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

63-3-5

AFSWC-TDR-63-11

SWC
TDR
63-11

SPINNING UNGUIDED ROCKET TRAJECTORY DIGITAL COMPUTER PROGRAM (SPURT)

by

Allen D. Dayton
Lt USAF

TECHNICAL DOCUMENTARY REPORT NUMBER AFSWC-TDR-63-11

February 1963

Test Directorate
AIR FORCE SPECIAL WEAPONS CENTER
Air Force Systems Command
Kirtland Air Force Base
New Mexico

Aerospace Environmental Research Program

Best Available Copy

HEADQUARTERS
AIR FORCE SPECIAL WEAPONS CENTER
Air Force Systems Command
Kirtland Air Force Base
New Mexico

FOREWORD

The author wishes to acknowledge the assistance of Mr. Carl S. Christensen, who did much of the original logic and programing of SPURT. Mr. Christensen, now with the Aerospace Corporation, Los Angeles, California, was formerly a lieutenant at the Air Force Special Weapons Center.

ABSTRACT

SPURT is a five-degree-of-freedom trajectory digital computer program for spinning unguided space probe vehicles. The program was written for the Control Data Corporation, 1604 digital computer.

SPURT will compute trajectories for a vehicle up to a maximum of ten stages and has provision for computing the trajectories of the separated stages.

This generalized program computes the trajectory over an oblate spheroidal, rotating Earth with atmosphere, in a geocentric rectangular coordinate system. All input and output data are in geodetic coordinates.

Coasting flight trajectories are computed in two subroutines. The first is a Keplerian solution, which also computes orbital elements and "look angles" for various tracking stations. The second uses three-degree-of-freedom point mass equations solved by numerical integration.

The program will prepare two special output tapes. One is used in plotting output data and the other is used to prepare a special tape for the Atlantic Missile Range.

PUBLICATION REVIEW

This report has been reviewed and is approved.

ALBERT L. HALEY
Colonel USAF
Director, Test Directorate

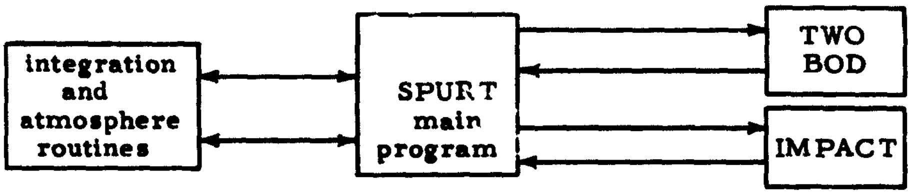
JOHN J. DISHUCK
Colonel USAF
DCS/Plans & Operations

CONTENTS

1. INTRODUCTION.

The Spinning Unguided Rocket Trajectory (SPURT) digital computer program is designed to provide a generalized computer program for calculating trajectories of spinning unguided space probe vehicles such as the SLV-1B. Many trajectory programs are available[1-5] but none meet the demands required by the Space Vehicle Branch (SWTTS) of AFSWC. The program is written in a mixture of FORTRAN and CODAP for use on the CDC-1604 computer and utilizes subroutines of the CO-OP library. The program is designed to handle up to a ten stage vehicle for both powered and unpowered flight with provisions to calculate the trajectories of the separated "expended" stages.

The main program computes the powered portion of the trajectory and controls entry into the various subroutines. All unpowered flight trajectories are computed in the two subroutines "TWO BOD" and "IMPACT." Both of these subroutines have the capability to calculate the trajectory of all the separated stages and of the payload. TWO BOD is a Keplerian trajectory program with provisions for calculating look angles of various tracking stations, while IMPACT integrates the equation of a point mass with drag.



The main program uses a five-degree-of-freedom, three-dimensional system with the sixth degree constrained by a table of spin rates that are read into the program. An oblate rotating earth and associated gravitational potential, standard 1962 atmosphere, and altitude-dependent wind provisions are also incorporated into the program. The position vector is calculated in a rotating Earth-centered coordinate system, while the angular positions are calculated in a launch-centered coordinate system.

The procedure includes provisions for coasting periods between stages, which are terminated by time. Thrust is computed from thrust vs. time tables and corrected for atmosphere back pressure.

Aerodynamic forces and moments about the center of mass are interpolated from a table of Mach number dependent coefficients.

Computations are carried out using either the Adams or the Runge-Kutta method of numerical integration. Both methods can be used in either fixed or variable step size-mode.

The program has the provision for writing two special output tapes. One, a plot tape, is designed to be used with a special plot program for use on the AFSWC plotter and can plot any output variable against any other output variable. The second, a BATT tape, is used with the BATT program to prepare magnetic tapes to meet specified Atlantic Missile Range formats.

