (' CB( 1) = ' ,3D17.8) C1330600
215 FORMAT (' CB( 1) = CB( 1) + THA(' ,I2,' ) * QF(' ,I2,' ) ' ) C1330700
216 FORMAT (30X, ' CA(' ,I2,' ) = CA(' ,I2,' ) + THA(' ,I2,' ) * FLA(' ,I2,' ) ' ) C1330800
217 FORMAT (30X, ' XI(' ,I2,' ) = XI(' ,I2,' ) + THA(' ,I2,' ) * FLE(' ,I2,' ) ' ) C1330900
218 FORMAT (' FLAC(' ,I2,' ) = XMC(' ,I2,' ) * FLA(' ,I2,' ) = ' ,3D12.5) C1331000
219 FORMAT (' FLGC(' ,I2,' ) = XMC(' ,I2,' ) * FLG(' ,I2,' ) = ' ,3D12.5) C1331100
220 FORMAT (30X, ' CA(' ,I2,' ) = CA(' ,I2,' ) ' ) C1331200
221 FORMAT (30X, ' XI(' ,I2,' ) = XI(' ,I2,' ) ' ) C1331300
222 FORMAT (30X, ' CA(' ,I2,' ) = ' ,3D12.5) C1331400
223 FORMAT (40X,3D12.5) C1331500
224 FORMAT (30X, ' XI(' ,I2,' ) = ' ,3D12.5) C1331600
225 FORMAT (77,25X, ' FLEXIBLE BODY PARAMETERS FOR BODY',I3) C1331700
      RETURN C1331800
      END C1331900

```

```

C C1400000
SUBROUTINE RATE C1400100
C USE TO COMPUTE ALL LINER AND ANGULAR VELOCITY VECTORS REQUIRED C1400200
C C1400300
C RCMC(I,K) = FOR BODY K RIGID- ANGULAR VELOCITY OF BODY K FIXED C1400400
C AXES RELATIVE TO BODY JCN(K) FIXED AXES; C1400500
C FOR BODY K POINT MASS- LINEAR VELOCITY OF POINT MASS C1400600
C REALATIVE TO AXES FIXED AT HINGE POINT K-1 C1400700
C CCMC(I,K) = FOR BODY K RIGID- ANGULAR VELOCITY OF BODY K FIXED C1400800
C AXES RELATIVE TO FRAME OF COMPUTATION C1400900
C FOR BODY K POINT MASS- EQUALS RCMC(I,K) C1401000
C FCMC(I,K) = FOR BODY K RIGID- ANGULAR VELOCITY OF BODY K FIXED C1401100
C AXES RELATIVE TO INERTIAL FRAME C1401200
C FOR BODY K POINT MASS- EQUALS RCMC(I,K) C1401300
C DCMC(I,K) = SUM OF INERTIAL DERIVATIVES(FIRST) OF FREE VECTORS C1401400
C AT HINGE POINT K-1 C1401500
C IMPLICIT REAL*8(A-F,O-Z,4) C1401600
C C1401700
C C1401800
C LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLD, LEQU, LINIT(1) C1401900
C LOGICAL LFRNGE, LTRNSI, LYDIV, LEQUIV, LTRAN C1402000
C (28) LFRANV, LRAE, ALXUY, LETA, LFORGU, C1402100
C * LQFDUT, LDCT, LANGLE, LSETUP, LSIMQ C1402200
C C1402300
C C1402400
C INTEGER C1402500
C * ANCRK, CT1, CT2, CT3, CT4, CT5, FCCN, PCEN, C1402600
C * SCNDUM, SCN, SCRDM, SCR, SFRDM, SFR, SG, C1402700

```

	* SI	, SIG	, SIXDUM	, SIX	, SKDUM	, SK	, SL	, SLK	, C140280C
	* SMA	, SMCDUM	, SMC	, SMV	, SQK	, SPIDUM	, SPI	, SQF	, C1402900
	* SGL	, SR	, SSCN	, SSIX	, SVA	, SVB	, SVD	, SVI	, C1403000
	* SVM	, SVP	, SVQ	, SXM	, SXT	, TURQ	, SMAL	, SEU	, C1403100
	* SC	, SCG	, NFLXB	, SFLX	, SFXM	, NMCDS	, SFCC	, SCC	, C1403200
	* IINIT(1)		, IZINIT(1)		, SD				, C1403300
C									, C1403400
C									, C1403500
C									, C1403600
	REAL#8								, C1403700
	* ANGC	(33)	, CNF	(3,10)	, ETIC	(3,10)	, ETMC	(3,10)	, C1403800
	* FLO	(3,20)	, FLE	(3,3,20)	, FLH	(3,3,20)			, C1403900
	* THADD	(33)	, YMCD	(3,2,11)	, RINIT	(1)	, RZINIT	(1)	, C1404000
C									, C1404100
C									, C1404200
	COMMON /LDEBUG/	LRNGE	, LTRNSI	, LVDIV	, LEQJIV	, LTRAN			, C1404300
	*		, LTRANV	, LRATE	, LXDY	, LETA	, LTORQU		, C140440C
	*		, LCFDOT	, LDCT	, LANGLE	, LSETUP	, LSIMO		, C1404500
C									, C1404600
C									, C1404700
	COMMON /LOGIC/	FG1	, FG2	, FG3	, FG4	, FG5	, INEFF	, RBLO(10)	, C1404800
C									, C1404900
C									, C140500C
	COMMON /INTG/	AWGRK(200)							, C140510C
	* CT1		, CT2		, CT3		, CT4		, C140520C
	* CTS		, FCON	(33)	, JCCN	(10)	, LCCN	(22)	, C140530C
	* MC	(10)	, NBI		, NBDU		, NCTC		, C140540C
	* NFER		, NFKC		, NFRC		, NLOK		, C1405500
	* NMV		, NMO		, NMJA		, NSVP		, C1405600
	* NSVG		, FCON	(11)	, SD		, SFR	(33)	, C1405700
	* SG		, SI	(55)	, SIG		, SL		, C140580C
	* SLK	(33)	, SMA	(10)	, SGK	(11)	, SQF	(11)	, C140590C
	* SCL	(11)	, SMV		, SR		, SSCN		, C1406000
	* SSIX		, SVA		, SVB		, SVD		, C1406100
	* SVI		, SVM		, SVP	(22)	, SVQ	(33)	, C1406200
	* SXM	(3,10)	, SXT		, TURQ	(97)	, SMAL		, C1406300
	* SEU		, NTQ		, SC	(33)	, SCG		, C1406400
	* NFLXB		, SFLX		, SFXM	(10)	, NMODS		, C1406500
	* SFCC		, SCC	(10)					, C140660C
C									, C1406700
C									, C1406800
	COMMON /INTGZ/								, C1406900
	* SCNDLM		, SCN	(5)	, SCRDM		, SCR	(9)	, C1407000
	* SFKDUM		, SFK	(5)	, SIXDUM		, SIX	(9)	, C1407100
	* SKDLM		, SK	(5)	, SPIDUM		, SPI	(9)	, C1407200
	* SPCDLM		, SMC	(5)					, C1407300
C									, C1407400
C									, C1407500
	COMMON /REAL/								, C1407600
	* CA	(3,10)	, CAC	(3,10)	, CLM	(10)	, CGMC	(3,11)	, C1407700
	* DCMC	(3,11)	, ETC	(3,11)	, ETM	(33)	, FLMC	(3,11)	, C1407800
	* GAM	(3,66)	, F		, HM	(3,10)	, HMC	(3,10)	, C1407900
	* HMCN	(10)	, PHI	(3,11)	, PLM	(10)	, QF	(3,33)	, C1408000
	* QFC	(3,33)	, QL	(3,22)	, QLC	(3,22)	, ROMC	(3,11)	, C1408100
	* THADW	(10)	, THAW	(10)	, THA	(33)	, THAD	(33)	, C1408200
00SSB									, C1408300
	* XIC	(3,3,10)	, XMAS	(10)	, XMB	(3,3,33)	, XMT	(3,3,10)	, C140840C
	* TLG	(33)	, FLA	(3,20)	, FLB	(3,20)	, FLC	(3,20)	, C1408500
	* FLJ	(3,3,20)	, FLJ	(3,3,20)	, CAU	(3,10)	, XID	(3,3,10)	, C140860C
	* FLIRC	(3,10)	, FLCRC	(3,10)	, FLAC	(3,20)	, FLQC	(3,20)	, C1408700

```

* FLCM (20) , ZETA (20) , FCF (3,3,40), FCK (3,40) , C1408800
* TIMEND C1408900
C C1409000
C C1409100
CCMCM /REALZ/ C1409200
* CBDLM (1,3) , CB (3,10) , CBDUM(1,3) , CBC (3,10) , C1409300
* XPCCLM(1,1,9) , XMC (3,3,10), CBN(3) C1409400
C C1409500
C C1409600
EQUIVALENCE (ETM(1),THACC(1)) ,(XMN(1,1),ANGD(1)) , C1409700
* (XMN(1,3),YMCD(1,1,1)) ,(XMN(1,6),CNF(1,1)) , C1409800
* (XMN(1,8),ETIC(1,1)) ,(XMN(1,10),ETMC(1,1)) , C1409900
* (FLB(1,1),FLG(1,1)) ,(FLE(1,1,1),FLD(1,1,1)) , C1410000
* (FLH(1,1,1),FLJ(1,1,1)) , C1410100
* (FG1,LINIT(1)) ,(CA(1,1),RINIT(1)) , C1410200
* (CBDUM(1,1),RZINIT(1)) ,(AWORK(1),IINIT(1)) , C1410300
* (SCNDLM,IZINIT(1)) C1410400
DIMENSION X1(3),X2(3) C1410500
DIMENSION TEM1(33),TEM2(33) C1410600
REAL*8 EFGMC(33), ECCMC(33), ERGMC(33), EDGMC(33) C1410700
INTEGER SET(10) C1410800
LOGICAL CTAIN C1410900
C C1411000
C USE SINGLE DIMENSION ARRAYS WHEN ADVANTAGEOUS TO INCREASE C1411100
C COMPLETION SPEED C1411200
EQUIVALENCE(FCMC(1,1),TEM1(1)) , (CCMC(1,1),TEM2(1)) C1411300
EQUIVALENCE (EFGMC(1),FCMC(1,1)) , (ECGMC(1),COMC(1,1)) , C1411400
* (ERGMC(1),FCMC(1,1)) , (EDGMC(1),DOMC(1,1)) C1411500
C C1411600
EQUIVALENCE (LRATE,LEQU) C1411700
C C1411800
C C1411900
IF (LEQU)PRINT 200 C1412000
IF (.NOT.LEQU) GO TO 1000 C1412100
IF (INERF) GO TO 33 C1412200
PRINT 201 C1412300
GO TO 34 C1412400
33 PRINT 202 C1412500
34 CONTINUE C1412600
C EVALUTE RELATIVE VELOCITY VECTORS C1412700
1000 CONTINUE C1412800
M = 1 C1412900
DO 1 K= 1,NB1 C1413000
IF(K.EQ.1) GO TO 2 C1413100
M = M+3-PCON(K-1) C1413200
2 CONTINUE C1413300
IK1 = 3*K-3 C1413400
DO 31 I=1,3 C1413500
IK = IK1+I C1413600
31 ERGMC(IK) = 0.00 C1413700
IF (LEQU)PRINT 221, K C1413800
IK = IK1 + 1 C1413900
ITERM = M+2-PCON(K) C1414000
IF (M.GT.ITERM)GO TO 1 C1414100
DO 3 MM=M,ITERM
CALL SCLV(THAD(MM),QFC(1,MM),X1) C1414200
CALL VECADD(X1,ERGMC(IK),ERGMC(IK)) C1414300
IF (LEQU)PRINT ,205, K,K,MM,MM C1414400
3 CONTINUE C1414500
1 CONTINUE C1414600
C1414700

```

IF(.NOT.LEQU) GO TO 1001	C1414800
FRINT 214	C1414900
DO 25 K=1,NB1	C1415000
25 FRINT 215, K,(R0MC(J,K),J=1,3)	C1415100
FRINT 214	C1415200
1001 CONTINUE	C1415300
C	C1415400
C EVALUATE ANGULAR VELOCITY BODY K TO INERTIAL AND COMPUTING FRAME	C1415500
C	C1415600
C TAKE CARE OF RIGID BODY ANGULAR RATES	C1415700
CALL UNPAC(SET,NSET,SR)	C1415800
CC 7 I=1,3	C1415900
7 F0MC(1,I) = R0MC(I,I)	C1416000
IF(LEQU)PRINT 219	C1416100
NSETM1 = NSET - 1	C1416200
IF(NSETM1.EQ.0) GO TO 5042	C1416300
DO 8 JJ = 1,NSETM1	C1416400
J = NSETM1 - (JJ-1)	C1416500
K = SET(J)	C1416600
CALL VECADD(F0MC(1,JCCN(K)),R0MC(1,K),F0MC(1,K))	C1416700
IF(LEQU)JK = JCCN(K)	C1416800
IF(LEQU)PRINT 209, K,JK,K	C1416900
8 CONTINUE	C1417000
5042 CONTINUE	C1417100
IF(INERF) GO TO 11	C1417200
CC 9 I=1,3	C1417300
9 ECCMC(1) = G.DC	C1417400
IF(LEQU)PRINT 214	C1417500
IF(LEQU)PRINT 222	C1417600
NSETM1 = NSET - 1	C1417700
IF(NSETM1.EQ.0) GO TO 5043	C1417800
DO 10 JJ = 1, NSETM1	C1417900
J = NSETM1 - (JJ - 1)	C1418000
K = SET(J)	C1418100
CALL VECADD(C0MC(1,JCCN(K)),F0MC(1,K),C0MC(1,K))	C1418200
IF(LEQU)JK = JCCN(K)	C1418300
IF(LEQU)PRINT 210, K,JK,K	C1418400
10 CONTINUE	C1418500
5043 CONTINUE	C1418600
IF(LEQU)PRINT 214	C1418700
GO TO 12	C1418800
C	C1418900
C C0MC = F0MC VIA EQUIVALENCE FOR INERTIAL COMPUTING FRAME	C1419000
11 ITERM = 3 * NB0D	C1419100
CC 13 I = 1, ITERM	C1419200
13 TEM2(I) = TEM1(I)	C1419300
IF(.NOT.LEQU) GO TO 1002	C1419400
FRINT 214	C1419500
DO 26 K=1,NB0D	C1419600
26 PRINT 220, K,K	C1419700
FRINT 214	C1419800
1002 CONTINUE	C1419900
12 CONTINUE	C1420000
C	C1420100
2000 IF TAKE CARE OF LINEAR OSCILLATORS AND CENTER OF MASS MOTION	C1420200
CALL UNPAC(SET,NSET,SL)	C1420300
IF(NSET.EQ.0) GO TO 5000	C1420400
DO 23 J=1,NSET	C1420500
K = SET(J)	C1420600
IF(LEQU) PRINT 223, K,JCCN(K)	C1420700

IF(LEQU) PRINT 224, K, JCCN(K)	C1420800
DO 6 I=1,3	C1420900
FOMC(I,K) = FOMC(I,JCCN(K))	C1421000
COMC(I,K) = COMC(I,JCCN(K))	C1421100
23 CONTINUE	C1421200
5000 CONTINUE	C1421300
FOMC(I,NE1) = FOMC(I,NE1)	C1421400
COMC(I,NE1) = COMC(I,NE1)	C1421500
IF(LEQU) PRINT 208, NB1, NE1, NB1	C1421600
C	C1421700
IF(.NOT.LEQU) GO TO 1003	C1421800
DO 28 K=1,NB1	C1421900
28 PRINT 217, K, (COMC(J,K), J=1,3), K, (FCMC(J,K), J=1,3)	C1422000
PRINT 214	C1422100
1003 CONTINUE	C1422200
C	C1422300
EVALUTE COMPONENTS OF ACCELERATION ASSOCIATED WITH COORDINATE	C1422400
FRAME ROTATION	C1422500
C	C1422600
IF(CT4.NE.1) GO TO 15	C1422700
DO 15 K=1,NB1	C1422800
IK = 3*K-3	C1422900
DO 14 I=1,3	C1423000
IK = IK+1	C1423100
14 EDCMC(IK) = 0.00	C1423200
IF(LEQU) PRINT 212, K	C1423300
15 CONTINUE	C1423400
C	C1423500
CYCLE THROUGH ALL BODIES	C1423600
M = 1	C1423700
DO 16 K=1,NBCD	C1423800
IK1 = 3*K-3	C1423900
IF(LEQU) PRINT 203	C1424000
IF(K.EQ.1) GO TO 17	C1424100
M = M+3-PCDN(K-1)	C1424200
17 IF(.NOT.FBLO(K)) GO TO 16	C1424300
DO 21 I=1,3	C1424400
IK = IK1+1	C1424500
21 EDCMC(IK) = 0.00	C1424600
IF(LEQU) PRINT 212, K	C1424700
C TAKE CARE OF COMPONENT DUE TO ROTATION OF FRAME K RELATIVE TO INER	C1424800
CYCLE THROUGH FREE COORDINATE VECTORS AT HINGE POINT K-1	C1424900
IK = IK1 + 1	C1425000
MMM = M + 2 - PCDN(K)	C1425100
IF(M.GT.MMM) GO TO 5044	C1425200
DO 18 MM= M,MMM	C1425300
MF = FCCN(MM)	C1425400
IF(MF.LT.0) GO TO 18	C1425500
IF(MF.EQ.0) GO TO 16	C1425600
CALL VECRUS (FCMC(1,MF),GFC(1,MM),X1)	C1425700
CALL SCLV(THAD(MM),X1,X1)	C1425800
CALL VECADD(EDCMC(IK),X1,EDCMC(IK))	C1425900
IF(LEQU) PRINT 213, K,K,MM,MF,MM	C1426000
GO TO 16	C1426100
ME = FCCN(MM-1)	C1426200
M1 = MM-1	C1426300
CALL SCLV(THAD(M1),GFC(1,M1),X1)	C1426400
IF(MF.EQ.0) GO TO 4	C1426500
CALL VECADD(FOMC(1,MF),X1,X1)	C1426600
4 CONTINUE	C1426700

CALL VECROS (X1,QFC(1,MM),X2)	C1426800
CALL SCLV(THAD(MM),X2,X2)	C1426900
CALL VECADD(EDGMC(1K),X2,EDCMC(1K))	C1427000
IF(MF.EQ.0) GO TO 5	C1427100
IF(LEQU)PRINT 216, K,K,MM,MM,M1,M1,MM	C1427200
GO TO 16	C1427300
5 CONTINUE	C1427400
IF(LEQU)PRINT 204, K,K,MM,M1,M1,MM	C1427500
5044 CONTINUE	C1427600
16 CONTINUE	C1427700
IF(.NOT.LEQU) RETURN	C1427800
PRINT 214	C1427900
DO 25 M=1,NB1	C1428000
29 PRINT 227, M,(DMC(J,M),J=1,3)	C1428100
C	C1428200
C	C1428300
200 FORMAT (' SUBROUTINE RATE ENTERED ')	C1428400
201 FORMAT (/, ' COMPUTING FRAME IS BODY 1 ',/)	C1428500
202 FORMAT (/, ' COMPUTING FRAME IS INERTIALLY FIXED ',/)	C1428600
203 FORMAT (' ')	C1428700
204 FORMAT (' DCMC(',12,') = DCMC(',12,') + THAD(',12,')*THAD(',12,')	C1428800
*QFC(',12,') X QFC(',12,') ')	C1428900
205 FORMAT (' RCMC(',12,') = RCMC(',12,') + THAD(',12,')*QFC(',12,')	C1429000
*')	C1429100
208 FORMAT (' FCMC(',12,') = CCMC(',12,') = RCMC(',12,') ')	C1429200
209 FORMAT (' FCMC(',12,') = FCMC(',12,') + RCMC(',12,') ')	C1429300
210 FORMAT (' CCMC(',12,') = CCMC(',12,') + RCMC(',12,') ')	C1429400
212 FORMAT (' DCMC(',12,') = 0 ')	C1429500
213 FORMAT (' DCMC(',12,') = DCMC(',12,') + THAD(',12,')*(FCMC(',12,')	C1429600
*') X QFC(',12,')')	C1429700
214 FORMAT(2(/))	C1429800
215 FORMAT (' RCMC(',12,') = ',3D17.8)	C1429900
216 FORMAT (' DCMC(',12,') = DCMC(',12,') + THAD(',12,')*((FCMC(',12,')	C1430000
*') + THAD(',12,')*QFC(',12,') X QFC(',12,')')	C1430100
217 FORMAT (' CCMC(',12,') = ',3D17.8, ' FCMC(',12,') = ',3D17.8)	C1430200
219 FORMAT (' FCMC( 1) = RCMC( 1) ')	C1430300
220 FORMAT (' CCMC(',12,') = FCMC(',12,') ')	C1430400
221 FORMAT (' RCMC(',12,') = 0 ')	C1430500
222 FORMAT (' CCMC( 1) = 0 ')	C1430600
223 FORMAT (' FCMC(',12,') = FCMC(',12,') ')	C1430700
224 FORMAT (' CCMC(',12,') = CCMC(',12,') ')	C1430800
227 FORMAT (' DCMC(',12,') = ',3D17.8)	C1430900
RETURN	C1431000
END	C1431100
C	C1500000
1 SUERCUTINE XDY	C1500100
C COMPUTES VALUES OF VECTORS AND DYADS USED TO DEFINE SYSTEM INERT	C1500200
C TENSOR MATRIX	C1500300
C	C1500400
C LET:	C1500500
C GAM(I,KL) = COMPONENTS OF VECTOR FROM HINGE POINT K-1 TO THE	C1500600
C CENTER OF MASS OF EDDY LAMBA WHERE	C1500700
C KL = KTO(NBDD+1,K-1,LAMBA)	C1500800
C XDIC(I,J,KI) = COMPONENTS OF TENSOR OF RANK TWO IN ROW K, CCL I	C1500900
C OF THE SYSTEM MATRIX OF INERTIA TENSORS	C1501000

```

C                                     WHERE
C                                     KI = KI(NBUD+1,K,1)
C
C NOTE:
C THE SYSTEM MATRIX OF INERTIA TENSORS IS SYMMETRIC THUS ONLY
C LOWER TRIANGULAR PORTION COMPUTED
C IMPLICIT REAL*8(A-H,O-Z,F)
C
C LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLD, LEQU, LINIT(1)
C LOGICAL LFLNGE, LTRNSI, LVDIV, LEQUIV, LTRAN,
* LTRANV, LRATE, LXDY, LETA, LTORQU,
* LQFDDT, LDCT, LANGLE, LSETUP, LSIMG
C
C INTEGER
C * AWORK, CT1, CT2, CT3, CT4, CT5, FCCN, PCCN,
* SCNDUM, SCN, SCR, SCDUM, SCR, SFDUM, SFK, SFR, SG,
* SI, SIG, SIXDUM, SIX, SKDUM, SK, SL, SLK,
* SMA, SMCDUM, SMC, SMV, SOK, SPIDUM, SPI, SQF,
* SGL, SR, SSCN, SSIX, SVA, SVB, SVD, SVI,
* SVM, SVP, SVQ, SXM, SXT, TORQ, SMAL, SEU,
* SC, SCG, NFLXB, SFLX, SFXM, NMGDS, SFCC, SCC,
* IINIT(1), IZINIT(1), SD
C
C REAL*8
C * ANGD (33), CNF (3,10), ETIC (3,10), ETMC (3,10),
* FLQ (3,20), FLE (3,3,20), FLH (3,3,20),
* TFAED (33), YMCD (3,2,11), RINIT (1), RZINIT(1)
C
C COMMON /LDEBUG/ LFLNGE, LTRNSI, LVDIV, LEQUIV, LTRAN,
* LTRANV, LRATE, LXDY, LETA, LTORQU,
* LQFDDT, LDCT, LANGLE, LSETUP, LSIMG
C
C COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RELO(10)
C
C COMMON /INTG/ AWORK(200),
* CT1, CT2, CT3, CT4,
* CT5, FCCN (33), JCCN (10), LCCN (22),
* MC (10), NB1, NBUD, NCTC,
* NFER, NFKC, NFRG, NLDR,
* NMV, NMG, NMOA, NSVP,
* NSVG, FCCN (11), SC, SFR (33),
* SC, SI (55), SIG, SL,
* SLK (33), SMA (10), SOK (11), SQF (11),
* SGL (11), SMV, SR, SSCN,
* SSIX, SVA, SVB, SVD,
* SVI, SVM, SVP (22), SVQ (33),
* SXM (3,10), SXT, TORQ (97), SMAL,
* SEU, NTQ, SC (33), SCG,
* NFLXB, SFLX, SFXM (10), NMGDS
* SFCC
C
C COMMON /INTGZ/
* SCNDUM, SCN (5), SCDUM, SCR (9),
* SFKCLM, SFK (5), SIXDUM, SIX (9)

```

```

* SKDLM      , SK      (9)      , SPIDUM      , SPI      (9)      , C1507100
* SMCCLM     , SMC     (9)      ,              ,          , C1507200
C1507300
C1507400
C1507500
COMMON /REAL/
* CA      (3,10) , CAC      (3,10) , CLM      (10) , CCMC      (3,11) , C1507600
* DCMC      (3,11) , ETC      (3,11) , ETM      (3,3) , FCMC      (3,11) , C1507700
* GAM      (3,66) , F          , HM      (3,10) , HMC      (3,10) , C1507800
* HMC      (10) , PHI      (3,11) , PLM      (10) , GF      (3,33) , C1507900
* QFC      (3,33) , GL      (3,22) , QLC      (3,22) , RCMC      (3,11) , C1508000
* T          , THA      (33) , THAD      (33) , C1508100
* THADW      (10) , THAW      (10) , XDIC      (3,3,66) , XI      (3,3,10) , C1508200
* XIC      (3,3,10) , XMAS      (10) , XMN      (3,3,33) , XMT      (3,3,10) , C1508300
* TLG      (33) , FLA      (3,20) , FLB      (3,20) , FLC      (3,20) , C1508400
* FLD      (3,3,20) , FLJ      (3,3,20) , CAU      (3,10) , XIU      (3,3,10) , C1508500
* FLIRC      (3,10) , FLCRC      (3,10) , FLAC      (3,20) , FLQC      (3,20) , C1508600
* FLCW      (20) , ZETA      (20) , FCF      (3,3,40) , FCK      (3,40) , C1508700
* TIMEND
C1508800
C1508900
C1509000
C1509100
COMMON /REALZ/
* CEDUM      (1,3) , CB      (3,10) , CBCDUM      (1,3) , CBC      (3,10) , C1509200
* XMCCLM      (1,1,9) , XMC      (3,3,10) , CBN      (3)
C1509300
C1509400
C1509500
EQUIVALENCE (ETM(1),THAD(1)) , (XMN(1,1),ANGD(1)) , C1509600
* (XMN(1,3),YMCD(1,1,1)) , (XMN(1,6),CNF(1,1)) , C1509700
* (XMN(1,8),ETIC(1,1)) , (XMN(1,10),ETMC(1,1)) , C1509800
* (FLB(1,1),FLC(1,1)) , (FLE(1,1,1),FLD(1,1,1)) , C1509900
* (FLH(1,1,1),FLJ(1,1,1)) , C1510000
* (FG1,LINIT(1)) , (CA(1,1),RINIT(1)) , C1510100
* (CBDUM(1,1),FZINIT(1)) , (AWRK(1),IINIT(1)) , C1510200
* (SCNDUM,IZINIT(1))
C1510300
C1510400
INTEGER ST1(10),ST2(10),ST3(10),ST4(10)
DIMENSION TEM(3),TEM1(3,3),TEM2(3,3),TEM3(3),TEM4(3,3)
EQUIVALENCE (LXDY,LEQU)
C1510500
C1510600
C1510700
C1510800
C1510900
C1511000
C1511100
C1511200
C1511300
C1511400
C1511500
C1511600
C1511700
C1511800
C1511900
C1512000
C1512100
C1512200
C1512300
C1512400
C1512500
C1512600
C1512700
C1512800
C1512900
C1513000
20 CALL UNPAC(ST3,NST3,SXT)

```



```

      IF(LEQU)PRINT 202, SXT,(ST3(I),I=1,NST3)          C1513100
      CALL UNPAC(ST4,NST4,SVD)                          C1513200
      IF(LEQU)PRINT 203, SVD,(ST4(I),I=1,NST4)         C1513300
C
C      C1513400
C      COMPLETE VECTORS FROM HINGE POINT K-1 TO BODY LAMEA CENTER OF MASS C1513500
C      STORE UPPER TRIANGULAR                          C1513600
C
C      C1513700
C      AFTER FIRST PASS RECOMPUTE ONLY THOSE GAM VECTORS WHICH ARE TIME V C1513800
C
C      C1513900
22  CONTINUE                                           C1514000
      IF(NST4.EQ.0) GO TO 5045                          C1514100
      DO 3 LP=1,NST4                                    C1514200
      L = ST4(LP)                                       C1514300
      KLL = KTC(NB1,L-1,L)                             C1514400
      DO 4 I=1,3                                       C1514500
      4  GAM(I,KLL) = CAC(I,L)                          C1514600
      IF(LEQU)PRINT 229                                C1514700
      IF(LEQU)L1 = L-1                                  C1514800
      IF(LEQU)PRINT 204, KLL,L1,L,L                   C1514900
      LL = L                                            C1515000
      5  K = JCCN(LL)                                   C1515100
      IF(K.EQ.C) GO TO 3                                C1515200
      KL = KTC(NB1,K-1,L)                              C1515300
      CALL VECADD(CBC(1,LL),GAM(1,KLL),GAM(1,KL))      C1515400
      IF(LEQU)K1 = K-1                                 C1515500
      IF(LEQU)PRINT 205, KL,K1,L,KLL,LL               C1515600
      KLL = KL                                          C1515700
      LL = K                                            C1515800
      GO TO 5                                           C1515900
      3  CONTINUE                                       C1516000
5045 CONTINUE                                         C1516100
      IF(.NOT. LEQU) GO TO 1000                        C1516200
      PRINT 230                                         C1516300
      DO 23 K=1,NBOD                                    C1516400
      PRINT 225                                         C1516500
      DO 23 L=K,NBOD                                    C1516600
      KL = KTC(NB1,K-1,L)                              C1516700
      K1 = K-1                                          C1516800
      23 PRINT 231, K1,L,KL,(GAM(I,KL),I=1,3)         C1516900
      PRINT 230                                         C1517000
1000 CONTINUE                                         C1517100
C
C      C1517200
C      COMPLETE ELEMENTS OF SYSTEM INERTIA TENSOR MATRIX C1517300
C
C      C1517400
C      AFTER FIRST PASS RECOMPUTE ONLY TIME VARYING ELEMENTS OF XDIC C1517500
C      1) IF K CONTAINED IN SXT, ELEMENTS OF ROW K FROM COLUMN 1 TO K C1517600
C      ARE EITHER ZERO OR ASSUMED TIME VARYING, INTERNAL LOGIC, SUM C1517700
C      UN SET SPI(K-1), SKIPS ZER. ELEMENTS           C1517800
C      2) IF K CONTAINED IN SXT, ELEMENT IN ROW NBOD+1 COLUMN K IS C1517900
C      ASSUMED TIME VARYING                           C1518000
C      3) IF K NOT CONTAINED IN SXT ABOVE ELEMENTS ASSUMED CONSTANT AND C1518100
C      EQUAL TO VALUE DEFINED ON FIRST PASS THROUGH   C1518200
C      4) IF HINGE K-1 IS A RIGID HINGE INERTIA TENSORS IN ROW K NOT, C1518300
C      NEEDED FOR DYNAMICS HOWEVER THEY ARE NECESSARY FOR CONSTRAINT C1518400
C      TORQUES, NO LOGIC TO AVOID THIS SINCE IT WILL BE HARDLY EVER C1518500
C
C      C1518600
C      SXT STORED IN ST3 ARRAY                          C1518600
      IF(NST3.EQ.0) GO TO 5046                          C1518700
      DO 6 KP=1,NST3                                    C1518800
      IF(LEQU)PRINT 230                                C1518900
      K = ST3(KP)                                       C1519000

```

```

IN = KTC(NB0D,0,K-1)
C SET SI(IN) DEFINES BODY LABELS ON PATH FROM HINGE POINT ZERO TO C1519100
C CENTER OF MASS OF BODY K C1519200
CALL UNPAC(ST1,NST1,SI(IN)) C1519300
IF(RELU(K)) GO TO 7 C1519400
C C1519500
C C1519600
C BODY K IS A LINEAR OSCILLATOR C1519700
IF(LEQU)PRINT 206, K C1519800
IF(LEQU)PRINT 232, K C1519900
IF(LEQU)PRINT 207,K,(ST1(JJ),JJ=1,NST1) C1520000
IF(LEQU)PRINT 229 C1520100
C FILL UP ROW K TO DIAGCNAL C1520200
IF(NST1.EQ.0) GO TO 6 C1520300
DO 8 II=1,NST1 C1520400
I = ST1(II) C1520500
KI = KTI(NB1,K,I) C1520600
IF(I.EQ.K) GO TO 5 C1520700
IK = KTC(NB1,I-1,K) C1520800
CALL SCLV(XMAS(K),GAM(1,IK),TEM) C1520900
CALL DYCF(TEM,XDIC(1,1,KI)) C1521000
IF(LEQU)II = I-1 C1521100
IF(LEQU)PRINT 208, KI,K,I,K,IK,K,II,K C1521200
GO TO 8 C1521300
5 IF(C14.NE.1) GO TO 2 C1521400
NK = KTI(NB1,NB1,K) C1521500
DO 10 M=1,3 C1521600
XDIC(M,M,KI) = XMAS(K) C1521700
10 XDIC(M,M,NK) = XMAS(K) C1521800
IF(LEQU)PRINT 210, KI,K,I,K C1521900
IF(LEQU)PRINT 210, NK,NB1,K,K C1522000
8 CONTINUE C1522100
GO TO 6 C1522200
C C1522300
7 CONTINUE C1522400
C BODY K IS A RIGID BODY C1522500
IF(LEQU)PRINT 211, K C1522600
IF(LEQU)PRINT 232, K C1522700
IF(LEQU)PRINT 229 C1522800
IF(LEQU)PRINT 207,K,(ST1(JJ),JJ=1,NST1) C1522900
DO 11 I=1,3 C1523000
TEM3(I) = 0 C1523100
DO 11 J=1,3 C1523200
11 TEM1(I,J) = 0 C1523300
C C1523400
C SPI(K-1) = SET OF BODIES IN NEST K-1 CONTRIBUTING TO PSUECC INER C1523500
C TENSORS OF NEST K-1 C1523600
CALL UNPAC(ST2,NST2,SPI(K-1)) C1523700
IF(LEQU)PRINT 212, K,K,(ST2(M),M=1,NST2) C1523800
IF(LEQU)PRINT 229 C1523900
IF(LEQU)PRINT 217 C1524000
IF(NST2.EQ.0) GO TO 5047 C1524100
DO 12 LL=1,NST2 C1524200
L = ST2(LL) C1524300
KL = KTO(NB1,K-1,L) C1524400
CALL SCLV(XMAS(L),GAM(1,KL),TEM) C1524500
CALL VECADD(TEM3,TEM,TEM3) C1524600
IF(LEQU)K1 = K-1 C1524700
IF(LEQU)PRINT 213, L,KL,L,K1,L C1524800
IF(.NOT.FGLQ(L)) GO TO 12 C1524900
CALL DYADD(TEM1,XIC(1,1,L),TEM1) C1525000

```

	IF(LEQU)PRINT 215, L	C1525100
12	CONTINUE	C1525200
5047	CONTINUE	C1525300
	NK = KT1(NB1,NB1,K)	C1525400
	CALL CYCF(TEM3,XDIC(1,1,NK))	C1525500
	IF(LEQU)PRINT 218, NK,NB1,K	C1525600
C	FILL CUT ROW K OVER TC DIAGONAL	C1525700
	IF(NST1.EQ.0) GO TO 6	C1525800
	DO 13 I1=1,NST1	C1525900
	I = ST1(I1)	C1526000
	K1 = KT1(NB1,K,1)	C1526100
	DO 14 M=1,3	C1526200
	DO 14 N=1,3	C1526300
14	TEM2(M,N) = 0	C1526400
	IF(LEQU)PRINT 220	C1526500
	IF(NST2.EQ.0) GO TO 5048	C1526600
	DO 15 LL=1,NST2	C1526700
	L = ST2(LL)	C1526800
	KL = KTC(NB1,K-1,L)	C1526900
	IL = KTC(NB1,I-1,L)	C1527000
	CALL SUECP(GAM(1,KL),GAM(1,IL),XMAS(L),TEM4)	C1527100
	CALL CYADD(TEM2,TEM4,TEM2)	C1527200
	IF(LEQU)KI = K-1	C1527300
	IF(LEQU)I1 = I-1	C1527400
	IF(LEQU)PRINT 221, L,K1,L,I1,L,I1,L,K1,L,KL,IL,L	C1527500
	IF(LEQU)PRINT 222	C1527600
15	CONTINUE	C1527700
5048	CONTINUE	C1527800
	CALL CYADD(TEM1,TEM2,XCIC(1,1,KI))	C1527900
	IF(LEQU)PRINT 224, KI,K,1	C1528000
13	CONTINUE	C1528100
6	CONTINUE	C1528200
5046	CONTINUE	C1528300
	IF(CT4.NE.1) GO TO 17	C1528400
	NN = KT1(NB1,NB1,NB1)	C1528500
	DO 16 K=1,NB0D	C1528600
	DO 16 I=1,3	C1528700
16	XDIC(1,I,NN) = XDIC(1,I,NN) + XMAS(K)	C1528800
	IF(LEQU)PRINT 226, NN,NB1,NB1	C1528900
17	CONTINUE	C1529000
	IF(.NOT. LEQU) GO TO 1001	C1529100
	PRINT 230	C1529200
	DO 24 K=1,NB1	C1529300
	DO 24 I=K,NB1	C1529400
	IK = KT1(NB1,I,K)	C1529500
	PRINT 229	C1529600
	PRINT 233, (XDIC(1,L,IK),L=1,3)	C1529700
	PRINT 234, I,K,IK,(XDIC(2,L,IK),L=1,3)	C1529800
	PRINT 233, (XDIC(3,L,IK),L=1,3)	C1529900
24	CONTINUE	C1530000
1001	CONTINUE	C1530100
C		C1530200
202	FORMAT (' SXT = ',Z8,' ELEMENTS OF SXT = ',1015 )	C1530300
203	FORMAT (' SVD = ',Z8,' ELEMENTS OF SVD = ',1015 )	C1530400
204	FORMAT (' GAM(',I2,') = GAM(',I2,',' ',I2,') = CAC(',I2,') ')	C1530500
205	FORMAT (' GAM(',I2,') = GAM(',I2,',' ',I2,') = GAM(',I2,') + CBC(',I2,')	C1530600
	*2,') ')	C1530700
206	FORMAT (' BODY ',I2,' IS A LINER OSCILLATOR ')	C1530800
207	FORMAT (' NON-ZERO COLUMNS OF ROW ',I2,' OVER TO DIAGONAL ARE ',10	C1530900
	*15,/) )	C1531000

```

20E FORMAT (' XDIC(',I2,') = XDIC(',I2,',',I2,') = XMAS(',I2,')*DYOP(GC1E3110C
      *AM(',I2,') = XMAS(',I2,')*CYJP(GAM(',I2,',',I2,') ) ) C1E31200
210 FGMAT (' XDIC(',I2,') = XDIC(',I2,',',I2,') = XMAS(',I2,')*1 ') C1E31300
211 FGMAT (' BCDY ',I2,' IS A RIGID BODY ') C1E31400
212 FORMAT (' TO GET INERTIA TENSORS IN ROW ',I2,' COLUMNS 1 THROUGH C1E31500
      ',I2,' SLM OVER BCDIES ',I2,') C1E31600
213 FORMAT (EX,' TEM3 = TEM3 + XMAS(',I2,')*GAM(',I2,') = TEM3 + XMAS(C1E31700
      ',I2,')*GAM(',I2,',',I2,') ') C1E31800
215 FORMAT (EX,' TEM1 = TEM1 + XIC(',I2,') ') C1E31900
217 FORMAT (EX,' TEM1 = 0 , TEM3 = 0 ') C1E32000
218 FORMAT (' XDIC(',I2,') = XDIC(',I2,',',I2,') = SKEW(TEM3) ') C1E32100
220 FORMAT (EX,' TEM2 = 0 ') C1E32200
221 FORMAT (EX,' TEM4 = XMAS(',I2,')*( GAM(',I2,',',I2,').GAM(',I2,',',I2,')C1E32300
      *,I2,')*1 - GAM(',I2,',',I2,')GAM(',I2,',',I2,') ) = SUEOP(GAM(',I2,C1E32400
      ',',GAM(',I2,').XMAS(',I2,') ) ) C1E32500
222 FORMAT (EX,' TEM2 = TEM2 + TEM4 ') C1E32600
224 FORMAT (' XDIC(',I2,') = XDIC(',I2,',',I2,') = TEM1 + TEM2 ') C1E32700
226 FORMAT (' XDIC(',I2,') = XDIC(',I2,',',I2,') = (TOTAL MASS)*1 ') C1E32800
228 FORMAT ('1 SUBROUTINE XCY ENTERED ') C1E32900
229 FORMAT (' ') C1E33000
230 FORMAT (3(/)) C1E33100
231 FORMAT (' GAM(',I2,',',I2,') = GAM(',I2,') = ',3D17.8) C1E33200
232 FORMAT (/,' CCMPLTE ELEMENTS IN ROW ',I2,' OF INERTIA MATR C1E33300
      *X ',/) C1E33400
233 FGMAT (27X,3D17.8) C1E33500
234 FORMAT (' XDIC(',I2,',',I2,') = XDIC(',I2,') = ',3D17.8) C1E33600
      RETURN C1E33700
      END C1E33800

```

```

C SUBROUTINE ETA C1600000
C USE TO COMPUTE GYROSCOPIC CROSS COUPLING TERMS DUE TO C1600100
C 1) INERTIA CROSS COUPLING C1600200
C 2) CENTRIPITAL CROSS COUPLING C1600300
C 3) CORIOLIS CROSS COUPLING C1600400
C LET: C1600500
C ETC(I,K) = COMPONENTS OF GYROSCOPIC CROSS COUPLING TORQUE C1600600
C ON NEXT K-1 C1600700
C C1600800
C C1600900
C IMPLICIT REAL*8(A-F,O-Z,*) C1601000
C LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLC, LEQU, LINIT(1) C1601100
C LOGICAL LRUNGE, LTRNSI, LVDIV, LEQUIV, LTRAN, C1601200
* LTRANV, LRATE, LXDY, LETA, LTOFOU, C1601300
* LGFDOT, LDCT, LANGLE, LSETUP, LSIMO C1601400
C C1601500
C C1601600
C C1601700
C C1601800
C C1601900
C C1602000
C C1602100
C C1602200
C INTEGER C1602300
* ANQK, CT1, CT2, CT3, CT4, CT5, FCON, PCON, C1602400
* SCNCUM, SCN, SCRDM, SCR, SFKDUM, SFK, SFR, SG, C1602500
* SI, SIG, SIXDUM, SIX, SKDUM, SK, SL, SLK, C1602600

```