The program running time is approximately 3 to 4 minutes for a four-stage vehicle similar to the SLV-1B. The integration and atmosphere routines are compiled separately in octal locations 6000 to 7514. The rest of the program is compiled in fixed binary mode starting at octal location 10050. The two parts are then put together to be read into the computer. This method has the advantage that the integration and atmosphere routines are not recompiled every time a change is made in the rest of the program.

The program is stored in core according to the following octal addresses.

6000 - 7142	Integration Routine		
7300 - 7514	Atmosphere Routine		
7660 - 10026	SQRTF, EXPF, & LOGF Routines		
10050 - 34102	SPURT Routine	(main program)	
34103 - 34262	SETTAB Routine		subroutines
34263 - 34350	ECLOCK Routine		subroutines
34351 - 34364	SCLOCK Routine		subroutines
34365 - 56750	TWO BOD Routine		subroutines
56751 - 61516	IMPACT Routine		subroutines
61517 - 61615	GEODED Routine		subroutines
61616 - 62006	ROTATE Routine		subroutines
62013 - 67420	LIBRARY Routines		
70037 - 71343	COMMON		

2. EQUATIONS.

SYMBOLS

		Dimensions
A	Axial moment of inertial	ft^2 - slug
A_E	EXIT area of nozzle	in^2
A_w	Azimuth angle* of the wind	deg
Ax	Azimuth angle of the X_L axis	deg
a_E	Equatorial radius of the Earth	ft
B	Longitudinal moment of inertia	ft^2-slug
$C \triangleq$	$1/\sqrt{1-(2f - f^2)\sin^2 \theta_G}$ (Ref 6)	ft
C_D	Drag coefficient	None
C_{DB}	Powered flight drag coefficient	None
C_{DC}	Coasting flight drag coefficient	None
C_{N_α}	$\left(\frac{\partial C_N}{\partial \alpha}\right)$ Normal force coefficient with respect to angle of attack	per radian
C_{M_α}	Moment coefficient with respect to angle of attack	None
C_P	Center of pressure of the missile**	ft
D	Diameter of the missile	ft
D	Total drag on the missile	lb_f
d	Derivative of a variable	None
d	Reference length	ft
e_E	Eccentricity of the Earth	None
$\vec{F}$	Total force vector on the missile	lb_f
f	Flattening of the Earth	None

* All azimuth angles are measured clockwise from true north.
** Measured from the tail.

		Dimensions
G	Missile center of gravity*	ft
g_E	Acceleration of gravity constant	$32.174 \frac{lb_m - ft}{lb_f - sec^2}$
G_i	Initial center of gravity of the missile*	ft
$\vec{G}_M$	Overturning moment of the missile	ft-lbs
GM	Gravitational constant of the Earth	ft^3/sec^2
G_P	Center of gravity of the propellant*	ft
G_x, G_y, G_z	Gravitational attraction components	ft/sec^2
$\vec{H}$	Angular momentum vector	lb-ft-sec
H_G	Geodetic altitude	ft
i, j, k	Unit vectors along X, Y, Z	None
J	Earth oblateness constant	None
K	Earth oblateness constant = f(J)	ft^2
K_{AP}	Propellant axial radius of gyration	ft
K_{BP}	Propellant transverse radius of gyration	ft
M	Mass of the missile	slugs
M_i	Initial mass of the missile	slugs
M_P	Mass of the propellant	slugs
M. N.	Mach number of the missile	None
N	Spin rate of the vehicle	rad/sec
N, E	North and east directions at the launch site	None
P_a	Pressure of the atmosphere	lb_f/in^2
P_{aT}	Pressure of the atmosphere at which the thrust is measured	lb_f/in^2
P_E	Constant in the R_E equation	None
$\vec{R}$	Position vector of the missile	ft

* Measured from the tail.

		Dimensions
R_E	Radius of the Earth	ft
S	Reference area of the missile	ft^2
$s \triangleq$	$C\ (1 - f^2)$ (Ref 6)	ft
$\vec{T}$	Thrust vector of the missile	lb_f
t	Time - independent variable	sec
T_{AS}	Reference temperature of the atmosphere	^oK
T_A	Temperature of the atmosphere	^oK
T_{Tt}	Thrust known for input data	lb_f
T_{TV}	Vacuum thrust of the missile	lb_f
V	Velocity of the missile	ft/sec
V_{SD}	Speed of sound	ft/sec
V_{SDS}	Reference speed of sound	ft/sec
V_x, V_y, V_z	Velocity components along X, Y, Z	ft/sec
V_{wx}, V_{wy}, V_{wz}	Wind velocity components along X, Y, Z	ft/sec
V_1, V_2	Velocity components along axis 1 & 2	ft/sec
X, Y, Z	Geocentric coordinate system	ft
X_L, Y_L, Z_L	Launch coordinate system	ft
X_1, X_2, X_3	Missile nonrotating coordinate system	None
X_1, X_2', X_3'	Missile rotating coordinate system	None