```

* SMA , SMCDUM, SMC , SMV , SSK , SPIDUM, SPI , SQF , C1602700
* SQL , SR , SSCN , SSIK , SVA , SVB , SVD , SVI , C1602800
* SVM , SVP , SVQ , SXM , SXT , TORQ , SMAL , SEU , C1602900
* SC , SCG , NFLXB , SFLX , SFXM , NMUDS , SFCC , SCC , C1603000
* IINIT(1) , IZINIT(1) , SD C1603100
C C1603200
C C1603300
C C1603400
REAL*8
* ANGE (33) , CNF (3,10) , ETIC (3,10) , ETMC (3,10) , C1603500
* FLO (3,20) , FLE (3,3,20) , FLH (3,3,20) , C1603600
* THAD (33) , YMCD (3,2,11) , RINIT (1) , RZINIT(1) C1603700
C C1603800
C C1603900
COMMON /LDEBUG/ LRUNGE , LTRNSI , LVDIV , LEQUIV , LTRAN , C1604000
* LTRANV , LRATE , LXDY , LETA , LTOFQU , C1604100
* LQFDDT , LDCT , LANGLE , LSETUP , LSIMQ C1604200
C C1604300
C C1604400
COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RELO(10) C1604500
C C1604600
C C1604700
COMMON /INTG/ AWORK(200), C1604800
* CT1 , CT2 , CT3 , CT4 , C1604900
* CT5 , FCON (33) , JCCN (10) , LCCN (22) , C1605000
* MC (10) , NE1 , NBCD , NCTC , C1605100
* NFER , NFKC , NFRC , NLOR , C1605200
* NMV , NMO , NMOA , NSVP , C1605300
* NSVG , FCCN (11) , SD , SFR (33) , C1605400
* SG , SI (55) , SIG , SL , C1605500
* SLK (33) , SMA (10) , SSK (11) , SQF (11) , C1605600
* SGL (11) , SMV , SR , SSCN , C1605700
* SSIK , SVA , SVB , SVD , C1605800
* SVI , SVM , SVP (22) , SVQ (33) , C1605900
* SXM (3,10) , SXT , TORQ (97) , SMAL , C1606000
* SEU , NTQ , SC (33) , SCG , C1606100
* NFLXB , SFLX , SFXM (10) , NMUDS , C1606200
* SFCC , SCC (10) C1606300
C C1606400
C C1606500
C C1606600
COMMON /INTG2/
* SCNDUM , SCN (5) , SCRDUM , SCR (9) , C1606700
* SFKDUM , SFK (5) , SIXDUM , SIX (9) , C1606800
* SKDUM , SK (5) , SPIDUM , SPI (9) , C1606900
* SMCDUM , SMC (5) C1607000
C C1607100
COMMON /REAL/ C1607200
* CA (3,10) , CAC (3,10) , CLM (10) , CCMC (3,11) , C1607300
* DCMC (3,11) , ETC (3,11) , ETM (33) , FUMC (3,11) , C1607400
* GAM (3,66) , H , HM (3,10) , HMC (3,10) , C1607500
* HMCN (10) , PHI (3,11) , PLM (10) , QF (3,33) , C1607600
* QFC (3,33) , GL (3,22) , QLC (3,22) , QMCM (3,11) , C1607700
* T , THA (33) , THAD (33) , C1607800
* THADW (10) , THAW (10) , XDIC (3,3,66) , XI (3,3,10) , C1607900
* XIC (3,3,10) , XMAS (10) , XMN (33,33) , XMT (3,3,10) , C1608000
* TLG (33) , FLA (3,20) , FLBA (3,20) , FLC (3,20) , C1608100
* FLD (3,3,20) , FLJ (3,3,20) , CAO (3,10) , XID (3,3,10) , C1608200
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLQC (3,20) , C1608300
* FLCM (20) , ZETA (20) , FCF (3,3,40) , FCK (3,40) , C1608400
* TIMEND C1608500
C C1608600

```

```

C
COMMON /REALZ/
* CEDLM (1,3) , CE (3,10) , CBDDUM(1,3) , CbC (3,10) ,
* XPCDUM(1,1,9) , XMC (3,3,10), CbN(3)
C1E0E700
C1E08800
C1E08900
C1E09000
C1E09100
C1E09200
C
EQUIVALENCE (ETM(1),THACC(1)) ,(XMN(1,1),ANGD(1)) ,
* (XMN(1,3),YMCD(1,1,1)) ,(XMN(1,6),CNF(1,1)) ,
* (XMN(1,8),ETIC(1,1)) ,(XMN(1,10),ETMC(1,1)) ,
* (FLB(1,1),FLO(1,1)) ,(FLE(1,1,1),FLD(1,1,1)),
* (FLH(1,1,1),FLJ(1,1,1)) ,
* (FG1,LINIT(1)) ,(CA(1,1),RINIT(1)) ,
* (CBDUM(1,1),RZINIT(1)) ,(AWORK(1),IINIT(1)) ,
* (SCNDUM,IZINIT(1))
C1E09300
C1E09400
C1E09500
C1E09600
C1E09700
C1E09800
C1E09900
C1E10000
C1E10100
C
INTEGER ST1(10),ST2(10),ST3(10),ST4(10),SET(10),SFXMN
DIMENSION TEM(3),TEM1(3),TEM2(3),TEM3(3),TEM4(3),TEM5(3),TEM6(3)
DIMENSION TEM7(3,3),TEM8(3,3),TEM9(3),TEM10(3),FLEC(3,3)
EQUIVALENCE (LETA,LEQU)
C1E10200
C1E10300
C1E10400
C1E10500
C
IF(LEQU)PRINT 240
DO 1 I=1,3
TEM3(I) = 0
TEM6(I) = 0
1 ETC(I,NB1) = 0
C1E10600
C1E10700
C1E10800
C1E10900
C1E11000
C1E11100
C1E11200
C
USE SETS SSIX AND SSCN TO REDUCE REDUNDANCY IN CROSS COUPLING CCMP
C1E11300
C
COMPLETE INERTIAL CROSS COUPLING CONTRIBUTIONS FOR EACH BODY
C1E11400
IF(LEQU)PRINT 233
CALL UNPAC(ST1,NST1,SSIX)
C1E11500
IF(NST1.EQ.0) GO TO 5049
C1E11600
DO 30 II=1,NST1
C1E11700
I = ST1(II)
C1E11800
CALL VXDYUV(FOMC(1,I),XIC(1,1,1),ETIC(1,1))
C1E11900
IF(LEQU)PRINT 239, I,1,1,1,(ETIC(J,1),J=1,3)
C1E12000
C1E12100
30 CONTINUE
C1E12200
5049 CONTINUE
C1E12300
C1E12400
C1E12500
C
COMPLETE FORCE ASSOCIATED WITH CENTRIPITAL ACCELERATION OF EACH
C1E12600
BODY IN THE SET SSCN
C1E12700
CALL UNPAC(ST2,NST2,SSCN)
C1E12800
IF(LEQU)PRINT 234
C1E12900
IF(NST2.EQ.0) GO TO 5050
C1E13000
DO 31 III=1,NST2
C1E13100
II=NST2-(III-1)
C1E13200
I = ST2(II)
C1E13300
IF(I.EQ.1) GO TO 31
C1E13400
CALL TRIPVP(FOMC(1,JCON(I)),CBC(1,1),CNF(1,1))
C1E13500
IF(JCON(I).EQ.1) GO TO 32
C1E13600
CALL VECADD(CNF(1,JCON(I)),CNF(1,1),CNF(1,1))
C1E13700
IF(LEQU)PRINT 235, I,JCON(I),JCON(I),JCON(I),I
C1E13800
GO TO 31
C1E13900
32 CONTINUE
C1E14000
IF(LEQU)PRINT 236, I,JCON(I),JCON(I),I
C1E14100
31 CONTINUE
C1E14200
5050 CONTINUE
C1E14300
IF(LEQU)PRINT 229
C1E14400
IF(NST2.EQ.0) GO TO 5051
C1E14500
DO 33 III=1,NST2
C1E14600

```

II=NST2-(III-1)	C1614700
I = ST2(II)	C1614800
IF(I.EQ.1) GO TO 33	C1614900
CALL TRIPVP(FOMC(1,I),CAC(1,I),TEM)	C1615000
IF(LEQU)PRINT 237, I,I,I,I,I,I	C1615100
35 CALL VECADD(CNF(1,I),TEM,CNF(1,I))	C1615200
CALL SCLV(XMAS(1),CNF(1,I),CNF(1,I))	C1615300
33 CONTINUE	C1615400
5051 CONTINUE	C1615500
IF(.NOT. LEQU) GO TO 1000	C1615600
PRINT 229	C1615700
IF(NST2.EQ.0) GO TO 1000	C1615800
DO 36 II=1,NST2	C1615900
I = ST2(II)	C1616000
IF(I.EQ.1) GO TO 36	C1616100
PRINT 238, I,(CNF(J,I),J=1,3)	C1616200
36 CONTINUE	C1616300
1000 CONTINUE	C1616400
C	C1616500
C	C1616600
C	C1616700
C	C1616800
DEFORMATION CONTRIBUTIONS TO CROSS COUPLING	C1616800
IF(NFLXB.EQ.0) GO TO 38	C1616900
MN = 0	C1617000
CALL CNPAC(SET,NSET,SFLX)	C1617100
IF(LEQU) PRINT 252, (SET(I),I=1,NSET)	C1617200
DO 40 NN=1,NSET	C1617300
N = SET(NSET+1-NN)	C1617400
IF(LEQU) PRINT 253, N	C1617500
DO 37 I=1,3	C1617600
TEM9(I) = 0.	C1617700
DO 37 J=1,3	C1617800
37 TEM8(I,J) = 0.	C1617900
IF(LEQU) PRINT 231	C1618000
SFXM = SFXM(N)	C1618100
DO 39 M=1,SFXM	C1618200
MN = MN + 1	C1618300
CALL DYDF(FLQC(1,MN),TEM7)	C1618400
CALL TENTERN(FLE(1,1,MN),XMC(1,1,N),FLEC)	C1618500
CALL DYADD(FLEC,TEM7,TEM7)	C1618600
CALL SCLD(THAD(NFER+MN),TEM7,TEM7)	C1618700
CALL DYADD(TEM8,TEM7,TEM8)	C1618800
CALL SCLV(THAD(NFER+MN),FLAC(1,MN),TEM10)	C1618900
CALL VECADD(TEM9,TEM10,TEMS)	C1619000
IF(.NOT.LEQU) GO TO 5001	C1619100
PRINT 251, MN,N,MN,N	C1619200
NFGX = NFER+MN	C1619300
PRINT 252, NFGX,MN,MN	C1619400
PRINT 243, NFGX,MN	C1619500
5001 CONTINUE	C1619600
39 CONTINUE	C1619700
CALL DYDCTV(TEM8,FCMC(1,N),FLIRC(1,N))	C1619800
CALL SCLV(2.000,TEM9,TEMS)	C1619900
501051 CALL SCLV(XMAS(N),TEM9,TEMS)	C1620000
CALL VECROS(FCMC(1,N),TEMS,FLCRC(1,N))	C1620100
CALL VECADD(ETIC(1,N),FLIRC(1,N),ETIC(1,N))	C1620200
CALL VECADD(CNF(1,N),FLCRC(1,N),CNF(1,N))	C1620300
IF(.NOT.LEQU) GO TO 5000	C1620400
PRINT 244, N,N,(FLIRC(I,N),I=1,3)	C1620500
PRINT 246, N,N,N,(ETIC(I,N),I=1,3)	C1620600

	PRINT 245, N,N,N,(FLCRC(I,N),I=1,3)	C1620700
	PRINT 247, N,N,N,(CNF(I,N),I=1,3)	C1620800
500C	CONTINUE	C1620900
4C	CONTINUE	C1621000
3E	CONTINUE	C1621100
C		C1621200
C		C1621300
C		C1621400
C	ELIMINATE REDUNDANT COMPTATION IN MCMENTUM WHEEL CROSS	C1621500
C	CCUPLING COMPUTATION BY ETMC	C1621600
C		C1621700
	IF(NMC.EQ.0) GO TO SCE2	C1621800
	IF(LEQU)PRINT 208	C1621900
	DO 10 M=1,NMD	C1622000
	CALL VECROS (FGMC(1,MO(M)),FMC(1,M),ETMC(1,M))	C1622100
	CALL SCLV(HMGM(M),ETMC(1,M),ETMC(1,M))	C1622200
	IF(LEQU)PRINT 207, M,M,MC(M),M,(ETMC(1,M),I=1,3)	C1622300
1C	CONTINUE	C1622400
5052	CONTINUE	C1622500
C		C1622600
	DO 2 K=1,NBOD	C1622700
	DO 3 I=1,3	C1622800
3	ETC(I,K) = 0	C1622900
	K1 = K-1	C1623000
	IF(LEQU)PRINT 225, K1	C1623100
C	SET UP SUMMATION SETS FOR NEST K-1	C1623200
C	INERTIA CROSS CCUPLING	C1623300
	CALL UNPAC(ST1,NST1,SIX(K1))	C1623400
	IF(LEQU)PRINT 200, K1,SIX(K1),(ST1(I),I=1,NST1)	C1623500
C	CENTRIPITAL CROSS CCUPLING	C1623600
	CALL UNPAC(ST2,NST2,SCR(K1))	C1623700
	IF(LEQU)PRINT 201, K1,SCR(K1),(ST2(I),I=1,NST2)	C1623800
C	CERRILLIS CROSS COUPLING	C1623900
	CALL UNPAC(ST3,NST3,SCR(K1))	C1624000
	IF(LEQU)PRINT 202, K1,SCR(K1),(ST3(I),I=1,NST3)	C1624100
C	MCMENTUM WHEEL CROSS CCUPLING	C1624200
	CALL UNPAC(ST4,NST4,SMC(K1))	C1624300
	IF(,NOT, LEQU) GO TO 1001	C1624400
	PRINT 209, K1,SMC(K1),(ST4(I),I=1,NST4)	C1624500
C		C1624600
C		C1624700
	PRINT 229	C1624800
	PRINT 203, K	C1624900
	IF(K.NE.1) GO TO 5	C1625000
	PRINT 203, NB1	C1625100
	PRINT 230	C1625200
5	CONTINUE	C1625300
C	INERTIA CROSS COUPLING	C1625400
	PRINT 241	C1625500
	PRINT 226	C1625600
1001	CONTINUE	C1625700
	IF(,NOT,RBLU(K)) GO TO 4	C1625800
	IF(LEQU)PRINT 204, K,K	C1625900
C	SUM OVER ONLY THOSE EDDIES OF NEST K-1 WHICH SIGNIFICANTLY	C1626000
C	CONTRIBUTE TO INERTIAL CROSS COUPLING IN EQUATIONS OF	C1626100
C	MOTION OF NEST K-1	C1626200
	IF(NST1.EQ.0) GO TO 4	C1626300
	DO 6 LL=1,NST1	C1626400
	L = ST1(LL)	C1626500
	IF(LEQU)PRINT 205, K,K,L	C1626600



6	CALL VEC SUB(ETC(1,K),ETIC(1,L),ETC(1,K))	C1626700
C	NOTE REDUNDANT VECTOR ADDITIONS LESS COSTLY THAN LOGIC NEEDED TO	C1626800
C	AVOID THEM	C1626900
4	CONTINUE	C1627000
	IF(LEQU)PRINT 216, K,(ETC(I,K),I=1,3)	C1627100
C		C1627200
C	CENTRIFITAL CROSS COUPLING	C1627300
	IF(LEQU)PRINT 241	C1627400
	IF(LEQU)PRINT 227	C1627500
	IF(NST2.EQ.0) GO TO 7	C1627600
	DO 8 I=1,3	C1627700
	TEM4(I) = 0	C1627800
8	TEM2(I) = 0	C1627900
C	SUM OVER ONLY THOSE ECCIES OF NEST K-1 WHICH SIGNIFICANTLY	C1628000
C	CONTRIBUTE TO CENTRIFITAL CROSS COUPLING IN EQUATIONS OF	C1628100
C	MOTION OF NEST K-1	C1628200
	IF(LEQU)PRINT 206	C1628300
	IF(NST2.EQ.0) GO TO 5053	C1628400
	DO 9 LL=1,NST2	C1628500
	L = ST2(LL)	C1628600
	LC = L	C1628700
	IF(LC.EQ.1) GO TO 9	C1628800
	IF(RELO(K)) GO TO 22	C1628900
	CALL VEC SUB(TEM4,CNF(1,L),TEM4)	C1629000
	IF(LEQU)PRINT 211, L	C1629100
	GO TO 9	C1629200
22	IF(K.NE.1) GO TO 14	C1629300
	CALL VEC SUB(TEM3,CNF(1,L),TEM3)	C1629400
	IF(LEQU)PRINT 212, L	C1629500
14	KL = KTC(NB1,K-1,L)	C1629600
	CALL VEC FOS (GAM(1,KL),CNF(1,L),TEM)	C1629700
	CALL VEC SUB(TEM4,TEM,TEM4)	C1629800
	IF(LEQU)K1 = K-1	C1629900
	IF(LEQU)PRINT 213, K1,L,L	C1630000
	IF(LEQU)PRINT 229	C1630100
9	CONTINUE	C1630200
5053	CONTINUE	C1630300
	CALL VEC ADD(ETC(1,K),TEM4,ETC(1,K))	C1630400
	IF(LEQU)PRINT 214, K, K, (ETC(I,K),I=1,3)	C1630500
	IF(K.NE.1) GO TO 7	C1630600
	CALL VEC ADD(ETC(1,NB1),TEM3,ETC(1,NB1))	C1630700
	IF(LEQU)PRINT 215, NB1,NE1, (ETC(I,NB1),I=1,3)	C1630800
7	CONTINUE	C1630900
C		C1631000
C	CORRILIS CROSS COUPLING	C1631100
	IF(NST3.EQ.0) GO TO 15	C1631200
	IF(LEQU)PRINT 241	C1631300
	IF(LEQU)PRINT 228	C1631400
	DO 16 I=1,3	C1631500
16	TEM5(I) = 0	C1631600
C	SUM OVER ONLY THOSE BODIES OF NEST K-1 WHICH SIGNIFICANTLY	C1631700
C	CONTRIBUTE TO CORIOLIS CROSS COUPLING IN EQUATIONS OF	C1631800
C	MOTION OF NEST K-1	C1631900
	IF(LEQU)PRINT 217	C1632000
	IF(NST3.EQ.0) GO TO 5054	C1632100
	DO 17 LL=1,NST3	C1632200
	L = ST3(LL)	C1632300
	CALL VEC FOS (FCMC(1,L),FCMC(1,L),TEM)	C1632400
	DO 18 I=1,3	C1632500
18	TEM2(I) = 2.D0*XMAS(L)*TEM(I)	C1632600

```

      IF(LEQU)PRINT 219, L,L,L                                C1632700
      IF(RBLJ(K)) GO TO 20                                    C1632800
      DO 21 I=1,3                                             C1632900
21  TEM5(I) = -TEM2(I)                                        C1633000
      IF(LEQU)PRINT 221                                       C1633100
      GO TO 17                                                C1633200
2C  KL = KTC(NB1,K-1,L)                                       C1633300
      CALL VECROS (GAM(1,KL),TEM2,TEM)                       C1633400
      CALL VECSUB(TEM5,TEM,TEM5)                              C1633500
      IF(LEQU)K1 = K- 1                                       C1633600
      IF(LEQU)PRINT 222, K1,L                                C1633700
      IF(K.NE.1) GO TO 15                                     C1633800
      CALL VECSUB(TEM6,TEM2,TEM6)                             C1633900
      IF(LEQU)PRINT 220                                       C1634000
15  CONTINUE                                                C1634100
      IF(LEQU)PRINT 229                                       C1634200
17  CONTINUE                                                C1634300
5054 CONTINUE                                               C1634400
      CALL VECADD(ETC(1,K),TEM5,ETC(1,K))                   C1634500
      IF(LEQU)PRINT 223, K,K,(ETC(1,K),I=1,3)              C1634600
      IF(K.NE.1) GO TO 15                                     C1634700
      CALL VECADD(ETC(1,NB1),TEM6,ETC(1,NB1))              C1634800
      IF(LEQU)PRINT 224, NB1,NE1,(ETC(1,NB1),I=1,3)       C1634900
15  CONTINUE                                                C1635000
C                                                                C1635100
C                                                                C1635200
C                                                                C1635300
      MOMENTUM WHEEL CROSS COUPLING
      IF(NST4.EQ.0) GO TO 5055                                C1635400
      IF(LEQU)PRINT 241                                       C1635500
      IF(LEQU)PRINT 210                                       C1635600
C                                                                C1635700
      DO 11 MM=1,NST4                                         C1635800
      M = ST4(MM)                                             C1635900
      IF(LEQU)PRINT 218, K,K,M                                C1636000
      CALL VECSUB(ETC(1,K),ETMC(1,M),ETC(1,K))              C1636100
11  CONTINUE                                                C1636200
5055 CONTINUE                                               C1636300
      IF(LEQU)PRINT 216, K,(ETC(1,K),I=1,3)                C1636400
C                                                                C1636500
C                                                                C1636600
C                                                                C1636700
      2 CONTINUE                                             C1636800
C                                                                C1636900
200 FORMAT (5X,' SIX(1,12,1) = 1,28,1 CONTRIBUTORS TO INERTIAL CROSS C1637000
      *COUPLING ARE BODIES 1,1015)                          C1637100
201 FORMAT (5X,' SCN(1,12,1) = 1,28,1 CONTRIBUTORS TO CENTRIPITAL CRO C1637200
      *SS COUPLING ARE BODIES 1,1015)                       C1637300
202 FORMAT (5X,' SCR(1,12,1) = 1,28,1 CONTRIBUTORS TO CORIOLIS CROSS C1637400
      *COUPLING ARE BODIES 1,1015)                          C1637500
203 FORMAT (' CROSS COUPLING TORQUE ETC(1,12,1) = 0 ')     C1637600
204 FORMAT (' ETC(1,12,1) = ETC(1,12,1) - ETC(1,12,1) ') C1637700
205 FORMAT (' ETC(1,12,1) = ETC(1,12,1) - ETC(1,12,1) ') C1637800
206 FORMAT (20X,' TEM2 = 0, TEM4 = 0 ')                     C1637900
207 FORMAT (5X,' ETMC(1,12,1) = FCMC(1,12,1) * (FCMC(1,12,1) X HMC(1,12,1) C1638000
      *FCMC(1,12,1)) = 1,3D17.8)                            C1638100
208 FORMAT (//,5X,' INERTIAL CROSS COUPLING TERM FOR EACH MOMENTUM WHEEL C1638200
      *EL 1,/)                                               C1638300
209 FORMAT (5X,' SMC(1,12,1) = 1,28,1 CONTRIBUTORS TO MOMENTUM WHEEL C1638400
      *CROSS COUPLING ARE WHEELS 1,1015)                   C1638500
210 FORMAT (' MOMENTUM WHEEL CROSS COUPLING EFFECTS ')     C1638600

```

```

211 FORMAT (20X,' TEM4 = TEM4 - CNF('',12,'') ') C163E700
212 FORMAT (20X,' TEM3 = TEM3 - CNF('',12,'') ') C163B800
213 FORMAT (20X,' TEM4 = TEM4 - GAM('',12,'','',12,'') X CNF('',12,'') ') C163B900
214 FORMAT (' ETC('',12,'') = ETC('',12,'') + TEM4 = ',3D17.8) C1639000
215 FORMAT (' ETC('',12,'') = ETC('',12,'') + TEM3 = ',3D17.8) C1639100
216 FORMAT (' ETC('',12,'') = ',3D17.8) C1639200
217 FORMAT (40X,' TEM5 = C ') C1639300
218 FORMAT (' ETC('',12,'') = ETC('',12,'') - ETMC('',12,'') ') C1639400
219 FORMAT (40X,' TEM2 = 2*XMAS('',12,'') * FOMC('',12,'') X RCMC('',12,'') C1639500
*) ') C1639600
220 FORMAT (40X,' TEM6 = TEM6 - TEM2 ') C1639700
221 FORMAT (40X,' TEM5 = -TEM2 ') C1639800
222 FORMAT (40X,' TEM5 = TEM5 - GAM('',12,'','',12,'') X TEM2 ') C1639900
223 FORMAT (' ETC('',12,'') = ETC('',12,'') + TEM5 = ',3D17.8) C1640000
224 FORMAT (' ETC('',12,'') = ETC('',12,'') + TEM6 = ',3D17.8) C1640100
225 FORMAT(7(/)) C1640200
*) BODY LABELS OF THOSE BODIES WHICH SIGNIFICANTLY CONTRIB C1640300
*)RIEUTE TO GYROSCOPIC CROSS COUPLING TORQUES ON NEXT ',12,/) C1640400
226 FORMAT (' INERTIA CROSS COUPLING EFFECTS ') C1640500
227 FORMAT ( 20X,' CENTRIPITAL CROSS COUPLING EFFECTS ') C1640600
228 FORMAT ( 40X,' CCRICLIS CROSS COUPLING EFFECTS ') C1640700
229 FORMAT (' ') C1640800
230 FORMAT (' TEM3 = 0, TEM6 = 0 ') C1640900
231 FORMAT (10X,' TEM8 = 0',45X,'TEM9 = 0 ',/) C1641000
232 FORMAT (10X,' TEM8 = TEM8 + THAD('',12,'')*(FLEC('',12,'') + SKEW(FLQC1641100
*(('',12,'')) ') C1641200
233 FORMAT (//,5X,' INERTIAL CROSS COUPLING CONTRIBUTIONS OF BODIES INC1641300
*SET SSIX ',/) C1641400
234 FORMAT (//,5X,' INERTIAL FORCE ASSOCIATED WITH CENTRIPITAL ACCELER1641500
*ATION OF CENTER OF MASS OF EACH BODY IN SSCN ',/) C1641600
235 FORMAT (8X,'CNF('',12,'') = CNF('',12,'') + FOMC('',12,'') X (FCMC('',12,'') C1641700
*) X CBC('',12,'') ') C1641800
236 FORMAT (8X,'CNF('',12,'') = ',10X,' FOMC('',12,'') X (FCMC('',12,'') C1641900
*) X CBC('',12,'') ') C1642000
237 FORMAT (8X,'CNF('',12,'') = XMAS('',12,'') * (CNF('',12,'') + FOMC('',12,'') C1642100
*) X (FCMC('',12,'') X CAC('',12,'') ') C1642200
238 FORMAT (8X,'CNF('',12,'') = ',3D17.8) C1642300
239 FORMAT (8X,'ETIC('',12,'') = FOMC('',12,'') X (XIC('',12,'') + FOMC('',12,'') C1642400
*,') = ',3D17.8) C1642500
240 FORMAT ('1 SUBROUTINE ETA ENTERED ',2(/)) C1642600
241 FORMAT (3(/)) C1642700
243 FORMAT (65X,'TEM9 = TEM9 + THAD('',12,'')*FLAC('',12,'')') C1642800
244 FORMAT (/, ' FLIRC('',12,'') = TEM8.FOMC('',12,'') = ',6X,3D12.5) C1642900
246 FORMAT (' ETIC('',12,'') = ETIC('',12,'') + FLIRC('',12,'') = ',3D12.5) C1643000
245 FORMAT (/,55X,'FLCRC('',12,'') = 2*XMAS('',12,'')*FOMC('',12,'') X TEM9 C1643100
*=',3D12.5) C1643200
247 FORMAT (53X,' CNF('',12,'') = CNF('',12,'') + FLCRC('',12,'') = ',9X,3D1 C1643300
*2.5) C1643400
251 FORMAT (10X,' FLEC('',12,'') = XMC('',12,'')*FLE('',12,'')*XMC('',12,'')* C1643500
**T ') C1643600
252 FORMAT (//,5X,' ELASTIC CROSS COUPLING CONTRIBUTIONS DUE TO FLEXIB C1643700
*ILITY OF BODIES',10IS,/) C1643800
253 FORMAT (///,35X,' ELASTIC DEFORMATION EFFECTS DUE TO BODY',15,/) C1643900
RETURN C1644000
ENC C1644100

```

C										C1700000
	SUER	CUTINE	TGR	QUE	(Y,YD,NEG)					C1700100
C										C1700200
C										C1700300
	IMPLICIT	REAL	#8	(A-F,0-Z,.)						C1700400
	LOGICAL	FG1, FG2, FG3, FG4, FG5, INERF, RBLO, LEQU, LINIT(1)								C1700500
	LOGICAL	LRUNGE, LTRNSI, LVDIV, LEQUV, LTRAN,								C1700600
	*	LTRANV, LRATE, LXDY, LETA, LTORQU,								C1700700
	*	LGFOOT, LDCT, LANGLE, LSETUP, LSIMQ								C1700800
C										C1700900
C										C1701000
	INTEGER									C1701100
	* AWRK	, CT1, CT2, CT3, CT4, CT5, FCON, PCON,								C1701200
	* SCNDUM	, SCN, SCRUM, SCR, SFKDUM, SFK, SFR, SG,								C1701300
	* SI	, SIG, SIXDUM, SIX, SKDUM, SK, SL, SLK,								C1701400
	* SMA	, SMCUM, SMC, SMV, SOK, SPIDUM, SPI, SQF,								C1701500
	* SGL	, SR, SSCN, SSIX, SVA, SVB, SVD, SVI,								C1701600
	* SVM	, SVP, SVQ, SXM, SXT, TORQ, SMAL, SEU,								C1701700
	* SC	, SCG, NFLXB, SFLX, SFXM, NMGDS, SFCC, SCC,								C1701800
	* IINIT(1)	, IZINIT(1), SD								C1701900
C										C1702000
C										C1702100
	REAL	#8								C1702200
	* ANGD	(33), CNF (3,10), ETIC (3,10), ETMC (3,10),								C1702300
	* FLQ	(3,20), FLE (3,3,20), FLH (3,3,20),								C1702400
	* THADD	(33), YMCD (3,2,11), RINIT (1), RZINIT(1)								C1702500
C										C1702600
C										C1702700
	COMMON /LDEBUG/	LRUNGE, LTRNSI, LVDIV, LEQUV, LTRAN,								C1702800
	*	LTRANV, LRATE, LXDY, LETA, LTORQU,								C1702900
	*	LQFOOT, LDCT, LANGLE, LSETUP, LSIMQ								C1703000
C										C1703100
C										C1703200
	COMMON /LOGIC/	FG1, FG2, FG3, FG4, FG5, INERF, RBLO(10)								C1703300
C										C1703400
C										C1703500
	COMMON /INTG/	AWRK (200),								C1703600
	* CT1	, CT2, CT3, CT4,								C1703700
	* CT5	, FCON (33), JCCN (10), LCCN (22),								C1703800
	* MC	(10), NBI, NBOD, NCTC,								C1703900
	* NFER	, NFKC, NFKC, NLGR,								C1704000
	* NMV	, NMO, NMOA, NSVP,								C1704100
	* NSVC	, PCON (11), SD, SFR (33),								C1704200
	* SG	, SI (55), SIG, SL,								C1704300
	* SLK	(33), SMA (10), SOK (11), SQF (11),								C1704400
	* SGL	(11), SMV, SR, SSCN,								C1704500
	* SSIX	, SVA, SVB, SVD,								C1704600
	* SVI	, SVM, SVP (22), SVQ (33),								C1704700
	* SXM	(3,10), SXT, TORQ (97), SMAL,								C1704800
	* SEU	, NTQ, SC (33), SCG,								C1704900
	* NFLXB	, SFLX, SFXM (10), NMGDS,								C1705000
	* SFCC	, SCC (10)								C1705100
C										C1705200
C										C1705300
	COMMON /INTG2/									C1705400
	* SCNDUM	, SCN (9), SCRUM, SCR (9),								C1705500
	* SFKDUM	, SFK (9), SIXDUM, SIX (9),								C1705600
	* SKDUM	, SK (9), SPIDUM, SPI (9),								C1705700
	* SMCCLM	, SMC (9)								C1705800
C										C1705900

```

C
COMMON /REAL/
* CA (3,10) , CAC (3,10) , CLM (10) , CCMC (3,11) , C1706000
* DCMC (3,11) , ETC (3,11) , ETM (33) , FCMC (3,11) , C1706100
* GAM (3,66) , F , HM (3,10) , HMC (3,10) , C1706200
* HNCM (10) , PHI (3,11) , PLM (10) , GF (3,33) , C1706300
* QFC (3,33) , GL (3,22) , QLC (3,22) , RCMC (3,11) , C1706400
* T , THA (33) , THAD (33) , C1706500
* THADW (10) , THAW (10) , XDIC (3,3,66) , XI (3,3,10) , C1706600
* XIC (3,3,10) , XMAS (10) , XMN (33,33) , XMT (3,3,10) , C1706700
* TUG (23) , FLA (3,20) , FLB (3,20) , FLC (3,20) , C1706800
* FLD (3,3,20) , FLJ (3,3,20) , CAD (3,10) , XIU (3,3,10) , C1706900
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLQC (3,20) , C1707000
* FLCN (20) , ZETA (20) , FCF (3,3,40) , FCK (3,40) , C1707100
* TIMEND C1707200
C C1707300
C C1707400
C C1707500
C C1707600
COMMON /REAL Z/ C1707700
* CEDUM (1,3) , CB (3,10) , CBCDUM(1,3) , CBC (3,10) , C1707800
* XCCDUM(1,1,9) , XMC (3,3,10) , CBN(3) C1707900
C C1708000
C C1708100
COMMON /SATELL/ C1708200
* DUMMY(100) C1708300
C C1708400
C C1708500
EQUIVALENCE (ETM(1),THAD(1)) ,(XMN(1,1),ANGD(1)) , C1708600
* (XMN(1,3),YMCC(1,1,1)) ,(XMN(1,6),CNF(1,1)) , C1708700
* (XMN(1,8),ETIC(1,1)) ,(XMN(1,10),ETMC(1,1)) , C1708800
* (FLB(1,1),FLQ(1,1)) ,(FLE(1,1,1),FLD(1,1,1)) , C1708900
* (FLH(1,1,1),FLJ(1,1,1)) , C1709000
* (FGI,LINIT(1)) ,(CA(1,1),RINIT(1)) , C1709100
* (CBCDUM(1,1),RZINIT(1)) ,(AWORK(1),IINIT(1)) , C1709200
* (SCNDUM,IZINIT(1)) C1709300
C C1709400
EQUIVALENCE (LTCRGU,LEGL) C1709500
C C1709600
DIMENSION Y(NEQ),YC(NEQ) C1709700
C C1709800
C C1709900
C C1710000
C C1710100
C SYMECL LISTING OF PARAMETERS USED FROM COMMON C1710200
C C1710300
NEED = TOTAL NUMBER OF RIGID BODIES AND POINT MASSES C1710400
NB1 = NBUD + 1 C1710500
M = GIMBAL AXIS LABEL C1710600
K-1 = HINGE POINT AT WHICH GIMBAL AXIS M IS LOCATED C1710700
JCON(K) = LABEL OF BODY INBOARD OF HINGE POINT K-1 C1710800
K = LABEL OF BODY OUTBOARD OF HINGE POINT K-1 C1710900
RELC(K) = TRUE IF BODY K IS A RIGID BODY, FALSE OTHERWISE C1711000
XMAS(K) = MASS OF BODY K, (M) C1711100
XIC(I,J,K) = INERTIA TENSOR OF BODY K ABOUT ITS CENTER OF MASS C1711200
RELATIVE TO FRAME OF COMPUTATION, (M*L**2) C1711300
GFC(I,M) = COMPONENTS RELATIVE TO COMPUTING FRAME OF UNIT C1711400
VECTOR ALONG GIMBAL AXIS M C1711500
THA(M) = DISPLACEMENT ABOUT OR ALONG GIMBAL AXIS M (R OR L) C1711600
THAD(M) = RATE ABOUT OR ALONG GIMBAL AXIS M (R/T OR L/T) C1711700
PHI(I,NB1) = RESULTANT EXTERNAL FORCE ACTING ON COMPOSITE SYSTEM CM. C1711800
PHI(I,K) = RESULTANT EXTERNAL TORQUE ON BODY K-1 C1711900

```

```

C      MW = MOMENTUM WHEEL LABEL                                C1712000
C      NO(MW) = BODY IN WHICH MOMENTUM WHEEL MW IS EMBEDDED   C1712100
C      HMC(I,MW) = COMPONENTS OF UNIT VECTOR ALONG SPIN AXIS OF WHEEL MW C1712200
C                  (RELATIVE TO COMPUTING FRAME)                C1712300
C      GAM(I,KL) = COMPONENTS OF VECTOR FROM HINGE POINT K-1 TO CENTER OF C1712400
C                  MASS OF BODY L, WHERE KL = KTO(NB1,K-1,L)    C1712500
C      SK(K-1) = CODED WORD, ALL BODIES IN NEST K-1           C1712600
C      SMC(K-1) = CODED WORD, ALL MOMENTUM WHEELS IN NEST K-1 C1712700
C      XMC(I,J,L) = TRANSFORMATION MATRIX, BODY L TO COMPUTING FRAME C1712800
C      XMC(I,J,0) = TRANSFORMATION MATRIX, INERTIAL TO COMPUTING FRAME C1712900
C      Y = SOLUTION ARRAY, CONTAINS INTEGRATED PARAMETERS     C1713000
C      YD = EQUATION ARRAY, SENT TO RUNGE FOR INTEGRATION      C1713100
C      NEQ = NUMBER OF FIRST ORDER DIFFERENTIAL EQUATIONS DEFINED C1713200
C                  OUTSIDE OF SUBROUTINE TORQUE                 C1713300
C                                                                C1713400
C                                                                C1713500
C                                                                C1713600
C      INPUT OF USER REQUIRED DATA FOR SUBROUTINE TORQUE      C1713700
C                                                                C1713800
C      THE USER MAY APPLY ONE OF THREE OPTIONS                C1713900
C      1) PREFERABLE, DEFINE ALL USER REQUIRED DATA ON        C1714000
C          'DATA' CARDS WITHIN SUBROUTINE TORQUE              C1714100
C      2) WRITE SUBROUTINE INTOR AND PASS ALL USER REQUIRED DATA C1714200
C          THROUGH COMMON IN /SATELL/                          C1714300
C      3) READ INPUT DATA ON FIRST PASS THROUGH TORQUE       C1714400
C          CT4 = 1 ON FIRST PASS THROUGH, STORE DATA         C1714500
C          IN /SATELL/                                         C1714600
C                                                                C1714700
C                                                                C1714800
C                                                                C1714900
C                                                                C1715000
C                                                                C1715100
C      REACTION TORQUE ACTING ACROSS OR ALONG GIMBAL AXIS M C1715200
C      AT HINGE POINT K-1 DUE TO:
C      LINEAR SPRINGS                                         C1715300
C      LINEAR VISCOUS DAMPERS                                 C1715400
C      MOTORS                                                 C1715500
C                                                                C1715600
C      LET:
C      SPR(M) = SPRING CONSTANT ABOUT OR ALONG GIMBAL AXIS M C1715700
C              (M*L**2/T**2 OR M/T**2)                        C1715800
C              C1715900
C      DPC(M) = DAMPING CONSTANT ABOUT OR ALONG GIMBAL AXIS M C1716000
C              (M*L**2/T OR M/T)                              C1716100
C      CLT(M) = CONTROL TORQUE APPLIED BY MOTOR ABOUT OR ALONG GIMBAL C1716200
C              AXIS M M*L**2/T**2 OR M*L/T**2                C1716300
C              C1716400
C      DIMENSION TEM(3)                                       C1716500
C      SPRING TORQUE
C      SPR(M) = USER INPUT                                     C1716600
C      A = SPR(M)*THA(M)                                        C1716700
C      CALL SCLV(A,QFC(1,M),TEM)                               C1716800
C      CALL VEC SUB(PHI(1,K),TEM,PHI(1,K))                     C1716900
C      DAMPER TORQUE
C      DPC(M) = USER INPUT                                     C1717000
C      CALL SCLV(A,QFC(1,M),TEM)                               C1717100
C      CALL VEC SUB(PHI(1,K),TEM,PHI(1,K))                     C1717200
C      MOTOR TORQUE
C      CLT(M) = FUNCTION OF STATE VARIABLES, USER DEF. C1717300
C      CALL SCLV(CLT(M),QFC(1,M),TEM)                          C1717400
C      CALL VEC ADD(PHI(1,K),TEM,PHI(1,K))                    C1717500
C      MOTOR TORQUE
C      CLT(M) = FUNCTION OF STATE VARIABLES, USER DEF. C1717600
C      CALL SCLV(CLT(M),QFC(1,M),TEM)                          C1717700
C      CALL VEC ADD(PHI(1,K),TEM,PHI(1,K))                    C1717800
C      MOTOR TORQUE
C      CLT(M) = FUNCTION OF STATE VARIABLES, USER DEF. C1717900
C      CALL SCLV(CLT(M),QFC(1,M),TEM)                          C1718000
C      CALL VEC ADD(PHI(1,K),TEM,PHI(1,K))                    C1718100

```

C		C1718000
C		C1718100
C		C1718200
C		C1718300
C		C1718400
C	REACTION TORQUES ON SYSTEM DUE TO A CONTROL TORQUE	C1718500
C	APPLIED TO MOMENTUM WHEEL MW	C1718600
C		C1718700
C	LET:	C1718800
C	CLM(MW) = CONTROL TORQUE APPLIED TO WHEEL MW ABOUT ITS SPIN AXIS	C1718900
C	USER DEFINED FUNCTION OF STATE VARIABLES (M*L**2/T**2)	C1719000
C		C1719100
C	C CLM(MW) = USER DEFINED FUNCTION OF STATE VARS.	C1719200
C		C1719300
C		C1719400
C		C1719500
C		C1719600
C		C1719700
C	REACTION TORQUES ON SYSTEM DUE TO A LOCALLY	C1719800
C	APPLIED EXTERNAL FORCE (I.E. A GAS JET)	C1719900
C		C1720000
C	LET:	C1720100
C	J = INTEGER LABEL ASSIGNED TO GAS JET	C1720200
C	L = BODY TO WHICH EXTERNAL FORCE DIRECTLY APPLIED	C1720300
C	RJ(I) = RADIUS VECTOR FROM CENTER OF MASS OF BODY L TO	C1720400
C	GAS JET J, (COMPONENTS RELATIVE TO BODY L COORDINATES)	C1720500
C	FJ(I) = COMPONENTS OF APPLIED FORCE DUE TO GAS JET J, (RELATIVE	C1720600
C	TO BODY L COORDINATES) USER DEFINED FUNCTION OF STATE	C1720700
C	VARIABLES (M*L/T**2)	C1720800
C		C1720900
C	DIMENSION RJC(3),FJC(3),TEM(3),TEM1(3),RJ(3),FJ(3)	C1721000
C	INTEGER S1(10),NS1	C1721100
C	LOGICAL CTAIN	C1721200
C	C RJ(I) = USER INFLT	C1721300
C	CALL VECTR(N,RJ,XMC(1,1,L),RJC)	C1721400
C	C FJ(I) = USER DEFINED FUNCTION OF STATE VARIABLES	C1721500
C	CALL VECTR(N,FJ,XMC(1,1,L),FJC)	C1721600
C	CALL VECADD(PHI(1,NE1),FJC,PHI(1,NE1))	C1721700
C	DO 3 K=1,NBOD	C1721800
C	CALL UNPAC(S1,NS1,SK(K-1))	C1721900
C	IF(.NOT.CTAIN(L,S1,NS1)) GO TO 3	C1722000
C	IF(RBL0(K)) GO TO 4	C1722100
C	CALL VECADD(PHI(1,K),FJC,PHI(1,K))	C1722200
C	GO TO 3	C1722300
C	4 KL = KTO(NB1,K-1,L)	C1722400
C	CALL VECADD(GAM(1,KL),RJC,TEM)	C1722500
C	CALL VECRCS(TEM,FJC,TEM1)	C1722600
C	CALL VECADD(PHI(1,K),TEM1,PHI(1,K))	C1722700
C	3 CONTINUE	C1722800
C		C1722900
C		C1723000
C		C1723100
C		C1723200
C		C1723300
C	REACTION TORQUES ON SYSTEM DUE TO GRAVITATIONAL	C1723400
C	EFFECTS ON AN EARTH BASED SYSTEM	C1723500
C		C1723600
C	LET:	C1723700
C	GRAV = ACCELERATION OF GRAVITY (L/T**2)	C1723800
C	BF(I) = COMPONENTS OF UNIT VECTOR FROM INERTIAL ORIGIN TO COMP.	C1723900
C	SYSTEM CENTER OF MASS, (RELATIVE TO INERTIAL FRAME)	