GREEK LETTERS

α	Angle between east and X_L axis	deg
δ_T	Thrust misalignment angle	rad
θ	Euler angle as shown in figure 2	deg
θ_C	Geocentric latitude	deg

		Dimensions
θ_G	Geodetic latitude	deg
ρ	Density of the atmosphere	$slugs/ft^3$
Σ	Summation sign	None
φ	Euler angle as shown in figure 2	deg
φ_A	Angle at which the thrust misalignment acts in the X_2, X_3 plane	rad
φ_T	Angle at which the thrust misalignment acts in the X_2', X_3' plane	rad
$\vec{\Omega}$	Angular velocity of missile in the X_1, X_2, X_3 coordinate system	rad/sec
$\vec{\omega}$	Angular velocity of the missile in the X_1', X_2', X_3' coordinate system	rad/sec
$\vec{\omega}_E$	$= \omega_E \vec{k}$ = Angular velocity of the Earth	rad/sec

SUBSCRIPT	REFERS TO
C	Geocentric
E	The Earth
G	Geodetic
i	Initial
L	Launch system
P	Propellant
S	Reference conditions
T	Thrust
V	Vacuum
X	X_L - Axis
x, y, z -	X, Y, Z Coordinate system
W	Wind
1, 2, 3	X_1, X_2, X_3 Coordinate system
α	Angle of attack

An arrow over a variable indicates that the variable is a vector.

A dot over a variable indicates the time derivative of that variable.

a. Linear equations of motion.

(1) Coordinate systems.

(a) Earth-centered coordinate system (X, Y, Z).

A right-handed, Earth-centered, Cartesian coordinate system is used. The X, Y, and Z axes, shown in figure 1 are oriented as follows:

The X axis is the node of the equatorial plane and the plane containing the Z axis and Greenwich Meridian.

The Y axis also lies in the equatorial plane and is at right angles to the X and Z axes.

The Z axis lies along the spin axis of the Earth and is at right angles to the X and Y axes.

(b) Launch coordinate system (X_L, Y_L, Z_L).

A right-hand coordinate system with the X_L, Y_L plane tangent to an oblate Earth at the launch point is used. The X_L, Y_L, Z_L axes are oriented as follows:

The X_L axis is in the direction of the initial launch azimuth.

The Y_L axis is counterclockwise 90° from the X_L axis.

The Z_L axis is positive along the local geodetic vertical.

(2) Development.

(a) Newton second law for a rotating coordinate system.

The fundamental equation of motion for a particle moving in a rotating coordinate system such as the Earth is

$$\frac{d^2\vec{R}}{dt^2} = \frac{\Sigma\vec{F}}{M} + 2\left(\frac{d\vec{R}}{dt} \times \vec{\omega}_E\right) - \vec{\omega}_E \times (\vec{\omega}_E \times \vec{R}) \qquad (1)$$

(Reference 2)

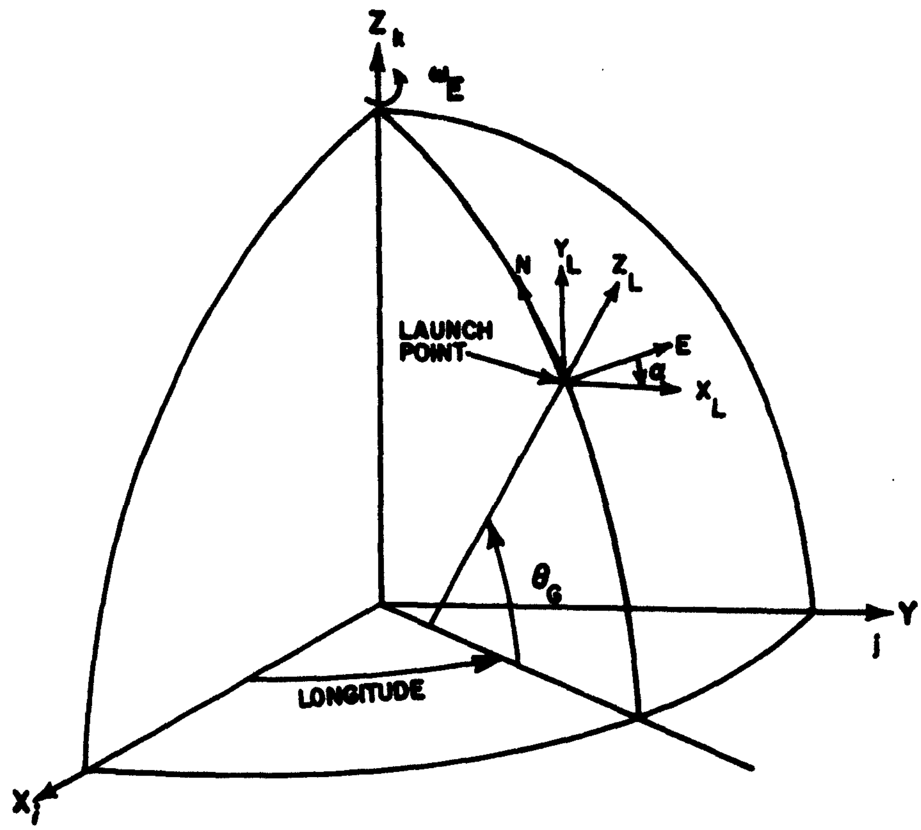


Figure 1. Coordinate systems for trajectory equations.