```

C          THAT IS, PARALLEL TO DIRECTION OF GRAVITY FORCE          C1724000
C
C          INTEGER S1(10),NS1          C1724100
C          DIMENSION TEM(3),BFC(3),HH(3)          C1724200
C          BH(I) = USER INPUT          C1724300
C          CALL VECTRN(BF,XMC(1,1,0),BHC)          C1724400
C          DO 4 K=1,NBOC          C1724500
C          GRAV = USER INPLT          C1724600
C          A = XMAS(K)*GRAV          C1724700
C          CALL SCLV(A,BFC,TEM)          C1724800
C          CALL VECSLB(PHI(1,NB1),TEM,PHI(1,NB1))          C1724900
C          IF(RBLO(K)) GO TO 5          C1725000
C          CALL VECSUB(PHI(1,K),TEM,PHI(1,K))          C1725100
C          GO TO 4          C1725200
C          5 CALL UNPAC(S1,NS1,SK(K-1))          C1725300
C          DO 4 LL=1,NS1          C1725400
C          L = S1(LL)          C1725500
C          KL = KTO(NB1,K-1,L)          C1725600
C          A = XMAS(L)*GRAV          C1725700
C          CALL SCLV(A,BFC,TEM)          C1725800
C          CALL VECROS (GAM(1,KL),TEM,TEM1)          C1725900
C          CALL VECSLB(PHI(1,K),TEM1,PHI(1,K))          C1726000
C          4 CONTINUE          C1726100
C          C1726200
C          C1726300
C          C1726400
C          C1726500
C          C1726600
C          C1726700
C          C1726800
C          C1726900
C          KEPLERIAN ORBIT          C1727000
C          LET:          C1727100
C          CE(1,0) = COMPONENTS OF VECTOR FROM EARTH'S CENTER TO COMPOSITE          C1727200
C          SYSTEM CENTER OF MASS, RELATIVE TO INERTIAL REFERENCE          C1727300
C          FRAME, ORBIT ASSUMED TO BE IN INERTIAL 2-3 PLANE          C1727400
C          ASM = SEMI-MAJOR AXIS OF ELLIPTIC ORBIT INERTIAL 2 DIRECTION          C1727500
C          ECC = ORBIT ECCENTRICITY          C1727600
C          TPP = TIME OF PERIHELION PASSAGE, (T)          C1727700
C          GEV = EARTH'S GRAVITATIONAL CONSTANT, (L**3/T**2)          C1727800
C          ETE = MEAN MOTION          C1727900
C          AME = MEAN ANOMALY          C1728000
C          ECE = ECCENTRIC ANOMALY          C1728100
C          TVE = TRUE ANOMALY          C1728200
C          BTC = MAGNITUDE OF CE(1,0), (L)          C1728300
C          C1728400
C          C1728500
C          C1728600
C          C1728700
C          C1728800
C          C1728900
C          C1729000
C          C1729100
C          C1729200
C          C1729300
C          C1729400
C          C1729500
C          C1729600
C          C1729700
C          C1729800
C          C1729900
C          C1729900

```



```

C      CB(2,0) = BTO*((CE - ECC)/(1 - ECC*CE))          C1730000
C      CB(3,0) = BTO*(SQRT(1-ECC**2)*SE/(1-ECC*CE))    C1730100
C      CALL VECTRN(CE(1,0),XMC(1,1,0),CBC(1,0))        C1730200
C                                                       C1730300
C                                                       C1730400
C                                                       C1730500
C                                                       C1730600
C                                                       C1730700
C      REACTION TORQUES ON ORBITING SYSTEM DUE TO     C1730800
C      GRAVITY GRADIENT EFFECTS                       C1730900
C      LET:                                            C1731000
C      CBC(I,1) = COMPONENTS OF VECTOR FROM COMPOSITE SYSTEM CENTER OF
C      MASS TO CENTER OF MASS OF BODY 1, RELATIVE TO   C1731100
C      COMPUTING FRAME                                C1731200
C      * NOTE FOR GRAVITY GRADIENT OPTION             C1731300
C      * CE(I,1) AND ITS INERTIAL DERIVATIVE         C1731400
C      * ARE REDEFINED TO CIRCUMVENT DIFFERENCE     C1731500
C      * OF LARGE NUMBER PROBLEMS, THAT IS          C1731600
C      * THEY ARE MEASURED FROM COMPOSITE CM TO      C1731700
C      * CM OF BODY 1 RATHER THAN FROM INERTIAL     C1731800
C      * ORIGIN TO CM OF BODY 1                     C1731900
C      * ORIGIN TO CM OF BODY 1                     C1732000
C      BH(I) = UNIT VECTOR FROM EARTH'S CENTER TO SYSTEM COMPOSITE
C      CENTER OF MASS, COMPONENTS RELATIVE TO INERTIAL FRAME C1732100
C      DEL(I,K) = COMPONENTS OF VECTOR FROM COMPOSITE SYSTEM CENTER OF
C      MASS TO CENTER OF MASS OF BODY K              C1732200
C      DFG(I,K) = COMPONENTS OF GRAVITY GRADIENT FORCE ACTING ON BODY K C1732300
C      SGG(I) = COMPACTED INTEGER WORD, THOSE BODIES IN THE NEXT I
C      WHICH SIGNIFICANTLY CONTRIBUTE TO GRAVITY GRADIENT EFF. C1732400
C      BTO = DISTANCE FROM EARTH'S CENTER TO COMPOSITE SYSTEM CM C1732500
C                                                       C1732600
C      DIMENSION DEL(3,10),DFG(3,10),BHC(3),BH(3),TEM(3),TEM1(3) C1732700
C      INTEGER SGG(0,9), S1(10)                     C1732800
C      C KEPLERIAN ORBIT MUST BE USED WITH GRAVITY GRADIENT OPTION C1732900
C      DO 10 I=1,3                                    C1733000
C      10 BHC(I) = CBC(I,0)/ETO                         C1733100
C      DO 7 K=1,NBOD                                  C1733200
C      KL = KTO(NB1,C,K)                               C1733300
C      CALL VECADD(CBC(I,1),GAM(1,KL),DEL(I,K))        C1733400
C      CALL VECDCI(BHC,DEL(I,K),A)                    C1733500
C      A = 3*A                                          C1733600
C      CALL SCLV(A,BHC,TEM)                            C1733700
C      CALL VEC SUB(DEL(I,K),TEM,TEM)                 C1733800
C      A = -GEV*MAS(K)/BTO**3                         C1733900
C      CALL SCLV(A,TEM,DFG(I,K))                      C1734000
C      7 CONTINUE                                     C1734100
C      DO 8 K=1,NBOD                                  C1734200
C      IF(RBLO(K)) GO TO 9                             C1734300
C      CALL VECADD(PHI(1,K),DFG(1,K),PHI(1,K))        C1734400
C      GO TO 8                                         C1734500
C      C SGG(I) = SK(I) IF ALL BODIES CONTRIBUTE TO GRAVITY GRAC. C1734600
C      C EFFECTS. IF NOT USE COMPACT TO CONSTRUCT C1734700
C      C SGG(I) FROM USER INPUT OR DEFINE DIRECTLY C1734800
C      C CALL UNPAC(S1,NS1,SGG(K-1))                  C1734900
C      DO 8 LL=1,NS1                                   C1735000
C      L=SI(LL)                                       C1735100
C      KL = KTO(NB1,K-1,L)                             C1735200
C      CALL VECRCS (GAM(1,KL),DFG(1,L),TEM)           C1735300
C      CALL VXDYCV(BHC,XIC(1,1,L),TEM1)              C1735400
C      A = 3*GEV/BTO**3                               C1735500
C      CALL SCLV(A,TEM1,TEM1)                         C1735600

```

```

C          CALL VECACD(TEM,TEM1,TEM)                                C1736900
C          CALL VECACD(PHI(1,K),TEM,PHI(1,K))                       C1736100
C          2 CONTINUE                                              C1736200
C                                                                    C1736300
C                                                                    C1736400
C                                                                    C1736500
C                                                                    C1736600
C                                                                    C1736700
C                                                                    C1736800
C          PARAMETERS DEFINED BY FIRST ORDER                       C1736900
C          DIFFERENTIAL EQUATIONS                                  C1737000
C          LET:                                                    C1737100
C          NTQ = TOTAL NUMBER OF FIRST ORDER DIFFERENTIAL EQUATIONS TO C1737200
C          BE SOLVED FOR USE IN SUBROUTINE TORQUE                   C1737300
C          TQ(N) = MAGNITUDE OF PARAMETER NUMBER N DEFINED WITHIN SLE. C1737400
C          TORQUE AT TIME T                                         C1737500
C          TQD(N) = TIME DERIVATIVE OF PARAMETER TQ(N). A USER DEFINED C1737600
C          FUNCTION OF THE SYSTEM'S STATE VARIABLES                 C1737700
C                                                                    C1737800
C          DIMENSION TQ(20),TQD(20),                               C1737900
C          C FOR THE PARAMETER N                                    C1738000
C          IF(CT4.NE.1) GO TO 11                                     C1738100
C          C Y(NEG+N) = TQ(N) = INITIAL VALUE FOR TQ(N)           C1738200
C          11 TQ(N) = Y(NEG+N)                                       C1738300
C          C TQD(N) = USER DEFINED FUNCTION OF STATE VARB.       C1738400
C          C                                                         C1738500
C          C AFTER DEFINITION OF LAST DIFFERENTIAL EQUATION      C1738600
C          C NTQ = TOTAL NUMBER OF FIRST ORDER DIFFERENTIAL      C1738700
C          C EQUATIONS TO BE SOLVED FOR USE IN TORQUE            C1738800
C          C DO 12 N=1,NTQ                                         C1738900
C          C 12 YD(NEG+N) = TQD(N)                                  C1739000
C                                                                    C1739100
C                                                                    C1739200
C                                                                    C1739300
C                                                                    C1739400
C                                                                    C1739500
C          THERMALLY INDUCED MOTION ABOUT GIMBAL AXIS M           C1739600
C          AT HINGE POINT K-1                                       C1739700
C          ASSUME:                                                  C1739800
C          ALL THERMALLY INDUCED DEFLECTION IS SMALL ANGLE        C1739900
C          RELATIVE TO THE SYSTEM'S NOMINAL ZERO STRESS STATE     C1740000
C          THERMALLY INDUCED DEFLECTION IS MODELLED AS A MOVEMENT C1740100
C          OF THE ZERO STRESS STATE                                C1740200
C          ACROSS ALL HINGE POINTS SUBJECT TO THERMAL DEFORMATION C1740300
C          SPRINGS AND DAMPERS ACT                                  C1740400
C          A REASONABLE MODEL OF THE THERMAL INPUT CAN BE DEFINED C1740500
C          IN TERMS OF THE SYSTEM'S STATE VARIABLES                C1740600
C          THERMAL EQUILIBRIUM POSITION ABOUT ANY GIMBAL AXIS IS DEF. C1740700
C          BY SOLUTION OF THE HEAT CONDUCTION EQUATION            C1740800
C          LET:                                                    C1740900
C          CSFR(M) = SPRING CONSTANT ACROSS GIMBAL AXIS M          C1741000
C          CDAMP(M) = DAMPING CONSTANT ACROSS GIMBAL AXIS M        C1741100
C          CTAU(M) = THERMAL TIME CONSTANT FOR DEFORMATION ABOUT GIMBAL C1741200
C          AXIS M, (T)                                             C1741300
C          CTEMPID TQ(N) = THERMAL EQUILIBRIUM POSITION FOR THERMAL DEFORMATION C1741400
C          ABOUT GIMBAL AXIS M, (RAD)                              C1741500
C          C TQD(N) = RATE OF CHANGE OF THERMAL EQUILIBRIUM POSITION ABOUT C1741600
C          GIMBAL AXIS M, FIRST ORDER DIFF. EQ., (RAD/T)         C1741700
C          C Tinp = THERMAL INPUT USER DEFINED FUNCTION OF STATE VARIABLES. C1741800
C          C (RAD/T)                                               C1741900

```

```

C 1742000
C DIMENSION TEM(3) 1742100
C C N = USER DEFINED LABEL, DEPENDS UPON EQUATION NUMBERING 1742200
C C SEQUENCE DEFINED WITHIN SUBROUTINE TORQUE 1742300
C IF(CT4.NE.1) GO TO 13 1742400
C C Y(NEQ+N) = TQ(N) = INITIAL VALUE FOR THERMAL DEFORMATION 1742500
C C ABOUT GIMBAL AXIS M, USER INPUT 1742600
C 13 TQ(N) = Y(NEQ+N) 1742700
C C T1NP = USER DEFINED THERMAL INPUT FOR THERMAL DEFORMATION 1742800
C C ABOUT GIMBAL AXIS M 1742900
C C TAU(M) = USER INPUT 1743000
C TQD(N) = -TQ(N)/TAL(M) + T1NP 1743100
C A = SPR(M)*((THA(M) - TQ(N))) 1743200
C CALL SCLV(A,QFC(1,M),TEM) 1743300
C CALL VEC SUB(PHI(1,K),TEM,PHI(1,K)) 1743400
C A = DPC(M)*THAD(M) 1743500
C CALL SCLV(A,QFC(1,M),TEM) 1743600
C CALL VEC SUB(PHI(1,K),TEM,PHI(1,K)) 1743700
C 1743800
C 1743900
C 1744000
C 1744100
C 1744200
C 1744300
C 1744400
C ZERO ALL ELEMENTS OF EXTERNAL TORQUE MATRIX 1744500
C DO 1 K=1,NB1 1744600
C DO 1 I=1,3 1744700
C 1 PHI(I,K) = 0.00 1744800
C DO 2 M=1,NMO 1744900
C 2 CLM(M) = 0.00 1745000
C 1745100
C RETURN 1745200
C END 1745300

```

```

C 1800000
C SUBROUTINE QFDC 1800100
C USED TO REDUCE THE SET OF NBOD+1+NMV VECTOR DYADIC EQUATIONS CF 1800200
C MCTION TO NFER+NMV SCALAR EQUATIONS 1800300
C 1800400
C LET 1800500
C SCF(K) = LOWEST MAGNITUDE FREE COORDINATE INDICE AT HINGE 1800600
C POINT K-1. EQUALS ZERO IF THREE CONSTRAINED AXES 1800700
C SCL(K) = LOWEST MAGNITUDE LOCKED COORDINATE INDICE AT HINGE 1800800
C POINT K-1. EQUALS ZERO IF THREE FREE AXES 1800900
C SCK(K) = BODY LABELS CN PATH FROM HINGE POINT ZERO TO C.M. 1801000
C OF BODY K. FOR K=NB1 IT IS SET OF ALL BODY LABELS 1801100
C IMPLICIT REAL*8(A-H,O-Z,1) 1801200
C 1801300
C 1801400
C LOGICAL FG1, FG2, FG3, FG4, FGS, INERF, RBLO, LEQU, LINIT(1) 1801500
C LOGICAL LRLNGE, LTRNSI, LVDIV, LEQUIV, LTRAN, 1801600
C * LTRANV, LRATE, LXDY, LETA, LTORQU, 1801700
C * LQFDC, LDCT, LANGLE, LSETUP, LSIMQ 1801800
C 1801900
C 1802000

```

	INTEGER								C1802100
	* AWDFK	, CT1	, CT2	, CT3	, CT4	, CT5	, FCON	, PCCN	, C1802200
	* SCNDUM	, SCN	, SCRDM	, SCR	, SFKDUM	, SFK	, SFR	, SG	, C1802300
	* SI	, SIG	, SIXDUM	, SIX	, SKDUM	, SK	, SL	, SLK	, C1802400
	* SMA	, SMCDUM	, SMC	, SMV	, SOK	, SPIDUM	, SPI	, SQF	, C1802500
	* SGL	, SR	, SSCN	, SSIX	, SVA	, SVB	, SVD	, SVI	, C1802600
	* SVM	, SVP	, SVQ	, SXM	, SXT	, TORQ	, SMAL	, SEU	, C1802700
	* SC	, SCG	, SFLXB	, SFLX	, SFXM	, NMCDS	, SFCC	, SCC	, C1802800
	* IINIT(1)		, IZINIT(1)		, SD	, SCXC(20)			, C1802900
C									, C1803000
C									, C1803100
	REAL *B								, C1803200
	* ANGE	(33)	, CNF	(3,10)	, ETIC	(3,10)	, ETMC	(3,10)	, C1803300
	* FLQ	(3,20)	, FLE	(3,3,20)	, FLH	(3,3,20)			, C1803400
	* THADD	(33)	, YMCD	(3,2,11)	, RINIT	(1)	, RZINIT	(1)	, C1803500
C									, C1803600
C									, C1803700
	COMMON /LDEBUG/	LRLNGE	, LTFNSI	, LYDIV	, LEQUIV	, LTRAN			, C1803800
	*	LTRANV	, LRATE	, LXDY	, LETA	, LTORQU			, C1803900
	*	LCFDDT	, LDCT	, LANGLE	, LSETUP	, LSIMO			, C1804000
C									, C1804100
C									, C1804200
	COMMON /LOGIC/	FG1, FG2, FG3, FG4, FG5, INERF, RBLJ(10)							, C1804300
C									, C1804400
C									, C1804500
	COMMON /INTG/	AWDFK(200)							, C1804600
	* CT1		, CT2		, CT3		, CT4		, C1804700
	* CT5		, FCON	(33)	, JCCN	(10)	, LCON	(22)	, C1804800
	* MC	(10)	, NE1		, NBOJ		, NCTC		, C1804900
	* NFER		, NFKC		, NFRC		, NLDR		, C1805000
	* NMV		, NMC		, NMOA		, NSVP		, C1805100
	* NSVG		, PCON	(11)	, SD		, SFR	(33)	, C1805200
	* SG		, SI	(55)	, SIG		, SL		, C1805300
	* SLK	(33)	, SMA	(10)	, SOK	(11)	, SQF	(11)	, C1805400
	* SGL	(11)	, SMV		, SR		, SSCN		, C1805500
	* SSIX		, SVA		, SVB		, SVD		, C1805600
	* SVI		, SVM		, SVP	(22)	, SVQ	(33)	, C1805700
	* SXM	(3,10)	, SXT		, TORQ	(97)	, SMAL		, C1805800
	* SEU		, NTQ		, SC	(33)	, SCG		, C1805900
	* NFLXB		, SFLX		, SFXM	(10)	, NMCDS		, C1806000
	* SFCC		, SCC	(10)					, C1806100
C									, C1806200
C									, C1806300
	COMMON /INTGZ/								, C1806400
	* SCNDUM		, SCN	(9)	, SCRDM		, SCR	(9)	, C1806500
	* SFKDUM		, SFK	(9)	, SIXDUM		, SIX	(9)	, C1806600
	* SKDUM		, SK	(9)	, SPIDUM		, SPI	(9)	, C1806700
	* SMCDUM		, SMC	(5)					, C1806800
C									, C1806900
C									, C1807000
	COMMON /REAL/								, C1807100
	* CA	(3,10)	, CAC	(3,10)	, CLM	(10)	, CUMC	(3,11)	, C1807200
	* DCMC	(3,11)	, ETC	(3,11)	, ETM	(33)	, FDMC	(3,11)	, C1807300
	* GAM	(3,66)	, H		, HM	(3,10)	, HMC	(3,10)	, C1807400
	* HMOM	(10)	, PHI	(3,11)	, PLM	(10)	, QF	(3,33)	, C1807500
	* QFC	(3,33)	, QL	(3,22)	, QLC	(3,22)	, RDMC	(3,11)	, C1807600
	* T				, THA	(33)	, THAD	(33)	, C1807700
	* THADW	(10)	, THAW	(10)	, XDIC	(3,3,66)	, XI	(3,3,10)	, C1807800
	* XIC	(3,3,10)	, XMAS	(10)	, XMN	(33,33)	, XMT	(3,3,10)	, C1807900
	* TLG	(33)	, FLA	(3,20)	, FLB	(3,20)	, FLC	(3,20)	, C1808000

```

* FLD (3,3,20), FLJ (3,3,20), CAO (3,10) , X10 (3,3,10), C1808100
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLOC (3,20) , C1808200
* FLCM (20) , ZETA (20) , FCF (3,3,40), FCK (3,40) , C1808300
* TIMEND C1808400
C C1808500
C C1808600
COMMON /REALZ/ C1808700
* CEDLM (1,3) , CE (3,10) , CBCDUM(1,3) , CBC (3,10) , C1808800
* X*CELM(1,1,9) , XPC (3,3,10), CBN(3) C1808900
C C1809000
C1809100
EQUIVALENCE (ETM(1),THADD(1)) ,(XMN(1,1),ANGD(1)) C1809200
* (XMN(1,3),YMCD(1,1,1)) ,(XMN(1,6),CNF(1,1)) C1809300
* (XMN(1,8),ETIC(1,1)) ,(XMN(1,10),ETMC(1,1)) C1809400
* (FLB(1,1),FLQ(1,1)) ,(FLE(1,1,1),FLD(1,1,1)) C1809500
* (FLH(1,1,1),FLJ(1,1,1)) , C1809600
* (FG1,LINIT(1)) ,(CA(1,1),RINIT(1)) C1809700
* (CEDUM(1,1),RZINIT(1)) ,(AWGRK(1),IINIT(1)) C1809800
* (SCNDUM,IZINIT(1)) ,(TORG(78),SCXC(1)) C1809900
C1810000
C1810100
C1810200
C1810300
C1810400
DIMENSION TEM1(3),TEM(3,11),XDD(3,11)
DIMENSION TEM2(3),TEM3(3),FTEM(3),FLHC(3,3)
DIMENSION EQFC(99), EXDIC(554)
REAL*8 FCUP(3,3,20),FCLP1(3,3),KCUP(3,20),KCUP1(3)
REAL*8 TMF(3,3), TMK(3)
INTEGER ST1(10),ST2(10),ST3(11)
INTEGER ST4(10)
INTEGER SFXMN
LOGICAL CTAIN
EQUIVALENCE (DCMC(1,1),TEM(1,1)), (EQFC(1),QFC(1,1)),
* (EXDIC(1),XDIC(1,1,1))
DATA IH/4HXMN(/
EQUIVALENCE (LQFDCT,LEQU)
C1811000
C1811100
C1811200
C1811300
C1811400
C1811500
C1811600
C1811700
C1811800
C1811900
C1812000
C1812100
C1812200
C1812300
C1812400
C1812500
C1812600
C1812700
C1812800
C1812900
C1813000
C1813100
C1813200
C1813300
C1813400
C1813500
C1813600
C1813700
C1813800
C1813900
C1814000

```

C	GET NON-ZERO ELEMENTS IN ROW K OF XDIC	C1814100
	IF(LEQU)PRINT 227	C1814200
	IF(LEQU)PRINT 205, K, (ST3(I), I=1, NST3)	C1814300
	IF(LEQU)PRINT 228	C1814400
	IF(NST3.EQ.0) GO TO 41	C1814500
	DO 40 I1=1, NST3	C1814600
	I = ST3(I1)	C1814700
	MBEG=SQF(K)	C1814800
	NBEG=SQF(I)	C1814900
	MTERM=SQF(K)+2-PCCN(K)	C1815000
	NTERM=SQF(I)+2-PCCN(I)	C1815100
	IF(MBEG.GT.MTERM) GO TO 43	C1815200
	DO 42 M=MBEG, MTERM	C1815300
	IF(NBEG.GT.NTERM) GO TO 45	C1815400
	DO 44 N=NBEG, NTERM	C1815500
	IF(N.GT.M) GO TO 44	C1815600
	KI = KT1(NB1, K, I)	C1815700
	ME = 3*(N-1)	C1815800
	NE = 3*(N-1)	C1815900
	KIE = 9*(KI-1)	C1816000
C	CALL VODYUV(QFC(1, M), XDIC(1, I, KI), QFC(1, N), XMN(M, N))	C1816100
C	REPLACE MULTI-SUBSCRIPT OPERATION WITH SINGLE SUBSCRIPT	C1816200
C	COMPUTATION OF VECTOR DOT CYAD DOT VECTOR	C1816300
	DO 10 J=1, J	C1816400
	TEM1(J) = EXDIC(KIE+J)*EQFC(NE+1) +	C1816500
	* EXDIC(KIE+3+J)*EQFC(NE+2) +	C1816600
	* EXDIC(KIE+6+J)*EQFC(NE+3)	C1816700
10	CONTINUE	C1816800
	XMN(M, N) = EQFC(ME+1)*TEM1(1) +	C1816900
	* EQFC(ME+2)*TEM1(2) +	C1817000
	* EQFC(ME+3)*TEM1(3)	C1817100
	IF(LEQU)PRINT 206, M, N, M, KI, N, M, K, I, N, XMN(M, N)	C1817200
44	CONTINUE	C1817300
45	CONTINUE	C1817400
42	CONTINUE	C1817500
43	CONTINUE	C1817600
40	CONTINUE	C1817700
41	CONTINUE	C1817800
7	CONTINUE	C1817900
C		C1818000
C		C1818100
C		C1818200
C	INCLUDE FLEXIBILITY TERMS	C1818300
	IF(NFLXB.EQ.C) GO TO 11	C1818400
C		C1818500
	CALL UNPAC(ST1, NST1, SFLX)	C1818600
	CALL UNPAC(ST4, NST4, SFCC)	C1818700
C		C1818800
C	ZERO OUT GYROSCOPIC TORQUE ARRAY ETM(M), M=NFER+1, ..., NFER+NMODS	C1818900
	NF1 = NFER + 1	C1819000
	NF2 = NFER + NMODS	C1819100
	DO 17 M=NF1, NF2	C1819200
	IF(LEQU) PRINT 252, M	C1819300
17	ETM(M) = 0.	C1819400
C		C1819500
C	CYCLE THRU ALL NESTS, FOR K=1 PICK UP DIAGONAL AND MOST OF IT	C1819600
C	RIGHT HAND SIDE OF EQUATION	C1819700
	DO 3 K=1, NBUD	C1819800
	CALL UNPAC(ST2, NST2, SK(K-1))	C1819900
	MN=0	C1820000

	DO 3 NN=1,NST1	C1E20100
	N = ST1(NST1+1-NN)	C1E20200
C	CHECK IS BODY N A FLEXIBLE BODY IN NEST K-1	C1E20300
	IF(C1AIN(N,ST2,NST2)) GO TO 4	C1E20400
	MN = MN+SFXM(N)	C1E20500
	GO TO 3	C1E20600
4	CONTINUE	C1E20700
	IF(.NOT.LEQU) GO TO 5000	C1E20800
	KM1 = K-1	C1E20900
	PRINT 224, N,KM1	C1E21000
5000	CONTINUE	C1E21100
C		C1E21200
C	FOR K=1 GET: 1)DIAGONAL ELEMENTS OF XMN(M,N)	C1E21300
C	2)CENTRIPITAL ACC. OF UNDEFORMED CM OF BODY N	C1E21400
C	3)MAJOR PORTION OF ETM(M)	C1E21500
C	M=NFER+1,.....NFER+NMDS	C1E21600
C		C1E21700
	IF(K.NE.1) GO TO 5	C1E21800
	TEM2(1) = 0.	C1E21900
	TEM2(2) = 0.	C1E22000
	TEM2(3) = 0.	C1E22100
	IF(LEQU) PRINT 241	C1E22200
C	CENTRIPITAL ACCELERATION UNDEFORMED POSITION OF BODY N CM.	C1E22300
	NI = N	C1E22400
	IF(NI.EQ.1) GO TO 12	C1E22500
	CALL VECTRNC(CAU(1,NI),XMC(1,1,NI),TEM3)	C1E22600
	IF(LEQU) PRINT 211, NI,NI,NI	C1E22700
	CALL TRIPVP(FOMC(1,NI),TEM3,TEM2)	C1E22800
	IF(LEQU) PRINT 212, NI,NI,NI	C1E22900
32	JNI = JCCN(NI)	C1E23000
	CALL TRIFVP(FOMC(1,JNI),CBC(1,NI),TEM3)	C1E23100
	CALL VECADD(TEM2,TEM3,TEM2)	C1E23200
	IF(LEQU) PRINT 213, JNI,JNI,NI	C1E23300
	NI = JNI	C1E23400
	IF(NI.NE.1) GO TO 32	C1E23500
12	CALL SOLV(XMAS(N),TEM2,TEM2)	C1E23600
	IF(LEQU) PRINT 222, N,(TEM2(J),J=1,3)	C1E23700
5	CONTINUE	C1E23800
C		C1E23900
C		C1E24000
C		C1E24100
C	CYCLE THRU ALL FLEXIBLE BODY MODES ASSOCIATED WITH BODY N IN	C1E24200
C	THE NEST K-1, FOR K=1 DO EXTRA COMPUTATIONS	C1E24300
C		C1E24400
	SFXMN = SFXM(N)	C1E24500
	DO 3 I1=1,SFXMN	C1E24600
	MN = MN+1	C1E24700
	M = NFER + MN	C1E24800
C	COMPLETE VECTOR ELEMENTS OF F MATRIX AS NEEDED DGN'T STORE IN	C1E24900
C	COMMON, STORE AS NEEDED IN FTEM TO SAVE STORAGE	C1E25000
	KN = KTC(NB1,K-1,N)	C1E25100
	CALL SOLV(XMAS(N),GAM(1,KN),TEM3)	C1E25200
	CALL VECROSS(TEM3,FLAC(1,MN),FTEM)	C1E25300
	CALL VECADD(FTEM,FLQC(1,MN),FTEM)	C1E25400
	IF(.NOT.LEQU) GO TO 5001	C1E25500
	KM1 = K-1	C1E25600
	PRINT 235, KM1,N,MN,N,KM1,N,MN,MN,(FTEM(J),J=1,3)	C1E25700
5001	CONTINUE	C1E25800
	IF(K.NE.1)GO TO 19	C1E25900
	CALL TETRNC(FLH(1,1,MN),XMC(1,1,N),FLHC)	C1E26000

```

      IF(.NOT.LEQU) GO TO 5002
      PRINT 22E
      PRINT 25C,          (FLFC(1,I),I=1,3)
      PRINT 253, MN,N,MN,N,(FLFC(2,I),I=1,3)
      PRINT 25C,          (FLFC(3,I),I=1,3)
5002 CONTINUE
19 CONTINUE
C
C   COMPLETE ELEMENTS OF XMN ARRAY
C
      IF(LEQU) PRINT 22E
      IF(PCCN(K).EQ.3) GO TO 46
      N1 = SQF(K)
      N2 = SQF(K) + 2 - PCCN(K)
      DO 6 L=N1,N2
      CALL VECCOT(QFC(1,L),FTEM,XMN(M,L))
      IF(LEQU) PRINT 236, M,L,L,KM1,N,MN,XMN(M,L)
6 CONTINUE
46 CONTINUE
C
C   THAT'S IT FOR XMN IF K.NE.1
C   IF(K.NE.1) GO TO 8
      XMN(M,M) = XMAS(N)
      IF(LEQU) PRINT 237, M,M,N,XMN(M,M)
C   GET TERMS ASSOCIATED WITH THE TRANSLATION EQUATION
      CALL SOLV(XMAS(N),FLAC(1,MN),TEM3)
      IF(PCCN(NB1).EQ.3) GO TO 8
      N1 = SQF(NB1)
      N2 = SQF(NB1) + 2 - PCCN(NB1)
      DO 13 L=N1,N2
      CALL VECCOT(QFC(1,L),TEM3,XMN(M,L))
      IF(LEQU) PRINT 219, M,L,L,N,MN,XMN(M,L)
13 CONTINUE
8 CONTINUE
C
C   CK CONCENTRATE ON EMT ARRAY NOW
C   PUT IN FTEM,QFC(DCT) TERMS
      CALL VECCOT(FTEM,CCMC(1,K),A)
      ETM(M) = ETM(M) - A
      IF(LEQU) PRINT 238, M,M,KM1,N,MN,K,ETM(M)
C   THAT'S IT IF K.NE.1
      IF(K.NE.1) GO TO 3
C
C   PUT IN SPRING-DASHPOT EFFECTS MODL MM
      ETM(M) = ETM(M) - XMAS(N)*(2.*ZETA(MN)*FLUM(MN)*THAD(M)
      *
      + FLUM(MN)**2*THA(M))
      IF(LEQU) PRINT 239, M,M,N,MN,MN,M,MN,M,ETM(M)
      CALL VDBYDV(FLMC(1,N),FLFC,FCMC(1,N),A)
      CALL VECCOT(TEM2,FLAC(1,MN),B)
      ETM(M) = ETM(M) + A - B
      IF(LEQU) PRINT 240, M,M,N,MN,N,MN,ETM(M)
3 CONTINUE
C
C
C   CHECK TO SEE IF COUPLING SIGNIFICANT
      IF(NST4.EQ.0) GO TO 11
      KF = 0
      MN = 0
      DO 35 K=1,NBOD
C   CHECK TO SEE IF BODY K FLEXIBLE

```



```

IF(SFXM(K).EQ.0) GO TO 35
CYCLE THROUGH ALL GENERALIZED COORDINATE EQUATIONS FOR BODY K
MMN = MN
(SF)MN = SFXM(K)
DO 33 N=1,SFXMN
MN = MN+1
CALL LNPAC(ST1,NST1,SCXC(MN))
IF(NST1.EQ.0) GO TO 33
IF(LEQU) PRINT 256, N,K,MN,(ST1(I),I=1,NST1)
MI = NFER+MN
DO 37 I=1,3
KCUP(I,MN) = 0.0
DO 37 J=1,3
37 FCUP(I,J,MN) = 0.0
IF(LEQU) PRINT 245
IF(LEQU) PRINT 242, MN,MN
DO 34 I=1,NST1
M = ST1(NST1+1-I)
MJ = MMN + M + NFER
KF = KF+1
CALL SCLD(THA(MJ),FCF(1,1,KF),TMF)
CALL DYADD(FCUP(1,1,MN),TMF,FCUP(1,1,MN))
IF(.NOT.LEQU) GO TO 5004
PRINT 246
PRINT 246, (FCUP(1,L,MN),L=1,3)
PRINT 247, MN,MN,MJ,KF,(FCUF(2,L,MN),L=1,3)
PRINT 246, (FCUP(3,L,MN),L=1,3)
5004 CALL SCLV(THAD(MJ),FCK(1,KF),TMK)
CALL VECADD(KCUP(1,MN),TMK,KCUP(1,MN))
IF(.NOT.LEQU) GO TO 5005
PRINT 245
PRINT 246, MN,MN,MJ,KF,(KCUF(L,MN),L=1,3)
5005 CONTINUE
34 CONTINUE
C TRANSFORM FCUP AND KCUP TO COMPUTING FRAME
CALL TENTRN(FCUP(1,1,MN),XMC(1,1,K),FCUP1)
CALL VECTRN(KCUP(1,MN),XMC(1,1,K),KCUP1)
IF(.NOT.LEQU) GO TO 5003
PRINT 245
PRINT 254, (FCUP1(1,L),L=1,3)
PRINT 257, K,MN,K,(FCUP1(2,L),L=1,3)
PRINT 254, (FCUP1(3,L),L=1,3)
PRINT 245
PRINT 255, K,MN,(KCUP1(L),L=1,3)
5003 CONTINUE
C
CALL VODYOV(FOMC(1,K),FCUP1,FOMC(1,K),A)
CALL VECCUT(FOMC(1,K),KCLP1,B)
ETM(MI) = ETM(MI) + A - 2.0*B
IF(.NOT.LEQU) GO TO 5007
PRINT 245
PRINT 249,MI,MI,K,K,K,ETM(MI)
5007 CONTINUE
33 CONTINUE
35 CONTINUE
11 CONTINUE
C THAT'S IT FOR MODAL COUPLING
C
C
C COMPUTATION FOR VARIABLE SPEED MOMENTUM WHEELS

```

```

C1E32100
C1E32200
C1E32300
C1E32400
C1E32500
C1E32600
C1E32700
C1E32800
C1E32900
C1E33000
C1E33100
C1E33200
C1E33300
C1E33400
C1E33500
C1E33600
C1E33700
C1E33800
C1E33900
C1E34000
C1E34100
C1E34200
C1E34300
C1E34400
C1E34500
C1E34600
C1E34700
C1E34800
C1E34900
C1E35000
C1E35100
C1E35200
C1E35300
C1E35400
C1E35500
C1E35600
C1E35700
C1E35800
C1E35900
C1E36000
C1E36100
C1E36200
C1E36300
C1E36400
C1E36500
C1E36600
C1E36700
C1E36800
C1E36900
C1E37000
C1E37100
C1E37200
C1E37300
C1E37400
C1E37500
C1E37600
C1E37700
C1E37800
C1E37900
C1E38000

```

$I + (I-1)*A =$

```

C      EXPAND DIMENSION OF MATRIX EQUATIONS AND COMPUTE SCALAR ELEMENTS C1838100
C      ASSOCIATED WITH PRESENCE OF MOMENTUM WHEELS C1838200
      CALL UNPAC(ST1,NST1,SMV) C1838300
      IF(NST1.EQ.0) GO TO 5057 C1838400
      IF(LEQU)PRINT 201,(ST1(I),I=1,NST1) C1838500
      CO 2 M=1,NST1 C1838600
      M=M-(M-1) C1838700
      NM = ST1(M) C1838800
      N = NFER+NST1+1-M+NM+NMCD5 C1838900
      XMN(M,M) = PLM(M) C1839000
      IF(LEQU)PRINT 203, M,M,M,XMN(M,M) C1839100
      CALL UNPAC(ST2,NST2,SC(NC(M))) C1839200
      ETM(M) = 0.D0 C1839300
      IF(LEQU)PRINT 204, M,ETM(M) C1839400
      IF(NST2.EQ.0) GO TO 5058 C1839500
      CO 1 I=1,NST2 C1839600
      I = ST2(I) C1839700
      CALL VECDOT(HMC(1,M),DNC(1,I),A) C1839800
      ETM(M) = ETM(M) - A*PLM(M) C1839900
      IF(LEQU)PRINT 207, M,M,M,M,I C1840000
      KTERM=SGF(I)+2-PCCN(I) C1840100
      KBEG=SQF(I) C1840200
      IF(KEEG.GT.KTERM) GO TO 48 C1840300
      CO 47 K=KBEG,KTERM C1840400
      N = K C1840500
      CALL VECDOT(HMC(1,M),OFC(1,K),A) C1840600
      XMN(M,N) = A*PLM(M) C1840700
      IF(LEQU)PRINT 209, M,N,M,M,K,XMN(M,N) C1840800
47 CONTINUE C1840900
48 CONTINUE C1841000
1 CONTINUE C1841100
5058 CONTINUE C1841200
      ETM(M) = ETM(M) + CLM(M) C1841300
      IF(LEQU)PRINT 210, M,M,M,ETM(M) C1841400
2 CONTINUE C1841500
5057 CONTINUE C1841600
C C1841700
C C1841800
C C1841900
C C1842000
      FILL IN UPPER TRIANGULAR PORTION C1842100
      NFMV = NFER+NMJDS+NMV C1842200
      CO 9 M=1,NFMV C1842300
      CO 9 N=M,NFMV C1842400
      XMN(M,N) = XMN(N,M) C1842500
      IF(.NOT.LEQU) GO TO 5006 C1842600
      PRINT 227 C1842700
      I1 = NFMV/4 C1842800
      I2 = NFMV - 4*I1 C1842900
      I3 = I1 + 1 C1843000
      CO 20 I=1,I3 C1843100
      I4 = 3 C1843200
      IF(I1.NE.I3) GO TO 22 C1843300
      IF(I2.EQ.0) GO TO 20 C1843400
      I4 = I2 - 1 C1843500
      I5 = 4*(I-1) + 1 C1843600
      CO 23 N=1,NFMV C1843700
      ISPI4 = I5 + I4 C1843800
      PRINT 200, (I,M,N,XMN(M,N),M=I5,ISPI4) C1843900
23 CONTINUE C1844000
      PRINT 22E C1844000

```

20	CONTINUE	C1844100
	PRINT 227	C1844200
C		C1844300
C	COMPUTE (NB1)X1 COLUMN MATRIX OF VECTORS RESULTING FROM DIFFERENTIATION	C1844400
C	VECTORS MOVING RELATIVE TO FRAME OF COMPUTATION	C1844500
C	DONE BY EQUIVALENCE (DCMC,TEM)	C1844600
	PRINT 208, (K,K,(TEM(I,K),I=1,3),K=1,NB1)	C1844700
	PRINT 227	C1844800
C		C1844900
C	MULTIPLY (NB1)X(NB1) INERTIA MATRIX XDIC WITH (NB1)X1 MATRIX TEM	C1845000
C	NOTE THAT XDIC STORED IN TRIANGULAR FORM PUT RESULT IN TEM	C1845100
	506C CONTINUE	C1845200
	200C CONTINUE	C1845300
	DO 24 K=1,NB1	C1845400
	DO 29 I=1,3	C1845500
	29 XCD(I,K) = 0	C1845600
	IF(LEQU)PRINT 215, K	C1845700
	IF(K.NE.NB1) GO TO 30	C1845800
	CALL UNPAC(ST1,NST1,SR)	C1845900
	NST2 = C	C1846000
	GC TO 31	C1846100
	30 CALL UNPAC(ST1,NST1,SI(K))	C1846200
	CALL UNPAC(ST2,NST2,SK(K-1))	C1846300
	ST2(NST2) = NB1	C1846400
	31 CONTINUE	C1846500
C		C1846600
C	START MATRIX MULTIPLICATION ROW K TO RIGHT OF DIAGONAL	C1846700
	IF(LEQU)PRINT 217,K,(ST2(I),I=1,NST2)	C1846800
	IF(NST2.EQ.0) GO TO 5059	C1846900
	DO 26 II=1,NST2	C1847000
C	ELEMENT K=I DELETE IT WAS TAKEN CARE OF ABOVE	C1847100
	I = ST2(II)	C1847200
C	CHECK FOR MULTIPLICATION BY ZERO	C1847300
C	RECALL FROM RATE THAT DCMC(NB1) = TEM1(NB1) = 0	C1847400
	IF(I.EQ.NB1) GO TO 26	C1847500
	IF(.NOT.RBLO(I)) GO TO 26	C1847600
	KI = KT1(NB1,K,I)	C1847700
	IF(LEQU)PRINT 220, K,K,KI,I,K,K,I,I	C1847800
C	NOTE THAT I,K TENSOR IS THE TRANSPOSE OF THE K,I TENSOR	C1847900
C	THIS HOWEVER DOES NOT INSURE THAT I,K IS SYMMETRIC, IN FACT	C1848000
C	IN GENERAL IT WILL NOT BE SO HENCE DYTOV IS USED	C1848100
	CALL DYTCV (XDIC(I,I,KI),TEM(I,I),TEM1)	C1848200
	CALL VECADD(XCD(I,K),TEM1,XCD(I,K))	C1848300
	26 CONTINUE	C1848400
	5059 CONTINUE	C1848500
	IF(LEQU)PRINT 228	C1848600
	IF(NST1.EQ.0)GO TO 506C	C1848700
	IF(LEQU)PRINT 216, K,(ST1(I),I=1,NST1)	C1848800
	DO 25 II=1,NST1	C1848900
	I = ST1(II)	C1849000
	IF(.NOT.RBLO(I)) GO TO 25	C1849100
C	BODY I RIGID BODY	C1849200
	KI = KT1(NB1,K,I)	C1849300
	IF(LEQU)PRINT 218, K,K,KI,I,K,K,I,I	C1849400
	CALL DYDCTV(XDIC(I,I,KI),TEM(I,I),TEM1)	C1849500
	CALL VECADD(XCD(I,K),TEM1,XCD(I,K))	C1849600
	25 CONTINUE	C1849700
	506C CONTINUE	C1849800
	IF(LEQU)PRINT 228	C1849900
C	FINISHED MATRIX MULTIPLICATION ROW K OVER TO DIAGONAL	C1850000

C		C1E50100
C	ADD UP GYRUSCOPIC, XGD AND EXTERNAL TORQUE ON NEST K-1	C1E50200
	IF (LEQU) PRINT 229, K, (ETC(I,K), I=1,3)	C1E50300
	IF (LEQU) PRINT 230, K, (XQC(I,K), I=1,3)	C1E50400
	IF (LEQU) PRINT 231, K, (PHI(I,K), I=1,3)	C1E50500
	CC 27 I=1,3	C1E50600
	27 ETC(I,K) = ETC(I,K) - XQC(I,K) + PHI(I,K)	C1E50700
	IF (LEQU) PRINT 232, K, K, K, K, (ETC(I,K), I=1,3)	C1E50800
	24 CONTINUE	C1E50900
	IF (.NOT. LEQU) GO TO 300C	C1E51000
	PRINT 227	C1E51100
	CC 15 K=1, NB1	C1E51200
	15 PRINT 229, K, (ETC(I,K), I=1,3)	C1E51300
	PRINT 227	C1E51400
	300C CONTINUE	C1E51500
C		C1E51600
C	COMPLETE (NFER) X1 COLUMN MATRIX OF TORQUE COMPONENTS ALONG	C1E51700
C	FREE COORDINATE AXES	C1E51800
	DO 2E K=1, NB1	C1E51900
	MBEG=SQF(K)	C1E52000
	MTERM=MEEG+2-PCUN(K)	C1E52100
	IF (MEEG.GT.MTERM) GO TO 5061	C1E52200
	DO 2E M=MBEG, MTERM	C1E52300
	CALL VECDOT(QFC(1,M), ETC(1,K), ETM(M))	C1E52400
	IF (LEQU) PRINT 224, M, M, K	C1E52500
	5061 CONTINUE	C1E52600
	2E CONTINUE	C1E52700
	IF (.NOT. LEQU) RETURN	C1E52800
	PRINT 227	C1E52900
	MTERM=NFER+NMV+NMCD5	C1E53000
	DO 1E M=1, MTERM	C1E53100
	1E PRINT 223, M, ETM(M)	C1E53200
C		C1E53300
C	REDUCTION OF VECTOR CYCLIC EQUATIONS TO SCALAR EQUATIONS COMPLETE	C1E53400
C		C1E53500
C	SUM OVER N=1, NFER+NMV+NMCD5	C1E53600
C		C1E53700
C	XMN(M,N)*TFADD(N) = ETM(M)	C1E53800
C		C1E53900
	200 FORMAT (4(2X, A4, I2, ', ', I2, ') = ', D15.8))	C1E54000
	201 FORMAT (//, ' LABELS OF VARIABLE SPEED MOMENTUM WHEELS ', I0I5, /)	C1E54100
	202 FORMAT (' BODY ', I2, ' TIED TO SYSTEM AT RIGID HINGE ')	C1E54200
	203 FORMAT (/, ' XMN(', I2, ', ', I2, ') = PLM(', I2, ') = ', D17.8)	C1E54300
	204 FORMAT (60X, ' ETM(', I2, ') = ', D17.8)	C1E54400
	205 FORMAT (' NON-ZERO COLUMNS IN ROW ', I2, ' OF XDIC OVER TO DIAGONAL	C1E54500
	*ELEMENT ARE ', I1I5)	C1E54600
	206 FORMAT (' XMN(', I2, ', ', I2, ') = QFC(', I2, ') . (XDIC(', I2, ') . QFC(', I2,	C1E54700
	*') = QFC(', I2, ') . (XDIC(', I2, ', ', I2, ') . QFC(', I2, ') = ', D17.8)	C1E54800
	207 FORMAT (60X, ' ETM(', I2, ') = ETM(', I2, ') - PLM(', I2, ') * HMC(', I2,	C1E54900
	*') . DCMC(', I2, ')')	C1E55000
	208 FORMAT (' TEM(', I2, ') = DCMC(', I2, ') = ', D17.8)	C1E55100
	209 FORMAT (' XMN(', I2, ', ', I2, ') = PLM(', I2, ') * HMC(', I2, ') . QFC(',	C1E55200
	*I2, ') = ', D17.8)	C1E55300
	210 FORMAT (60X, ' ETM(', I2, ') = ETM(', I2, ') + CLM(', I2, ') = ', D17.8)	C1E55400
	211 FORMAT (30X, ' CACC(', I2, ') = XMC(', I2, ') * CAG(', I2, ')')	C1E55500
	212 FORMAT (30X, ' TEM2 = FCMC(', I2, ') X (FCMC(', I2, ') X CACC(', I2, ')'	C1E55600
	*)	C1E55700
	213 FORMAT (30X, ' TEM2 = TEM2 + FJMC(', I2, ') X (FCMC(', I2, ') X CEC(',	C1E55800
	*I2, ')')	C1E55900
	214 FORMAT (' XMN = 0.0 ', ///)	C1E56000

```

215 FORMAT (' XDG(' ,I2,' ) = 0 ') C1856100
216 FORMAT (' ROW ',I2,' OF XDIC LEFT OF DIAGONAL USE ELEMENTS IN COL C1856200
      *MNS ',I0I5) C1856300
217 FORMAT (' ROW ',I2,' OF XDIC RIGHT OF DIAGONAL USE ELEMENTS IN COL C1856400
      *LMNS ',I0I5) C1856500
218 FORMAT (' XQD(' ,I2,' ) = XGD(' ,I2,' ) + XCIC(' ,I2,' ) . TEM(' ,I2,' ) C1856600
      * = ' ,XGD(' ,I2,' ) + XCIC(' ,I2,' ,I2,' ) . TEM(' ,I2,' ) ') C1856700
219 FORMAT (' XMN(' ,I2,' ,I2,' ) = QFC(' ,I2,' ) . (XMAS(' ,I2,' )*FLAC(' C1856800
      * ,I2,' ) = ' ,D12.5) C1856900
220 FORMAT (' XQD(' ,I2,' ) = XGD(' ,I2,' ) + XDIC(' ,I2,' )**T . TEM(' ,I2,' C1857000
      *) = XDU(' ,I2,' ) + XDIC(' ,I2,' ,I2,' ) . TEM(' ,I2,' ) ') C1857100
222 FORMAT (30X,' TEM2 = XMAS(' ,I2,' )*TEM2 = ' ,17X,3D12.5) C1857200
223 FORMAT (' ETM(' ,I2,' ) = ' ,3D17.8) C1857300
224 FORMAT (' ETM(' ,I2,' ) = GFC(' ,I2,' ) . ETC(' ,I2,' ) ') C1857400
226 FORMAT (' SUBROUTINE GFCUT ENTERED ',2(/)) C1857500
227 FORMAT (3(/)) C1857600
228 FORMAT (' ') C1857700
229 FORMAT (' ETC(' ,I2,' ) = ' ,3D17.8) C1857800
230 FORMAT (' XQD(' ,I2,' ) = ' ,3D17.8) C1857900
231 FORMAT (' PHI(' ,I2,' ) = ' ,3D17.8) C1858000
232 FORMAT (' ETC(' ,I2,' ) = ETC(' ,I2,' ) - XJD(' ,I2,' ) + PHI(' ,I2,' ) = C1858100
      * ' ,3D17.8' ,//) C1858200
233 FORMAT (' ETM(' ,I2,' ) = ' ,D17.8) C1858300
234 FORMAT (///,40X,' BODY ',I2,' IS A FLEXIBLE BODY IN NEST ',I2,/) C1858400
235 FORMAT (/,' FTEM(' ,I2,' ,I2,' ,I2,' ) = XMAS(' ,I2,' )*CAM(' ,I2,' , C1858500
      * ' ,I2,' ) X FLAC(' ,I2,' ) + FLQC(' ,I2,' ) = ' ,3D12.5) C1858600
236 FORMAT (' XMN(' ,I2,' ,I2,' ) = QFC(' ,I2,' ) . FTEM(' ,I2,' ,I2,' ,I C1858700
      * 2,' ) = ' ,6X,D12.5) C1858800
237 FORMAT (' XMN(' ,I2,' ,I2,' ) = XMAS(' ,I2,' ) = ' ,20X,D12.5) C1858900
238 FORMAT (/,30X,' ETM(' ,I2,' ) = ETM(' ,I2,' ) - FTEM(' ,I2,' ,I2,' , C1859000
      * I2,' ) . DCMC(' ,I2,' ) = ' ,38X,D12.5) C1859100
239 FORMAT (30X,' ETM(' ,I2,' ) = ETM(' ,I2,' ) - XMAS(' ,I2,' )*(2.*ZETA(' C1859200
      * ,I2,' )*FLOM(' ,I2,' )=THAD(' ,I2,' ) + FLJM(' ,I2,' )**2*THA(' ,I2,' ) = ' , C1859300
      * C12.5) C1859400
240 FORMAT (30X,' ETM(' ,I2,' ) = ETM(' ,I2,' ) + FCMC(' ,I2,' ) . FLHC(' ,I2,' C1859500
      *) . FCMC(' ,I2,' ) - TEM2 . FLAC(' ,I2,' ) = ' ,19X,D12.5) C1859600
241 FORMAT (30X,' TEM2 = 0 ') C1859700
242 FORMAT (30X,' FCUP(' ,I2,' ) = 0 KCUP(' ,I2,' ) = 0') C1859800
245 FORMAT (' ') C1859900
246 FORMAT (50X,3D12.5) C1860000
247 FORMAT (10X,' FCUP(' ,I2,' ) = FCUP(' ,I2,' ) + THA(' ,I2,' )*FCF(' ,I2,' C1860100
      *) = ' ,3D12.5) C1860200
248 FORMAT (10X,' KCUP(' ,I2,' ) = KCUP(' ,I2,' ) + THAD(' ,I2,' )*FCK(' ,I2,' C1860300
      *) = ' ,3D12.5) C1860400
249 FORMAT (30X,' ETM(' ,I2,' ) = ETM(' ,I2,' ) + FCMC(' ,I2,' ) . FCUP1 . FCMC(' C1860500
      * ' ,I2,' ) - 2.0*FOMC(' ,I2,' ) . KCUP1 = ' ,D12.5) C1860600
250 FORMAT (71X,3D12.5) C1860700
251 FORMAT (//,45X,' MODAL CROSS COUPLING SIGNIFICANT FOR BODY ',I2) C1860800
252 FORMAT (/,30X,' ETM(' ,I2,' ) = 0 ') C1860900
253 FORMAT (30X,' FLHC(' ,I2,' ) = XMC(' ,I2,' )*FLH(' ,I2,' )*XMC(' ,I2,' C1861000
      *)**T = ' ,3D12.5) C1861100
254 FORMAT (48X,3D12.5) C1861200
255 FORMAT (10X,' KCUP1 = XMC(' ,I2,' )*KCUP(' ,I2,' ) = ' ,3D12.5) C1861300
256 FORMAT (/,' EQUATION OF MOTION FOR MODE ',I2,' OF BODY ',I2,' (MOD C1861400
      *E NUMBER ',I2,' ) HAS CROSS COUPLING FROM MODES ',I0I4) C1861500
257 FORMAT (10X,' FCUP1 = XMC(' ,I2,' )*FCUP(' ,I2,' )*XMC(' ,I2,' )**T = ' , C1861600
      * 3D12.5) C1861700
C C1861800
C C1861900
RETURN C1862000

```

```

C
C SUERCUTINE DCT C1900000
C USED TO DEFINE DIFFERENTIAL EQUATIONS WHICH MUST BE INTEGRATED C1900100
C TO DEFINE DIRECTION COSINE MATRICES C1900200
C C1900300
C C1900400
C SD = ALL CONTIGUOUS PAIRS OF BODIES (K,JCON(K)) HAVING C1900500
C SIGNIFICANT RELATIVE MOTION. DELETION OF K FROM SD C1900600
C IMPLIES TRANSFORMATION MATRIX OF BODY K TO JCON(K) C1900700
C IS CONSTANT IN TIME, OR DEFINED BY SMALL ANGLE ASSUMPTION C1900800
C OR,EULER ANGLE TECHNIQUES C1900900
C C1901000
C C1901100
C C1901200
C IMPLICIT REAL*8(A-F,O-Z,*) C1901300
C LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLU, LEQU, LINIT(1) C1901400
C LOGICAL LRUNGE, LTRNSI, LVDIV, LEQUIV, LTRAN, C1901500
C * LTRANV, LRATE, LXDY, LETA, LTORQU, C1901600
C * LQFDOT, LDCT, LANGLE, LSETUP, LSIMQ C1901700
C C1901800
C C1901900
C C1902000
C INTEGER C1902100
C * AWRK, CT1, CT2, CT3, CT4, CT5, FCON, PCON, C1902200
C * SCNDUM, SCN, SCRUM, SCR, SFKUM, SFK, SFR, SG, C1902300
C * SI, SIG, SIXDUM, SIX, SKDUM, SK, SL, SLK, C1902400
C * SMA, SMCDUM, SMC, SMV, SOK, SPIDUM, SPI, SQF, C1902500
C * SQL, SR, SSCN, SSIX, SVA, SVB, SVD, SVI, C1902600
C * SVM, SVP, SVQ, SXN, SXT, TCRQ, SMAL, SEU, C1902700
C * SC, SCG, NFLXB, SFLX, SFXM, NMODS, SFCC, SCC, C1902800
C * IINIT(1), IZINIT(1), SD C1902900
C C1903000
C C1903100
C REAL*8 C1903200
C * ANGC (33), CNF (3,10), ETIC (3,10), ETMC (3,10), C1903300
C * FLO (3,20), FLE (3,3,20), FLH (3,3,20), C1903400
C * TFADD (33), YMCD (3,2,11), RINIT (1), RZINIT(1) C1903500
C C1903600
C COMMON /LDEBUG/ LRUNGE, LTRNSI, LVDIV, LEQUIV, LTRAN, C1903700
C * LTRANV, LRATE, LXDY, LETA, LTORQU, C1903800
C * LQFDOT, LDCT, LANGLE, LSETUP, LSIMQ C1903900
C C1904000
C C1904100
C COMMON /LDGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RBLU(10) C1904200
C C1904300
C C1904400
C COMMON /INTG/ AWRK(200), C1904500
C * CT1, CT2, CT3, CT4, C1904600
C * CT5, FCON(33), JCON(10), LCON(22), C1904700
C * MC (10), NBI, NBD, NCTC, C1904800
C * NFER, NFKC, NFRC, NLOR, C1904900
C * NV, NMO, NMCA, NSVP, C1905000
C * NSVC, FCON (11), SD, SFR (33), C1905100
C * SC, SI (5), SIG, SL, C1905200
C * SLK (3), SMA (10), SOK (11), SQF (11), C1905300

```

```

* SCL (11) , SMV , SR , SSCN , C1905400
* SEIX , SVA , SVB , SVD , C1905500
* SVI , SVM , SVP (22) , SVQ (33) , C1905600
* SXM (3,10) , SXT , TDFQ (97) , SMAL , C1905700
* SEU , NTQ , SC (33) , SCG , C1905800
* NFLXB , SFLX , SFXM (10) , NMGDS , C1905900
* SFCC , SCC (10) , C1906000
C
C
CCMMCN /INTGZ/
* SCNDLM , SCN (9) , SCRDM , SCR (9) , C1906100
* SFKDLN , SFK (9) , SIXDUM , SIX (9) , C1906200
* SKDUM , SK (9) , SPIDUM , SPI (9) , C1906300
* SMCCUM , SMC (9) , C1906400
C
C
CCMMCN /REAL/
* CA (3,10) , CAC (3,10) , CLM (10) , COMC (3,11) , C1906500
* DCMC (3,11) , ETC (3,11) , ETM (33) , FCMC (3,11) , C1906600
* GAM (3,66) , F , HM (3,10) , FMC (3,10) , C1906700
* HMCN (10) , PHI (3,11) , PLM (10) , QF (3,33) , C1906800
* QFC (3,33) , GL (3,22) , QLC (3,22) , RCMC (3,11) , C1906900
* T , THA (33) , THAD (33) , C1907000
* THADW (10) , THAW (10) , XDIC (3,3,66) , XI (3,3,10) , C1907100
* XIC (3,3,10) , XMAS (10) , XMN (3,3,33) , XMT (3,3,10) , C1907200
* TUG (33) , FLA (3,20) , FLB (3,20) , FLC (3,20) , C1907300
* FLD (3,3,20) , FLJ (3,3,20) , CAU (3,10) , XID (3,3,10) , C1907400
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLOC (3,20) , C1907500
* FLCN (20) , ZETA (20) , FCF (3,3,40) , FCK (3,40) , C1907600
* TIMEND , C1907700
C
C
COMMON /REALZ/
* CBDLM (1,3) , CE (3,10) , CBDUM(1,3) , CBC (3,10) , C1907800
* XMCDDM(1,1,9) , XMC (3,3,10) , CBN(3) , C1907900
C
C
EQUIVALENCE (ETM(1),THAD(1)) , (XMN(1,1),ANGD(1)) , C1908000
* (XMN(1,3),YMCD(1,1,1)) , (XMN(1,6),CNF(1,1)) , C1908100
* (XMN(1,8),ETIC(1,1)) , (XMN(1,10),ETMC(1,1)) , C1908200
* (FLB(1,1),FLQ(1,1)) , (FLE(1,1,1),FLD(1,1,1)) , C1908300
* (FLH(1,1,1),FLJ(1,1,1)) , C1908400
* (FGI,LINIT(1)) , (CA(1,1),RINIT(1)) , C1908500
* (CBDUM(1,1),FZINIT(1)) , (AWORK(1),IINIT(1)) , C1908600
* (SCNDLM,IZINIT(1)) , C1908700
C
C
INTEGER SET(10) , C1908800
EQUIVALENCE (LDCT,LEQU) , C1908900
C
C
C
C
IF(.NOT. LEQU) GO TO 1000 , C1909000
008000 PRINT 100 , C1909100
J0= 0 , C1909200
J1= 1 , C1909300
J2= 2 , C1909400
J3= 3 , C1909500
1000 CONTINUE , C1909600
N = C , C1909700
C1909800
C1909900
C1910000
C1910100
C1910200
C1910300
C1910400
C1910500
C1910600
C1910700
C1910800
C1910900
C1911000
C1911100
C1911200
C1911300

```

```

IF(LEQU) PRINT 104,INERF                                C1911400
CALL UNPAC(SET,NSET,SD)                                C1911500
IF(NSET.EQ.0) GO TO 5062                                C1911600
IF(LEQU) PRINT 103, SD,(SET(I),I=1,NSET)              C1911700
IF(INERF.UR.SET(NSET).NE.1) GO TO 2                   C1911800
C TRANS BODY 1 TO BODY 1 COMPUTING FRAME NOT NEEDED   C1911900
NSET = NSET - 1                                        C1912000
C ACCCLNT FOR INERTIAL TO BODY 1 TRANSFORMATION      C1912100
C ANGLAR VELOCITY INERTIAL TO BODY 1                C1912200
C = - ANGULAR VELOCITY BODY 1 TO INERTIAL           C1912300
C = - FOMC(I,1), I=1,2,3                             C1912400
N = N+1                                                C1912500
DO 4 I=1,2                                             C1912600
C GET *INCS SIGN IN EY REVERSEING VECTOR CROSS PRODUCT C1912700
CALL VECRCS (XMC(1,I,0),FOMC(1,I),YMC(1,I,N))        C1912800
IF(LEQU) PRINT 106, I,N,I,J,C,(YMC(J,I,N),J=1,3)     C1912900
4 CONTINUE                                             C1913000
IF(LEQU) PRINT 105                                    C1913100
2 IF(NSET.EQ.0) GO TO 5062                             C1913200
DO 3 KK=1,NSET                                        C1913300
N = N+1                                                C1913400
K = SET(KK)                                           C1913500
DO 6 I=1,2                                             C1913600
CALL VECRCS (CCMC(1,K),XMC(1,I,K),YMC(1,I,N))        C1913700
IF(LEQU) PRINT 107, I,N,K,I,K,(YMC(J,I,N),J=1,3)     C1913800
6 CONTINUE                                             C1913900
IF(LEQU) PRINT 105                                    C1914000
3 CONTINUE                                             C1914100
5062 CONTINUE                                         C1914200
100 FORMAT ('1 SUBROUTINE COT ENTERED ')              C1914300
103 FORMAT (' SD =',Z8,' SET ELEMENTS ',I0I5)        C1914400
104 FORMAT (' INERF = ',L10)                          C1914500
105 FORMAT (' ')                                       C1914600
106 FORMAT (' YMC(',I2,',',I2,',) = XMC(',I2,',',I2,',) X FOMC( 1) =',J0 C1914700
*,3C17.8)                                             C1914800
107 FORMAT (' YMC(',I2,',',I2,',) = CCMC(',I2,',) X XMC(',I2,',',I2,',) C1914900
*, = ',3D17.8)                                       C1915000
RETURN                                                C1915100
END                                                    C1915200

```

```

C SUERCUTINE ANGLE                                    C2000000
C USED TO SET UP DIFFERENTIAL EQUATIONS WHICH DEFINE C2000100
C ANGULAR DISPLACEMENT ABOUT FREE COORDINATES      C2000200
C RECALL                                              C2000300
C SFR(I) = FREE COORDINATE AXES ABOUT WHICH ANGLE TO C2000400
C BE COMPLETED                                     C2000500
C                                                    C2000600
C                                                    C2000700
C                                                    C2000800
C                                                    C2000900
C IMPLICIT REAL*8(A-H,O-Z,*)                          C2001000
C LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLO, LEQU, LINIT(1) C2001100
C LOGICAL LRUNGE, LTRNSI, LVDIV, LEQJIV, LTRAN,      C2001200
* LTRANV, LRATE, LXDY, LETA, LTORQU,                C2001300
* LGFDDT, LDCT, LANGLE, LSETUP, LSING                C2001400
C                                                    C2001500

```



									C2C01600
									C2C01700
									C2C01800
									C2C01900
									C2C02000
									C2C02100
									C2C02200
									C2C02300
									C2C02400
									C2C02500
									C2C02600
									C2C02700
									C2C02800
									C2C02900
									C2C03000
									C2C03100
									C2C03200
									C2C03300
									C2C03400
									C2C03500
									C2C03600
									C2C03700
									C2C03800
									C2C03900
									C2C04000
									C2C04100
									C2C04200
									C2C04300
									C2C04400
									C2C04500
									C2C04600
									C2C04700
									C2C04800
									C2C04900
									C2C05000
									C2C05100
									C2C05200
									C2C05300
									C2C05400
									C2C05500
									C2C05600
									C2C05700
									C2C05800
									C2C05900
									C2C06000
									C2C06100
									C2C06200
									C2C06300
									C2C06400
									C2C06500
									C2C06600
									C2C06700
									C2C06800
									C2C06900
									C2C07000
									C2C07100
									C2C07200
									C2C07300
									C2C07400
									C2C07500

```

* TUG (33) , FLA (3,20) , FLB (3,20) , FLC (3,20) , C2C07600
* FLD (3,3,20) , FLJ (3,3,20) , CAD (3,10) , XIU (3,3,10) , C2C07700
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLCC (3,20) , C2C07800
* FLCN (20) , ZETA (20) , FCF (3,3,40) , FCK (3,40) , C2C07900
* TIMEND C2C08000
C C2C08100
C C2C08200
C C2C08300
C CMMCN /REALZ/
* CEDLM (1,3) , CE (3,10) , CBCDUM(1,3) , CBC (3,10) , C2C08400
* XMCLM(1,1,9) , XMC (3,3,10) , CBN(3) C2C08500
C C2C08600
C C2C08700
C EQUIVALENCE (ETM(1),THAD(1)) ,(XMN(1,1),ANGD(1)) , C2C08800
* (XMN(1,3),YMCD(1,1,1)) ,(XMN(1,6),CNF(1,1)) , C2C08900
* (XMN(1,8),ETIC(1,1)) ,(XMN(1,10),ETMC(1,1)) , C2C09000
* (FLB(1,1),FLQ(1,1)) ,(FLE(1,1,1),FLD(1,1,1)) , C2C09100
* (FLH(1,1,1),FLJ(1,1,1)) , C2C09200
* (FSI,LINIT(1)) ,(CA(1,1),RINIT(1)) , C2C09300
* (CBDUM(1,1),RZINIT(1)) ,(AWURK(1),IINIT(1)) , C2C09400
* (SCNDLM,IZINIT(1)) C2C09500
C C2C09600
C EQUIVALENCE (LANGE,LEQU) C2C09700
C INTEGER S1(33),S2(33) C2C09800
C C2C09900
C C2C10000
C C2C10100
C C2C10200
C C2C10300
C IF(LEQU) PRINT 100 C2C10400
C C2C10500
C C2C10600
C PROCEED SEQUENTIALLY NOW DEFINING ONLY THOSE EQUATIONS CALLED FOR C2C10600
C TAKING SPECIAL NOTE OF 3-AXIS GIMBALS BETWEEN RIGID BODIES C2C10700
C IF(LEQU) PRINT 101, NFRC,(SFR(I),I=1,NFRC) C2C10800
C K = 1 C2C10900
C M = 1 C2C11000
C DO 5 N=1,NB1 C2C11100
C IF(N.EQ.1) GO TO 9 C2C11200
C M = M+3-FCUN(N-1) C2C11300
C 5 CONTINUE C2C11400
C MT = HIGHEST MAGNITUDE INDICE AT HINGE POINT N-1 C2C11500
C MT = M+2-PCUN(N) C2C11600
C 7 IF(SFR(K).GT.MT.OR.K.GT.NFRC) GO TO 5 C2C11700
C IF(N.EQ.NB1) GO TO 4 C2C11800
C IF(RELN(N).AND.PCCN(N).EQ.0) GO TO 6 C2C11900
C IF(RELN(N).AND.PCCN(N).EQ.1) GO TO 1 C2C12000
C 4 CONTINUE C2C12100
C IF(SFR(K).EQ.0) GO TO 5 C2C12200
C ANGD(K) = THAD(SFR(K)) C2C12300
C IF(LEQU) KK=SFR(K) C2C12400
C IF(LEQU) PRINT 102, K, KK, ANGD(K) C2C12500
C K = K+1 C2C12600
C GO TO 7 C2C12700
C C2C12800
C C2C12900
C SPECIAL CASE CHECK FOR SKEWED TWO AXIS GIMBAL C2C12900
C 1 CALL VECDOT(QFC(1,M),QFC(1,N+1),C) C2C13000
C IF(C.EQ.C.DO) GO TO 4 C2C13100
C CALL VECDOT(QFC(1,M),RCMC(1,N),A1) C2C13200
C CALL VECDOT(QFC(1,M+1),RCMC(1,N),A3) C2C13300
C ANGD(K) = (A1 - C*A3)/(1.DO - C**2) C2C13400
C ANGD(K+1) = (A3 - C*A1)/(1.DO - C**2) C2C13500

```

```

IF (.NOT. LEQU) GO TO 1001
M1 = M+1
K1 = K+1
FRINT 103, M,N,A1
FRINT 105, M1,N,A3
FRINT 106, M,M1,C
FRINT 107, K,ANGD(K)
FRINT 109, K1,ANGD(K1)
1001 CONTINUE
K = K+2
GO TO 5
C
C CONTINUE
C SPECIAL CASE THREE AXIS GIMBAL - NOTE LOGIC ASSUMES THAT
C SFR(K),SFR(K+1),SFR(K+2)
C ARE THE THREE GIMBAL AXES
CALL VECDOT(QFC(1,M),RCMC(1,N),A1)
CALL VECDOT(QFC(1,M+1),RCMC(1,N),A2)
CALL VECDOT(QFC(1,M+2),RCMC(1,N),A3)
CALL VECDOT(QFC(1,M),QFC(1,M+2),C)
ANGD(K) = (A1 - C*A3)/(1 - C**2)
ANGD(K+1) = A2
ANGD(K+2) = (A3 - C*A1)/(1 - C**2)
IF (.NOT. LEQU) GO TO 100C
M1 = M+1
M2 = M+2
K1 = K+1
K2 = K+2
FRINT 103, M,N,A1
FRINT 104, M1,N,A2
FRINT 105, M2,N,A3
FRINT 106, M,M2,C
FRINT 107, K,ANGD(K)
FRINT 108, K1,ANGD(K+1)
FRINT 109, K2,ANGD(K+2)
100C CONTINUE
K = K+3
5 CONTINUE
C
100 FORMAT ('1 SUBROUTINE ANGLE ENTERED ')
101 FORMAT ('110, ' ELEMENTS IN ARRAY SFR, THEY ARE ',3J13)
102 FORMAT (' ' ANGD(' ,I2,') = THAD(' ,I2,') = ',D17.8)
103 FORMAT (' ' A1 = QFC(' ,I2,') . RCMC(' ,I2,') = ',D17.8)
104 FORMAT (' ' A2 = QFC(' ,I2,') . RCMC(' ,I2,') = ',D17.8)
105 FORMAT (' ' A3 = QFC(' ,I2,') . RCMC(' ,I2,') = ',D17.8)
106 FORMAT (' ' C = QFC(' ,I2,') . QFC(' ,I2,') = ',D17.8)
107 FORMAT (' ' ANGD(' ,I2,') = (A1-C*A3)/(1-C**2) = ',D17.8)
108 FORMAT (' ' ANGD(' ,I2,') = A2 = ',D17.8)
109 FORMAT (' ' ANGD(' ,I2,') = (A3-C*A1)/(1-C**2) = ',D17.8)
110 FORMAT ('110, ' ELEMENTS IN ARRAY SMA, THEY ARE ',10I3)
RETURN
END
C
SUBROUTINE SETUP(Y,YD,NEG, SORT)
IF SORT = .TRUE. SORT CUT INTEGRATED QUANTITIES

```

```

C      = .FALSE. SET UP ONE DIMENSIONAL ARRAY OF EQUATIONS TOC2100300
C      BE INTEGRATED                                C2100400
C                                                    C2100500
C                                                    C2100600
C      IMPLICIT REAL*8(A-H,O-Z,1)                   C2100700
C                                                    C2100800
C                                                    C2100900
C      LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLO, LEQU, LINIT(1) C2101000
C      LOGICAL   LRLNGE, LTRNSI, LVDIV, LEQUIV, LTRAN,   C2101100
C      *         LTRANV, LRATE, LXDY, LETA, LTORQU,      C2101200
C      *         LQFDOT, LDCT, LANGLE, LSETUP, LSIMQ    C2101300
C                                                    C2101400
C                                                    C2101500
C      INTEGER                                       C2101600
C      * AWRK, CT1, CT2, CT3, CT4, CT5, FCCN, PCCN,      C2101700
C      * SCNDUM, SCN, SCRDM, SCR, SFKDUM, SFK, SFR, SG,   C2101800
C      * SI, SIG, SIXDUM, SIX, SKDUM, SK, SL, SLK,        C2101900
C      * SMA, SMCDUM, SMC, SMV, SUK, SPIDUM, SPI, SQF,    C2102000
C      * SGL, SR, SSCN, SSIX, SVA, SVE, SVD, SVI,        C2102100
C      * SVM, SVP, SVQ, SXM, SXT, TORQ, SMAL, SEU,       C2102200
C      * SC, SCG, NFLXB, SFLX, SFXM, NMODS, SFCC, SCC,   C2102300
C      * IINIT(1), IZINIT(1), SD                      C2102400
C                                                    C2102500
C                                                    C2102600
C      REAL*8                                         C2102700
C      * ANGE (33), CNF (3,10), ETIC (3,10), ETMC (3,10), C2102800
C      * FLO (3,20), FLE (3,3,20), FLH (3,3,20),        C2102900
C      * THADD (33), YMCD (3,2,11), RINIT (1), RZINIT(1) C2103000
C                                                    C2103100
C                                                    C2103200
C      COMMON /LDEBUG/ LRLNGE, LTRNSI, LVDIV, LEQUIV, LTRAN, C2103300
C      *         LTRANV, LRATE, LXDY, LETA, LTORQU,      C2103400
C      *         LQFDOT, LDCT, LANGLE, LSETUP, LSIMQ    C2103500
C                                                    C2103600
C                                                    C2103700
C      COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RBLO(10) C2103800
C                                                    C2103900
C                                                    C2104000
C      COMMON /INTG/ AWRK(184),ST1,N,M,MM,NST1,J,I,      C2104100
C      * CT1, CT2, CT3, CT4,                            C2104200
C      * CT5, FCCN (33), JCCN (10), LCCN (22),          C2104300
C      * MC (10), NB1, NBOD, NCTC,                      C2104400
C      * NFER, NFKC, NFRC, NLOR,                       C2104500
C      * NMV, NMO, NMUA, NSVP,                          C2104600
C      * NSVG, FCCN (11), SD, SFR (33),                C2104700
C      * SG, SI (55), SIG, SL,                         C2104800
C      * SLK (33), SMA (10), SUK (11), SQF (11),        C2104900
C      * SGL (11), SMV, SR, SSCN,                      C2105000
C      * SEI, SVA, SVB, SVD,                           C2105100
C      * SVI, SVM, SVP (22), SVQ (33),                 C2105200
C      * SXM (3,10), SXT, TORQ (97), SMAL,             C2105300
C      * SEU, NTQ, SC (33), SCG,                       C2105400
C      * NFLXB, SFLX, SFXM (10), NMODS,               C2105500
C      * SFCC, SCC (10)                                C2105600
C                                                    C2105700
C                                                    C2105800
C      COMMON /INTG2/                                   C2105900
C      * SCNDUM, SCN (9), SCRDM, SCR (9),              C2106000
C      * SFKDUM, SFK (9), SIXDUM, SIX (9),             C2106100
C      * SKDUM, SK (9), SPIDUM, SPI (9),              C2106200

```

```

      * SMCCDM      , SMC      (9)
C
C
      CGMMCN /REAL/
      * CA      (3,10) , CAC      (3,10) , CLM      (10) , CCMC      (3,11) ,
      * DCMC      (3,11) , ETC      (3,11) , ETM      (33) , FCMC      (3,11) ,
      * GAM      (3,66) , F      , HM      (3,10) , HMC      (3,10) ,
      * HVMC      (10) , PHI      (3,11) , PLM      (10) , QF      (3,33) ,
      * QFC      (3,33) , QL      (3,22) , QLC      (3,22) , RCMC      (3,11) ,
      * T      , THA      (33) , THAD      (33) ,
      * THADW      (10) , THAW      (10) , XDIC      (3,3,66) , XI      (3,3,10) ,
      * XIC      (3,3,10) , XMAS      (10) , XMN      (3,3,33) , XMT      (3,3,10) ,
      * TLG      (33) , FLA      (3,20) , FLB      (3,20) , FLC      (3,20) ,
      * FLD      (3,3,20) , FLJ      (3,3,20) , CAG      (3,10) , XIO      (3,3,10) ,
      * FLIRC      (3,10) , FLCRC      (3,10) , FLAC      (3,20) , FLOC      (3,20) ,
      * FLCN      (20) , ZETA      (20) , FCF      (3,3,40) , FCK      (3,40) ,
      * TIMEND
C
C
      CGMMCN /REALZ/
      * CEDLM      (1,3) , CE      (3,10) , CBCDUM      (1,3) , CBC      (3,10) ,
      * XMCCDM      (1,1,9) , XMC      (3,3,10) , CBN      (3)
C
C
      EQUIVALENCE (ETM(1),THADD(1)) , (XMN(1,1),ANGD(1)) ,
      * (XMN(1,3),YMCD(1,1,1)) , (XMN(1,6),CNF(1,1)) ,
      * (XMN(1,8),ETIC(1,1)) , (XMN(1,10),ETMC(1,1)) ,
      * (FLB(1,1),FLG(1,1)) , (FLE(1,1,1),FLD(1,1,1)) ,
      * (FLH(1,1,1),FLJ(1,1,1)) ,
      * (FJ1,LINIT(1)) , (CA(1,1),RINIT(1)) ,
      * (CBDUM(1,1),RZINIT(1)) , (AWORK(1),IINIT(1)) ,
      * (SCNDLM,IZINIT(1))
C
      INTEGER STI(10)
      LOGICAL SORT
      DIMENSION Y(NEQ),YC(NEQ)
      EQUIVALENCE (LSETUP,LEQU)
C
      IF(LEQU) PRINT 100
C
      ANGULAR OR LINEAR RATE RELATIVE TO FREE COORDINATE AXES
      DO 1 N=1,NFER
      IF(SCRT) GO TO 10
      YD(N) = THADD(N)
      IF(LEQU) PRINT 102, N,N,YD(N)
      GO TO 1
      10 THAD(N) = Y(N)
      IF(LEQU) PRINT 202,N,N,THAD(N)
      1 CONTINUE
      NEG = NFER
C
      GENERALIZED ELASTIC COORDINATE RATE EQUATIONS
      IF(NMODS.EQ.0) GO TO 5063
      DO 7 NN=1,NMCD5
      N = NEG+NN
      IF(SCRT) GO TO 9
      YD(N) = THADD(N)
      IF(LEQU) PRINT 102, N,N,YD(N)

```

	GO TC 7	C2112300
	5 THAD(N) = Y(N)	C2112400
	IF(LEQU) PRINT 202, N,N,THAD(N)	C2112500
	7 CONTINUE	C2112600
	5063 CONTINUE	C2112700
	NEG = NEG + NMODS	C2112800
C		C2112900
C		C2113000
C	RELATIVE ANGULAR MOMENTUM WHEEL	C2113100
	(CALL UNFAC(ST1,NST1,SMV)	C2113200
	IF(NST1.EQ.0) GO TC 5064	C2113300
	DO 2 MM=1,NST1	C2113400
	MM=NST1-(MMM-1)	C2113500
	M = ST1(MM)	C2113600
	N = NEQ+NST1+1-MM	C2113700
	IF(SCRT) GO TO 11	C2113800
	YD(N) = THADD(N)	C2113900
	IF(LEQU) PRINT 104, N,N,YD(N)	C2114000
	GO TC 2	C2114100
	11 THADW(M) = Y(N)	C2114200
	HCM(M) = PLM(M)*THADW(M)	C2114300
	IF(LEQU) PRINT 205, M,N,THADW(M)	C2114400
	IF(LEQU) PRINT 204, M,N,M,HCM(M)	C2114500
	2 CONTINUE	C2114600
	5064 CONTINUE	C2114700
	NEG = NEG + NST1	C2114800
C		C2114900
C	DISPLACEMENT ABOUT OR ALONG FREE COORDINATE AXES	C2115000
	IF(NFRC.EQ.0) GO TC 5065	C2115100
	DO 3 MM=1,NFRC	C2115200
	N = NEQ + MM	C2115300
	IF(SCRT) GO TO 12	C2115400
	YD(N) = ANGD(MM)	C2115500
	IF(LEQU) PRINT 106, N,MM,YD(N)	C2115600
	GO TC 3	C2115700
	12 M = SFR(MM)	C2115800
	THA(M) = Y(N)	C2115900
	IF(LEQU) PRINT 206, M,N,THA(M)	C2116000
	3 CONTINUE	C2116100
	5065 CONTINUE	C2116200
	NEG = NEG + NFRC	C2116300
C		C2116400
C		C2116500
C	GENERALIZED ELASTIC COORDINATE DISPLACEMENT EQUATIONS	C2116600
	IF(NMODS.EQ.0) GO TO 5062	C2116700
	DO 19 NN=1,NMODS	C2116800
	N = NEQ+NN	C2116900
	NST1 = NFER+NN	C2117000
	IF(SCRT) GO TO 20	C2117100
	YD(N) = THAD(NST1)	C2117200
	IF(LEQU) PRINT 103, N,NST1,YD(N)	C2117300
	GO TC 19	C2117400
	20 CONTINUE	C2117500
	THA(NST1) = Y(N)	C2117600
	IF(LEQU) PRINT 206, NST1,N,THA(NST1)	C2117700
	19 CONTINUE	C2117800
	5062 CONTINUE	C2117900
	NEG = NEG + NMODS	C2118000
C		C2118100
C		C2118200

C	DISPLACEMENT ABOUT WHEEL SPIN AXIS	C2118300
	IF(NMCA.EQ.0) GO TO 5066	C2118400
	CO 4 MM=1,NMOA	C2118500
	M = SMA(MM)	C2118600
	N = NEQ + MM	C2118700
	IF(SCRT) GO TO 13	C2118800
	YD(N) = THADW(M)	C2118900
	IF(LEQU) PRINT 108. N,M,YD(N)	C2119000
	GO TO 4	C2119100
13	CONTINUE	C2119200
	THAW(M) = Y(N)	C2119300
	IF(LEQU) PRINT 208. M,N,THAW(M)	C2119400
4	CONTINUE	C2119500
5066	CONTINUE	C2119600
	NEQ = NEQ + NMUA	C2119700
C		C2119800
C	DIRECTION COSINES	C2119900
	CALL UNPAC(ST1,NST1,SD)	C2120000
	N = NEQ	C2120100
	M = C	C2120200
	IF(NST1.EQ.0) GO TO 5067	C2120300
	IF(INERF.OR.ST1(NST1).NE.1) GO TO 5	C2120400
	NST1 = NST1 - 1	C2120500
	IF(SCRT) GO TO 14	C2120600
	M = M+1	C2120700
14	CO 8 J=1,2	C2120800
	CO 8 I=1,3	C2120900
	N = N+1	C2121000
	IF(SCRT) GO TO 15	C2121100
	YD(N) = YMCD(I,J,M)	C2121200
	IF(LEQU) PRINT 111. N,I,J,M,YD(N)	C2121300
	GO TO 8	C2121400
15	XMC(I,J,M) = Y(N)	C2121500
	IF(LEQU) PRINT 211. I,J,M,N,XMC(I,J,M)	C2121600
8	CONTINUE	C2121700
5	IF(NST1.EQ.0) GO TO 5067	C2121800
	CO 6 MM=1,NST1	C2121900
	IF(SCRT) GO TO 16	C2122000
	M = M+1	C2122100
	GO TO 17	C2122200
16	M = ST1(MM)	C2122300
17	CO 6 J=1,2	C2122400
	CO 6 I=1,3	C2122500
	N = N+1	C2122600
	IF(SCRT) GO TO 18	C2122700
	YD(N) = YMCD(I,J,M)	C2122800
	IF(LEQU) PRINT 111. N,I,J,M,YD(N)	C2122900
	GO TO 6	C2123000
18	XMC(I,J,M) = Y(N)	C2123100
	IF(LEQU) PRINT 211. I,J,M,N,XMC(I,J,M)	C2123200
6	CONTINUE	C2123300
5067	CONTINUE	C2123400
	NEC = N	C2123500
	IF(LEQU) PRINT 112. NEC	C2123600
100	FORMAT ('1 SUBROUTINE SETUP ENTERED')	C2123700
102	FORMAT (' YD(' ,I2,' ) = THACD(' ,I2,' ) = ' ,D20.8)	C2123800
103	FORMAT (' YD(' ,I2,' ) = THAC(' ,I2,' ) = ' ,D20.8)	C2123900
202	FORMAT (' THAD(' ,I2,' ) = Y(' ,I2,' ) = ' ,D20.8)	C2124000
104	FORMAT (' YD(' ,I2,' ) = THADD(' ,I2,' ) = ' ,D20.8)	C2124100
204	FORMAT (' HMOM(' ,I2,' ) = FLM(' ,I2,' ) * THACW(' ,I2,' ) = ' ,D20.8)	C2124200

205	FORMAT (' THADW(' ,I2,' ) = Y(' ,I2,' ) = ',D20.8)	C2124300
106	FORMAT (' YD(' ,I2,' ) = ANGD(' ,I2,' ) = ',D20.8)	C2124400
206	FORMAT (' THA(' ,I2,' ) = Y(' ,I2,' ) = ',D20.8)	C2124500
108	FORMAT (' YD(' ,I2,' ) = THACW(' ,I2,' ) = ',D20.8)	C2124600
208	FORMAT (' THAA(' ,I2,' ) = Y(' ,I2,' ) = ',D20.8)	C2124700
111	FORMAT (' YD(' ,I2,' ) = YMCC(' ,I2,' ,',I2,' ,',I2,' ) = ',D20.8)	C2124800
211	FORMAT (' XMC(' ,I2,' ,',I2,' ,',I2,' ) = Y(' ,I2,' ) = ',D20.8)	C2124900
112	FORMAT (' TCTAL NLMBER CF EQUATIONS, NEQ = ',I5)	C2125000
	RETURN	C2125100
	END	C2125200

C		C2200000
	SUBROUTINE OUTPUT	C2200100
C	GENERAL TYPE OUTPUT FOR NO PARTICULAR SATELLITE.	C2200200
C		C2200300
C		C2200400
	IMPLICIT REAL*8(A-F,O-Z,*)	C2200500
	LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLG, LEQU, LINIT(1)	C2200600
C		C2200700
C		C2200800
	INTEGER	C2200900
	* AWCNK , CT1 , CT2 , CT3 , CT4 , CT5 , FCUN , PCUN ,	C2201000
	* SCNDUM, SCN , SCR DUM, SCR , SFKDUM, SFK , SFR , SG ,	C2201100
	* SI , SIG , SIXDUM, SIX , SKDUM , SK , SL , SLK ,	C2201200
	* SMA , SMC DUM, SMC , SMV , SOK , SPIDUM, SPI , SQF ,	C2201300
	* SGL , SR , SSCN , SSIX , SVA , SVE , SVD , SVI ,	C2201400
	* SVM , SVP , SVO , SXM , SXT , TOHQ , SMAL , SEU ,	C2201500
	* SC , SCG , NFLXB , SFLX , SFXM , NMCDS , SFCC , SCC ,	C2201600
	* IINIT(1) , IZINIT(1) , SD	C2201700
C		C2201800
C		C2201900
C		C2202000
	REAL*8	C2202100
	* ANGD (33) , CNF (3,10) , ETIC (3,10) , ETMC (3,10) ,	C2202200
	* FLQ (3,20) , FLE (3,3,20) , FLH (3,3,20) ,	C2202300
	* THADW (33) , YMCD (3,2,11) , RINIT (1) , RZINIT(1)	C2202400
C		C2202500
C		C2202600
C		C2202700
C		C2202800
C		C2202900
	COMMON /LOGIC/ FG1, FG2, FG3, FG4, FG5, INERF, RBLG(10)	C2203000
C		C2203100
C		C2203200
	COMMON /INTG/ AWORK (200) ,	C2203300
	* CT1 , CT2 , CT3 , CT4 ,	C2203400
	* CT5 , FCUN (33) , JCON (10) , LCCN (22) ,	C2203500
	* MC (10) , NBI , NBDD , NCTC ,	C2203600
	* NFER , NFKC , NFRC , NLOR ,	C2203700
	* NNV , NMD , NMDA(B,OUTER,INVP) ,	C2203800
	* NSVC , PCUN (11) , SU , SFR (33) ,	C2203900
	* SG , SI (55) , SIG , SL ,	C2204000
	* SLK (33) , SMA (10) , SOK (11) , SGF (11) ,	C2204100
	* SGL (11) , SMV , SR , SSCN ,	C2204200
	* SSIX , SVA , SVB , SVD ,	C2204300
	* SVI , SVM , SVP (22) , SVQ (33) ,	C2204400