where $\vec{R}$ is the vector distance from the origin of the rotating X-Y-Z coordinate system to the particle, ($\vec{R} = \vec{i}X + \vec{j}Y + \vec{k}Z$). $\frac{d\vec{R}}{dt}$ and $\frac{d^2\vec{R}}{dt^2}$ are the velocity and acceleration, respectively, of the particle measured with respect to the rotating axes,

$$\frac{d\vec{R}}{dt} = \vec{i}\dot{X} + \vec{j}\dot{Y} + \vec{k}\dot{Z} \quad \text{and} \quad \frac{d^2\vec{R}}{dt^2} = \vec{i}\ddot{X} + \vec{j}\ddot{Y} + \vec{k}\ddot{Z}. \qquad (2)$$

$\vec{\omega}_E$ is the angular velocity of the Earth (axis system) and is along the Z axis, ($\vec{\omega} = \vec{k}\omega_E$). $\Sigma\vec{F}$ is the vector sum of the forces acting on the particle: thrust, drag, and gravity, ($\Sigma\vec{F} = \vec{i}\Sigma F_x + \vec{j}\Sigma F_y + \vec{k}\Sigma F_z$). M is the total instantaneous mass of the particle. $2\left(\frac{d\vec{R}}{dt} \times \vec{\omega}_E\right)$ is the coriolis pseudo-acceleration of the axis system due to the rotation of the Earth; $-\vec{\omega}_E \times (\vec{\omega}_E \times \vec{R})$ is the centrifugal pseudo-acceleration of the axis system due to the rotation of the Earth.

$$\frac{d\vec{R}}{dt} \times \vec{\omega}_E = \omega_E \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ \frac{dX}{dt} & \frac{dY}{dt} & \frac{dZ}{dt} \\ 0 & 0 & 1 \end{vmatrix}$$

$$2\left(\frac{d\vec{R}}{dt} \times \vec{\omega}_E\right) = 2\omega_E \left(\vec{i}\,\frac{dY}{dt} - \vec{j}\,\frac{dX}{dt}\right)$$

$$(\omega_E \times \vec{R}) = \omega_E \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ 0 & 0 & 1 \\ X & Y & Z \end{vmatrix} = \omega_E(-\vec{i}Y + \vec{j}X)$$

$$\vec{\omega}_E \times (\vec{\omega}_E \times \vec{R}) = \omega_E^2 \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ 0 & 0 & 1 \\ -Y & X & 0 \end{vmatrix} = \omega_E^2(-\vec{i}X - \vec{j}Y)$$

(3)

From equations 2 and 3 the components of equation 1 can be expressed:

$$\frac{d^2X}{dt^2} = \frac{\Sigma Fx}{M} + 2\omega_E \frac{dY}{dt} + \omega_E{}^2 X$$

$$\frac{d^2Y}{dt^2} = \frac{\Sigma Fy}{M} - 2\vec{\omega}_E \frac{dX}{dt} + \vec{\omega}_E{}^2 Y$$

$$\frac{d^2Z}{dt^2} = \frac{\Sigma Fz}{M} \qquad (4)$$

The program assumes that there are three forces acting on a rocket vehicle: thrust, drag, and gravity. Thrust is assumed to act along the longitudinal axis of the rocket vehicle. Drag is assumed to act opposite the velocity vector. Gravity is assumed to be directed toward the center of mass of an oblate Earth.

(b) Gravitational attraction equations.

The equations of gravitational attraction are obtained from reference 8, and converted to a Cartesian coordinate system. They are as follows:

$$G_x = -GM \frac{X}{R^3}\left[1 + \frac{3K}{R^2} - 15 \frac{KZ^2}{R^4}\right]$$

$$G_y = -GM \frac{Y}{R^3}\left[1 + \frac{3K}{R^2} - 15 \frac{KZ^2}{R^4}\right]$$

$$G_z = -GM \frac{Z}{R^3}\left[1 + \frac{9K}{R^3} - 15 \frac{KZ^2}{R^4}\right] \qquad (5)$$

Finally, the equations for total acceleration components along X, Y, and Z axes can be written as follows:

$$\ddot{X} = \frac{Tx}{M} - \frac{D}{M}\frac{\dot{X}}{\dot{R}} - G_x + 2\omega_E \dot{Y} + \omega_E^2 X$$

$$\ddot{Y} = \frac{Ty}{M} - \frac{D}{M}\frac{\dot{Y}}{\dot{R}} - G_y - 2\omega_E \dot{X} + \omega_E^2 Y$$

$$\ddot{Z} = \frac{Tz}{M} - \frac{D}{M}\frac{\dot{Z}}{\dot{R}} - G_z \qquad (6)$$

where

$$\dot{R} = (\dot{X}^2 + \dot{Y}^2 + \dot{Z}^2)^{1/2}$$

$$\ddot{R} = (\ddot{X}^2 + \ddot{Y}^2 + \ddot{Z}^2)^{1/2} \qquad (7)$$

(c) Thrust and mass.