```

* SXM (3,10) , EXT , TORQ (57) , SMAL , C2204500
* SEU , NTQ , SC (33) , SCG , C2204600
* NFLXB , SFLX , SFXM (10) , NMCD5 , C2204700
* SFCC , SCC (10) , C2204800
C
C2204900
C2205000
C2205100
COMMON /INTGZ/
* SCNDUM , SCN (5) , SCRUM , SCR (9) , C2205200
* SFKCLM , SFK (5) , SIXDUM , SIX (9) , C2205300
* SKDLM , SK (5) , SPIDUM , SPI (9) , C2205400
* SMCCLM , SNC (5) , C2205500
C
C2205600
C2205700
C2205800
COMMON /REAL/
* CA (3,10) , CAC (3,10) , CLM (10) , CCMC (3,11) , C2205900
* DCMC (3,11) , ETC (3,11) , ETM (33) , FUMC (3,11) , C2206000
* GAM (3,66) , F , HM (3,10) , FMC (3,10) , C2206100
* HNDM (10) , PHI (3,11) , PLM (10) , GF (3,32) , C2206200
* QFC (3,33) , QL (3,22) , QLC (3,22) , RUMC (3,11) , C2206300
* T , THA (33) , THAD (32) , C2206400
* THADW (10) , THAW (10) , XDIC (3,3,66) , XI (3,3,10) , C2206500
* XIC (3,3,10) , XMAS (10) , XMN (33,33) , XMT (3,3,10) , C2206600
* TLG (33) , FLA (3,20) , FLB (3,20) , FLC (3,20) , C2206700
* FLD (3,3,20) , FLJ (3,3,20) , CAD (3,10) , XIG (3,3,10) , C2206800
* FLIRC (3,10) , FLCRC (3,10) , FLAC (3,20) , FLQC (3,20) , C2206900
* FLCM (20) , ZETA (20) , FCF (3,3,40) , FCK (3,40) , C2207000
* TIMEND , C2207100
C
C2207200
C2207300
C2207400
COMMON /REALZ/
* CEDLM (1,3) , CB (3,10) , CBCDUM(1,3) , CBC (3,10) , C2207500
* XMCCLM(1,1,9) , XMC (3,3,10) , CBN(3) , C2207600
C
C2207700
C2207800
EQUIVALENCE (ETM(1),THAD(1)) ,(XMN(1,1),ANGD(1)) , C2207900
* (XMN(1,3),YMCD(1,1,1)) ,(XMN(1,6),CNF(1,1)) , C2208000
* (XMN(1,8),ETIC(1,1)) ,(XMN(1,10),ETMC(1,1)) , C2208100
* (FLB(1,1),FLG(1,1)) ,(FLE(1,1,1),FLD(1,1,1)) , C2208200
* (FLH(1,1,1),FLJ(1,1,1)) , C2208300
* (FG1,INIT(1)) ,(CA(1,1),RINIT(1)) , C2208400
* (CBDUM(1,1),EZINIT(1)) ,(AWORK(1),IINIT(1)) , C2208500
* (SCNDUM,IZINIT(1)) , C2208600
C
C2208700
C2208800
C2208900
INTEGER TIK
DIMENSION PHII(3,3) C2209000
DIMENSION TOMC(3,11) C2209100
DIMENSION SYSCM(3) C2209200
DIMENSION SYSIN(3,3) C2209300
DIMENSION HB(3,10),TK(10) C2209400
DIMENSION PUS(3,10),VEL(3,10) C2209500
DIMENSION TEM1(3),TEM2(3),TEM3(3),DERV(3,11) C2209600
DIMENSION TEM4(3) C2209700
DIMENSION HBCDY(3) , FINERT(3) C2209800
DIMENSION EPD(3,10),DHM(3,10) C2209900
DIMENSION EP(3),EI(3,3),EIC(3,3),FQD(3),FQDC(3),TEM5(3,3) C2210000
REAL*8 LM(3,10),LMT(3) C2210100
INTEGER SET(10),SFXMK C2210200
C
C2210300
C2210400

```

```

1 CONTINUE
PRINT 203, T
C
C
C      COMPLETE SYSTEM COMPOSITE CENTER OF MASS
TCTM = C.D0
DO 11 I=1,3
11 TEM1(I) = 0.D0
DO 15 K=1,NBOD
KO = KTG(NB1,0,K)
CALL SCLV(XMAS(K),CAM(1,KG),TEM2)
CALL VECADD(TEM1,TEM2,TEM1)
15 TOTM = TCTM + XMAS(K)
DO 16 I=1,3
16 SYSCM(I) = TEM1(I)/TOTM
PRINT 200, (SYSCM(I),I=1,3)
PRINT 222, TOTM
C
C
C      COMPLETE SYSTEM INERTIA TENSOR ABOUT COMPOSITE CENTER OF MASS
CALL SUECP(SYSCM,SYSCM,TCTM,SY SIN)
DO 4 I=1,3
DO 4 J=1,3
4 SYSIN(I,J) = XDIC(I,J,1) - SYSIN(I,J)
PRINT 211, (SYSIN(1,J),J=1,3)
PRINT 212, (SYSIN(2,J),J=1,3)
PRINT 211, (SYSIN(3,J),J=1,3)
PRINT 213
C
C
C      COMPLETE VECTOR FROM INERTIAL ORIGIN TO COMPOSITE SYSTEM
CENTER OF MASS RELATIVE TO COMPUTING FRAME
CALL VECADD(CBC(1,1),SYSCM,CBC(1,0))
C
C
C
C      FLEXIBILITY RELATED PARAMETERS
MN = 0
DO 30 K=1,NBCD
DO 32 I=1,3
EPD(I,K) = 0.
32 DHM(I,K) = 0.
IF(SFXM(K).EQ.C) GO TO 30
SFXMK = SFXM(K)
DO 31 I=1,SFXMK
MN = MN+1
NFMN = NFER + MN
CALL SCLV(THAD(NFMN),FLAC(1,MN),TEM1)
CALL VECADD(EPD(1,K),TEM1,EPD(1,K))
CALL SCLV(THAD(NFMN),FLQC(1,MN),TEM1)
CALL VECADD(DHM(1,K),TEM1,DFM(1,K))
31 CONTINUE
30 CONTINUE
C
C
C      COMPLETE INERTIAL ANGULAR MOMENTUM AND KINETIC ENERGY OF EACH
BODY AND OF THE COMPOSITE SYSTEM
TKIN = C.D0
DO 7 I=1,3

```

```

C2210500
C2210600
C2210700
C2210800
C2210900
C2211000
C2211100
C2211200
C2211300
C2211400
C2211500
C2211600
C2211700
C2211800
C2211900
C2212000
C2212100
C2212200
C2212300
C2212400
C2212500
C2212600
C2212700
C2212800
C2212900
C2213000
C2213100
C2213200
C2213300
C2213400
C2213500
C2213600
C2213700
C2213800
C2213900
C2214000
C2214100
C2214200
C2214300
C2214400
C2214500
C2214600
C2214700
C2214800
C2214900
C2215000
C2215100
C2215200
C2215300
C2215400
C2215500
C2215600
C2215700
C2215800
C2215900
C2216000
C2216100
C2216200
C2216300
C2216400

```

```

LMT(I) = 0.00                                C2216500
7  FBODY(I) = 0.00                            C2216600
DO 17 KKK=1,NBOD                              C2216700
  K=NBOD-(KKK-1)                             C2216800
  KK = K                                       C2216900
  JK = JCCN(K)                                C2217000
  IF(KK.NE.1) GO TO 28                       C2217100
  DO 25 I=1,3                                 C2217200
29 TEM1(I) = RDMC(I,NE1)                      C2217300
  GO TO 26                                     C2217400
26 CONTINUE                                   C2217500
C  COMPUTE LINEAR VELOCITY OF CENTER OF MASS OF BODY K PUT IN TEM1 C2217600
  IF(RELD(K)) GO TO 19                       C2217700
  CALL VECROS (FCMC(1,JK),CAC(1,K),TEM1)     C2217800
  CALL VECADD(RDMC(1,K),TEM1,TEM1)          C2217900
  GO TO 24                                     C2218000
19 CALL VECROS (FCMC(1,K),CAC(1,K),TEM1),    C2218100
24 CALL VECADD(RDMC(1,NB1),TEM1,TEM1)       C2218200
C  CHECK FOR END OF CHAIN                     C2218300
25 IF(JK.EQ.0) GO TO 26                     C2218400
  CALL VECROS (FCMC(1,JK),CEC(1,KK),TEM2)   C2218500
  CALL VECADD(TEM1,TEM2,TEM1)               C2218600
  KK = JK                                     C2218700
  JK = JCCN(KK)                              C2218800
  GO TO 25                                    C2218900
26 CONTINUE                                   C2219000
C  INTERIAL POSITION OF CENTER OF MASS IN TEM2 C2219100
C  KL = KTC(NB1,0,K)                          C2219200
  CALL VECADD(CEC(1,1),GAM(1,KL),TEM2)      C2219300
  DO 3 I=1,3                                  C2219500
  FOS(I,K) = TEM2(I)                         C2219600
3  VEL(I,K) = TEM1(I)                        C2219700
C  ADD RELATIVE VELOCITY OF CM, NON-ZERO IF BODY FLEXIBLE C2219800
C  CALL VECADD(VEL(I,K),EPD(I,K),VEL(I,K))    C2219900
  DO 13 I=1,3                                 C2220000
13 TEM1(I) = VEL(I,K)                       C2220100
C  START COMPUTATION OF ANGLAR MOMENTUM, LINEAR MOMENTUM AND KINETIC C2220200
C  CALL DYCDTV(XIC(1,1,K),FCMC(1,K),HB(1,K)) C2220300
  CALL VECDDOT(FCMC(1,K),HE(1,K),TK(K))     C2220400
  CALL VECDDOT(TEM1,TEM1,TEM)               C2220500
  TK(K) = .SDO*(TK(K) + XMAS(K)*TEM)        C2220600
  CALL VECROS (TEM2,TEM1,TEM3)               C2220700
  CALL SOLV(XMAS(K),TEM3,TEM3)               C2220800
  CALL VECADD(HB(1,K),TEM3,HB(1,K))         C2220900
  IF(.NOT.RBLD(K)) GO TO 27                  C2221000
  IF(NMD.EQ.0) GO TO 5068                    C2221100
  DO 27 M=1,NMD                              C2221200
  IF(MC(M).NE.K) GO TO 27                   C2221300
  CALL SOLV(HMDM(M),HMC(1,M),TEM3)          C2221400
  CALL VECADD(HB(1,K),TEM3,HE(1,K))         C2221500
  TK(K) = TK(K) + .SDO*HMC(M)**2/PLM(M)     C2221600
27 CONTINUE                                  C2221700
5068 CONTINUE                                C2221800
C  ADD FLEXIBILITY ADDITONS TO MOMENTUM AND ENERGY C2221900
C  C2222000
C  C2222100
C  C2222200
C  C2222300
C  C2222400

```

```

CALL VECADD(HB(1,K),CHM(1,K),HJ(1,K)) C222500
CALL VECDDCT(FGMC(1,K),CHM(1,K),TEM) C222600
TK(K) = TK(K) + TEM C222700
C C222800
C C222900
C ADD UP FOR SYSTEM ANGULAR MOMENTUM AND KINETIC ENERGY C2223000
TKIN = TKIN + TK(K) C2223100
CALL VECADD(HB(1,K),HBCDY,HEODY) C2223200
CALL SCLV(XMAS(K),VEL(1,K),LM(1,K)) C2223300
CALL VECADD(LMT,LM(1,K),LMT) C2223400
17 CONTINUE C2223500
CALL TRNSPS (XMC(1,1,0)) C2223600
CALL VECTRAN (HBODY,XMC(1,1,0),HINERT) C2223700
CALL TRNSPS (XMC(1,1,0)) C2223800
IF(.NOT.INERF) GO TO 2 C2223900
CALL TRNSPS (XMC(1,1,1)) C2224000
CALL VECTRAN (HINERT,XMC(1,1,1),HBODY) C2224100
CALL TRNSPS (XMC(1,1,1)) C2224200
2 CONTINUE C2224300
HMG = DSQRT(HINERT(1)**2 + HINERT(2)**2 + HINERT(3)**2) C2224400
PRINT 209, HMG C2224500
PRINT 201, (HEODY(I),I=1,3) C2224600
PRINT 202, (HINERT(I),I=1,3) C2224700
A = CSQRT(LMT(1)**2 + LMT(2)**2 + LMT(3)**2) C2224800
PRINT 217, A C2224900
PRINT 219, (LMT(I),I=1,3) C2225000
PRINT 215, TKIN C2225100
C C2225200
C C2225300
C COMPLETE INERTIAL ACCELERATIONS C2225400
DO 8 I=1,3 C2225500
6 TOMC(I,1) = DCMC(I,1) C2225600
IF(NECD.EQ.1) GO TO 5001 C2225700
DO 14 K=2,NBOD C2225800
CALL VECFOS (FGMC(1,K),RCMC(1,K),TEM1) C2225900
14 CALL VECSUB(DCMC(1,K),TEM1,TCMC(1,K)) C2226000
5001 CONTINUE C2226100
CALL VECFOS (FGMC(1,1),RCMC(1,NB1),TEM1) C2226200
CALL VECSUB(DCMC(1,NB1),TEM1,TCMC(1,NB1)) C2226300
M = 1 C2226400
DO 20 K=1,NB1 C2226500
IF(K.EQ.1) GO TO 21 C2226600
M = M+3-PCON(K-1) C2226700
21 DO 22 I=1,3 C2226800
22 TEM1(I) = 0 C2226900
MMTERM=M+2-PCON(K) C2227000
IF(M.GT.MMTERM) GO TO 20 C2227100
DO 23 MN=M,MMTERM C2227200
CALL SCLV(THADD(MN),OFC(1,MN),TEM2) C2227300
23 CALL VECADD(TEM1,TEM2,TEM1) C2227400
20 CALL VECADD(TEM1,TCMC(1,K),DERV(1,K)) C2227500
C C2227600
C C2227700
C C2227800
DO 5 K=1,NBOD C2227900
PRINT 204, K, (ROMC(I,K),I=1,3), (FOMC(I,K),I=1,3) C2228000
10 PRINT 207, (DERV(I,K),I=1,3) C2228100
PRINT 203, (ETC(I,K),I=1,3), (PHI(I,K),I=1,3) C2228200
PRINT 205, (CAC(I,K),I=1,3), (CJC(I,K),I=1,3) C2228300
PRINT 216, (PCS(I,K),I=1,3), (VEL(I,K),I=1,3) C2228400
PRINT 206, ((XMC(I,J,K),J=1,3), (XIC(I,J,K),J=1,3),I=1,3) C2228500

```

```

PRINT 214, (HB(J,K),J=1,3),TK(K)          C2228500
PRINT 21E, (LM(J,K),J=1,3)                C2228600
5 CONTINUE                                C2228700
PRINT 20E, (FOMC(I,NB1),I=1,3),(DERV(I,NB1),I=1,3) C2228E00
PRINT 223, (ETC(I,NB1),I=1,3),(PHI(I,NB1),I=1,3) C2228900
PRINT 224, (CBC(I,0),I=1,3)               C2229000
PRINT 221, ((XMC(I,J,0),J=1,3),I=1,3)    C2229100
PRINT 213                                  C2229200
IF(NMD.EQ.0) GO TO 41                      C2229300
DO 6 M=1,NMD                               C2229400
PRINT 21C, M,MMOM(M),M,CLM(M)            C2229500
6 CONTINUE                                C2229600
41 CONTINUE                                C2229700
PRINT 213                                  C2229800
IF(NMDS.EQ.0) GO TO 40                    C2229900
CALL UNPAC(SET,NSET,SFLX)                 C2230000
MN = 0                                     C2230100
DO 33 KK=1,NSET                            C2230200
K = SET(NSET+1-KK)                        C2230300
DO 34 I=1,3                                C2230400
EP(I) = C                                  C2230500
FOC(I) = 0                                 C2230600
FOCC(I) = 0                                C2230700
DO 34 J=1,3                                C2230800
EI(I,J) = 0                                C2230900
EID(I,J) = 0                               C2231000
34 CONTINUE                                C2231100
SF XMK = SF XM(K)                          C2231200
DO 35 M=1,SF XMK                           C2231300
MN = MN+1                                  C2231400
NFMN = NFER+MN                             C2231500
CALL SCLV(THA(NFMN),FLA(1,MN),TEM1)        C2231600
CALL VECADD(EP,TEM1,EP)                   C2231700
CALL SCLV(THAD(NFMN),FLO(1,MN),TEM1)      C2231800
CALL VECADD(FOG,TEM1,FOG)                 C2231900
CALL SCLD(THA(NFMN),FLE(1,1,MN),TEMS)    C2232000
CALL DYADD(EI,TEMS,EI)                    C2232100
CALL SCLD(THAD(NFMN),FLE(1,1,MN),TEMS)   C2232200
CALL DYALD(EID,TEMS,EID)                  C2232300
7 35 CONTINUE                              C2232400
PRINT 22E, (EP(I),I=1,3),(EPD(I,K),I=1,3) C2232500
PRINT 22E, K,(FOD(I),I=1,3)               C2232600
PRINT 22S, ((EI(I,J),J=1,3),(EID(I,J),J=1,3),I=1,3) C2232700
PRINT 23C, (FLIRC(I,K),I=1,3),(FLCRC(I,K),I=1,3) C2232800
PRINT 231,(DHM(I,K),I=1,3)                C2232900
33 CONTINUE                                C2233000
PRINT 213                                  C2233100
40 CONTINUE                                C2233200
DO 9 I=1,NFER                               C2233300
9 PRINT 22C, (I,THA(I),I,THAD(I),I,THADD(I),I,(OFC(J,I),J=1,3)) C2233400
N1 = NFER+1                                 C2233500
N2 = NFER+MN                                C2233600
IF(MN.EQ.0) RETURN                         C2233700
DO 12 I=N1,N2                               C2233800
12 PRINT 22E, I,THA(I),I,THAD(I),I,THADD(I) C2233900
200 FORMAT (' CENTER OF MASS = ',3D17.8,/) C2234000
201 FORMAT (' BODY = ',3D17.8)             C2234100
202 FORMAT (' INERT = ',3D17.8)           C2234200
203 FORMAT (' TIME = ',D15.5,/)           C2234300
204 FORMAT (' BODY ',12,4X,' FOMC = ',3D17.8,3X,' FCMC = ',3D17.8) C2234400

```

```

205 FORMAT (14X,'CAC = ',3D17.8,3X,'CBC = ',3D17.8) C2234500
206 FORMAT (14X,'XMC = ',3D17.8,3X,'XIC = ',3D17.8) C2234600
207 FORMAT (14X,'ACC = ',3D17.8) C2234700
208 FORMAT (/,3X,'DRIGIN',4X,'FMC = ',3D17.8,3X,'ACC = ',3D17.8) C2234800
209 FORMAT (' ANGULAR MOMENTUM = ',D20.8) C2234900
210 FORMAT (3X,'HMGM(','12,') = ',D17.8,10X,'CLM(','12,') = ',D17.8) C2235000
211 FORMAT (25X,3D17.8) C2235100
212 FORMAT (' SYSTEM INERTIA TENSOR = ',3D17.8) C2235200
213 FORMAT (' ') C2235300
214 FORMAT(14X,' HB = ',3D17.8,3X,' TK = ',D17.8) C2235400
215 FORMAT (/,' KINETIC ENERGY = ',D20.8) C2235500
216 FORMAT (14X,'PCS = ',3D17.8,3X,'VEL = ',3D17.8) C2235600
217 FORMAT (/,' LINEAR MOMENTUM = ',D20.8) C2235700
218 FORMAT (14X,' LM = ',3D17.8) C2235800
219 FORMAT (/,3X,'LBOCY = ',3D17.8) C2235900
220 FORMAT (3X,'THA(','12,') = ',D13.6,3X,'THAD(','12,') = ',D13.6,3X,'THA C2236000
      *CD(','12,') = ',D13.6,3X,'GFC(','12,') = ',3D13.6) C2236100
221 FORMAT (14X,'XMC = ',3D17.8) C2236200
222 FORMAT (' TOTAL SYSTEM MASS = ',D17.8,/) C2236300
223 FORMAT (14X,'ETC = ',3D17.8,3X,'PHI = ',3D17.8) C2236400
224 FORMAT (14X,'CBC = ',3D17.8) C2236500
225 FORMAT (3X,'THA(','12,') = ',D13.6,3X,'THAD(','12,') = ',D13.6,3X,'THA C2236600
      *CD(','12,') = ',D13.6) C2236700
226 FORMAT (/,3X,'FLEXIBLE',3X,' EP = ',3D17.8,3X,'EPP = ',3D17.8) C2236800
228 FORMAT (3X,' BODY',12,4X,' CD = ',3D17.8) C2236900
229 FORMAT (14X,' EI = ',3D17.8,3X,'EID = ',3D17.8) C2237000
230 FORMAT (14X,'FIR = ',3D17.8,3X,'FCR = ',3D17.8) C2237100
231 FORMAT (14X,'DHM = ',3D17.8) C2237200
      RETURN C2237300
      END C2237400

```

```

C SUBROUTINE OUTPSP C2300000
C GENERAL TYPE OUTPUT FOR NO PARTICULAR SATELLITE C2300100
C C2300200
C C2300300
C C2300400
C IMPLICIT REAL*8(A-H,O-Z,8) C2300500
C LOGICAL FG1, FG2, FG3, FG4, FG5, INERF, RBLD, LEQU, LINIT(1) C2300600
C C2300700
C C2300800
C INTEGER C2300900
C * AWORK , CT1 , CT2 , CT3 , CT4 , CT5 , FCCN , PCCN , C2301000
C * SCNDUM , SCN , SCRDUM , SCR , SFKDUM , SFK , SFR , SG C2301100
C * SI , SIG , SIXDUM , SIX , SKDUM , SK , SL , SLK C2301200
C * SVA , SMC0UM , SMC , SMV , SGK , SPIDUM , SPI , SQF C2301300
C * SCL , SR , SSCN , SSIX , SVA , SVB , SVD , SVI C2301400
C * SVM , SVP , SVQ , SVM , SXT , TORQ , SMAL , SEU C2301500
C * SC , SCG , NFLXB , SFLX , SFXM , NMUDS , SFCC , SCC C2301600
C * IINIT(1) , IZINIT(1) , SD C2301700
C C2301800
C C2301900
C C2302000
C REAL*8 C2302100
C * ANGC (33) , CNF (3,10) , ETIC (3,10) , ETMC (3,10) , C2302200
C * FLO (2,20) , FLE (3,3,20) , FLH (3,3,20) , C2302300
C * THAD (33) , YMCD (3,2,11) , RINIT (1) , RZINIT(1) C2302400

```



```

*          (CBCUM(1,1),RZINIT(1))      ,(AWORK(1),IINIT(1))      (2308500
*          (SCNDLM,IZINIT(1))          (2308600
C          (2308700
C          (2308800
C          (2308900
DIMENSION PHII(3,3)                    (2309000
DIMENSION TCMC(3,11)                   (2309100
DIMENSION SYSCM(3)                     (2309200
DIMENSION SYSIN(3,3)                   (2309300
DIMENSION HB(3,10),TK(10)              (2309400
DIMENSION POS(3,10),VEL(3,10)         (2309500
DIMENSION TEM1(3),TEM2(3),TEM3(3),DERV(3,11) (2309600
DIMENSION TEM4(3)                      (2309700
DIMENSION HBODY(3),                    FINERT(3) (2309800
DIMENSION EPC(3,10),DHM(3,10)         (2309900
DIMENSION EP(3),EI(3,3),EIC(3,3),FGD(3),FQDC(3),TEM5(3,3) (2310000
INTEGER SET(10),SFXMK                 (2310100
REAL*8 LM(3,10),LMT(3)                (2310200
INTEGER TIK                             (2310300
C          (2310400
C          (2310500
1 CONTINUE                              (2310600
PRINT 203, T                            (2310700
C          (2310800
C          (2310900
C          (2311000
COMPUTE SYSTEM COMPOSITE CENTER OF MASS (2311000
TOTM = 0.00                             (2311100
DO 11 I=1,3                              (2311200
11 TEM1(I) = 0.00                        (2311300
DO 15 K=1,NBOD                            (2311400
KO = KTC(NB1,0,K)                       (2311500
CALL SCLV(XMAS(K),GAM(1,KC),TEM2)        (2311600
CALL VECADD(TEM1,TEM2,TEM1)              (2311700
15 TOTM = TOTM + XMAS(K)                 (2311800
DO 16 I=1,3                              (2311900
16 SYSCM(I) = TEM1(I)/TOTM                (2312000
PRINT 200, (SYSCM(I),I=1,3)             (2312100
PRINT 222, TOTM                          (2312200
C          (2312300
C          (2312400
C          (2312500
COMPUTE VECTOR FROM INERTIAL ORIGIN TO COMPOSITE SYSTEM (2312500
CENTER OF MASS RELATIVE TO COMPUTING FRAME (2312600
CALL VECADD(CBC(1,1),SYSCM,CBC(1,0))    (2312700
C          (2312800
C          (2312900
C          (2313000
COMPUTE SYSTEM INERTIA TENSOR ABOUT COMPOSITE CENTER OF MASS (2313000
CALL SUECP(SYSCM,SYSCM,TOTM,SYSIN)      (2313100
DO 4 I=1,3                                (2313200
DO 4 J=1,3                                (2313300
4 SYSIN(I,J) = XDIC(I,J,1) - SYSIN(I,J) (2313400
PRINT 211, (SYSIN(1,J),J=1,3)           (2313500
PRINT 212, (SYSIN(2,J),J=1,3)           (2313600
PRINT 211, (SYSIN(3,J),J=1,3)           (2313700
PRINT 213                                (2313800
C          (2313900
C          (2314000
C          (2314100
FLEXIBILITY RELATED PARAMETERS          (2314100
MN = 0                                    (2314200
DO 30 K=1,NBOD                            (2314300
DO 32 I=1,3                              (2314400

```



```

EPD(1,K) = 0.                                C2314500
32 CHM(1,K) = 0.                                C2314600
   IF(SFXM(K).EQ.0) GO TO 30                   C2314700
   SFXMK = SFXM(K)                             C2314800
   DO 31 I=1,SFXMK                             C2314900
   MN = MN+1                                    C2315000
   NFMN = NFER + MN                            C2315100
   CALL SCLV(THAD(NFMN),FLAC(1,MN),TEM1)       C2315200
   CALL VECADD(EPD(1,K),TEM1,EPD(1,K))        C2315300
   CALL SCLV(THAD(NFMN),FLOC(1,MN),TEM1)       C2315400
   CALL VECADD(DHM(1,K),TEM1,DHM(1,K))         C2315500
31 CONTINUE                                    C2315600
30 CONTINUE                                    C2315700
C                                              C2315800
C                                              C2315900
C                                              C2316000
C ***** COMPUTE INERTIAL ANGULAR MOMENTUM AND KINETIC ENERGY OF EACH C2316100
C ***** BODY AND OF THE COMPOSITE SYSTEM C2316200
   TKIN = C.D0
   DO 7 I=1,3
   LMI(I) = C.D0
   7 FBCDY(I) = 0.D0
   DO 17 KKK=1,NBOD
   KK=NBCD-(KKK-1)
   KK = K
   JK = JCLN(K)
   IF(KK.NE.1) GO TO 28
   DO 29 I=1,3
29 TEM1(I) = ROMC(I,NB1)
   GO TO 26
28 CONTINUE
C ***** COMPUTE LINEAR VELOCITY OF CENTER OF MASS OF BODY K PUT IN TEM1 C2317600
   IF(RELO(K)) GO TO 19
   CALL VECROS (FGMC(1,JK),CAC(1,K),TEM1)
   CALL VECADD(ROMC(1,K),TEM1,TEM1)
   GO TO 24
19 CALL VECROS (FGMC(1,K),CAC(1,K),TEM1)
24 CALL VECADD(ROMC(1,NB1),TEM1,TEM1)
C ***** CHECK FOR END OF CHAIN
25 IF(JK.EQ.0) GO TO 26
   CALL VECROS (FGMC(1,JK),CBC(1,JK),TEM2)
   CALL VECADD(TEM1,TEM2,TEM1)
   KK = JK
   JK = JCLN(KK)
   GO TO 25
26 CONTINUE
C ***** INERTIAL POSITION OF CENTER OF MASS IN TEM2
   KL = KTC(NB1,C,K)
   CALL VECADD(CBC(1,1),GAM(1,KL),TEM2)
   DO 3 I=1,3
   FCS(I,K) = TEM2(I)
   3 VEL(I,K) = TEM1(I)
C *****
C ***** ADD RELATIVE VELOCITY OF CM, NON-ZERO IF BCDY FLEXIBLE, C2319900
   CALL VECADD(VEL(1,K),EPD(1,K),VEL(1,K))
   DO 13 I=1,3
13 TEM1(I) = VEL(1,K)
C *****

```

```

C      START COMPUTATION OF ANGULAR MOMENTUM, LINEAR MOMENTUM AND KINETIC C2320500
      CALL DYDCDV(XIC(1,1,K),FCMC(1,K),HB(1,K)) C2320600
      CALL VECCDT(FCMC(1,K),HB(1,K),TK(K)) C2320700
      CALL VECCDT(FCMC(1,K),HB(1,K),TK(K)) C2320800
      CALL VECCDT(FCMC(1,K),HB(1,K),TK(K)) C2320900
      TK(K) = .5D0*(TK(K) + XMAS(K)*TEM) C2321000
      CALL VECROS (TEM2,TEM1,TEM3) C2321100
      CALL SCLV(XMAS(K),TEM3,TEM3) C2321200
      CALL VECADD(HB(1,K),TEM3,FB(1,K)) C2321300
      IF(.NOT.RBLD(K)) GO TO 27 C2321400
      IF(NMC.EQ.0) GO TO 5068 C2321500
      DO 27 M=1,NM0 C2321600
      IF(MC(M).NE.K) GO TO 27 C2321700
      CALL SCLV(HMOM(M),HMC(1,M),TEMJ) C2321800
      CALL VECADD(HB(1,K),TEM3,FE(1,K)) C2321900
      TK(K) = TK(K) + .5D0*HMC(M)**2/PLM(M) C2322000
27 CONTINUE C2322100
5068 CONTINUE C2322200
C      C2322300
C      C2322400
C      ADD FLEXIBILITY ADDITIONS TO MOMENTUM AND ENERGY C2322500
      CALL VECADD(HB(1,K),DHM(1,K),HB(1,K)) C2322600
      CALL VECCDT(FCMC(1,K),DHM(1,K),TEM) C2322700
      TK(K) = TK(K) + TEM C2322800
C      C2322900
C      ADD UP FOR SYSTEM ANGULAR MOMENTUM AND KINETIC ENERGY C2323000
      TKIN = TKIN + TK(K) C2323100
      CALL VECADD(HB(1,K),FBODY,FBODY) C2323200
      CALL SCLV(XMAS(K),VEL(1,K),LM(1,K)) C2323300
      CALL VECADD(LMT,LM(1,K),LMT) C2323400
17 CONTINUE C2323500
      CALL TRNSPS (XMC(1,1,0)) C2323600
      CALL VECTR (HBODY,XMC(1,1,0),HINERT) C2323700
      CALL TRNSPS (XMC(1,1,0)) C2323800
      IF(.NOT.INERT) GO TO 2 C2323900
      CALL TRNSPS (XMC(1,1,1)) C2324000
      CALL VECTR (HINERT,XMC(1,1,1),HBODY) C2324100
      CALL TRNSPS (XMC(1,1,1)) C2324200
2 CONTINUE C2324300
      HMG = DSQRT(HINERT(1)**2 + HINERT(2)**2 + HINERT(3)**2) C2324400
      PRINT 209, HMG C2324500
      PRINT 201, (HBCDY(I),I=1,3) C2324600
      PRINT 202, (HINERT(I),I=1,3) C2324700
      A = DSQRT(LMT(1)**2 + LMT(2)**2 + LMT(3)**2) C2324800
      PRINT 217, A C2324900
      PRINT 219, (LMT(I),I=1,3) C2325000
      PRINT 215, TKIN C2325100
C      C2325200
C      C2325300
C      C2325400
C      COMPLETE INERTIAL ACCELERATIONS C2325500
      DO 8 I=1,3 C2325600
      TCMC(I,1) = DOMC(I,1) C2325700
      IF(NECD.EQ.1) GO TO 5001 C2325800
      DO 14 K=2,NB0D C2325900
      CALL VECROS (FCMC(1,K),RCMC(1,K),TEM1) C2326000
      CALL VECSUB(DOMC(1,K),TEM1,TCMC(1,K)) C2326100
5001 CONTINUE C2326200
      CALL VECROS (FCMC(1,1),RCMC(1,NB1),TEM1) C2326300
      CALL VECSUB(DOMC(1,NB1),TEM1,TCMC(1,NB1)) C2326400
      M = 1

```

DO 20 K=1,NB1	C2326500
IF(K.EQ.1) GC TO 21	C2326600
M = A+3-FCDN(K-1)	C2326700
21 DO 22 I=1,3	C2326800
22 TEM1(I) = 0	C2326900
MMTERM=M+2-PCON(K)	C2327000
IF(M.GT.MMTERM) GC TO 20	C2327100
DO 23 MM=M,MMTERM	C2327200
CALL SCLV(THADD(MM),QFC(1,MM),TEM2)	C2327300
23 CALL VECADD(TEM1,TEM2,TEM1)	C2327400
20 CALL VECADD(TEM1,TCMC(1,K),CERV(1,K))	C2327500
C	C2327600
C	C2327700
DO 5 K=1,NB0D	C2327800
PRINT 204, K, (RUMC(1,K),I=1,3), (FCMC(1,K),I=1,3)	C2327900
10 PRINT 207, (DERV(1,K),I=1,3)	C2328000
PRINT 223, (ETC(1,K),I=1,3), (PHI(1,K),I=1,3)	C2328100
PRINT 205, (CAC(1,K),I=1,3), (CBC(1,K),I=1,3)	C2328200
PRINT 216, (POS(1,K),I=1,3), (VEL(1,K),I=1,3)	C2328300
PRINT 206, ((XMC(1,J,K),J=1,3), (XIC(1,J,K),J=1,3), I=1,3)	C2328400
PRINT 214, (HB(J,K),J=1,3), TK(K)	C2328500
PRINT 218, (LM(J,K),J=1,3)	C2328600
E CONTINUE	C2328700
PRINT 208, (FCMC(1,NB1),I=1,3), (DERV(1,NB1),I=1,3)	C2328800
PRINT 223, (ETC(1,NB1),I=1,3), (PHI(1,NB1),I=1,3)	C2328900
PRINT 224, (CBC(1,C),I=1,3)	C2329000
PRINT 221, ((XMC(1,J,0),J=1,3), I=1,3)	C2329100
PRINT 213	C2329200
IF(NMC.EQ.0) GC TO 41	C2329300
DO 6 M=1,NMO	C2329400
PRINT 210, M, HMUN(M), M, CLM(M)	C2329500
6 CONTINUE	C2329600
PRINT 213	C2329700
41 CONTINUE	C2329800
IF(NMODS.EQ.0) GO TO 40	C2329900
CALL UNPAC(SET,NSET,SFLX)	C2330000
MN = C	C2330100
DO 33 KK=1,NSET	C2330200
K = SET(NSET+1-KK)	C2330300
DO 34 I=1,3	C2330400
EP(I) = C	C2330500
FQC(I) = 0	C2330600
FQCC(I) = 0	C2330700
DO 34 J=1,3	C2330800
EI(I,J) = 0	C2330900
EID(I,J) = 0	C2331000
34 CONTINUE	C2331100
SFXMK = SFXM(K)	C2331200
DO 35 M=1,SFXMK	C2331300
MN = MN+1	C2331400
NFMN = NFER+MN	C2331500
CALL SCLV(THA(NFMN),FLA(1,MN),TEM1)	C2331600
CALL VECADD(EP,TEM1,EP)	C2331700
CALL SCLV(THAD(NFMN),FLQ(1,MN),TEM1)	C2331800
CALL VECADD(FQD,TEM1,FQD)	C2331900
CALL SCLC(THA(NFMN),FLE(1,1,MN),TEMS)	C2332000
CALL DYADD(EI,TEMS,EI)	C2332100
CALL SCLC(THAD(NFMN),FLE(1,1,MN),TEMS)	C2332200
CALL DYADD(EID,TEMS,EID)	C2332300
35 CONTINUE	C2332400

```

PRINT 226, (EP(I),I=1,3),(EPD(I,K),I=1,3) C2332500
PRINT 228, K,(FUD(I),I=1,3) C2332600
PRINT 229, ((EI(I,J),J=1,3),(EID(I,J),J=1,3),I=1,3) C2332700
PRINT 230, (FLIRC(I,K),I=1,3),(FLCRC(I,K),I=1,3) C2332800
PRINT 231,(DHM(I,K),I=1,3) C2332900
33 CONTINUE C2333000
PRINT 213 C2333100
40 CONTINUE C2333200
DO 9 I=1,NFER C2333300
5 PRINT 220, (I,THA(I),I,THAD(I),I,THADD(I),I,(QFC(J,I),J=1,3)) C2333400
N1 = NFER+1 C2333500
N2 = NFER+MN C2333600
IF(MK.EQ.0) RETURN C2333700
DO 12 I=N1,N2 C2333800
12 PRINT 225, I,THA(I),I,THAD(I),I,THADD(I) C2333900
2000 FORMAT (' CENTER OF MASS = ',3D17.8,/) C2334000
201 FORMAT (/,3X,'HBOCY = ',3D17.8,/) C2334100
202 FORMAT (3X,'HINERT = ',3D17.8) C2334200
203 FORMAT (' TIME = ',D15.8,/) C2334300
204 FORMAT (/,3X,'BODY ',I2,4X,'FCMC = ',3D17.8,3X,'FCMC = ',3D17.8) C2334400
205 FORMAT (14X,'CAC = ',3D17.8,3X,'CBC = ',3D17.8) C2334500
206 FORMAT (14X,'XMC = ',3D17.8,3X,'XIC = ',3D17.8) C2334600
207 FORMAT (14X,'ACC = ',3D17.8) C2334700
208 FORMAT (/,3X,'ORIGIN',4X,'FCMC = ',3D17.8,3X,'ACC = ',3D17.8) C2334800
209 FORMAT (' ANGULAR MOMENTUM = ',D20.8) C2334900
210 FORMAT (3X,'HMCM(' ,I2,') = ',D17.8,10X,'CLM(' ,I2,') = ',D17.8) C2335000
211 FORMAT (25X,3D17.8) C2335100
212 FORMAT (' SYSTEM INERTIA TENSOR = ',3D17.8) C2335200
213 FORMAT (' ') C2335300
214 FORMAT (14X,' HB = ',3D17.8,3X,' TK = ',D17.8) C2335400
215 FORMAT (/,' KINETIC ENERGY = ',D20.8) C2335500
216 FORMAT (14X,'PCS = ',3D17.8,3X,'VEL = ',3D17.8) C2335600
217 FORMAT (/,' LINEAR MOMENTUM = ',D20.8) C2335700
218 FORMAT (14X,' LM = ',3D17.8) C2335800
219 FORMAT (/,3X,'LBOCY = ',3D17.8) C2335900
220 FORMAT (3X,'THA(' ,I2,') = ',D13.6,3X,'THAD(' ,I2,') = ',D13.6,3X,'THA C2336000
*CD(' ,I2,') = ',D13.6,3X,'QFC(' ,I2,') = ',3D13.6) C2336100
221 FORMAT (14X,'XMC = ',3D17.8) C2336200
222 FORMAT (' TOTAL SYSTEM MASS = ',D17.8,/) C2336300
223 FORMAT (14X,'ETC = ',3D17.8,3X,'PHI = ',3D17.8) C2336400
224 FORMAT (14X,'CBC = ',3D17.8) C2336500
225 FORMAT (3X,'THA(' ,I2,') = ',D13.6,3X,'THAD(' ,I2,') = ',D13.6,3X,'THA C2336600
*CD(' ,I2,') = ',D13.6) C2336700
226 FORMAT (/,3X,'FLEXIBLE',3X,' EP = ',3D17.8,3X,'EPP = ',3D17.8) C2336800
228 FORMAT (3X,' BODY',I2,4X,' GD = ',3D17.8) C2336900
229 FORMAT (14X,' EI = ',3D17.8,3X,'EID = ',3D17.8) C2337000
230 FORMAT (14X,'FIR = ',3D17.8,3X,'FCR = ',3D17.8) C2337100
231 FORMAT (14X,'DHM = ',3D17.8) C2337200
RETURN C2337300
END C2337400

```

DATA SECTION

END DATA

```

C SUERCUTINE SIMQ(D,C,N,NC) C2400000
IMPLICIT REAL*8(A-H,O-Z,*) C2400100
LOGICAL LEQU C2400200
LOGICAL LBRUNGE , LTRANSI , LVDIV , LEQUIV , LTRAN C2400300
C2400400

```

```

*          LTRFNV , LRATE , LXDY , LETA , LTRFQU ,      C2400500
*          LQFDDT , LDCT , LANGLE , LSETUP , LSIMQ      C2400600
C          C2400700
DIMENSION D(1),C(1)                                    C2400800
DIMENSION DD(33,33),CC(33)                             C2400900
C          USED TO OBTAIN SOLUTION OF A SET OF SIMULTANEOUS LINEAR EQUATIONS C2401000
C          C*X = C                                       C2401100
C          C2401200
C          DESCRIPTION OF PARAMETERS                       C2401300
C          D = COEFFICIENT MATRIX STORED COLUMN-WISE DESTROYED DURING C2401400
C          COMPUTATION                                     C2401500
C          N = NUMBER OF EQUATIONS                        C2401600
C          ND = DIMENSION OF ARRAYS D AND C IN THE CALLING SUBROUTINE C2401700
C          C = VECTOR OF ORIGINAL CONSTANTS OF LENGTH N, THESE ARE C2401800
C          REPLACED BY THE SOLUTION VECTOR                C2401900
C          C2402000
COMMON /LDEBUG/ LRUNGE , LTRNSI , LVDIV , LEQUIV , LTRAN , C2402100
*          LTRFNV , LRATE , LXDY , LETA , LTRFQU ,      C2402200
*          LQFDDT , LDCT , LANGLE , LSETUP , LSIMQ      C2402300
EQUIVALENCE (LSIMQ,LFQU)                                C2402400
IF(LEQU) PRINT 107                                     C2402500
C          C2402600
C          TAKE ARRAY D FROM DIMENSION NDXND AS DEFINED IN CALLING SUBROUTINE C2402700
C          TO AN NXXN ARRAY                               C2402800
C          IJ = 0                                         C2402900
C          DO 14 J=1,N                                    C2403000
C          J3 = (J-1)*ND                                  C2403100
C          DO 15 I=1,N..                                  C2403200
C          IJ = IJ + 1                                     C2403300
C          C(IJ) = C(I+J3)                                 C2403400
C          IF(LEQU) DD(I,J) = D(IJ)                       C2403500
C          15 CONTINUE                                    C2403600
C          IF(LEQU) CC(J) = C(J)                          C2403700
C          14 CONTINUE                                    C2403800
C          C2403900
C          FORWARD SOLUTION                               C2404000
C          C2404100
C          TOL = 0.00                                     C2404200
C          KS = 0                                         C2404300
C          JJ = -N                                        C2404400
C          DO 65 J=1,N..                                  C2404500
C          JY = J+1                                       C2404600
C          JJ = JJ+N+1                                     C2404700
C          EIGA = 0.00                                     C2404800
C          IT = JJ-J                                       C2404900
C          DO 33 I=J,N..                                  C2405000
C          C2405100
C          SEARCH FOR MAXIMUM COEFFICIENT IN COLUMN      C2405200
C          C2405300
C          IJ = IT+1                                       C2405400
C          IF(DABS(EIGA) - DABS(D(IJ))) 21,33,33          C2405500
C          21 EIGA = D(IJ)                                  C2405600
C          IMAX = I                                         C2405700
C          33 CONTINUE                                    C2405800
C          C2405900
C          TEST FOR PIVOT LESS THAN TOLERANCE            C2406000
C          C2406100
C          IF(DABS(EIGA) - TOL) 35,35,40                 C2406200
C          35 KS = 1                                       C2406300
C          PRINT 103                                       C2406400

```

STCP	02476500
RETURN	C2406600
C	C2406700
C INTERCHANGE ROWS IF NECESSARY	C2406800
C	C2406900
40 I1 = J + N*(J-2)	C2407000
IT = IMAX - J	C2407100
DO 50 K=J,N	C2407200
I1 = I1 + N	C2407300
I2 = I1 + IT	C2407400
SAVE = D(I1)	C2407500
D(I1) = D(I2)	C2407600
D(I2) = SAVE	C2407700
C	C2407800
C DIVIDE EQUATION BY LEADING COEFFICIENT	C2407900
C	C2408000
50 C(I1) = C(I1)/BIGA	C2408100
SAVE = C(IMAX)	C2408200
C(IMAX) = C(J)	C2408300
C(J) = SAVE/BIGA	C2408400
C	C2408500
C ELIMINATE NEXT VARIABLE	C2408600
C	C2408700
IF(J-N) 55,70,55	C2408800
55 IQS = N*(J-1)	C2408900
DO 65 IX=JY,N	C2409000
IXJ = IQS + IX	C2409100
IT = J - IX	C2409200
DO 60 JX = JY,N	C2409300
IXJX = N*(JX-1) + IX	C2409400
JJX = IXJX + IT	C2409500
60 C(IXJX) = D(IXJX) - (D(IXJ)*D(JJX))	C2409600
65 C(IX) = C(IX) - (C(J)*C(IXJ))	C2409700
C	C2409800
C BACK SOLUTION	C2409900
C	C2410000
70 NY = N-1	C2410100
IT = N*N	C2410200
DO 80 J=1,NY	C2410300
IA = IT - J	C2410400
IB = N - J	C2410500
IC = N	C2410600
DO 80 K=1,J	C2410700
C(IB) = C(IB) - D(IA)*C(IC)	C2410800
IA = IA-N	C2410900
80 IC = IC-1	C2411000
C	C2411100
C	C2411200
IF(.NOT. LEQU) GO TO 1000	C2411300
TEST1 = C.DO	C2411400
DO 1 I=1,N	C2411500
TEST2 = C.DO	C2411600
DO 2 J=1,N	C2411700
2 TEST2 = TEST2 + D(I,J)*C(J)	C2411800
TEST2 = TEST2 - C(I)	C2411900
PRINT 100, I, TEST2	C2412000
TEST1 = TEST1 + TEST2**2	C2412100
1 CONTINUE	C2412200
TEST1 = CSQRT(TEST1)	C2412300
PRINT 100, TEST1	C2412400

```

1000 CONTINUE
C
100 FORMAT (6D15.5)
103 FORMAT (' MATRIX IS SINGULAR GARBAGE FOLLOWS ')
104 FORMAT (' O(.,I4,.) = XPN(.,I2,.,.,I2,.) = ',D20.10)
105 FORMAT (' ')
106 FORMAT (' C(.,I2,.) = ',D20.10)
107 FORMAT ('! SUBROUTINE SING ENTERED ')
108 FORMAT (' ERRCR IN ROW',I2,.' = ',D17.8)
109 FORMAT (' NCRM OF ERRCR VECTOR = ',D17.8)
RETURN
END

```

```

C2412500
C2412600
C2412700
C2412800
C2412900
C2413000
C2413100
C2413200
C2413300
C2413400
C2413500
C2413600

```

```

C
SUBRCUTINE RUNGE(T,H,Y,YC,N1,N2,TEM)
IMPLICIT REAL*8(A-H,O-Z,4)
LOGICAL LEQU
LOGICAL LRUNGE , LTRNSI , LVDIV , LEQUIV , LTRAN ,
* LTRFANV , LRATE , LXDY , LETA , LTRORQU ,
* LQFDCT , LDCT , LANGLE , LSETUP , LSIMO
COMMON /LDEBEG/ LRUNGE , LTRNSI , LVDIV , LEQUIV , LTRAN ,
* LTRFANV , LRATE , LXDY , LETA , LTRORQU ,
* LQFDCT , LDCT , LANGLE , LSETUP , LSIMO
EQUIVALENCE (LRUNGE,LEQU)
DIMENSION Y(1),YC(1),TEM(2,1)
C ACCEPTS SYSTEM STATE AT TIME T
C RETURNS SYSTEM STATE AT TIME T+H
C SYSTEM STATE Y(N)
C DERIVATIVE OF Y(N) IS YD(N)
C SUBRCUTINE DYN COMPUTES YD(N) AS REQUIRED BY RUNGE KUTTA
C NUMBER OF STATE VARIABLES IS N1 + N2
C N1 = NUMBER OF DYNAMIC EQUATIONS
C N2 = NUMBER OF EQUATIONS FROM TORQUE
C TEMPORARY STORAGE AREA TEM(2,N) NOT TO BE USED IN DYN
N = N1 + N2
IF(LEQU)PRINT 202
IF(LEQU)PRINT 200, (I,Y(I),I,YD(I),I,TEM(1,I),I,TEM(2,I),I=1,N)
K = 0
DO 1 I=1,N
TEM(1,I) = Y(I)
1 TEM(2,I) = YD(I)
CD = H/2
A = CD
2 T = T + CD
3 DO 4 I=1,N
4 Y(I) = TEM(1,I)+A*YD(I)
IF(LEQU)PRINT 201, (I,Y(I),I,YD(I),I,TEM(1,I),I,TEM(2,I),I=1,N)
CALL DYN(Y,YD,N1)
IF(LEQU)PRINT 200, (I,Y(I),I,YD(I),I,TEM(1,I),I,TEM(2,I),I=1,N)
K=K+1
IF(K.EQ.3) GO TO 7
DO 5 I=1,N
5 TEM(2,I) = TEM(2,I) + 2*YC(I)
IF(K.EQ.1) GO TO 3
A = H
GO TO 2

```

```

C2500000
C2500100
C2500200
C2500300
C2500400
C2500500
C2500600
C2500700
C2500800
C2500900
C2501000
C2501100
C2501200
C2501300
C2501400
C2501500
C2501600
C2501700
C2501800
C2501900
C2502000
C2502100
C2502200
C2502300
C2502400
C2502500
C2502600
C2502700
C2502800
C2502900
C2503000
C2503100
C2503200
C2503300
C2503400
C2503500
C2503600
C2503700
C2503800
C2503900
C2504000
C2504100
C2504200

```

```

7 A = A/6
CO 6 I=1,N
TEM(2,I) = A*(TEM(2,I) + YD(I))
E Y(I) = TEM(1,I) + TEM(2,I)
IF(LEQU)PRINT 201, (I,Y(I),I,YD(I),I,TEM(1,I),I,TEM(2,I),I=1,N)
(CALL DYN(Y,YD,N))
IF(LEQU)PRINT 200, (I,Y(I),I,YD(I),I,TEM(1,I),I,TEM(2,I),I=1,N)
200 FORMAT (' IN RUNGE Y(',I2,') = ',D18.8,' YD(',I2,') = ',D18.8,'
*EM(1,',I2,') = ',D18.8,' TEM(2,',I2,') = ',D18.8)
201 FORMAT (' OUT RUNGE Y(',I2,') = ',D18.8,' YD(',I2,') = ',D18.8,'
*EM(1,',I2,') = ',D18.8,' TEM(2,',I2,') = ',D18.8)
202 FORMAT ('1')
RETURN
END

```

```

C SUBROUTINE UNCAGE(SCG,SC,T,TUG)
IMPLICIT REAL*8(A-H,O-Z,S)
INTEGER SCG,SC(1),SCG1
DIMENSION TUG(1)
C TEST TO SEE IF TIME TO UNCAGE ANY DEGREE OF FREEDOM
C IF YES DO SO AND REINLEER SC ARRAY
SCG1 = SCG
CO 6 I=1,SCG1
IF(T.LT.TUG(SC(I))) GO TO 6
SCG = SCG-1
PRINT 100, SC(I),T,SCG
SC(I) = 0
E CONTINUE
IF(SCG.EQ.SC) RETURN
J=0
CO 7 I=1,SCG1
IF(SC(I).EQ.0) GO TO 7
J=J+1
SC(J)=SC(I)
IF(I.EQ.J) GO TO 7
SC(I) = 0
7 CONTINUE
100 FORMAT (' NOTICE ABOUT FREE VECTOR ',I2, ' UNCAGED AT T = ',E15.5,
*' NOTICE STILL CAGED AECLT ',I2, ' FREE VECTORS ')
RETURN
END

```

```

SUBROUTINE COMPR(XMN,THADD,N,SC,SCG,LG)
IMPLICIT REAL*8(A-H,O-Z,S)
DIMENSION XMN(33,1),THADD(1)
INTEGER SC(1),SCG
LOGICAL LG(1)
SET UP LOGIC FLAGS
NS = N + SCG
CO 7 I=1,NS
7 LG(I) = .TRUE.

```



```

CO 8 I=1,SCG
LG(SC(I)) = .FALSE.
II = 0
CO 9 I=1,NS
IF(.NOT.LG(I)) GO TO 9
II = II+1
THADD(II) = THADD(I)
JJ = 0
CO 10 J=1,NS
IF(.NOT.LG(J)) GO TO 10
JJ = JJ+1
XMN(II,JJ) = XMN(I,J)
10 CONTINUE
9 CONTINUE
RETURN
END

```

```

C2E03600
C2E03700
C2E03800
C2E03900
C2E04000
C2E04100
C2E04200
C2E04300
C2E04400
C2E04500
C2E04600
C2E04700
C2E04800
C2E04900
C2E05000
C2E05100

```

```

SUBROUTINE UNPRS(THADD,N,SCG,LG)
IMPLICIT REAL*8(A-H,O-Z,*)
DIMENSION THADD(1)
INTEGER SCG
LOGICAL LG(1)
I2 = N
NN = N+SCG
CO 11 I=1,NN
II = NN+1-I
IF(LG(II)) GO TO 12
THADD(II) = C.O
GO TO 11
12 THADD(II) = THADD(12)
I2 = I2-1
11 CONTINUE
RETURN
END

```

```

C2E05200
C2E05300
C2E05400
C2E05500
C2E05600
C2E05700
C2E05800
C2E05900
C2E06000
C2E06100
C2E06200
C2E06300
C2E06400
C2E06500
C2E06600
C2E06700
C2E06800

```

```

SUBROUTINE COMPAC(SET,NSET,S)
IMPLICIT REAL*8(A-H,O-Z,*)
INTEGER SET(NSET),A(24),S,AB
DATA A/ Z1 , Z2 , Z4 , Z8 ,
* Z10 , Z20 , Z40 , Z80 ,
* Z100 , Z200 , Z400 , Z800 ,
* Z1000 , Z2000 , Z4000 , Z8000 ,
* Z10000 , Z20000 , Z40000 , Z80000 ,
* Z100000 , Z200000 , Z400000 , Z800000 /
DATA AB/ Z1000000 /
C TAKES THE SET OF INTEGERS STORED IN SET(NSET) AND COMPACTS
C THEM INTO THE SINGLE CODED INTEGER WORD S. THE SET OF
C INTEGERS IN ARRAY SET MUST BE DISTINCT FROM EACH OTHER
C AND LIE BETWEEN 1 AND 24 INCLUSIVE.
S = NSET * AB
IF(NSET.EQ.0)GO TO 2

```

```

C2E06900
C2E07000
C2E07100
C2E07200
C2E07300
C2E07400
C2E07500
C2E07600
C2E07700
C2E07800
C2E07900
C2E08000
C2E08100
C2E08200
C2E08300
C2E08400
C2E08500

```

CO 1 K=1,NSET	C2608600
1 S= S + A(SET(K))	C2608700
RETURN	C2608800
2 SET(I)=C	C2608900
RETURN	C2609000
END	C2609100
SUBROUTINE UNPAC(SET,NSET,S)	C2609200
IMPLICIT REAL*8(A-F,O-Z,I)	C2609300
INTEGER SET(1),A(24),S,AB,TS	C2609400
DATA A/ Z1 , Z2 , Z4 , Z8 ,	C2609500
* Z10 , Z20 , Z40 , Z80 ,	C2609600
* Z100 , Z200 , Z400 , Z800 ,	C2609700
* Z1000 , Z2000 , Z4000 , Z8000 ,	C2609800
* Z10000 , Z20000 , Z40000 , Z80000 ,	C2609900
* Z100000 , Z200000 , Z400000 , Z800000/	C2610000
DATA AB/ Z1000000/	C2610100
C	C2610200
C DECODES THE CODED WORD S TO OBTAIN THE ELEMENTS OF SET(NSET)	C2610300
C ELEMENTS OF SET RETURNED IN DECREASING ORDER OF MAGNITUDE	C2610400
C SET(1).GT.SET(2).GT. .... .GT. SET(NSET)	C2610500
NSET = S/AB	C2610600
IF(NSET.EQ.0) GO TO 5	C2610700
I=C	C2610800
TS= S-NSET*AB	C2610900
KN=25	C2611000
CO 1 K=1,24	C2611100
IF ( TS-A(KN-K) ) 1,3,2	C2611200
2 I=I+1	C2611300
SET(I) = KN-K	C2611400
TS = TS-A(KN-K)	C2611500
1 CONTINUE	C2611600
3 I=I+1	C2611700
SET(I)=KN-K	C2611800
4 RETURN	C2611900
5 SET(I)=0	C2612000
RETURN	C2612100
END	C2612200
INTEGER FUNCTION KTO(N,J,K)	C2612300
IMPLICIT REAL*8(A-F,O-Z,I)	C2612400
IF(K.GE.J) GO TO 1	C2612500
KTC=K*(N-1)+J+1-K*(K-1)/2	C2612600
GO TO 2	C2612700
KTC = J*(N-1) + K + 1 - J*(J-1)/2	C2612800
2 RETURN	C2612900
END	C2613000

```

INTEGER FUNCTION KT1(N,J,K)
IMPLICIT REAL*8(A-F,O-Z,*)
IF(K.GE.J) GO TO 1
KT1=(K-1)*(N-1)+J-(K-1)*(K-2)/2
GO TO 2
1 KT1 = (J-1)*(N-1) + K - (J-1)*(J-2)/2
2 RETURN
END

```

C2613100  
C2613200  
C2613300  
C2613400  
C2613500  
C2613600  
C2613700  
C2613800

```

LOGICAL FUNCTION CTAIN(I,S,N)
IMPLICIT REAL*8(A-F,O-Z,*)
INTEGER S(N)
C
C CTAIN = .TRUE. IF BODY LABEL I CONTAINED IN SET S(N)
C = .FALSE. IF NOT
C
C ELEMENTS IN SET(N) MUST BE POSITIVE NON-ZERO INTEGERS IN
C EITHER ASCENDING OR DECENDING ORDER OF MAGNITUDE
C
IF(N.EQ.0.OR.I.EQ.0) GO TO 1
N1 = N+1
IF(S(N) - S(1)) 2,2,3
2 K = N
L = -1
GO TO 4
3 K = 1
L = 1
4 IF(S(K)-I) 5,6,1
5 K = K+L
IF(K.EQ.0.OR.K.EQ.N1) GO TO 1
GO TO 4
1 CTAIN = .FALSE.
RETURN
6 CTAIN = .TRUE.
RETURN
END

```

C2613900  
C2614000  
C2614100  
C2614200  
C2614300  
C2614400  
C2614500  
C2614600  
C2614700  
C2614800  
C2614900  
C2615000  
C2615100  
C2615200  
C2615300  
C2615400  
C2615500  
C2615600  
C2615700  
C2615800  
C2615900  
C2616000  
C2616100  
C2616200  
C2616300  
C2616400

```

SUBROUTINE VECTRN (VA,XMT,VAD)
PERFORMS TRANSFORMATION OF COORDINATES FOR VECTORS
XMT = MATRIX, TRANSFORMS VECTORS TO COMPUTING FRAME COORDINATES
FROM BODY LAMBDA FIXED COORDINATES
VA = VECTOR INPUTED RELATIVE TO BODY LAMBDA FIXED COORDINATES
VAD = VECTOR COMPONENTS COMPUTED RELATIVE TO COMPUTING FRAME
CORROTEQUATION SOLVED IS
VAD = (XMT) * VA
IMPLICIT REAL*8(A-F,O-Z,*)
DIMENSION VA(3),XMT(3,3),VAD(3)
DO 1 I=1,3
VAD(I)=0

```

C2700000  
C2700100  
C2700200  
C2700300  
C2700400  
C2700500  
C2700600  
C2700700  
C2700800  
C2700900  
C2701000  
C2701100  
C2701200  
C2701300  
C2701400  
C2701500

```

CO 1 J=1,3
1 VAC(I)=VAD(I)+XMT(I,J)*VA(J)
RETURN
END

```

C2701600  
C2701700  
C2701800  
C2701900

```

SUBROUTINE TENTRN(XI,XMT,XID)
C TRANSFORM 3X3 TENSORS WITH CHECK FOR SYMMETRY
IMPLICIT REAL*8(A-H,O-Z,S)
DIMENSION XI(3,3),XMT(3,3),XID(3,3)
LOGICAL FLAG
FLAG = .TRUE.
IF( XI(1,2).EQ.XI(2,1)
* .AND.XI(1,3).EQ.XI(3,1)
* .AND.XI(2,3).EQ.XI(3,2)) GO TO 2
FLAG = .FALSE.
2 CO 1 I=1,3
II = I
IF(FLAG) GO TO 3
II = 1
3 CO 1 J=II,3
XID(I,J) = 0
CO 1 L=1,3
CO 1 M=1,3
XID(I,J) = XID(I,J) + XMT(I,L)*XI(L,M)*XMT(J,M)
1 CONTINUE
IF(.NOT.FLAG) RETURN
XID(2,1) = XID(1,2)
XID(3,1) = XID(1,3)
XID(3,2) = XID(2,3)
RETURN
END

```

C2702000  
C2702100  
C2702200  
C2702300  
C2702400  
C2702500  
C2702600  
C2702700  
C2702800  
C2702900  
C2703000  
C2703100  
C2703200  
C2703300  
C2703400  
C2703500  
C2703600  
C2703700  
C2703800  
C2703900  
C2704000  
C2704100  
C2704200  
C2704300  
C2704400  
C2704500

```

SUBROUTINE VECNRM (V)
IMPLICIT REAL*8(A-H,O-Z,S)
DIMENSION V(3)
C USED TO NORMALIZE VECTORS TO UNITY
A = V(1)**2 + V(2)**2 + V(3)**2
IF(A.NE.0) GO TO 1
PRINT *,C
100 FORMAT (' GIMBAL LOCK CONDITION. NORMALIZATION SKIPPED ERRORS PRO
*EALY FOLLOW ')
RETURN
1 A = DSQRT(A)
CO 2 I=1,3
2 V(I) = V(I)/A
RETURN
END

```

C2704600  
C2704700  
C2704800  
C2704900  
C2705000  
C2705100  
C2705200  
C2705300  
C2705400  
C2705500  
C2705600  
C2705700  
C2705800  
C2705900  
C2706000

```

SUBROUTINE MATMUL (A,B,C,N)                                C2706100
IMPLICIT REAL*8(A-H,O-Z,S)                               C2706200
DIMENSION A(N,N),E(N,N),C(N,N)                          C2706300
C=0.00000000000000000000000000000000000000000000000 C2706400
C=0.00000000000000000000000000000000000000000000000 C2706500
A*B = C                                                    C2706600
DO 1 I=1,N                                                C2706700
DO 1 J=1,N                                                C2706800
C(I,J) = 0.0000000000000000000000000000000000000000 C2706900
DO 2 L=1,N                                                C2707000
C(I,J) = C(I,J) + A(I,L)*B(L,J)                         C2707100
1 CONTINUE                                                C2707200
RETURN                                                    C2707300
END

```

```

SUBROUTINE TRNSPS (XMT)                                  C2707400
C=0.00000000000000000000000000000000000000000000000 C2707500
IMPLICIT REAL*8(A-H,O-Z,S)                               C2707600
DIMENSION XMT(3,3),TEMP(3,3)                             C2707700
DO 1 I=1,3                                                C2707800
DO 1 J=1,3                                                C2707900
1 TEMP(I,J) = XMT(I,J)                                   C2708000
DO 2 I=1,3                                                C2708100
DO 2 J=1,3                                                C2708200
2 XMT(I,J) = TEMP(J,I)                                   C2708300
RETURN                                                    C2708400
END                                                        C2708500

```

```

SUBROUTINE ROT(A,C,JA,TME)                               C2708600
IMPLICIT REAL*8(A-H,O-Z,S)                               C2708700
C=0.00000000000000000000000000000000000000000000000 C2708800
GENERAL EULER ANGLE ROTATION MATRIX                      C2708900
DIMENSION TME(3,3),V(3),W(3)                            C2709000
S = C/DABS(C)                                             C2709100
DO 24 N=1,3                                               C2709200
W(N) = S                                                  C2709300
24 V(N) = .CDO                                           C2709400
JJA = IABS(JA)                                           C2709500
V(JJA) = A*JA/JJA                                       C2709600
W(JJA) = 1.0000000000000000000000000000000000000000 C2709700
TME(1,1) = DSQRT(1.00 - V(2)**2 - V(3)**2)*W(1)         C2709800
TME(2,2) = DSQRT(1.00 - V(1)**2 - V(3)**2)*W(2)         C2709900
TME(3,3) = DSQRT(1.00 - V(1)**2 - V(2)**2)*W(3)         C2710000
TME(1,2) = -V(3)                                         C2710100
TME(2,1) = V(3)                                          C2710200
TME(1,3) = V(2)                                          C2710300
TME(3,1) = -V(2)                                         C2710400
TME(2,3) = -V(1)                                         C2710500
TME(3,2) = V(1)                                          C2710600
RETURN                                                    C2710700
END

```

```

C          SUBROUTINE VECADD(V1,V2,S)
C          ADDS VECTOR V1 TO V2 RESULT IN S
C          IMPLICIT REAL*8(A-H,O-Z,S)
C          DIMENSION V1(3),V2(3), S(3)
C          S(1) = V1(1) + V2(1)
C          S(2) = V1(2) + V2(2)
C          S(3) = V1(3) + V2(3)
C          RETURN
C          END

```

```

C2800000
C2800100
C2800200
C2800300
C2800400
C2800500
C2800600
C2800700
C2800800
C2800900

```

```

C          SUBROUTINE VECSUB(V1,V2,C)
C          SUBTRACTS VECTORS V1-V2=C
C          IMPLICIT REAL*8(A-H,O-Z,S)
C          DIMENSION V1(3),V2(3),C(3)
C          C(1) = V1(1) - V2(1)
C          C(2) = V1(2) - V2(2)
C          C(3) = V1(3) - V2(3)
C          RETURN
C          END

```

```

C2801000
C2801100
C2801200
C2801300
C2801400
C2801500
C2801600
C2801700
C2801800

```

```

C          SUBROUTINE SCLV(SC,V,P)
C          SCALAR * VECTOR
C          IMPLICIT REAL*8(A-H,O-Z,S)
C          DIMENSION V(3),P(3)
C          P(1) = SC*V(1)
C          P(2) = SC*V(2)
C          P(3) = SC*V(3)
C          RETURN
C          END

```

```

C2801900
C2802000
C2802100
C2802200
C2802300
C2802400
C2802500
C2802600
C2802700

```

```

C          SUBROUTINE VECDDT(V1,V2,D)
C          VECTOR DOT PRODUCT
C          IMPLICIT REAL*8(A-H,O-Z,S)
C          DIMENSION V1(3),V2(3)
C          D = V1(1)*V2(1) + V1(2)*V2(2) + V1(3)*V2(3)
C          RETURN
C          END

```

```

C2802800
C2802900
C2803000
C2803100
C2803200
C2803300
C2803400

```

```

C          SUBROUTINE VECROS (V1,V2,C)
C          VECTOR CROSS PRODUCT C = V1 X V2
C          IMPLICIT REAL*8(A-H,O-Z,S)
C          DIMENSION V1(3),V2(3),C(3)
C          C(1) = V1(2)*V2(3)- V1(3)*V2(2)

```

```

C2803500
C2803600
C2803700
C2803800
C2803900

```

```

C(2) = V1(3)*V2(1) - V1(1)*V2(3)
C(3) = V1(1)*V2(2) - V1(2)*V2(1)
RETURN
END

```

C2804000  
C2804100  
C2804200  
C2804300

```

SUBROUTINE TRIPVP(V1,V2,V)

```

C2804400

```

C
C   COMPLETE STANDARD VECTOR TRIPLE PRODUCT

```

C2804500  
C2804600

```

C
C       V = V1X(V1XV2)
C       = V1*(V1.V2) - V2*(V1.V1)

```

C2804700  
C2804800  
C2804900

```

C
C   IMPLICIT REAL*8(A-H,O-Z,S)
C   DIMENSION V1(3),V2(3),V(3)

```

C2805000  
C2805100  
C2805200

```

A = V1(1)*V2(1) + V1(2)*V2(2) + V1(3)*V2(3)
E = V1(1)*V1(1) + V1(2)*V1(2) + V1(3)*V1(3)

```

C2805300  
C2805400

```

V(1) = V1(1)*A - V2(1)*B
V(2) = V1(2)*A - V2(2)*B

```

C2805500  
C2805600

```

V(3) = V1(3)*A - V2(3)*B

```

C2805700

```

RETURN
END

```

C2805800  
C2805900

```

SUBROUTINE DYADD(C1,D2,D)

```

C2806000

```

C   ADDS TWO DYADS

```

C2806100

```

C       C = D1 + C2

```

C2806200

```

C   IMPLICIT REAL*8(A-H,C-Z,S)

```

C2806300

```

C   DIMENSION D1(3,3), D2(3,3), D(3,3)

```

C2806400

```

DO 1 I=1,3

```

C2806500

```

DO 1 J=1,3

```

C2806600

```

1 D(I,J) = D1(I,J) + D2(I,J)

```

C2806700

```

RETURN

```

C2806800

```

END

```

C2806900

```

SUBROUTINE SCLC(A,C,T)

```

C2807000

```

C   IMPLICIT REAL*8(A-H,O-Z,S)

```

C2807100

```

C   DIMENSION D(3,3),T(3,3)

```

C2807200

```

C   MULTIPLY SCALAR BY A TENSOR

```

C2807300

```

T(1,1) = A*D(1,1)

```

C2807400

```

T(2,1) = A*D(2,1)

```

C2807500

```

T(3,1) = A*D(3,1)

```

C2807600

```

T(1,2) = A*D(1,2)

```

C2807700

```

T(2,2) = A*D(2,2)

```

C2807800

```

T(3,2) = A*D(3,2)

```

C2807900

```

T(1,3) = A*D(1,3)

```

C2808000

```

T(2,3) = A*D(2,3)

```

C2808100

```

T(3,3) = A*D(3,3)

```

C2808200

```

RETURN

```

C2808300

```

END

```

C2808400

SUBROUTINE DYDOTV(A,V,D)	C2808500
SCALAR DOT PRODUCT OF DYAD AND VECTOR	C2808600
D = A.V	C2808700
IMPLICIT REAL*8(A-H,O-Z,S)	C2808800
DIMENSION D(3),A(3,3),V(3)	C2808900
DO 1 I=1,3	C2809000
C(I) = 0	C2809100
DO 1 J=1,3	C2809200
1 C(I) = C(I) + A(I,J)*V(J)	C2809300
RETURN	C2809400
END	C2809500
SUBROUTINE VXDYQV(V1,DY,V)	C2809600
COMPUTES VECTOR X (DYAD . VECTOR)	C2809700
V = V1 X (DY . V1)	C2809800
IMPLICIT REAL*8(A-H,O-Z,S)	C2809900
DIMENSION V1(3),V2(3),V(3),DY(3,3)	C2810000
DO 1 K=1,3	C2810100
V2(K) = 0.D0	C2810200
DO 1 J=1,3	C2810300
1 V2(K) = V2(K) + DY(K,J)*V1(J)	C2810400
V(1) = V1(2)*V2(3) - V1(3)*V2(2)	C2810500
V(2) = V1(3)*V2(1) - V1(1)*V2(3)	C2810600
V(3) = V1(1)*V2(2) - V1(2)*V2(1)	C2810700
RETURN	C2810800
END	C2810900
SUBROUTINE DYTQV(D,X1,X)	C2811000
USE TO TAKE SCALAR DOT PRODUCT OF TRANSPOSE OF	C2811100
TENSOR D WITH VECTOR X1	C2811200
NEEDED SINCE TENSORS IN SYMMERIC MATRIX OF INERTIA TENSORS ARE IN	C2811300
NON-SYMMETRIC	C2811400
IMPLICIT REAL*8(A-H,O-Z,S)	C2811500
DIMENSION D(3,3),X1(3),X(3)	C2811600
DO 1 I=1,3	C2811700
X(I) = 0	C2811800
DO 1 J=1,3	C2811900
1 X(I) = X(I) + D(J,I)*X1(J)	C2812000
CONTINUE	C2812100
RETURN	C2812200
END	C2812300
SUBROUTINE VCDYQV(V1,DY,V2,X)	C2812400
COMPUTES THE SCALAR TRIPLE PRODUCT	C2812500
VECTOR . (DYAD . VECTOR)	C2812600
V1.(DY.V2)	C2812700
IMPLICIT REAL*8(A-H,C-Z,S)	C2812800
DIMENSION V1(3),DY(3,3),V2(3),TEM(3)	C2812900



```

CO 1 K=1,3 C2E13000
TEM(K) = 0.D0 C2E13100
CO 1 J=1,3 C2E13200
1 TEM(K) = TEM(K) + DY(K,J)*V2(J) C2E13300
X = C C2E13400
CO 2 J=1,3 C2E13500
2 X = X + V1(J)*TEM(J) C2E13600
RETURN C2E13700
END C2E13800

```

```

SUERCUTINE DYCF(V,C) C2E13900
C TRANSFORMS VECTOR V1 INTO SKEW DYAD C2E14000
IMPLICIT REAL*8(A-H,O-Z,*) C2E14100
DIMENSION V(3),C(3,3) C2E14200
C(1,1) = 0 C2E14300
D(1,2) = V(3) C2E14400
C(1,3) = -V(2) C2E14500
C(2,1) = -V(3) C2E14600
C(2,2) = 0 C2E14700
C(2,3) = V(1) C2E14800
C(3,1) = V(2) C2E14900
C(3,2) = -V(1) C2E15000
C(3,3) = 0 C2E15100
RETURN C2E15200
END C2E15300

```

```

SUERCUTINE SLEOP(V1,V2,XM,D) C2E15400
IMPLICIT REAL*8(A-H,O-Z,*) C2E15500
DIMENSION V1(3),V2(3),D(3,3) C2E15600
C USED TO COMPUTE THE PSUECC INERTIA TENSOR C2E15700
C OF BODY LAMBA WITH RESPECT TO THE ORIGIN OF NEST K-1 AND C2E15800
C THE HINGE POINT I-1 WHICH IS ON THE TOPOLOGICAL PATH FROM C2E15900
C BODY 1 TO BODY LAMBA C2E16000
C BLOCK G SUPPER GAMBA,SUE K-1,I-1 EQUATION 2-55 OF X-732-71-70 C2E16100
C D = XM*((V1.V2)*1 - V2 V1) C2E16200
C C2E16300
C XM - SCALAR C2E16400
C V1 - VECTOR C2E16500
C V2 - VECTOR C2E16600
C 1 - UNIT DYAD C2E16700
C * - SCALAR MULTIPLICATION C2E16800
C . - VECTOR SCALAR MULTIPLICATION C2E16900
C ELANK - TENSOR MULTIPLICATION C2E17000
C NOTE THAT IN GENERAL THE PSUEDD INERTIA TENSOR IS NON SYMMETRIC C2E17100
C C2E17200
C(1,1) = XM*(V1(2)*V2(2) + V1(3)*V2(3)) C2E17300
C(1,2) = -XM*V2(1)*V1(2) C2E17400
C(1,3) = -XM*V2(1)*V1(3) C2E17500
C(2,1) = -XM*V2(2)*V1(1) C2E17600
C(2,2) = XM*(V2(1)*V1(1) + V2(3)*V1(3)) C2E17700
C(2,3) = -XM*V2(2)*V1(3) C2E17800
C(3,1) = -XM*V2(3)*V1(1) C2E17900

```

```

C(3,2) = -XM*V2(3)*V1(2)
C(3,3) = XM*(V2(1)*V1(1) + V2(2)*V1(2))
RETURN
END

```

```

C2818000
C2818100
C2818200
C2818300

```

```

C
SUBROUTINE QUTMUL (Q1,Q2,P)
IMPLICIT REAL*8(A-H,O-Z,4)
DIMENSION Q1(4),Q2(4),P(4)
MULTIPLIES TWO QUATERNIONS AND PUTS PRODUCT IN P
      P=Q1*Q2
WHERE
      * = QUATERNION MULTIPLICATION
A0 = Q1(1)
A1 = Q1(2)
A2 = Q1(3)
A3 = Q1(4)
E0 = Q2(1)
E1 = Q2(2)
E2 = Q2(3)
E3 = Q2(4)
P(1) = A0*E0 - A1*E1 - A2*E2 - A3*E3
P(2) = A0*E1 + A1*E0 + A2*E3 - A3*E2
P(3) = A0*E2 - A1*E3 + A2*E0 + A3*E1
P(4) = A0*E3 + A1*E2 - A2*E1 + A3*E0
RETURN
END

```

```

C2900000
C2900100
C2900200
C2900300
C2900400
C2900500
C2900600
C2900700
C2900800
C2900900
C2901000
C2901100
C2901200
C2901300
C2901400
C2901500
C2901600
C2901700
C2901800
C2901900
C2902000
C2902100

```

```

SUBROUTINE QUATOP(QF,THA,ZT)
IMPLICIT REAL*8(A-H,O-Z,4)
DIMENSION QF(3),ZT(4)
COMPUTES ROTATION QUATERNION FROM EIGENVECTOR QF AND ROTATION
ANGLE THA, EQUATION A-11 OF X-732-71-89
ZT(1) = DCOS(THA/2)
S = DSIN(THA/2)
DO 1 J=2,4
1 ZT(J) = QF(J-1)*S
RETURN
END

```

```

C2902200
C2902300
C2902400
C2902500
C2902600
C2902700
C2902800
C2902900
C2903000
C2903100
C2903200

```

```

SUBROUTINE TRANSO(GML,SL)
IMPLICIT REAL*8(A-H,O-Z,4)
DIMENSION GML(4),SL(3,3)
MATRIX OPERATOR CONSTRUCTS THE QUATERNION TRANSFORMATION MATRIX
FROM THE COMPONENTS OF THE QUATERNION GML(K)
THE EQUATION EVALUATE IS THE TRANSPOSE OF A-18 OF X-732-71-89
      THAT IS, LET

```

```

C2903300
C2903400
C2903500
C2903600
C2903700
C2903800
C2903900
C2904000

```

```

C           GML(K) = (E0,E1,E2,E3)                                (2904100
C           THEN                                                    (2904200
C           (2904300
C   EC**2+E1**2-E2**2-E3**2      2*(EC*EJ+E1*E2)      2*(E1*E3-E0*E2)      (2904400
C   2*(E1*E2-E0*E3)      E0**2-E1**2+E2**2-E3**2      2*(E2*E3+E0*E1)      (2904500
C   2*(E1*E3+E0*E2)      2*(E2*E3-E0*E1)      E0**2-E1**2-E2**2+E3**2 (2904600
C           (2904700
C           SL(1,1)      SL(1,2)      SL(1,3)                                (2904800
C   =      SL(2,1)      SL(2,2)      SL(2,3)                                (2904900
C           SL(3,1)      SL(3,2)      SL(3,3)                                (2905000
C           (2905100
C           (2905200
E00 = GML(1)*GML(1)                                (2905300
E01 = GML(1)*GML(2)                                (2905400
E02 = GML(1)*GML(3)                                (2905500
E03 = GML(1)*GML(4)                                (2905600
E11 = GML(2)*GML(2)                                (2905700
E12 = GML(2)*GML(3)                                (2905800
E13 = GML(2)*GML(4)                                (2905900
E22 = GML(3)*GML(3)                                (2906000
E23 = GML(3)*GML(4)                                (2906100
E33 = GML(4)*GML(4)                                (2906200
SL(1,1) = E00 + E11 - E22 - E33                    (2906300
SL(1,2) = 2*(E03 + E12)                            (2906400
SL(1,3) = 2*(E13 - E02)                            (2906500
SL(2,1) = 2*(E12 - E03)                            (2906600
SL(2,2) = E00 - E11 + E22 - E33                    (2906700
SL(2,3) = 2*(E23 + E01)                            (2906800
SL(3,1) = 2*(E13 + E02)                            (2906900
SL(3,2) = 2*(E23 - E01)                            (2907000
SL(3,3) = E00 - E11 - E22 + E33                    (2907100
RETURN                                              (2907200
END                                                (2907300

```

**APPENDIX B**

**SUPPORT PROGRAMS FOR N-BOD2**

**FLEXIBLE BODY CAPABILITY**

**APPENDIX B**  
**SUPPORT PROGRAMS FOR N-BOD2**  
**FLEXIBLE BODY CAPABILITY**

The flexible body options available in N-BOD2 require the user to obtain several resultant mode dependent parameters. These parameters will, for most applications, be computed from lumped parameter data obtained via the standard in-house finite element program. NASTRAN is the standard program at NASA/GSFC; however, other installations make use of other programs. Therefore, as a guide for the interested user, the following two programs are submitted and can either be used directly or be extensively modified to suit the situation existing at the particular computer installation that N-BOD2 will be applied.

The first program is actually the NASTRAN Executive Control Deck for a specially written DMAP program. This DMAP program will output all lumped parameter data necessary for the generation of the required N-BOD2 resultant mode dependent parameters. The program DISCOS referred to in the DMAP listing is another flexible body program being prepared by H. P. Frisch for release at GSFC. It requires similar data but in a completely different format.

This DMAP program is operational on the Macneal-Schwendler version 38 of NASTRAN and the Level 15.5 NASA version with one trivial change in the functional module GP4. (Built-in NASTRAN error messages will point out the change.)

The DMAP program was written so that, in addition to modes and frequencies (real), NASTRAN could be forced to output modal mass, stiffness, and damping matrices. Several print and punch options were also built in so that the user could see more clearly some of the resultant mass stiffness and damping matrices internally generated by NASTRAN. The interested user is referred to a NASTRAN Level 15 User's Manual (Reference 1) and Programmer's Manual (Reference 2) for a detailed description of the DMAP program. Basically it is a combination of rigid formats 3, 9, and 12.