The mass M of the rocket vehicle and the thrust T_{Tt} at air pressure P_{at} are assumed to be known functions of time. In general these functions will be nonlinear and discontinuous. The thrust T corresponding to air pressure P_a may be computed from the equation

$$T = T_{Tt} + (P_{at} - P_a)A_e \qquad (8)$$

where A_e is the nozzle exit area. (A_e is measured in square inches; therefore, P_{at} and P_a are absolute pressures in lb/in^2.)

In this trajectory-computation program, the thrust function is evaluated by table look-up and interpolation. Since this function is discontinuous, a set of tables is needed for each stage of the rocket vehicle. The tables in the working storage are changed each time a stage is dropped.

The thrust function T_{Tt} vs. t is given, but the mass function M vs. t is computed by

$$M = M_i - M_{Pi} \frac{\int_{t_i}^{t} T_{T_v}\, dt}{\int_{t_i}^{t_b} T_{T_v}\, dt} \tag{9}$$

where $$T_{Tv} = T_{Tt} + P_{at} A_e \tag{10}$$

Subscript i indicates values at ignition, b denotes values at burnout, and M_P is the mass of the propellant. For clarity, subscript n denoting the configuration number has been omitted from the subscripted symbols.

(d) Aerodynamic drag.

The aerodynamic drag may be computed from the equation

$$D = \tfrac{1}{2} \rho SV^2 C_D (M.N.) \tag{11}$$

where

ρ = Density of the air about the rocket vehicle, slugs/ ft^3

$C_D (M.N.)$ = Drag coefficient, assumed to be a function of Mach number only in this report (dimensionless)

S = Cross-section area of the rocket vehicle, ft^2

M.N. = Mach number = V/V_{SD}

V_{SD} = Velocity of sound in the air about the rocket vehicle, ft/sec.

It should be noted that each stage of the rocket vehicle could have a different diameter. The diameter of the largest stage in the assembly that has not dropped off will be used.

Two drag coefficients for each configuration of the rocket vehicle are needed. It is assumed that the drag of a configuration during powered flight is different from (less than) the drag before ignition or after burnout by the amount of the base drag. The drag coefficients for the powered and coasting conditions will be denoted by C_{DB} and C_{DC}, respectively, with numerical subscripts added to denote the configuration or stage number. Thus, up to six different drag coefficients will be needed for a three-stage rocket vehicle: C_{DB1}, C_{DB2}, C_{DB3}, C_{DC1}, C_{DC2}, C_{DC3}. C_{DC1} and/or C_{DC2} are not needed, of course, if the first and/or second configuration do not coast.

The neglect of yaw angle in determining the drag coefficient is justified by the fact that the thrust will dominate the motion so that fairly large errors in the aerodynamics will have little effect on the trajectory. Also, the yaw angle (and its effect on the coefficients) will presumably be small.

(e) Atmosphere and wind.

The values of ρ, V_{SD}, and P_a are available from the COESA 1962 model atmosphere[15] or from launch site soundings. Actually V_{SD} is not measured directly; instead, the air temperature T_a is recorded and V_{SD} is computed from the equation

$$V_{SD} = V_{SDs} \left(T_a / T_{as} \right)^{1/2} \tag{12}$$

where V_{SDs} is the standard velocity of sound (ft/sec) corresponding to standard air temperature T_{as}, and the units of T_a and T_{as} should be ^oK.

If a wind is blowing, it will have an important effect on the trajectory of a multistage unguided rocket vehicle. The effect is most pronounced during the first stage, and decreases thereafter. After burnout of the last stage, the effect will be fairly small and can be neglected. The wind may be taken into account by computing the velocity components from the following equations:

$$V_X = \dot{X} - V_{WX}$$

$$V_Y = \dot{Y} - V_{WY}$$

$$V_Z = \dot{Z} - V_{WZ} \tag{13}$$

where V_{WX}, V_{WY}, V_{WZ} are the wind components in the Earth-centered axis system. The total velocity V with respect to the air mass may be computed from

$$V = (V_X^2 + V_Y^2 + V_Z^2)^{1/2} \tag{14}$$

Equations 13 follow from the definition of V_X, V_Y, and V_Z as components of the velocity of the rocket vehicle with respect to the air mass.