```

ID      FRISCH, DISCOS N-BOD2 NASTRAN
DIAG    i4
TIME    6000
APP     DMAP
BEGIN   DISCOS AND N-BOD2 DATA VIA NASTRAN $
FILE    $
GP1     GEOM1,GEOM2,/GPL,EQEXIN,GPDT,CSTM,BGPDT,SIL/V,N,LUSET/ C,N,123/
        V,N,NOGPDT $
SAVE    LUSET,NOGPDT $
CHKPNT  GPL,EQEXIN,GPDT,CSTM,BGPDT,SIL $
PARAM   //C,N,NOP/V,Y,PRTGD=1 $
COND    LBL10,PRTGD $
TABPT   GPL,EQEXIN,GPDT,BGPDT,SIL// $
TABPRT  GPL//C,N,GPL $
TABPRT  GPDT//C,N,GPDT $
TABPRT  BGPDT//C,N,BGPDT $
LABEL   LBL10 $
PURGE   USET,GM,GO,KAA,BAA,MAA,K4AA,PST,KFS,OP,EST/NOGPDT $
CHKPNT  USET,GM,GO,KAA,BAA,MAA,K4AA,PST,KFS,OP,EST $
GP2     GEOM2,EQEXIN/ECT $
CHKPNT  ECT $
TA1,    ,ECT,EPT,BGPDT,SIL,,CSTM/EST,,GEI,ECPT,GPCT/V,N,LUSET/ C,N,
        123/V,N,NOSIMP/C,N,O/V,N,NOGENL/V,N,GENEL $
SAVE    NOGENL,NOSIMP,GENEL $
CHKPNT  EST,GEI,ECPT,GPCT $
PURGE   K4GG,GPST,OGPST,MGG,BGG, K4NN,K4FF,K4AA,MNN,MFF,MAA,BNN,BFF,
        BAA,KGGX/NOSIMP/ OGPST/GENEL $
CHKPNT  K4GG,GPST,OGPST,MGG,BGG,K4NN,K4FF,K4AA,MNN,MFF,MAA,BNN,BFF,
        BAA,KGGX $
SMA1    CSTM,MPT,ECPT,GPCT, DIT/KGGX,K4GG,GPST/V,N,NOGENL/ V,N,NOK4GG$
SAVE    NOK4GG $
CHKPNT  KGGX,K4GG,GPST $
PARAM   //C,N,NOP/V,Y,PRTKG=-1 $
COND    LBL14,PRTKG $
MATPRN  KGGX,,,// $
LABEL   LBL14 $
PURGE   K4NN,K4GG,K4FF,K4AA/NOK4GG $
CHKPNT  K4NN,K4GG,K4FF,K4AA $
SMA2    CSTM,MPT,ECPT,GPCT,DIT/MGG,BGG/V,Y,WTMASS=1.0/V,N,NOMGG/ V,N,
        NOBGG/V,Y,COUPMASS=-1 $
SAVE    NOMGG,NOBGG $
CHKPNT  MGG,BGG $
PARAM   //C,N,NOP/V,Y,PRTMG=-1 $
COND    LBL11,PRTMG $
MATPRN  MGG,,,// $
LABEL   LBL11 $
PARAM   //C,N,NOP/V,Y,PRTMGD=1 $
COND    LBL15,PRTMGD $
DIAGONAL MGG/MMG/C,Y,OPT=COLUMN/V,Y,POWER=1. $
MATPRN  MMG,,,// $
LABEL   LBL15 $
PURGE   BNN,BFF,BAA,BGG/NOBGG $
CHKPNT  BNN,BFF,BAA,BGG $
COND    LBL1,GRDPNT $
GPWG    BGPDT,CSTM,EQEXIN,MGG/OGPWG/V,Y,GRDPNT=-1/V,Y,WTMASS $
CHKPNT  OGPWG $
OPF     OGPWG,,,,,//V,N,CARDNO $
SAVE    CARDNO $
LABEL   LBL1 $
PARAM   //C,N,MPY/V,N,NSKIP/C,N,O/C,N,O $

```

GP4 CASECC,GEOM4,EQEXIN,SIL,GPDT,BGPD,CSTM/RG,,USET,/V,N,LUSET/  
 V,N,MPCF1/V,N,MPCF2/V,N,SINGLE/V,N,OMIT/V,N,REACT/V,N,NSKIP/  
 V,N,REPEAT/V,N,NOSET/V,N,NOL/V,N,NOA/C,N,O \$  
 SAVE MPCF1,SINGLE,OMIT,NOSET,REACT,MPCF2,NSKIP,REPEAT,NOL,NOA \$  
 CHPNT RG,USET \$  
 PURGE GM,GMD/MPCF1/GO,KOQB,LOO,UOO,MOQB,MOAB,GOD/OMIT/KFS,PST,QP/  
 SINGLE \$  
 CHPNT GM,GMD,GO,KOQB,LOO,UOO,MOQB,MOAB,GOD,KFS,PST,QP \$  
 EQUIV KGGX,KNN/MPCF1/MGG,MNN/MPCF1/ BGG,BNN/MPCF1/K4GG,K4NN/MPCF1 \$  
 CHPNT KNN,MNN,BNN,K4NN,OGPST \$  
 GPSP GPL,GPST,USET,SIL/OGPST \$  
 OFP OGPST,,,,//V,N,CARDNO \$  
 SAVE CARDNO \$  
 COND LBL2,MPCF1 \$  
 MCE1 USET,RG/GM \$  
 MCE2 USET,GM,KGGX,MGG,BGG,K4GG/KNN,MNN,BNN,K4NN \$  
 CHPNT GM,KNN,MNN,BNN,K4NN \$  
 LABEL LBL2 \$  
 EQUIV KNN,KFF/SINGLE/MNN,MFF/SINGLE/BNN,BFF/SINGLE/K4NN,K4FF/SINGLE \$  
 CHPNT KFF,MFF,BFF,K4FF \$  
 COND LBL3,SINGLE \$  
 SCE1 USET,KNN,MNN,BNN,K4NN/KFF,KFS, ,MFF,BFF,K4FF \$  
 LABEL LBL3 \$  
 EQUIV KFF,KAA/OMIT/ MFF,MAA/OMIT/BFF,BAA/OMIT/K4FF,K4AA/OMIT \$  
 CHPNT KAA,MAA,BAA,K4AA \$  
 COND LBL20,OMIT \$  
 SMP1 USET,KFF,,BFF,K4FF/GO,KAA,KOQB,LOO,UOO,,,,BAA,K4AA \$  
 CHPNT GO,KAA,KOQB,LOO,UOO,BAA,K4AA \$  
 SMP2 USET,GO,MFF/MAA \$  
 CHPNT MAA \$  
 LABEL LBL20 \$  
 PURGE KRR,KLR,DM,MLR,MR/REACT/CM/MPCF1/GO/OMIT/KFS/SINGLE/OG/NOSET \$  
 COND LBL21,REACT \$  
 RBMG1 USET,KAA,MAA/KLL,KLR,KRR,MLL,MLR,MRR \$  
 CHPNT KLL,KLR,KRR,MLL,MLR,MRR \$  
 RBMG2 KLL/LLL,ULL \$  
 CHPNT ULL,LLL \$  
 RBMG3 LLL,ULL,KLR,KRR/DM \$  
 CHPNT DM \$  
 RBMG4 DM,MLL,MLR,MRR/MR \$  
 CHPNT MR \$  
 LABEL LBL21 \$  
 DPD DYNAMICS,GPL,SIL,USET/GPLD,SILD,USSETD,,,,,EED,EQDYN/V,N,  
 LUSET/V,N,LUSETD/V,N,NOTFL/V,N,NOFLT/V,N,NOPSDL/V,N,NOFRL/V,N,  
 NOMLFT/V,N,NOTRL/V,N,NOEED/C,N,123/V,N,NOUE \$  
 SAVE NOEED,NOUE \$  
 CHPNT GPLD,SILD,USSETD,EED,EQDYN \$  
 COND ERRO2,NOEED  
 READ KAA,MAA,MR,DM,EED,USET,CASECC/LAMA,PHIA,MI,OEIGS/C,N,MODES/V,  
 N,NEIGV \$  
 SAVE NEIGV\$  
 CHPNT LAMA,PHIA,MI,OEIGS \$  
 ADD MI,/OI/C,N,(1.-10,0.0) \$  
 OFP LAMA,OEIGS,,,,//V,N,CARDNO \$  
 COND FINIS,NEIGV \$  
 SDR1 USET,,PHIA,,,GO,GM,,KFS,,/PHIG,,QG/C,N,1/C,N,REIG \$  
 CHPNT PHIG,QG \$  
 PARAM //C,N,NOP/V,Y,PRTM=1 \$  
 COND LBL12,PRTM \$  
 MATPRN PHIG,,,,// \$

```

LABEL LBL12 $
SDR2 CASECC,CSTM,MPT,DIT,EQEXIN,SIL,,,BGPDT,LAMA,OG,PHIG,EST,/,OQG1,
OPHIG,OES1,DEF1,PPHIG/C,N,REIG $
CHKPNT OOG1,OPHIG,OES1,DEF1,PPHIG $
OFF OPHIG,OQG1,DEF1,OES1,>//V,N,CARDNO $
SAVE CARDNO $
EQUIV GO,GOD/NOUE/GM,GMD/NOUE $
CHKPNT GOD,GMD $
MTRXIN CASECC,MATPOOL,EQDYN,,/K2PP,M2PP,B2PP/V,N,LUSETD/V,N,NOK2PP/V,
N,NOM2PP/V,N,NOB2PP $
SAVE NOK2PP,NOM2PP,NOB2PP $
CHKPNT K2PP,M2PP,B2PP $
PARAM //C,N,AND/V,N,KDEKA/V,N,NOUE/V,N,NOK2PP $
PARAM //C,N,AND/V,N,MDEMA/V,N,NOUE/V,N,NOM2PP $
PARAM //C,N,AND/V,N,KDEK2/V,N,NOGENL/V,N,NOSIMP $
PURGE K2DD/NOK2PP/M2DD/NOM2PP/B2DD/NOB2PP $
EQUIV M2PP,M2DD/NOA/B2PP,B2DD/NOA/K2PP,K2DD/NOA/MAA,MDD/MDEMA/ KAA,
KDD/KDEKA
GKAD USETD,GM,GO,KAA,BAA,MAA,K4AA,K2PP,M2PP,B2PP/KDD,BDD,MDD,GMD,
GOD,K2DD,M2DD,B2DD/C,N,TRANRESP/C,N,DISP/C,N,DIRECT/C,Y,G=0.0/
C,Y,W3=0.0/C,Y,W4=0.0/V,N,NOK2PP/V,N,NOM2PP/V,N,NOB2PP/ V,N,
MPCF1/V,N,SINGLE/V,N,OMIT/V,N,NOUE/V,N,NOK4GG/V,N,NORGG/ V,N,
KDEK2/C,N,-1 $
CHKPNT M2DD,B2DD,K2DD,MDD,KDD,BDD,GMD,GOD $
GKAM USETD,PHIA,OI,LAMA,DIT,MDD,BDD,KDD,CASECC/MHH,BHH,KHH,
PHIDH/V,N,NOUE/C,Y,LMODES=0/C,Y,LFREQ=0.0/ C,Y,HFREQ=0.0/C,N,
1/C,N,1/C,N,1/V,N,NONCPU/V,N,FMODE $
CHKPNT MHH,BHH,KHH,PHIDH $
PARAM //C,N,NOP/V,Y,PRTMM=1 $
COND LBL13,PRTMM $
MATPRN KHH,BHH,MHH,PHIDH,//$ $
LABEL LBL13 $*
PARAM //C,N,NOP/V,Y,PCHGD=-1 $
COND LBL6,PCHGD $
TABPCH GPL,BGPDT,,,//C,N,GP/C,N,BG $
LABEL LBL6 $
PARAM //C,N,NOP/V,Y,PCHMG=-1 $
COND LBL7,PCHMG $
OUTPUT3 MGG,,,,//C,N,0/C,Y,N1=MMG $
LABEL LBL7 $
PARAM //C,N,NOP/V,Y,PCHMD=-1 $
COND LBL8,PCHMD $
OUTPUT3 PHIG,,,,//C,N,0/C,Y,N1=PHG $
LABEL LBL8 $
PARAM //C,N,NOP/V,Y,PCHMM=-1 $
COND LBL9,PCHMM $
OUTPUT3 KHH,BHH,MHH,,,//C,N,0/C,Y,N1=KKH/C,Y,N2=BBH/C,Y,N3=MMH $
LABEL LBL9 $
JUMP FINIS$
LABEL ERRO2 $
PRTPARM //C,N,-2/C,N,MODES $
LABEL FINIS $
END $
CEND

```



NASTRAN is extremely inflexible in allowing the user control over output data formatting. Consequently, a preprocessor program must be available to read the NASTRAN generated data. The next program is designed to not only read the data but also to process the data and compute the input data required by both N-BOD2 and DISCOS in respectively acceptable format.

For N-BOD2 users, the resultant mode dependent parameters are outputted on the line printer and annotated with the same acronyms used in the coding of the read data statements in subroutine INOPT.

```
C
C
C   DISCOS, N-BOD2 PREPROCESSOR OF NASTRAN GENERATED DATA
C
C   PURPOSE:
C       1) READ ONE OR MORE NASTRAN GENERATED TAPES
C       2) WRITE ONE INPUT TAPE FOR DISCOS PROGRAM
C           CONTAINING FLEXIBLE BODY DATA FOR ALL
C           FLEXIBLE BODIES IN SIMULATION
C       3) COMPUTE RESULTANT MODE DEPENDENT PARAMETERS
C           FOR GSFC MULTI-FLEXIBLE BODY PROGRAM.
C           N-BOD2, THIS SECTION OF NO INTEREST TO
C           DISCOS USERS
C
C
C   DEFINITIONS:
C
C   DATA SET 1 = SCRATCH FILE USED TO PROCESS NASTRAN DATA
C
C   DATA SET 2 = DISCOS INPUT TAPE (THE OUTPUT TAPE OF THIS PROGRAM)
C   THE CARD OUTPUT OF THIS PROGRAM ASSUMES THAT DATA SET 14 IN
C   DISCOS WILL BE RESERVED FOR THE FLEXIBLE BODY DATA
C
C   DATA SET 3 = SCRATCH FILE USED TO REDUCE CORE REQUIREMENTS
C
C   DATA SET 11 = LOCATION IN SYSTEM OF NASTRAN TAPE ASSOCIATED WITH
C   THE FLEXIBLE BODY IN DISCOS HAVING LOWEST MAGNITUDE
C   INTEGER LABEL
C
C   DATA SET 12 = LOCATION IN SYSTEM OF NASTRAN TAPE ASSOCIATED WITH
C   THE NEXT FLEXIBLE BODY IN THE DISCOS MODEL
C
C   .
C   .
C   ETC.   PROGRAM DIMENSIONED FOR A MAXIMUM OF 6 NASTRAN TAPES
C
C   IMPLICIT REAL*8 (A-H,O-Z)
C
C   DATA IDL/1H$/
C   DATA ISTR,IBLK/1H*,1H /
C   DATA IGPL,IBGP,IMGG /4HGPL ,4HRGPD,4HMGG /
```

```

DATA ID,IT,II/1HD,1HT,1HI/
DATA IR,IE,IS/1HR,1HE,1HS/
C
INTEGER INPT(80), IGRID(500), IA(20), IB(20),
* ICOT(5), IY(4)
INTEGER MSAVE(100),MSTOTL,HINGE
C
REAL*8 X(500), Y(500), Z(500),
* INERTA(6,6,500), PHI(6,500,12),
* KHH(12,12), BHH(12,12),
* MHH(12,12), XX(4), XY(4)
C
REAL*8 AMD(500,72)
C
COMMON /WORK/ AMD
C
EQUIVALENCE (AMD(1,1),PHI(1,1,1))
EQUIVALENCE (AMD(1,1),X(1))
EQUIVALENCE (AMD(1,2),Y(1))
EQUIVALENCE (AMD(1,3),Z(1))
EQUIVALENCE (AMD(1,4),INERTA(1,1,1))
C
LOGICAL LECHO,LPUNCH,LCHECK,LNAST
LOGICAL LRSTRT
C
C
100 FORMAT (I5,3L5,2I5,L5)
101 FORMAT (4A4,7I8,2A4)
102 FORMAT (2A4,8I8)
103 FORMAT (4A4,5I8,E16.9)
104 FORMAT (2A4,2E16.9,1I6,E16.9)
105 FORMAT (2A4,2E16.9)
106 FORMAT (I10)
108 FORMAT (4A4,18,2I16,E16.8)
109 FORMAT (10E13.5)
110 FORMAT (6E15.5)
111 FORMAT (12I5)
112 FORMAT ('1',I5,3L5,2I5,L5)
113 FORMAT (1X,20I5)
114 FORMAT ('1',5X,'*****. MODAL DATA SELECTED FROM NASTRAN INPUT TAPE',
*E',I4,' FOR PROCESSING *****',//)
115 FORMAT (12X,'MODE',I4,' LISTED BELOW IS MODE',I4,' ON THE NASTRAN
*DATA TAPE')
200 FORMAT (' NJOINT =',I5,' NUMBER OF GRID POINT LOCATIONS TO BE READ
* =',I5,' THESE SHOULD BE EQUAL')
201 FORMAT (' NUMBER GRID POINT X-LOCATION Y-LOCATION
* Z-LOCATION ')
202 FORMAT (6X,I5,7X,I5,2X,3E15.5)
300 FORMAT (80A1)
302 FORMAT (4A4,18,2I16,E16.8,2A4)
303 FORMAT (2A4,4E16.8,2A4)
400 FORMAT (2X,80A1)
500 FORMAT (' ANOMOLOUS EOF MARK ON TAPE, LISTING FOLLOWS ')
501 FORMAT (/, ' INTERNAL GRID POINT ',I5,' EXTERNAL GRID POINT',I5,'
* N THE NASTRAN BULK DATA DECK')
502 FORMAT (' X,Y,Z COORDINATES')
503 FORMAT (6E20.10)
504 FORMAT (' LUMPED INERTIA MATRIX')
505 FORMAT (' PHI (6 MODAL COMPONENTS FOR EACH SELECTED MODE)')

```

```

506 FORMAT (//, ' KHH ')
507 FORMAT ('1',5X, '***** LUMPED PARAMETER AND MODAL DATA TO BE PROCESSED FOR DISCOS AND N-ROD2 INPUT DATA *****',///)
508 FORMAT (//, ' BHH ')
509 FORMAT (//, ' MHH ')
510 FORMAT (//////)
511 FORMAT (2X, 'EXIT DECODE ',4(I8,E16.8))
512 FORMAT (2X, 'ENTER DECODE ',2I8,4E16.8)
600 FORMAT (I5,3E20.8)

```

C  
C  
C  
C  
C  
C  
C  
C  
C  
C  
C  
C  
C  
C  
C  
C  
C  
C  
C

```

READ TWO INPUT CARDS PER NASTRAN TAPE
NTAPE = DATA SET WHERE TAPE IS TO
        BE FOUND NTAPE=11,12,...
LECHO = .TRUE. PRINT ECHO OF PROCESSED DATA
        = .FALSE. PRINT ONLY IF ERROR FOUND ON TAPE
LPUNCH = .TRUE. PUNCH PLOT DATA
        = .FALSE. DON'T
LCHECK = .TRUE. PRINT DATA IN/OUT OF DECODE
        = .FALSE. DON'T
HINGE = GRID POINT NUMBER OF HINGE POINT AS
        DEFINED BY USER IN NASTRAN BULK DATA
MSTOTL = TOTAL NUMBER OF FLEXIBLE BODY MODES TO
        PROCESS FOR DISCOS AND N-ROD2 DATA
LNAST = .TRUE. PRINT NASTRAN DATA INPUT
        = .FALSE. DON'T
MSAVE = MODE SAVE ARRAY
        MSAVE(K) = 0 SKIP MODE K DATA
        K PROCESS MODE K DATA

```

```

REWIND 2
1 CONTINUE
READ(5,100,END=4) NTAPE,LECHO,LPUNCH,LCHECK,HINGE,MSTOTL,LNAST
WRITE(6,112) NTAPE,LECHO,LPUNCH,LCHECK,HINGE,MSTOTL,LNAST
READ(5,111) (IA(I),I=1,MSTOTL)
DO 11 I=1,100
11 MSAVE(I) = 0
DO 12 I=1,MSTOTL
12 MSAVE(IA(I)) = I
WRITE(6,113) MSAVE
REWIND NTAPE

```

C  
C  
C  
C  
C  
C  
C  
C  
C  
C  
C

```

LRSTRT = .FALSE. NO RESTART CARDS ON NTAPE
        = .TRUE. RSTART CARDS ON NTAPE
ICNTRS = 0
LRSTRT = .FALSE.
9 READ (NTAPE,300,END=8) INPT
IF(LNAST) WRITE(6,400) INPT
CHECK FOR RESTART CARDS
IF(IR.NE.INPT(1)) GO TO 9
IF(IE.NE.INPT(2)) GO TO 9
IF(IS.NE.INPT(3)) GO TO 9
IF(IT.NE.INPT(4)) GO TO 9
COME HERE FOR RESTART CARDS, COUNT THEM AND SET FLAG
LRSTRT = .TRUE.
ICNTRS = ICNTRS+1
13 READ (NTAPE,300,END=8) INPT
IF(LNAST) WRITE(6,400) INPT
IF(ID.EQ.INPT(1).AND.IT.EQ.INPT(2).AND.II.EQ.INPT(3)) GO TO 9
ICNTRS = ICNTRS+1
GO TO 13

```

```

8 CONTINUE
REWIND NTAPE
REWIND 1
REWIND 3
ICNT = 0
IBLK = 0
DO 7 I=1,5
7 ICOT(I) = 0

```

```

C
C
C      READ ALL DTI CARDS FROM NASTRAN TAPE NTAPE TO OBTAIN
C      1) GRID POINT NUMBERING SEQUENCE SET UP BY NASTRAN
C      2) GRID POINT LOCATIONS

```

```

C      **** NOTE ****

```

```

C      NASTRAN WILL RENUMBER GRID POINTS DEFINED BY USER
C      INTO A SEQUENTIAL SET, THIS IS USED FOR INTERNAL NASTRAN
C      COMPUTATION AND WILL BE USED BY DISCOS

```

```

C      SKIP RESTART CARDS IN FRONT OF GOOD DATA
C      IF(.NOT.LRSTRT) GO TO 2
C      DO 14 I=1,ICNTRS
14 READ (NTAPE,300,END=3) INPT
2 READ (NTAPE,101,END=3) (IA(I),I=1,4),(IB(I),I=1,7),(IA(I),I=5,6)
C      CHECK FOR TABLE HEADING
C      IF(IA(3).EQ.IGPL) GO TO 50
C      IF(IA(3).EQ.IBGP) GO TO 60
C      IF(IA(3).EQ.IMGG) GO TO 70
C      GO TO 2

```

```

C      ERROR ON TAPE
C      3 CONTINUE
C      WRITE(6,500)
6 REWIND NTAPE
5 READ (NTAPE,300,END=4) INPT
WRITE(6,400) INPT
GO TO 5
4 STOP

```

```

C      COME HERE TO PROCESS GPL TABLE DATA
50 CONTINUE
ICNT = ICNT + 1
IF(ICNT.NE.1) GO TO 51
NJOINT = IB(2)
C      NJOINT = NUMBER OF GRID POINTS IN MODEL
GO TO 2
51 IF(ICNT.NE.2)GO TO 52
DO 53 J=1,6
53 IGRID(J) = IB(J+1)
NC = (NJOINT-6)/8
DO 54 J=1,NC
JJ = 7 + 8*(J-1)
JJ7 = JJ+7
54 READ(NTAPE,102) (IA(I),I=1,2),(IGRID(I),I=JJ,JJ7)
JJ = 6 + 8*NC
JJ1 = JJ+1
IF(JJ.EQ.NJOINT) GO TO 2
READ(NTAPE,102) (IA(I),I=1,2),(IGRID(I),I=JJ1,NJOINT)
GO TO 2

```

```

C      IGRID(I) =USER DEFINED GRID POINT NUMBER

```

```

C                               WHICH NASTRAN HAS RELABLED TO
C                               BE INTERNAL NUMBER 1
52 ICNT = 0
   NC = (NJOINT-3)/4
   IF(NJOINT.EQ.4*NC+3) GO TO 16
   NC = NC + 1
16 DO 15 I=1,NC
15 READ (NTAPE,300) INPT
   GO TO 2

C
C
C                               COME HERE TO PROCESS BGPDT CARDS
60 CONTINUE
   IF(IB(2).EQ.NJOINT) GO TO 61
   ERROR
   WRITE(6,200) NJOINT,IB(2)
   GO TO 6
61 READ (NTAPE,300) INPT
   READ (NTAPE,103) (IA(I),I=1,4),(IB(I),I=1,5),X(1)
   DO 62 J=2,NJOINT
62 READ (NTAPE,104) (IA(I),I=1,2),Y(J-1),Z(J-1),IB(1),X(J)
   J = NJOINT
   READ (NTAPE,105) (IA(I),I=1,2),Y(J),Z(J)
   GO TO 2
   SAVE LOCATION DATA, WRITE IT ON TAPE 3 AFTER INERTIA DATA

C
C                               X(J) = X-COORDINATE OF GRID POINT IGRID(J)
C                               Y(J) = Y-COORDINATE OF GRID POINT IGRID(J)
C                               Z(J) = Z-COORDINATE OF GRID POINT IGRID(J)

C
C                               COME HERE TO PROCESS ALL DMI CARDS
C                               STEP 1:
C                               COUNT CARDS IN EACH GROUP
C                               ASCERTAIN FORMAT OF EACH CARD
C                               REWRITE ON SCRATCH FILE ALL CARDS
C                               PRECEDED BY THEIR FORMAT CODE
C                               STEP 2:
C                               REWIND SCRATCH FILE
C                               READ CARDS OFF SCRATCH FILE WITH KNOWN FORMAT
C                               LOAD ALL ARRAYS, SKIPPING ALL UNWANTED MODES
70 CONTINUE
   REWIND 1
   IFORM = 301
   WRITE(1,106) IFORM
   WRITE(1,101) (IA(I),I=1,4),(IB(I),I=1,7),(IA(I),I=5,6)
   IBLK = 1
   ICOT(IBLK) = 1
   IF(IB(2).NE.6.OR.IB(6).NE.IB(7).OR.IB(6).NE.NJOINT*6) GO TO 71
   DO 76 I=1,NJOINT
   DO 76 J1=1,6
   DO 76 J2=1,6
76 INERTA(J1,J2,I) = 0.0
   GO TO 72
71 CONTINUE
   ERROR
   WRITE(6,101) (IA(I),I=1,4),(IB(I),I=1,7),(IA(I),I=5,6)
   GO TO 6
72 READ(NTAPE,300,END=80) INPT
   IF(INPT(4).NE.ISTR) GO TO 73

```

```

      ICOT(IBLK) = ICOT(IBLK) + 1
      IFORM = 302
      GO TO 74
73  IF(INPT(1).EQ.ISTR) GO TO 75
C      END OF DATA BLOCK IBLK
C      CHECK FOR 'END OF CHECKPOINT DICTIONARY'
      IF(INPT(1).EQ.IDL) GO TO 72
      IBLK = IBLK + 1
      ICOT(IBLK) = ICOT(IBLK) + 1
      IFORM = 301
      GO TO 74
75  CALL INTERP(INPT,IFORM)
      ICOT(IBLK) = ICOT(IBLK) + 1
74  WRITE(1,106) IFORM
      WRITE(1,300) INPT
      GO TO 72
80  END FILE 1

C
C      ALL DMI CARDS READ, FORMAT DEDUCED AND
C      REWRITTEN ON SCRATCH FILE 1
C
C      FORMAT CODES:
C      301 = FORMAT (4A4,7I8,2A4)
C      302 = FORMAT (4A4,I8,2I16,E16.8,2A4)
C      303 = FORMAT (2A4,4E16.8,2A4)
C      304 = FORMAT (2A4,3E16.8,I16,2A4)
C      305 = FORMAT (2A4,2E16.8,I16,E16.8,2A4)
C      306 = FORMAT (2A4,E16.8,I16,2E16.8,2A4)
C      307 = FORMAT (2A4,E16.8,I16,E16.8,I16,2A4)
C      308 = FORMAT (2A4,I16.8,3E16.8,2A4)
C      309 = FORMAT (2A4,I16.8,2E16.8,I16.8,2A4)
C      310 = FORMAT (2A4,I16,E16.8,I16,E16.8,2A4)
C
C      START TO PROCESS DMI CARDS
C
      REWIND 1
      IF(IBLK.LE.5) GO TO 20
C      ERROR IBLK MUST BE LESS THAN OR EQUAL TO 5
81  REWIND 1
83  READ (1,300,END=90) INPT
      WRITE(6,400) INPT
      GO TO 83
90  STOP
20  CONTINUE
      DO 22 IBK=1,IBLK
      ICBLK = ICOT(IBK)
      DO 21 JJ =1,ICBLK
      IF(JJ.NE.1) GO TO 23
C      READ FIRST CARD OF DATA BLOCK
      READ (1,106) IFORM
      READ (1,101) (IA(I),I=1,4),(IB(I),I=1,7),(IA(I),I=5,6)
      IF(IBK.NE.1) GO TO 40
C
C      INERTIA MATRIX
C      IF(IB(2).NE.6.OR.IB(6).NE.IR(7).OR.IB(6).NE.NJOINT*6) GO TO 81
C      SO FAR SO GOOD
      GO TO 21
40  IF(IBK.NE.2) GO TO 41
C
C      MODE SHAPES
C      IF(IB(6).NE.6*NJOINT) GO TO 81

```

```

      NMODES = IB(7)
C
C
      NMODES = NUMBER OF MODES
      MSTOTL = NUMBER OF MODES TO PROCESS
      DO 42 I1=1,6
      DO 42 I2=1,NJOINT
      DO 42 I3=1,MSTOTL
42  PHI(I1,I2,I3) = 0.0
      GO TO 21
C
C
      MODAL MASS, STIFFNESS AND DAMPING MATRICES
41  IF(IB(6).NE.IB(7).OR.IB(7).NE.NMODES) GO TO 71
      DO 43 I1=1,MSTOTL
      DO 43 I2=1,MSTOTL
      IF(IBK.NE.3) GO TO 44
      KHH(I1,I2) = 0.0
      GO TO 43
44  IF(IBK.NE.4) GO TO 45
      BHH(I1,I2) = 0.0
      GO TO 43
45  MHH(I1,I2) = 0.0
43  CONTINUE
      GO TO 21
C
C
      READ IN DATA CARDS
23  CONTINUE
      READ (1,106) IFORM
      IF(IFORM.NE.302) GO TO 24
      READ (1,302) (IA(I),I=1,4),IB(1),NNR,NNC,XX(1),(IA(I),I=5,6)
      NSR = MSAVE(NNR)
      NSC = MSAVE(NNC)
      IF(IBK.NE.1) GO TO 25
      N1T = (NNC-1)/6 + 1
      N2T = (NNR-1)/6 + 1
      IF(N1T.EQ.N2T) GO TO 26
C
      ERROR
      WRITE(6,302) (IA(I),I=1,4),IB(1),NNR,NNC,XX(1),(IA(I),I=5,6)
      GO TO 81
26  NT = N1T
      NC = MOD(NNC-1,6) + 1
      NR = MOD(NNR-1,6) + 1
      INERTA(NR,NC,NT) = XX(1)
      GO TO 21
25  IF(IBK.NE.2) GO TO 27
      NT = (NNC-1)/6 + 1
      NE = MOD(NNC-1,6) + 1
      IF(NSR.EQ.0) GO TO 21
      PHI(NE,NT,NSR) = XX(1)
      GO TO 21
27  IF(IBK.NE.3) GO TO 28
      IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 21
      KHH(NSR,NSC) = XX(1)
      GO TO 21
28  IF(IBK.NE.4) GO TO 29
      IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 21
      BHH(NSR,NSC) = XX(1)
      GO TO 21
29  CONTINUE
      IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 21
      MHH(NSR,NSC) = XX(1)
      GO TO 21

```

```

C
24 CONTINUE
C      FORMAT 4E16.8 FOR XX WILL READ INTEGER I WHICH IS RIGHT
C      JUSTIFIED AS I * 1.0E-08
C      THIS IS EASILY UNDONE IN DECODE
READ (1,303) (IA(I),I=1,2),(XX(I),I=1,4),(IA(I),I=3,4)
IF(LCHECK) WRITE (6,512) IFORM,NNC,(XX(I),I=1,4)
CALL DECODE(IFORM,XX,NNC,XY,IY,NY)
IF(LCHECK) WRITE (6,511) (IY(I),XY(I),I=1,NY)
DO 30 I=1,NY
IF(IBK.NE.1) GO TO 31
NC = MOD(IY(I)-1,6) + 1
INERTA(NR,NC,NT) = XY(I)
GO TO 30
31 IF(IBK.NE.2) GO TO 32
NT = (IY(I)-1)/6 + 1
NE = MOD(IY(I)-1,6) + 1
IF(NSR.EQ.0) GO TO 30
PHI(NE,NT,NSR) = XY(I)
GO TO 30
32 IF(IBK.NE.3) GO TO 33
NSC = MSAVE(IY(I))
IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 30
KHH(NSR,NSC) = XY(I)
GO TO 30
33 IF(IBK.NE.4) GO TO 34
NSC = MSAVE(IY(I))
IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 30
BHH(NSR,NSC) = XY(I)
GO TO 30
34 CONTINUE
NSC = MSAVE(IY(I))
IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 30
MHH(NSR,NSC) = XY(I)
30 CONTINUE
21 CONTINUE
C
IF(IBK.NE.1) GO TO 63
REWIND 3
WRITE (3) (((INERTA(I,J,N),I=1,6),J=1,6),N=1,NJOINT)
WRITE (3) (X(N),Y(N),Z(N),N=1,NJOINT)
IF(.NOT.LPUNCH) GO TO 10
WRITE (7,600) (IGRID(N),X(N),Y(N),Z(N),N=1,NJOINT)
10 CONTINUE
WRITE(6,507)
DO 64 N=1,NJOINT
WRITE (6,501) N,IGRID(N)
WRITE (6,502)
WRITE (6,503) X(N),Y(N),Z(N)
WRITE (6,504)
WRITE (6,503) ((INERTA(I,J,N),J=1,6),I=1,6)
64 CONTINUE
WRITE (6,510)
GO TO 22
63 IF(IBK.NE.2) GO TO 65
WRITE (3) (((PHI(I,N,J),I=1,6),J=1,MSTOTL),N=1,NJOINT)
IF(.NOT.LPUNCH) GO TO 67
DO 68 J=1,MSTOTL
WRITE (7,600) (IGRID(N),(PHI(I,N,J),I=1,3),N=1,NJOINT)

```



```

68 CONTINUE
67 CONTINUE
  WRITE (6,114) NTAPE
  DO 17 I=1,NMODES
    IF(MSAVE(I).EQ.0) GO TO 17
    WRITE (6,115) MSAVE(I),I
17 CONTINUE
  WRITE (6,510)
  DO 66 N=1,NJOINT
    WRITE (6,501) N,IGRID(N)
    WRITE (6,505)
    WRITE (6,503) ((PHI(I,N,J),I=1,6),J=1,MSTOTL)
66 CONTINUE
  WRITE (6,510)
  GO TO 22
65 IF(IBK.NE.3) GO TO 69
  WRITE (3) ((KHH(I,J),I=1,MSTOTL),J=1,MSTOTL)
  WRITE (6,506)
  WRITE (6,503) ((KHH(I,J),J=1,MSTOTL),I=1,MSTOTL)
  WRITE (6,510)
  GO TO 22
69 IF(IBK.NE.4) GO TO 77
  WRITE (3) ((BHH(I,J),I=1,MSTOTL),J=1,MSTOTL)
  WRITE (6,508)
  WRITE (6,503) ((BHH(I,J),J=1,MSTOTL),I=1,MSTOTL)
  WRITE (6,510)
  GO TO 22
77 IF(IBK.NE.5) GO TO 22
  WRITE (3) ((MHH(I,J),I=1,MSTOTL),J=1,MSTOTL)
  WRITE (6,509)
  WRITE (6,503) ((MHH(I,J),J=1,MSTOTL),I=1,MSTOTL)
  WRITE (6,510)
22 CONTINUE

```

C  
C

```

          DATA READY FOR DISCOS PROCESSING
CALL DISCOS(IGRID,NJOINT,MSTOTL,ITAPE,LECHO)
CALL NBOD2 (IGRID,NJOINT,MSTOTL,ITAPE,LECHO,HINGE)
IF(ITAPE.LE.6) GO TO 1
STOP
END

```

C  
C

```

SUBROUTINE DISCOS(IGRID,NJ,NM,ICNT,LECHO)
IMPLICIT REAL*8 (A-H,O-Z)

LOGICAL LECHO

REAL*8 X(500),Y(500),Z(500),INERTA(6,6,500),IRUNNO
REAL*8 PHI(6,500,12),KHH(12,12),RHH(12,12),LOC(6)
REAL*8 AMS(6),JMASS(500,1),JINER(500,6),AIN(6),SMM(6)
REAL*8 SMSM(500,3),LOCA(500,3),MOD(6,6),AMD(500,13),KH(6),BH(6)
REAL*8 AMP(500,72)
COMMON /WORK/ AMP
EQUIVALENCE (AMP(1,1),X(1))
EQUIVALENCE (AMP(1,2),Y(1))
EQUIVALENCE (AMP(1,3),Z(1))

```

EQUIVALENCE (AMP(1,4),INERTA(1,1,1))  
EQUIVALENCE (AMP(1,1),PHI(1,1,1))

C  
C  
INTEGER IGRID(500)

COMMON /WORK2/ AMD

EQUIVALENCE (AMD(1,1),JMASS(1,1)), (AMD(1,1),JINER(1,1))  
EQUIVALENCE (AMD(1,7),SMSM(1,1)), (AMD(1,10),LOCA(1,1))  
EQUIVALENCE (AMD(1,13),KHH(1,1)), (AMD(145,13),BHH(1,1))

C  
DATA IFST/0/

DATA AMS/6HJMASS1,6HJMASS2,6HJMASS3,6HJMASS4,6HJMASS5,  
\* 6HJMASS6/

DATA FLX/6HFLXDAT/

DATA AIN/6HINERT1,6HINERT2,6HINERT3,6HINERT4,6HINERT5,  
\* 6HINERT6/

DATA SMM/6HSTMSM1,6HSTMSM2,6HSTMSM3,6HSTMSM4,6HSTMSM5,  
\* 6HSTMSM6/

DATA LOC/6HLOCAT1,6HLOCAT2,6HLOCAT3,6HLOCAT4,6HLOCAT5,  
\* 6HLOCAT6/

DATA MOD/6HXDIS1,6HYDIS1,6HZDIS1,6HTHA11,6HTHA21,6HTHA31,

\* 6HXDIS2,6HYDIS2,6HZDIS2,6HTHA12,6HTHA22,6HTHA32,

\* 6HXDIS3,6HYDIS3,6HZDIS3,6HTHA13,6HTHA23,6HTHA33,

\* 6HXDIS4,6HYDIS4,6HZDIS4,6HTHA14,6HTHA24,6HTHA34,

\* 6HXDIS5,6HYDIS5,6HZDIS5,6HTHA15,6HTHA25,6HTHA35,

\* 6HXDIS6,6HYDIS6,6HZDIS6,6HTHA16,6HTHA26,6HTHA36 /

DATA KH/6HKHH1,6HKHH2,6HKHH3,6HKHH4,6HKHH5,6HKHH6 /

DATA BH/6HBHH1,6HBHH2,6HBHH3,6HBHH4,6HBHH5,6HBHH6 /

DATA IRUNNO/6HPREPRS/

C  
700 FORMAT (A6,' 0 14PREPRS ')

C  
C INITIALIZE TAPE, START COUNTER

REWIND 3

IF(IFST.NE.0) GO TO 1

CALL INTAPE(2,FLX)

ICNT = 0

IFST = 1

1 CONTINUE

ICNT = ICNT+1

C  
C PROCESS MASS MATRIX DATA

READ (3) (((INERTA(I,J,N),I=1,6),J=1,6),N=1,NJ)

READ (3) (X(N),Y(N),Z(N),N=1,NJ)

DO 2 I=1,NJ

IF(INERTA(1,1,I).NE.INERTA(2,2,I)) GO TO 3

IF(INERTA(2,2,I).NE.INERTA(3,3,I)) GO TO 3

IF(INERTA(3,3,I).LT.0) GO TO 3

JMASS(I,1) = INERTA(3,3,I)

GO TO 2

3 WRITE(6,500) I,(INERTA(II,II,I),II=1,3)

500 FORMAT (' ERROR MASS MATRIX POINT ',I5,' JMASS =',3E20.10)

JMASS(I,1) = (INERTA(1,1,I)+INERTA(2,2,I)+INERTA(3,3,I))/3.000

2 CONTINUE

C  
C PUNCH READ MATRIX INPUT CARD

PUT MASS MATRIX ON TAPE

WRITE(7,700) AMS(ICNT)

IF(LECHO) CALL WRITE(JMASS,NJ,1,AMS(ICNT),500)

CALL WTAPE(JMASS,NJ,1,AMS(ICNT),500,2)

```

C          PROCESS INERTIA DYAD DATA
DO 4 I=1,NJ
  JINER(I,1) = INERTA(4,4,I)
  JINER(I,2) = INERTA(5,5,I)
  JINER(I,3) = INERTA(6,6,I)
  JINER(I,4) = -INERTA(4,5,I)
  JINER(I,5) = -INERTA(4,6,I)
  4 JINER(I,6) = -INERTA(5,6,I)
  WRITE(7,700) AIN(ICNT)
  IF(LECHO) CALL WRITE(JINER,NJ,6,AIN(ICNT),500)
  CALL WTAPE(JINER,NJ,6,AIN(ICNT),500,2)

C          PROCESS STATIC MASS MOMENT DATA
C          PROCESS JOINT LOCATION DATA
DO 5 I=1,NJ
  SMSM(I,1) = INERTA(2,6,I)
  SMSM(I,2) = -INERTA(1,6,I)
  SMSM(I,3) = INERTA(1,5,I)
  LOCA(I,1) = X(I)
  LOCA(I,2) = Y(I)
  5 LOCA(I,3) = Z(I)
  WRITE(7,700) SMM(ICNT)
  IF(LECHO) CALL WRITE(SMSM ,NJ,3,SMM(ICNT),500)
  CALL WTAPE(SMSM,NJ,3,SMM(ICNT),500,2)
  WRITE(7,700) LOC(ICNT)
  IF(LECHO) CALL WRITE(LOCA ,NJ,3,LOC(ICNT),500)
  CALL WTAPE(LOCA,NJ,3,LOC(ICNT),500,2)

C          PROCESS MODAL DATA
C          READ (3) ((PHI(I,N,J),I=1,6),J=1,NM),N=1,NJ)
DO 6 K=1,6
DO 7 I=1,NJ
DO 7 M=1,NM
  7 AMD(I,M) = PHI(K,I,M)
  WRITE(7,700) MOD(K,ICNT)
  IF(LECHO) CALL WRITE(AMD,NJ,NM,MOD(K,ICNT),500)
  CALL WTAPE(AMD,NJ,NM,MOD(K,ICNT),500,2)
  6 CONTINUE

C          MODAL MATRICES
C          READ (3) ((KHH(I,J),I=1,NM),J=1,NM)
C          READ (3) ((BHH(I,J),I=1,NM),J=1,NM)
WRITE(7,700) KH(ICNT)
IF(LECHO) CALL WRITE(KHH,NM,NM,KH(ICNT),12)
CALL WTAPE(KHH,NM,NM,KH(ICNT),12,2)
WRITE(7,700) BH(ICNT)
WRITE(6,100)
IF(LECHO) CALL WRITE(BHH,NM,NM,BH(ICNT),12)
CALL WTAPE(BHH,NM,NM,BH(ICNT),12,2)

C          CALL LTAPE(2)
CALL RTAPE(IRUNNO,AMS(ICNT),AMD,NJ,1,500,12,2)
CALL WRITE(AMD,NJ,1,AMS(ICNT),500)
CALL RTAPE(IRUNNO,AIN(ICNT),AMD,NJ,6,500,12,2)
CALL WRITE(AMD,NJ,6,AIN(ICNT),500)
CALL RTAPE(IRUNNO,SMM(ICNT),AMD,NJ,3,500,12,2)
CALL WRITE(AMD,NJ,3,SMM(ICNT),500)
CALL RTAPE(IRUNNO,LOC(ICNT),AMD,NJ,3,500,12,2)
CALL WRITE(AMD,NJ,3,LOC(ICNT),500)
DO 8 K=1,6

```

```

CALL RTAPE(IRUNNO,MOD(K,ICNT),AMD,NJ,NM,500,12,2)
8 CALL WRITE(AMD,NJ,NM,MOD(K,ICNT),500)
CALL RTAPE(IRUNNO,KH(ICNT),KHH,NM,NM,12,12,2)
CALL WRITE(KHH,NM,NM,KH(ICNT),12)
CALL RTAPE(IRUNNO,BH(ICNT),BHH,NM,NM,12,12,2)
CALL WRITE(BHH,NM,NM,BH(ICNT),12)
100 FORMAT ('1',5X,'***** PROCESSED LUMPED PARAMETER AND MODAL DATA N
*OW ON DATA SET 2 (DISCOS INPUT TAPE) *****',///)
RETURN
END

```

```

C SUBROUTINE NBOD2 (IGRID,NJ,NM,ICNT,LECHO,HINGE)
C COMPUTE RESULTANT MODE DEPENDENT PARAMETERS FOR N-BOD2
C IMPLICIT REAL*8 (A-H,O-Z)
C
C LOGICAL LECHO
C
C COMMON /WORK/ AMP
C REAL*8 AMP(500,72)
C COMMON /WORK2/ AMD
C REAL*8 AMD(500,13)
C
C REAL*8 AMS(6),FLX,AIN(6),SMM(6),LOC(6),MOD(6,6),KH(6),BH(6)
C REAL*8 JMASS(500),JLOC(3,500),CMLOC(3),JINERT(3,3,500),
* PHIT(3,500,12),PHIR(3,500,12),SMO(3,500),
* FLOM(12),ZETA(12),BDINER(3,3),
* FLA(3,12),FLB(3,12),FLC(3,12),FLD(3,3,12),
* FLJ(3,3,12),
* FCF(3,3,12,12),FCK(3,12,12),
* TEM(3),TEM1(3,3),ADM(500,12,6),TFM(3),
* KHH(12,12),BHH(12,12),IRUNNO
C INTEGER IGRID(500)
C INTEGER HINGE,INTH
C
C DATA AMS/6HJMASS1,6HJMASS2,6HJMASS3,6HJMASS4,6HJMASS5,
* 6HJMASS6/
C DATA AIN/6HINERT1,6HINERT2,6HINERT3,6HINERT4,6HINERT5,
* 6HINERT6/
C DATA SMM/6HSTMSM1,6HSTMSM2,6HSTMSM3,6HSTMSM4,6HSTMSM5,
* 6HSTMSM6/
C DATA LOC/6HLOCAT1,6HLOCAT2,6HLOCAT3,6HLOCAT4,6HLOCAT5,
* 6HLOCAT6/
C DATA MOD/6HXDIS1,6HYDIS1,6HZDIS1,6HTHA11,6HTHA21,6HTHA31,
* 6HXDIS2,6HYDIS2,6HZDIS2,6HTHA12,6HTHA22,6HTHA32,
* 6HXDIS3,6HYDIS3,6HZDIS3,6HTHA13,6HTHA23,6HTHA33,
* 6HXDIS4,6HYDIS4,6HZDIS4,6HTHA14,6HTHA24,6HTHA34,
* 6HXDIS5,6HYDIS5,6HZDIS5,6HTHA15,6HTHA25,6HTHA35,
* 6HXDIS6,6HYDIS6,6HZDIS6,6HTHA16,6HTHA26,6HTHA36 /
C DATA KH/6HKHH1,6HKHH2,6HKHH3,6HKHH4,6HKHH5,6HKHH6 /
C DATA BH/6HBHH1,6HBHH2,6HBHH3,6HBHH4,6HBHH5,6HBHH6 /
C DATA IRUNNO/6HPREPRS/
C
C EQUIVALENCE (AMP(1,1),PHIT(1,1,1)),(AMP(1,37),PHIR(1,1,1)),
* (AMP(1,1),ADM(1,1,1)),
* (AMP(1,13),ADM(1,1,2)),

```

```

*          (AMP(1,25),ADM(1,1,3)),
*          (AMP(1,37),ADM(1,1,4)),
*          (AMP(1,49),ADM(1,1,5)),
*          (AMP(1,61),ADM(1,1,6))
EQUIVALENCE (AMD(1,1),JMASS(1)), (AMD(1,2),JLOC(1,1)),
*          (AMD(1,5),SMO(1,1)), (AMD(1,5),JINERT(1,1,1))
C
C          FIND INTERNAL NASTRAN NUMBER ASSOCIATED
C          WITH THE HINGE POINT (GRID POINT 'HINGE')
DO 1 N=1,NJ
IF(IGRID(N).EQ.HINGE) GO TO 2
1 CONTINUE
WRITE(6,100) HINGE
RETURN
2 INTH = N
C
C          INTH = INTERNAL NASTRAN NUMBER
C          OF HINGE POINT
C
C          GET DATA OFF OF TAPE 2 TO COMPUTE CENTER OF MASS
C
CALL RTAPE(IRUNNO,LOC(ICNT),AMP,NJ,3,500,72,2)
C
C          JLOC(I,N) = I-TH COORDINATE OF GRID POINT IGRID(N)
C          VECTOR RELATIVE TO HINGE POINT HINGE
DO 3 N=1,NJ
DO 3 I=1,3
JLOC(I,N) = AMP(N,I) - AMP(INTH,I)
3 CONTINUE
C
CALL RTAPE(IRUNNO,AMS(ICNT),AMP,NJ,1,500,72,2)
C
C          JMASS(N) = LUMPED MASS AT GRID POINT IGRID(N)
DO 4 N=1,NJ
JMASS(N) = AMP(N,1)
4 CONTINUE
C
IF(.NOT.LECHO) GO TO 17
WRITE(6,101)
WRITE(6,102) (N,IGRID(N),N,(JLOC(I,N),I=1,3),N,JMASS(N),N=1,NJ)
17 CONTINUE
C
C          COMPUTE CENTER OF MASS LOCATION
TOTMAS = 0.0
CMLOC(1) = 0.0
CMLOC(2) = 0.0
CMLOC(3) = 0.0
C
C          READ GRID POINT INERTIA TENSORS AND MASS MOMENTS
C
CALL RTAPE(IRUNNO,SMM(ICNT),AMP,NJ,3,500,72,2)
DO 11 N=1,NJ

```

```

DO 11 I=1,3
SMO(I,N) = AMP(N,I)
IF(SMO(I,N).EQ.0.0) GO TO 11
WRITE(6,107) (N,(SMO(I,N),I=1,3))
11 CONTINUE
IF(LECHO) CALL WRITE(SMO,3,NJ,6H SMO ,3)

```

C

```

CALL RTAPE(IRUNNO,AIN(ICNT),AMP,NJ,6,500,72,2)
DO 10 N=1,NJ
JINERT(1,1,N) = AMP(N,1)
JINERT(1,2,N) = -AMP(N,4)
JINERT(1,3,N) = -AMP(N,5)
JINERT(2,1,N) = -AMP(N,4)
JINERT(2,2,N) = AMP(N,2)
JINERT(2,3,N) = -AMP(N,6)
JINERT(3,1,N) = -AMP(N,5)
JINERT(3,2,N) = -AMP(N,6)
JINERT(3,3,N) = AMP(N,3)
10 CONTINUE
IF(LECHO) CALL WRITE(JINERT,9,NJ,6HJINERT,9)

```

C  
C  
C  
C  
C  
C

```

COMPUTE UNDEFORMED BODY INERTIA TENSOR
RELATIVE TO BODY CENTER OF MASS

```

```

DO 20 I=1,3
DO 20 J=1,3
20 BDINER(I,J) = 0.0
DO 19 N=1,NJ
CALL VEC SUB(JLOC(1,N),CMLDC,TEM)
CALL SUEOP(TEM,TEM,JMASS(N),TEM1)
CALL DYADD(BDINER,TEM1,BDINER)
CALL DYADD(BDINER,JINERT(1,1,N),BDINER)
19 CONTINUE

```

C  
C  
C  
C

```

READ IN MODE SHAPES AND RESORT THEM

```

```

DO 6 I=1,6
CALL RTAPE(IRUNNO,MOD(I,ICNT),ADM(1,1,I),NJ,NM,500,12,2)
6 CONTINUE
REWIND 3
WRITE(3) (((ADM(N,M,I),I=1,3),N=1,NJ),M=1,NM)
WRITE(3) (((ADM(N,M,I),I=4,6),N=1,NJ),M=1,NM)
REWIND 3
READ(3) (((PHIT(I,N,M),I=1,3),N=1,NJ),M=1,NM)
READ(3) (((PHIR(I,N,M),I=1,3),N=1,NJ),M=1,NM)
REWIND 3

```

C

```

IF(.NOT.LECHO) GO TO 18
WRITE(6,104)
DO 7 M=1,NM
WRITE(6,105)
WRITE(6,106) (N,M,(PHIT(I,N,M),I=1,3),
* (N,M,(PHIR(I,N,M),I=1,3),N=1,NJ)
7 CONTINUE
18 CONTINUE

```

C

```

WRITE(6,118)

```

```

WRITE(6,103) TOTMAS
WRITE(6,115)
WRITE(6,119) (BDINER(1,J),J=1,3)
WRITE(6,120) (BDINER(2,J),J=1,3)
WRITE(6,121) (BDINER(3,J),J=1,3)
WRITE(6,115)
WRITE(6,122) (CMLOC(I),I=1,3)
WRITE(6,115)
WRITE(6,125) INTH, HINGE
WRITE(6,115)

C
C      COMPUTE RESULTANT PARAMETERS
C
DO 8 M=1,NM
DO 8 I=1,3
FLA(I,M) = 0.0
FLB(I,M) = 0.0
FLC(I,M) = 0.0
DO 8 J=1,3
FLJ(I,J,M) = 0.0
FLD(I,J,M) = 0.0
8 CONTINUE
DO 9 M=1,NM
DO 9 MM=1,NM
DO 9 I=1,3
FCK(I,M,MM) = 0.0
DO 9 J=1,3
FCF(I,J,M,MM) = 0.0
9 CONTINUE

C
C      AMASS = 1.0/TOTMAS
C
DO 12 M=1,NM
DO 13 J=1,NJ

CALL SCLV(JMASS(J),PHIT(1,J,M),TEM)
CALL VECADD(FLA(1,M),TEM,FLA(1,M))

CALL VEC SUB(JLOC(1,J),CMLOC,TEM)
CALL SUEOP(PHIT(1,J,M),TEM,JMASS(J),TEM1)
CALL DYADD(FLD(1,1,M),TEM1,FLD(1,1,M))

CALL VECROS(JLOC(1,J),PHIT(1,J,M),TEM)
CALL SCLV(JMASS(J),TEM,TEM)
CALL VECADD(FLB(1,M),TEM,FLB(1,M))

IF(JINERT(1,1,J).EQ.0.0.AND.
*   JINERT(2,2,J).EQ.0.0.AND.
*   JINERT(3,3,J).EQ.0.0) GO TO 13

CALL DYDOTV(JINERT(1,1,J),PHIR(1,J,M),TEM)
CALL VECADD(FLC(1,M),TEM,FLC(1,M))

CALL VECXDY(PHIR(1,J,M),JINERT(1,1,J),TEM1)
CALL DYADD(FLJ(1,1,M),TEM1,FLJ(1,1,M))
13 CONTINUE

C
C      RESULTANT UNCOUPLED MODE DEPENDED PARAMETERS
C

```

```

CALL SCLV(AMASS,FLA(1,M),FLA(1,M))
CALL SCLV(AMASS,FLR(1,M),FLR(1,M))
12 CONTINUE

```

C  
C  
C  
C

READ MODAL STIFFNESS AND DAMPING DATA

```

CALL RTAPE(IRUNNO,KH(ICNT),KHH,NM,NM,12,12,2)
CALL RTAPE(IRUNNO,BH(ICNT),BHH,NM,NM,12,12,2)

```

C

```

DO 16 M=1,NM
WRITE (6,116) M
FLOM(M) = DSQRT(KHH(M,M))
ZETA(M) = BHH(M,M)/(2.0*FLOM(M))
WRITE(6,115)
WRITE(6,123) M,FLOM(M)
WRITE(6,115)
WRITE(6,124) M,ZETA(M)
WRITE(6,115)
WRITE(6,108) (M,(FLA(I,M),I=1,3))
WRITE(6,115)
WRITE(6,109) (M,(FLB(I,M),I=1,3))
WRITE(6,115)
WRITE(6,110) (M,(FLC(I,M),I=1,3))
WRITE(6,115)
WRITE(6,126) (M,(FLD(1,J,M),J=1,3))
WRITE(6,111) (M,(FLD(I,J,M),J=1,3),I=2,3)
WRITE(6,115)
WRITE(6,127) (M,(FLJ(1,J,M),J=1,3))
WRITE(6,112) (M,(FLJ(I,J,M),J=1,3),I=2,3)
WRITE(6,115)

```

```

16 CONTINUE
WRITE (6,105)

```

C  
C  
C

RESULTANT COUPLED MODE DEPENDENT PARAMETERS

```

DO 14 NN=1,NM
DO 14 MM=1,NM

```

C

```

DO 15 J=1,NJ
CALL SUEOP(PHIT(1,J,NN),PHIT(1,J,MM),JMASS(J),TEM1)
CALL DYADD(FCF(1,1,MM,NN),TEM1,FCF(1,1,MM,NN))

```

C

```

CALL DYDOTV(JINERT(1,1,J),PHIR(1,J,MM),TEM)
CALL VECROS(TEM,PHIR(1,J,NN),TFM)
HF = .5
CALL SCLV(HF,TFM,TEM)
CALL VECROS(PHIR(1,J,MM),PHIR(1,1,NN),TFM)
CALL SCLV(JMASS(J),TFM,TFM)
CALL VECADD(TEM,TFM,TEM)
CALL VECADD(FCK(1,MM,NN),TEM,FCK(1,MM,NN))

```

```

15 CONTINUE
WRITE (6,117) MM,NN
WRITE(6,115)
WRITE(6,128) (MM,NN,(FCF(1,J,MM,NN),J=1,3))
WRITE(6,113) (MM,NN,(FCF(I,J,MM,NN),J=1,3),I=2,3)
WRITE(6,115)
WRITE(6,114) (MM,NN,(FCK(I,MM,NN),I=1,3))

```

```

14 CONTINUE

```



C

```

100 FORMAT (1X, ' ** ERROR ** CANNOT FIND HINGE POINT', I5, ' IN GRID P
*OINT TABLE, RETURN OUT OF SUBROUTINE NBOD2')
101 FORMAT (////, ' GRID POINT LABELING, LOCATION AND MASS TABLES', //)
102 FORMAT (' IGRID(', I4, ') =', I5, 9X, ' JLOC(', I4, ') =', 3E15.7, 9X,
* ' JMASS(', I4, ') =', E15.7)
5 CONTINUE
A = 1.0/TOTMAS
CALL SCLV(A, CMLOC, CMLOC)
C
DO 5 N=1, NJ
TOTMAS = TOTMAS + JMASS(N)
CALL SCLV(JMASS(N), JLOC(1, N), TEM)
CALL VECADD(CMLOC, TEM, CMLOC)
8 Y(I) = X(I+1)
NY = 3
NR = IY(3)
RETURN
7 IF(IFORM.NE.307) GO TO 9
IY(1) = NR+1
Y(1) = X(1)
NY = 1
NR = X(2)*FAC
IF(NR.EQ.0) RETURN
IY(2) = NR
Y(2) = X(3)
NY = 2
NR = X(4)*FAC
IF(NR.EQ.0) RETURN
NR = NR-1
RETURN
9 IF(IFORM.NE.308) GO TO 10
NR = X(1)*FAC
IF(NR.EQ.0) GO TO 14
DO 11 I=1, 3
IY(I) = NR-1+I
11 Y(I) = X(I+1)
NY = 3
NR = IY(3)
RETURN
10 IF(IFORM.NE.309) GO TO 12
NR = X(1)*FAC
IF(NR.EQ.0) GO TO 14
DO 13 I=1, 2
IY(I) = NR-1+I
13 Y(I) = X(I+1)
NY = 2
NR = X(4)*FAC
IF(NR.EQ.0) RETURN
NR = NR-1
RETURN
103 FORMAT (' XMAS =', E15.7, 30X, ' (TOTAL MASS OF BODY)')
104 FORMAT (////, ' MODE SHAPE TABLE (INTERNAL NUMBERING)', //)
105 FORMAT (////)
106 FORMAT (' PHIT(', I4, ', ', I2, ') =', 3E15.7, 5X,
* ' PHIR(', I4, ', ', I2, ') =', 3E15.7)
107 FORMAT (////, ' WARNING MASS MOMENT AT GRID POINT NON-ZERO, THAT IS
* SMM ', I4, ') =', 3E15.7, /, ' THIS EFFECT IGNORED BY N-BOD2. N-BOD2
* ASSUMES EACH GRID POINT AT ELEMENT MASS CENTER ')

```

```

108 FORMAT (' FLA(',I2,') = ',3E15.7,' (COEFFICIENT TO FIND CENTER OF
* MASS LOCATION AFTER DEFORMATION)')
109 FORMAT (' FLB(',I2,') = ',3E15.7,' (COEFFICIENT TO FIND ANGULAR M
* OMENTUM DUE TO DEFORMATION)')
110 FORMAT (' FLC(',I2,') = ',3E15.7,' (COEFFICIENT TO FIND ANGULAR M
* OMENTUM DUE TO DEFORMATION)')
111 FORMAT (' FLO(',I2,') = ',3E15.7)
112 FORMAT (' FLJ(',I2,') = ',3E15.7)
113 FORMAT (' FCF(',I2,',',I2,') = ',3E15.7)
114 FORMAT (' FCK(',I2,',',I2,') = ',3E15.7,' (COEFFICIENT FOR CORIOLI
* S TORQUE DUE TO DEFORMATION)')
115 FORMAT (' ')
116 FORMAT (/3X,'FOR MODE',I3,' THE RESULTANT UNCOUPLED MODE DEPENDEN
* T PARAMETERS REQUIRED FOR N-BOD2 INPUT ARE')
117 FORMAT (/3X,'FOR MODES',I3,' AND',I3,' THE RESULTANT MODE DEPENDE
* NT CROSS COUPLING PARAMETERS REQUIRED FOR N-BOD2 INPUT ARE')
118 FORMAT ('1',5X,'***** RESULTANT FLEXIBLE BODY DATA REQUIRED FOR IN
* PUT TO N-BOD2 *****',///)
119 FORMAT (' ',3E15.7,' (UNDEFORMED INERTIA TENSOR OF THE')
120 FORMAT (' XI = ',3E15.7,' FLEXIBLE BODY IN BODY COORDINATES')
121 FORMAT (' ',3E15.7,' RELATIVE TO BODY CENTER OF MASS)')
122 FORMAT (' CA = ',3E15.7,' (CENTER OF MASS VECTOR, HINGE POINT T
* O CM)')
123 FORMAT (' FLOM(',I2,') = ',E15.7,' RAD/SEC (MODAL FREQUENCY)')
124 FORMAT (' ZETA(',I2,') = ',E15.7,' (MODAL DAMPING ZETA)')
125 FORMAT (' CB = VECTOR IN CONTIGUOUS BODY TO HINGE POINT DEFINED
* AT INTERNAL GRID POINT',I5,' (EXTERNAL GRID POINT',I5,' IN BULK D
* ATA)')
126 FORMAT (' FLD(',I2,') = ',3E15.7,' (COEFFICIENT TO FIND INERTIA T
* ENSOR AFTER DEFORMATION)')
127 FORMAT (' FLJ(',I2,') = ',3E15.7,' (COEFFICIENT FOR CENTRIPITAL T
* ORQUE DUE TO DEFORMATION)')
128 FORMAT (' FCF(',I2,',',I2,') = ',3E15.7,' (COEFFICIENT FOR CENTRIP
* ITAL TORQUE DUE TO DEFORMATION)')
RETURN
END

```

#### SUBROUTINE INTERP(INPT,IMAT)

```

C
C THIS ROUTINE SETS UP A CODE SEQUENCE WHICH WILL BE USED
C TO READ MIXED INTEGER AND FLOATING POINT DATA FROM
C THE DMI CARDS
C BLANK FIELD ON LAST CARD WILL BE INTERPRETED AS AN INTEGER
C BUT READ LATER IN AS ZERO, A CHECK FOR WHICH IS MADE
C

```

```

INTEGER INPT(80)
DATA IDOT,IEE/1H.,1HE/
IF(INPT(12).EQ.IDOT.AND.INPT(21).EQ.IEE) GO TO 1
IF(INPT(44).EQ.IDOT.AND.INPT(53).EQ.IEE) GO TO 2
IMAT = 310

```

```

C
C 310 - FORMAT I,E,I,E

```

```

RETURN

```

```

2 IF(INPT(60).EQ.IDOT.AND.INPT(69).EQ.IEE) GO TO 3
IMAT = 309

```

```

C
C 309 - FORMAT I,E,E,I

```

```

RETURN
3 IMAT = 308
C
RETURN
1 IF(INPT(28).EQ.IDOT.AND.INPT(37).EQ.IEE) GO TO 4
IF(INPT(60).EQ.IDOT.AND.INPT(69).EQ.IEE) GO TO 5
IMAT = 307
C
RETURN
5 IMAT = 306
C
RETURN
4 IF(INPT(44).EQ.IDOT.AND.INPT(53).EQ.IEE) GO TO 6
IMAT = 305
C
RETURN
6 IF(INPT(60).EQ.IDOT.AND.INPT(69).EQ.IEE) GO TO 7
IMAT = 304
C
RETURN
7 IMAT = 303
C
RETURN
END

```

SUBROUTINE DECODE(IFORM,X,NR,Y,IY,NY)

```

C
C*** ERROR IN TERMINOLOGY ELEMENTS COMING IN BY ROWS,
C NR AND IY(I) REFER TO COLUMN NUMBER
C
C OUTPUT OF DECODE
C NY = NUMBER OF REAL NUMBERS ON DMI CARD
C IY(I) = ROW LOCATION FOR I-TH REAL NUMBER ON DMI CARD
C Y(I) = I-TH REAL NUMBER
C NR = NEXT REAL NUMBER TO BE PUT IN ROW NR+1 UNLESS
C FIELD 1 OF NEXT CARD CONTAINS AN INTEGER
C

```

```

REAL*8 X(4), Y(4)
INTEGER IY(4)
FAC = 1.00001D+08
IF(IFORM.NE.303) GO TO 1
DO 2 I=1,4
IY(I) = NR+I
2 Y(I) = X(I)
NY = 4
NR = IY(4)
RETURN
1 IF(IFORM.NE.304) GO TO 3
DO 4 I=1,3
IY(I) = NR+I
4 Y(I) = X(I)
NY = 3
NR = X(4)*FAC-1
RETURN
3 IF(IFORM.NE.305) GO TO 5

```

```

      DD 6 I=1,2
      IY(I) = NR+I
6     Y(I) = X(I)
      NY = 2
      NR = X(3)*FAC
      IF(NR.EQ.0) RETURN
      IY(3) = NR
      Y(3) = X(4)
      NY = 3
      RETURN
5     IF(IFORM.NE.306) GO TO 7
      IY(1) = NR+1
      Y(1) = X(1)
      NY = 1
      NR = X(2)*FAC
      IF(NR.EQ.0) RETURN
      DO 8 I=2,3
      IY(I) = NR-2+I
12    IF(IFORM.NE.310) GO TO 14
      NR = X(1)*FAC
      IF(NR.EQ.0) GO TO 14
      IY(1) = NR
      Y(1) = X(2)
      NR = X(3)*FAC
      IF(NR.EQ.0) RETURN
      IY(2) = NR
      Y(2) = X(4)
      NY = 2
      RETURN
14   WRITE(6,100) IFORM
100  FORMAT (' IFORM =',I5,' ERROR')
      STOP
      END

```

```

SUBROUTINE INTAPE (NTAPE,TAPEID)
REAL*8 TAPEID,BUF,EOT
DATA IZ1,BUF,EOT/1,0,D 0,3HEOT/
DATA NOT / 6/

```

```

C
C INITIALIZE TAPE FOR SUBROUTINE WTAPE.
C CALLS FORMA SUBROUTINE PAGEHD.
C CODED BY RF HRUDA. JULY 1968.
C REVISED BY R A PHILIPPUS. APRIL 1969.
C
C SUBROUTINE ARGUMENTS (ALL INPUT)
C NTAPE = NUMBER OF TAPE. (E.G. 10).
C TAPEID = TAPE IDENTIFICATION. (E.G. T1234). (A6 FORMAT).
C
2001 FORMAT (//// 14H LOGICAL UNIT 12, 7H, TAPE A6,
*          23H, HAS BEEN INITIALIZED.)
C
      REWIND NTAPE
      WRITE (NTAPE) TAPEID,IZ1,EOT,(BUF,I=1,16)
      REWIND NTAPE

```

WRITE (NOT,2001) NTAPE,TAPEID

C

RETURN  
END

SUBROUTINE WTAPE (A,NRA,NCA,ANAME,KR,NTAPE)  
REAL\*8 A,ANAME,IRUNNO,DATE,  
\* BUF,EOT,DENSE,TAPEID,IEOTCK  
DIMENSION A(KR,1)  
DATA IRUNNO,DATE/6HPREPRS,6HDATE /  
DATA BUF,EOT,DENSE/O.D 0,3HEOT,5HDENSE/  
DATA NOT / 6/

C

C WRITE MATRIX A ON TAPE.

C INITIALIZE TAPE WITH SUBROUTINE INTAPE.

C REWIND TAPE BEFORE FIRST USE OF THIS SUBROUTINE.

C NOTE...THIS ROUTINE IS DESIGNED SPECIFICALLY FOR WRITING ON A DISK

C

C (EG CDC-6400 DISK). USING THIS ROUTINE TO WRITE ON A PHYSICAL

C TAPE DIRECTLY (IE WITHOUT USING THE DISK AS AN INTERMEDIARY)

C WILL PROBABLY GIVE POOR RESULTS (DUE TO THE TOLERANCE

C CHARACTERISTICS OF A TAPE DRIVE) AND SHOULD BE AVOIDED IF AT

C ALL POSSIBLE.

C ...THE CDC-6400 DISK IS AUTOMATICALLY ENDFILED AFTER EACH WRITE.

C CODED BY W A BENFIELD. MARCH 1966.

C REVISED BY R A PHILIPPUS. APRIL 1969.

C REVISED BY RF HRUDA. NOVEMBER 1970.

C

C SUBROUTINE ARGUMENTS (ALL INPUT)

C A = MATRIX TO BE WRITTEN ON TAPE. SIZE(NRA,NCA).

C NRA = NUMBER OF ROWS OF MATRIX A.

C NCA = NUMBER OF COLS OF MATRIX A.

C ANAME = MATRIX IDENTIFICATION. (A6 FORMAT).

C KR = ROW DIMENSION OF A IN CALLING PROGRAM.

C NTAPE = NUMBER OF TAPE. (E.G. 10).

C

C INTERNAL VARIABLES THAT ARE PUT ON TAPE (TRANSFERRED THRU COMMON).

C IRUNNO IS RUN NUMBER OF PROBLEM. (A6 FORMAT).

C DATE IS DATE. (A6 FORMAT). FOR EXAMPLE 15FE65.

C

C IF (NRA .LT. 1 .OR. NCA .LT. 1) GO TO 999

C

C SEARCH TAPE FOR END OF WRITTEN DATA.

10 READ (NTAPE) TAPEID,LN,IEOTCK

IF (IEOTCK .EQ. EOT) GO TO 20

READ (NTAPE)

GO TO 10

C

C END OF WRITTEN DATA HAS BEEN FOUND.

20 BACKSPACE NTAPE

WRITE (NTAPE) TAPEID,LN,BUF,IRUNNO,ANAME,NRA,NCA,DATE,DENSE,

\* (BUF,I=1,10)

WRITE (NTAPE) ((A(I,J),I=1,NRA),J=1,NCA)

LN = LN + 1

WRITE (NTAPE) TAPEID,LN,EOT,(BUF,I=1,16)

BACKSPACE NTAPE  
RETURN

C  
999 WRITE (NOT,1000)  
1000 FORMAT (1H1,43HERROR IN SUBROUTINE WTAPE, PROGRAM STOPPED.)  
STOP  
END

SUBROUTINE LTAPE (NTAPE)  
REAL\*8 TAPEID, IRUNNO, ANAME, IEOTCK, DATE, ITYPE, ICHK, EOT,  
\* DENSE, SPARSE, SPART  
DATA NOT / 6/  
DATA EOT, DENSE, SPARSE, SPART / 3HEOT, 5HDENSE, 6HSPARSE, 5HSPART /

C  
C LIST HEADINGS OF MATRICES ON TAPE.  
C CALLS FORMA SUBROUTINE PAGEHD.  
C CODED BY RF HRUDA. JULY 1968. REVISED NOVEMBER 1970.  
C REVISED BY R. A. PHILIPPUS. APRIL 1969.

C  
C SUBROUTINE ARGUMENTS (ALL INPUT)  
C NTAPE = NUMBER OF TAPE. (E.G. 10).

C  
2001 FORMAT (//36X35HLISTING OF MATRICES ON LOGICAL UNITI3,7H, TAPE A6)  
2002 FORMAT (//30X35HLISTING OF MATRICES ON LOGICAL UNITI3,7H, TAPE A6,  
\* 12H (CONTINUED))  
2003 FORMAT (27X69(1H-)/27X3HNO.3X7HRUN NO.4X4HNAMES5X5HNROWS4X5HNCOLS4X  
\* 4HDATE6X3HNNZ3X9HPARTITION/  
\* 27X3H---3X6H-----4X6H-----4X5H-----  
\* 4X5H-----3X6H-----5X3H---3X9H----- / )  
2004 FORMAT (25XI5,3XA6,4XA6,3XI5,4XI5,4X46,3XI5,3XI4,1H/I4)  
2005 FORMAT (/27X12HEND OF LIST.)

C  
REWIND NTAPE  
READ (NTAPE) TAPEID  
REWIND NTAPE  
L=0

C  
12 CONTINUE  
IF (L .EQ. 0) WRITE (NOT,2001) NTAPE,TAPEID  
IF (L .NE. 0) WRITE (NOT,2002) NTAPE,TAPEID  
WRITE (NOT,2003)  
NLINE=1

13 L=L+1  
READ (NTAPE) TAPEID, LN, IEOTCK, IRUNNO, ANAME, NR, NC, DATE, ITYPE, NNZ,  
\* NP, NPT  
IF (L .EQ. 1) ICHK = IRUNNO  
IF (ICLK .EQ. IRUNNO) GO TO 15  
NLINE=NLINE+1  
WRITE (NOT,2004)  
ICLK = IRUNNO

15 IF (IEOTCK .EQ. EOT) GO TO 30  
READ (NTAPE)  
IF (ITYPE .EQ. DENSE ) WRITE (NOT,2004)  
\* LN, IRUNNO, ANAME, NR, NC, DATE

```

IF (ITYPE .EQ. DENSE ) GO TO 20
IF (ITYPE .EQ. SPARSE) WRITE (NOT,2004)
* LN,IRUNNO,ANAME,NR,NC,DATE,NNZ
IF (ITYPE .EQ. SPARSE) GO TO 20
IF (ITYPE .EQ. SPART ) WRITE (NOT,2004)
* LN,IRUNNO,ANAME,NR,NC,DATE,NNZ,NP,NPT
IF (ITYPE .EQ. SPART ) GO TO 20
WRITE (NOT,2004) LN,IRUNNO,ANAME,NR,NC,ITYPE
20 NLINE=NLINE+1
IF(NLINE.GT.43) GO TO 12
GO TO 13

```

```

C
30 WRITE (NOT,2004) LN,IEOTCK
WRITE (NOT,2005)
REWIND NTAPE
RETURN
END

```

```

SUBROUTINE RTAPE (IARUNO,IANAME, A,NRA,NCA, KR,KC,NTAPE)
REAL*8 A,IARUNO,IANAME,TAPEID,IEOTCK,ITRUNO,ITNAME,
* DATE,ITYPE,DENSE,EOT
DIMENSION A(KR,1)
DATA NOT / 6 /
DATA DENSE,EOT / 5HDENSE,3HEOT /

```

```

C
C READ MATRIX A FROM TAPE BY IDENTIFICATION OF IARUNO,IANAME.
C CALLS FORMA SUBROUTINES LTAPE,PAGEHD,ZZHOMB.
C CODED BY WA BENFIELD. JUNE 1966.
C LAST REVISION BY R F HRUDA. SEPTEMBER 1971.
C

```

```

C SUBROUTINE ARGUMENTS

```

```

C IARUNO = INPUT RUN NUMBER OF MATRIX A. (A6 FORMAT).
C IANAME = INPUT MATRIX IDENTIFICATION. (A6 FORMAT).
C A = OUTPUT MATRIX READ FROM TAPE. SIZE(NRA,NCA).
C NRA = OUTPUT NUMBER OF ROWS OF MATRIX A. WILL BE READ FROM TAPE.
C NCA = OUTPUT NUMBER OF COLS OF MATRIX A. WILL BE READ FROM TAPE.
C KR = INPUT ROW DIMENSION OF A IN CALLING PROGRAM.
C KC = INPUT COL DIMENSION OF A IN CALLING PROGRAM.
C NTAPE = INPUT NUMBER OF TAPE. (E.G. 10).
C

```

```

3001 FORMAT (29H1RTAPE CANNOT FIND RUNNO = A6. /
* 21X 8HANAME = A6 / 29X 6H----- )

```

```

C
C NTIME = 0
C SEARCH TAPE FOR CORRECT HEADING.
5 READ (NTAPE) TAPEID, LN, IEOTCK, ITRUNO, ITNAME, NRA, NCA, DATE, ITYPE, NNZ
IF (ITRUNO .EQ. IARUNO .AND. ITNAME .EQ. IANAME) GO TO 10
IF (IEOTCK .EQ. EOT) GO TO 20
READ (NTAPE)
GO TO 5

```

```

C
C MATRIX HAS BEEN FOUND.
10

```

```

                                ERROR=1
IF (ITYPE .NE. DENSE .AND. NNZ .NE. 0) GO TO 999
                                ERROR=2
IF (NRA.GT.KR .OR. NCA.GT.KC) GO TO 999
READ (NTAPE) ((A(I,J),I=1,NRA),J=1,NCA)
RETURN
C
C MATRIX CANNOT BE FOUND. SEARCH TAPE ONCE MORE.
20 NTIME = NTIME+1
                                ERROR=3
IF (NTIME .EQ. 2) GO TO 998
REWIND NTAPE
GO TO 5
C
998 WRITE (NOT,3001) IARUND,IANAME
999 CALL LTAPE (NTAPE)
WRITE (NOT,3002) NERROR
3002 FORMAT (1H1,43HERROR IN SUBROUTINE RTAPE, PROGRAM STOPPED,
*          10H NERROR = ,I3)
STOP
END

SUBROUTINE WRITE (A,NR,NC,ANAME,KR)
REAL*8 A,ANAME
DIMENSION A(KR,1)
DATA NOT / 6/
C
C WRITE MATRIX OF REAL NUMBERS ON PAPER.
C REQUIRES 123 COLUMN (MINIMUM) PRINTER.
C UP TO 10 DATA FIELDS PER LINE. PRINTS ONLY NON-ZERO FIELD ROWS.
C CALLS FORMA SUBROUTINE PAGEHD.
C CODED BY RL WÖHLEN. DECEMBER 1968.
C
C SUBROUTINE ARGUMENTS (ALL INPUT)
C A = MATRIX TO BE PRINTED. SIZE(NR,NC).
C NR = NUMBER OF ROWS IN MATRIX A.
C NC = NUMBER OF COLS IN MATRIX A.
C ANAME = MATRIX IDENTIFICATION. (A6 FORMAT).
C KR = ROW DIMENSION OF A IN CALLING PROGRAM.
C
2010 FORMAT (//15H OUTPUT MATRIX A6,2X 1H(14,2H X 14,2H ) //
*          10X,10(7X,1H( 12,1H))//)
2020 FORMAT (//15H OUTPUT MATRIX A6,2X 1H(14,2H X 14,2H )
*          3X, 9HCONTINUED //10X,10(7X,1H( 12,1H))//)
2030 FORMAT (1X,2I5,2X,1P10D11.3)
2040 FORMAT (14HOEND OF WRITE.)
C
C PULL UP A NEW PAGE FOR MATRIX AND PRINT MATRIX NAME.
WRITE (NOT,2010) ANAME,NR,NC,(L,L=1,10)
NLINE = 0
C
DO 60 I=1,NR
NZERO = 0
JS = 1
10 JE = JS+9

```



```

      IF (JE .GT. NC) JE=NC
C   SEE IF ELEMENTS ARE ZERO.
      DO 20 J=JS,JE
      IF (A(I,J) .NE. 0.D 0) GO TO 30
20  CONTINUE
      GO TO 40
30  NLINE = NLINE+1
      IF (NLINE .LE. 44) GO TO 35
      WRITE (NOT,2020) ANAME,NR,NC,(L,L=1,10)
      NLINE = 1
35  WRITE (NOT,2030) I,JS,(A(I,J), J=JS,JE)
      NZERO = 1
40  IF (JE .EQ. NC) GO TO 50
      JS = JS+10
      GO TO 10
C   SKIP A SPACE BETWEEN EACH ROW IF THERE ARE MORE THAN 10 COLUMNS
C   AND SOMETHING HAS BEEN WRITTEN.
50  IF (NC.LE.10 .OR. NZERO.EQ.0 .OR. I.EQ.NR) GO TO 60
      NLINE = NLINE+1
      WRITE (NOT,2030)
60  CONTINUE
C
      WRITE (NOT,2040)
      RETURN
      END

```

```

SUBROUTINE VECADD(V1,V2,S)
C   ADDS VECTOR V1 TO V2 RESULT IN S.
      IMPLICIT REAL*8(A-H,O-Z,$)
      DIMENSION V1(3),V2(3), S(3)
      S(1) = V1(1) + V2(1)
      S(2) = V1(2) + V2(2)
      S(3) = V1(3) + V2(3)
      RETURN
      END

```

```

SUBROUTINE VECSUB(V1,V2,D)
C   SUBTRACTS VECTORS V1-V2=D
      IMPLICIT REAL*8(A-H,O-Z,$)
      DIMENSION V1(3),V2(3),D(3)
      D(1) = V1(1) - V2(1)
      D(2) = V1(2) - V2(2)
      D(3) = V1(3) - V2(3)
      RETURN
      END

```

```

C   SUBROUTINE SCLV(SC,V,P)
    SCALAR * VECTOR
    IMPLICIT REAL*8(A-H,O-Z,$)
    DIMENSION V(3),P(3)
    P(1) = SC*V(1)
    P(2) = SC*V(2)
    P(3) = SC*V(3)
    RETURN
    END

```

```

C   SUBROUTINE VECDOT(V1,V2,D)
    VECTOR DOT PRODUCT
    IMPLICIT REAL*8(A-H,O-Z,$)
    DIMENSION V1(3),V2(3)
    D = V1(1)*V2(1) + V1(2)*V2(2) + V1(3)*V2(3)
    RETURN
    END

```

```

C   SUBROUTINE VECROS (V1,V2,C)
    VECTOR CROSS PRODUCT C = V1 X V2
    IMPLICIT REAL*8(A-H,O-Z,$)
    DIMENSION V1(3),V2(3),C(3)
    C(1) = V1(2)*V2(3) - V1(3)*V2(2)
    C(2) = V1(3)*V2(1) - V1(1)*V2(3)
    C(3) = V1(1)*V2(2) - V1(2)*V2(1)
    RETURN
    END

```

```

C   SUBROUTINE TRIPVP(V1,V2,V)
C   COMPUTES STANDARD VECTOR TRIPLE PRODUCT
C
C        $V = V1 \times (V1 \times V2)$ 
C        $= V1 * (V1 \cdot V2) - V2 * (V1 \cdot V1)$ 
C
C   IMPLICIT REAL*8(A-H,O-Z,$)
C   DIMENSION V1(3),V2(3),V(3)
C   A = V1(1)*V2(1) + V1(2)*V2(2) + V1(3)*V2(3)
C   B = V1(1)*V1(1) + V1(2)*V1(2) + V1(3)*V1(3)
C   V(1) = V1(1)*A - V2(1)*B
C   V(2) = V1(2)*A - V2(2)*B
C   V(3) = V1(3)*A - V2(3)*B
C   RETURN
C   END

```

```

SUBROUTINE DYADD(D1,D2,D)
C   ADDS TWO DYADS
C   D = D1 + D2
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION D1(3,3), D2(3,3), D(3,3)
DO 1 I=1,3
DO 1 J=1,3
1 D(I,J) = D1(I,J) + D2(I,J)
RETURN
END

```

```

SUBROUTINE SCLD(A,D,T)
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION D(3,3),T(3,3)
C   MULTIPLY SCALAR BY A TENSOR
T(1,1) = A*D(1,1)
T(2,1) = A*D(2,1)
T(3,1) = A*D(3,1)
T(1,2) = A*D(1,2)
T(2,2) = A*D(2,2)
T(3,2) = A*D(3,2)
T(1,3) = A*D(1,3)
T(2,3) = A*D(2,3)
T(3,3) = A*D(3,3)
RETURN
END

```

```

SUBROUTINE DYDOTV(A,V,D)
C   SCALAR DOT PRODUCT OF DYAD AND VECTOR
C   D = A.V
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION D(3),A(3,3),V(3)
DO 1 I=1,3
D(I) = 0
DO 1 J=1,3
1 D(I) = D(I) + A(I,J)*V(J)
RETURN
END

```

```

SUBROUTINE VXDYOV(V1,DY,V)
C   COMPUTES VECTOR X (DYAD . VECTOR)
C   V = V1 X (DY . V1)
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION V1(3),V2(3),V(3),DY(3,3)
DO 1 K=1,3

```

```

V2(K) = 0.00
DO 1 J=1,3
1 V2(K) = V2(K) + DY(K,J)*V1(J)
V(1) = V1(2)*V2(3) - V1(3)*V2(2)
V(2) = V1(3)*V2(1) - V1(1)*V2(3)
V(3) = V1(1)*V2(2) - V1(2)*V2(1)
RETURN
END

```

```

SUBROUTINE DYTOV (D,X1,X)
C USE TO TAKE SCALAR DOT PRODUCT OF TRANSPOSE OF
C TENSOR D WITH VECTOR X1
C NEEDED SINCE TENSORS IN SYMMERIC MATRIX OF INERTIA TENSORS ARE IN
C NON SYMMETRIC
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION D(3,3),X1(3),X(3)
DO 1 I=1,3
X(I) = 0
DO 1 J=1,3
X(I) = X(I) + D(J,I)*X1(J)
1 CONTINUE
RETURN
END

```

```

SUBROUTINE VODYOV(V1,DY,V2,X)
C COMPUTES THE SCALAR TRIPLE PRODUCT
C VECTOR . (DYAD . VECTOR)
C V1.(DY.V2)
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION V1(3),DY(3,3),V2(3),TEM(3)
DO 1 K=1,3
TEM(K) = 0.00
DO 1 J=1,3
1 TEM(K) = TEM(K) + DY(K,J)*V2(J)
X = 0
DO 2 J=1,3
2 X = X + V1(J)*TEM(J)
RETURN
END

```

```

SUBROUTINE DYOP(V,D)
C TRANSFORMS VECTOR V1 INTO SKEW DYAD
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION V(3),D(3,3)
D(1,1) = 0
D(1,2) = V(3)

```

```

D(1,3) = -V(2)
D(2,1) = -V(3)
D(2,2) = 0
D(2,3) = V(1)
D(3,1) = V(2)
D(3,2) = -V(1)
D(3,3) = 0
RETURN
END

```

```

SUBROUTINE SUEOP(V1,V2,XM,D)
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION V1(3),V2(3),D(3,3)
C
C   USED TO COMPUTE THE PSUEDO INERTIA TENSOR
C   OF BODY LAMBA WITH RESPECT TO THE ORIGIN OF NEST K-1 AND
C   THE HINGE POINT I-1 WHICH IS ON THE TOPOLOGICAL PATH FROM
C   BODY 1 TO BODY LAMBA
C   BLOCK G SUPPER GAMBA, SUB K-1, I-1 EQUATION 2-55 OF X-732-71-70
C   D = XM*((V1.V2)*1 - V2 V1)
C
C   XM - SCALAR
C   V1 - VECTOR
C   V2 - VECTOR
C   1 - UNIT DYAD
C   * - SCALAR MULTIPLICATION
C   . - VECTOR SCALAR MULTIPLICATION
C   BLANK - TENSOR MULTIPLICATION
C   NOTE THAT IN GENERAL THE PSUEDO INERTIA TENSOR IS NON SYMMETRIC
C
D(1,1) = XM*(V1(2)*V2(2) + V1(3)*V2(3))
D(1,2) = -XM*V2(1)*V1(2)
D(1,3) = -XM*V2(1)*V1(3)
D(2,1) = -XM*V2(2)*V1(1)
D(2,2) = XM*(V2(1)*V1(1) + V2(3)*V1(3))
D(2,3) = -XM*V2(2)*V1(3)
D(3,1) = -XM*V2(3)*V1(1)
D(3,2) = -XM*V2(3)*V1(2)
D(3,3) = XM*(V2(1)*V1(1) + V2(2)*V1(2))
RETURN
END

```

```

SUBROUTINE VECXDY(P,T,D)
REAL*8 P(3), T(3,3), D(3,3)
C
C   COMPUTES VECTOR CROSS DYAD
C   D = P X T
C
D(1,1) = P(2)*T(3,1) - P(3)*T(2,1)
D(1,2) = P(2)*T(3,2) - P(3)*T(2,2)
D(1,3) = P(2)*T(3,3) - P(3)*T(2,3)
D(2,1) = P(3)*T(1,1) - P(1)*T(3,1)

```

```
D(2,2) = P(3)*T(1,2) - P(1)*T(3,2)
D(2,3) = P(3)*T(1,3) - P(1)*T(3,3)
D(3,1) = P(1)*T(2,1) - P(2)*T(1,1)
D(3,2) = P(1)*T(2,2) - P(2)*T(1,2)
D(3,3) = P(1)*T(2,3) - P(2)*T(1,3)
RETURN
END
```

## REFERENCES

1. McCormick, Caleb W., "The NASTRAN User's Manual," NASA Document SP-222, September 1970.
2. Douglas, Frank J., "The NASTRAN Programmer's Manual," NASA Document SP-223, September 1970.

## BIBLIOGRAPHIC DATA SHEET

1. Report No. NASA TP-1145	2. Government Accession No.	3. Recipient's Catalog No.	
4. Title and Subtitle  The N-BOD2 User's and Programmer's Manual		5. Report Date February 1978	
		6. Performing Organization Code 712	
7. Author(s) Harold P. Frisch		8. Performing Organization Report No. G 7702 F-22	
9. Performing Organization Name and Address  Goddard Space Flight Center Greenbelt, Maryland 20771		10. Work Unit No.	
		11. Contract or Grant No.	
		13. Type of Report and Period Covered  Technical Paper	
12. Sponsoring Agency Name and Address  National Aeronautics and Space Administration Washington, D.C. 20546		14. Sponsoring Agency Code	
15. Supplementary Notes			
16. Abstract  A general purpose digital computer program has been developed entitled N-BOD2. The program is designed to aid in the analysis of spacecraft attitude dynamics; however it is applicable to a much broader class of problems. The program provides the analyst with the capability of automatically deriving and numerically solving the equations of motion of any system that can be modeled as a topological tree of coupled rigid bodies, flexible bodies, point masses, and symmetrical momentum wheels. Two modes of output are available; the composite system equations of motion may be outputted on a line printer in a symbolic form that may be easily translated into common vector-dyadic notation, or the composite system equations of motion may be solved numerically and any desirable set of system state-variables outputted as a function of time.			
17. Key Words (Selected by Author(s)) Spacecraft attitude dynamics, Stabilization and control, Rigid body dynamics, Flexible body dynamics, Applied mathematics		18. Distribution Statement  STAR Category 18 Unclassified-Unlimited	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 306	22. Price* \$11.75



National Aeronautics and  
Space Administration

Washington, D.C.  
20546

Official Business

Penalty for Private Use, \$300

SPECIAL FOURTH CLASS MAIL  
BOOK

Postage and Fees Paid  
National Aeronautics and  
Space Administration  
NASA-451



**NASA**

**POSTMASTER: If Undeliverable (Section 158  
Postal Manual) Do Not Return**

---