It is assumed that the wind velocity is a function of H_G only and is horizontal (that is, $V_{WZ_L} = 0$). If the wind velocity is exactly horizontal, then $V_{WZ_L} \neq 0$ for $R_{XY_L} \neq 0$. Setting $V_{WZ_L} = 0$ may be thought of as a flat-earth approximation, but unless R_{XY_L} is very large, it is doubtful whether V_W can be measured accurately enough for the approximation to be questionable.

The meteorological data taken before a launch should include the wind velocity V_W and the wind azimuth angle A_W as well as ρ, T_a, and P_a vs. altitude. By convention A_W is defined to be the azimuth angle, measured clockwise from the North, from which the wind is blowing; then A_W is also the azimuth angle, measured clockwise from the South, to which the wind is blowing. It will be seen that V_{WX_L} and V_{WY_L} may be computed from the following equations:

$$V_{WX_L} = V_W \cos (A_W - A_{X_L})$$

$$V_{WY_L} = V_W \sin (A_W - A_{X_L}) \tag{15}$$

V_{WX_L} and V_{WY_L} are then rotated into the Earth-centered components V_{WX}, V_{WY} and V_{WZ}.

The presentation of the linear acceleration equations for the burning period has now been completed.

b. Angular equations of motion.

(1) Coordinate systems.

(a) Nonrotating body axis (X_1, X_2, X_3).

A right-hand Cartesian coordinate system which is fixed to, but does not spin with the body. The X_1 axis is along the spin axis. The X_2 axis lies in the vertical plane, while the X_3 axis lies in the horizontal plane. This axis system is related to the reversed ($-Y_L$) launch coordinate system by the two Euler angles θ and φ.

(b) Rotating body axis (X_1, X_2', X_3').

A body fixed coordinate system with the same origin as the X_1, X_2, X_3 system. Axes X_2' and X_3' are along the transverse principal axis of the rocket vehicle.

(2) Development.

(a) Body axis equations.

The angular acceleration equations of the rocket vehicle may be developed from the vector equation of reference 1. The only torque considered is the overturning moment. All other moments including pitch and jet damping are assumed to be small and will be neglected; the rocket vehicle is spun to reduce the effects of any misalignments, asymmetries, or unbalances, but such a spin is not large enough to develop an appreciable Magnus moment. The vector equation of angular motion is then as follows:

$$\frac{d\vec{H}}{dt} = \frac{\partial\vec{H}}{\partial t} + \vec{\Omega} \times \vec{H} = \vec{G}_M \qquad (16)$$

where

$\vec{H}$ = Angular momentum vector, lb-ft-sec

$\vec{G}_M$ = Overturning moment (vector), lb-ft

$\vec{\Omega}$ = Angular velocity of the X_1, X_2, X_3 system, rad/sec

The angular velocity of the missile is denoted by $\vec{\omega}$; the components of $\vec{\omega}$ in the X_1, X_2, X_3 system are $\Omega_1 + N, \Omega_2, \Omega_3$, where N is the axial spin of the missile and Ω_1, Ω_2, Ω_3 are the components of $\vec{\Omega}$. Then equations for $\vec{\Omega}$ and $\vec{H}$ are as follows: (reference 1.)

$$\vec{\Omega} = \vec{I}_1 \Omega_1 + \vec{I}_2 \Omega_2 + \vec{I}_3 \Omega_3$$

$$\vec{H} = \vec{I}_1 \ A(\Omega_1 + N) + \vec{I}_2 \ B\Omega_2 + \vec{I}_3 \ B\Omega_3 \tag{17}$$

where A = moment of inertia of the rocket vehicle about the longitudinal principal axis, slug-ft^2

B = moment of inertia of the rocket vehicle about a transverse axis through the center of mass, slug-ft^2

It is assumed that the mass distribution is symmetric, so the moment of inertia is the same about any transverse axis through the center of mass.

(b) Moments of inertia.

It is assumed that an internal-burning solid propellant is used. Approximate values of A and B during burning are computed by use of the following formulas:

$$A = A_i - k_{AP}^2 \ (M_i - M) \tag{18}$$

$$B = B_i - M_i \ (G - G_i)(G_i - G_P) - (M_i - M)k_{BP}^2 \tag{19}$$

where

$$G = G_i + (G_i - G_P)(M_i - M)/M \tag{20}$$

As before, subscript i indicates values at time of ignition and P denotes a property of the propellant; k_{AP} is the radius of gyration of the propellant grain about its longitudinal axis (ft), K_{BP} is the radius of gyration of the propellant grain about a transverse axis through its center of mass (ft), and G is the distance from the base to the center of mass of the rocket vehicle (ft). The quantities G_P, k_{AP} and k_{BP} are assumed to be constant for an internal burning grain; A_i, B_i, G_i, and M_i are, of course, constant, so M is the only variable.

Equations 19 and 20 cannot be used for an end-burning grain unless formulas are added for k_{BP} and G_P. None of these equations apply for liquid propellant rockets.

(c) Aerodynamic moment.

The expression for $\vec{G}_M$ is as follows:

$$\vec{G}_M = \tfrac{1}{2}\rho\, d\, S\, V C_{M\alpha} \left(\vec{i}_2 V_3 - \vec{i}_3 V_2\right) \qquad (21)$$

For the present application $C_{M\alpha}$ is the static stability derivative (dimensionless) and is a negative number which can be computed by use of the equation:

$$C_{M\alpha} = \frac{C_{N\alpha}(C_p - G)}{d} \qquad (22)$$

where

$C_{N\alpha}$ = Normal force coefficient (dimensionless)

C_p = Distance from the base of the rocket to the normal force center of pressure, feet

In this report $C_{M\alpha}$, $C_{N\alpha}$, and C_p, are assumed to be functions of Mach number only. In the trajectory-computation program these functions and C_D are evaluated by table look-up and interpolation.

(d) Euler angle relation.

If equations 17 and 21 are substituted into equation 16, the X_2 and X_3 components of the resulting vector equation will be as follows:

$$B\dot{\Omega}_2 + \dot{B}\Omega_2 + (A - B)\,\Omega_1\Omega_3 + AN\Omega_3 = \tfrac{1}{2}\rho V d S\, C_{M\alpha} V_3$$

$$B\dot{\Omega}_3 + \dot{B}\Omega_3 + (B - A)\,\Omega_1\Omega_2 - AN\Omega_2 = -\tfrac{1}{2}\rho V d S\, C_{M\alpha} V_2 \qquad (23)$$

It is desirable to replace Ω_1, Ω_2, Ω_3, $\dot{\Omega}_2$, $\dot{\Omega}_3$ in these equations by functions of φ and θ. This may be done by use of the following relations:

$$\Omega_1 = -\dot{\varphi}\sin\theta$$

$$\Omega_2 = -\dot{\theta}$$
$$\Omega_3 = -\dot{\varphi} \cos \theta \qquad (24)$$

These equations neglect the angular velocity of the Earth (ω_E); they are written by inspection of figure 2. It will be seen that

$$\dot{\Omega}_2 = -\ddot{\theta}$$
$$\dot{\Omega}_3 = -\ddot{\varphi} \cos \theta + \dot{\varphi}\,\dot{\theta} \sin \theta \qquad (25)$$

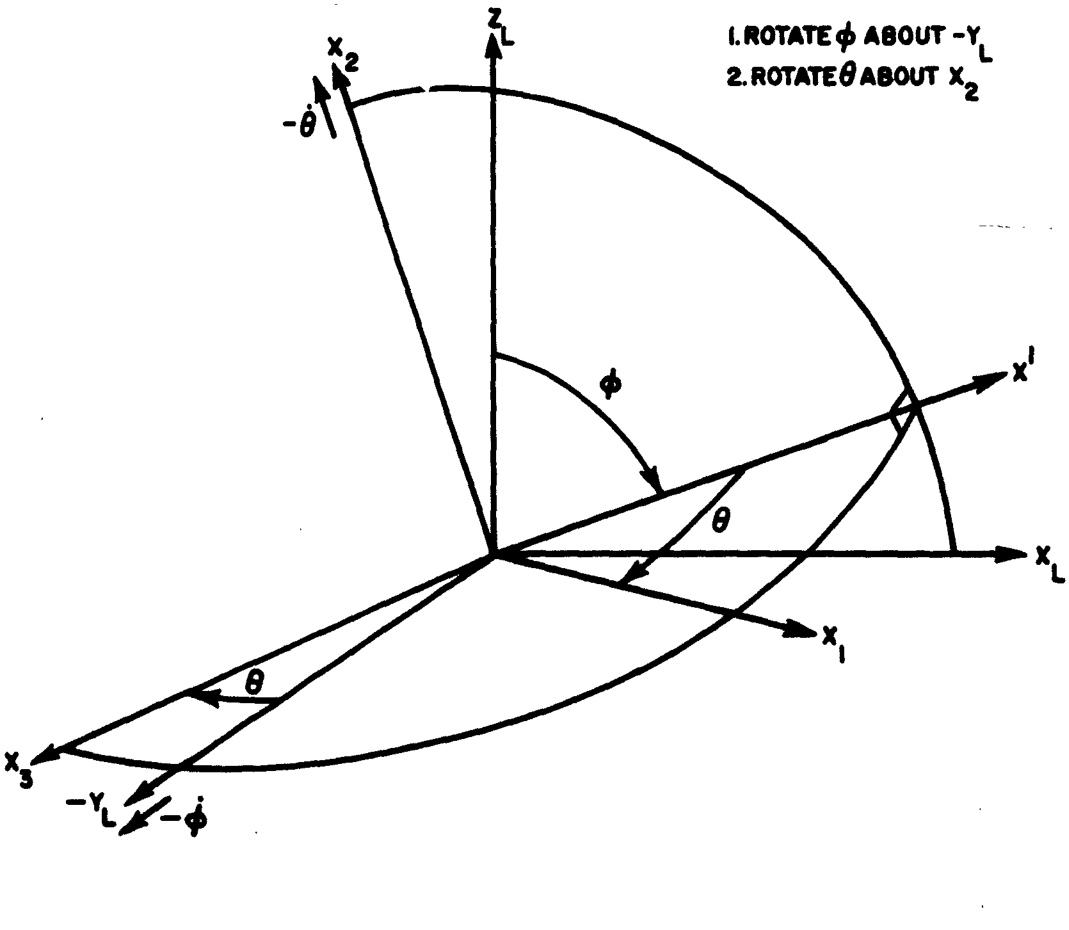


Figure 2. Coordinate system for angular equations.

Substitution of equations 24 and 25 into equations 23 yields the required angular acceleration equations as follows:

$$-B\ddot{\theta} - \dot{B}\dot{\theta} + (A - B)\,\dot{\varphi}^2 \sin\theta\cos\theta - AN\,\dot{\varphi}\cos\theta$$

$$= \tfrac{1}{2}\rho\, VSd\, C_{M\alpha}\, V_3$$

$$-B\ddot{\varphi}\cos\theta - \dot{B}\dot{\varphi}\cos\theta + (2B - A)\,\dot{\varphi}\dot{\theta}\sin\theta + AN\,\dot{\theta}$$

$$= -\tfrac{1}{2}\rho\, VSd\, C_{M\alpha}\, V_2 \qquad (26)$$

These equations are dynamically exact, except for the neglect of ω_E which is negligibly small compared to $\dot{\phi}$, $\dot{\theta}$, and N.

Equations 26 may be put in a form more suitable for computation by dividing through by B. The equations become

$$\ddot{\theta} + (\dot{B}/B)\,\dot{\theta} + (1 - (A/B))\,\dot{\varphi}^2 \sin\theta\cos\theta + (A/B)N\,\dot{\phi}\cos\theta$$

$$= (-\tfrac{1}{2}\rho\, VdS\, C_{M\alpha})\,\frac{VV_3}{B}$$

$$\ddot{\varphi}\cos\theta + (\dot{B}/B)\,\dot{\varphi}\cos\theta - (2 - (A/B))\,\dot{\varphi}\dot{\theta}\sin\theta - (A/B)N\dot{\theta}$$

$$= (\tfrac{1}{2}\rho\, dS\, C_{M\alpha})\,\frac{VV_2}{B} \qquad (27)$$

(e) $\dot{B}$ Terms.

For an internal burning solid-propellant rocket, the following formulas are used to compute the $\dot{B}$ terms:

$$\dot{B} = \dot{M}\left[(M_i^2/M^2)(G_i - G_P)^2 + k_{BP}^2\right]$$

where

$$\dot{M} = -T_{Tv}\left\{M_{Pi}\Big/\int_{ti}^{tb} T_{Tv}\,dt\right\} \qquad (28)$$

These formulas are easily derived from equations 9 and 19.

(f) Body-cross wind components.

A coordinate transformation is needed to compute V_2 and V_3 from V_{X_L}, V_{Y_L}, V_{Z_L}. The following equations can be written from inspection of figure 2:

$$V_2 = -V_{X_L} \cos\varphi + V_{Z_L} \sin\varphi$$

$$V_3 = -V_{X_L} \sin\varphi \sin\theta - V_{Y_L} \cos\theta - V_{Z_L} \cos\varphi \sin\theta \tag{29}$$

This completes the derivation of the equations of motion during burning.

c. Summary of equations of motion.

If the rocket is assumed to have thrust misalignments, additional terms are added to the equations to account for the new forces and moments created. For convenience, the equations are restated here with the new terms added.

$$\ddot{X} = \frac{1}{M}\left[T_x - \frac{D\dot{X}}{\dot{R}}\right] + G_x + 2\omega_E \dot{Y} + \omega_E^2 X$$

$$\ddot{Y} = \frac{1}{M}\left[T_y - \frac{D\dot{Y}}{\dot{R}}\right] + G_y - 2\omega_E \dot{X} + \omega_E^2 Y$$

$$\ddot{Z} = \frac{1}{M}\left[T_z - \frac{D\dot{Z}}{\dot{R}}\right] + G_z$$

$$\ddot{\varphi} = -\frac{\dot{B}}{B}\dot{\varphi} + \left\{\left[\left(2 - \frac{A}{B}\right)\dot{\varphi}\sin\theta + \frac{A}{B}N\right]\dot{\theta} + \frac{\pi}{8}\rho C_{N\alpha}(C_P - G)\frac{D^2 V V_2}{B} + \frac{T G \delta_T \cos\varphi_A}{B}\right\}\frac{1}{\cos\theta}$$

$$\ddot{\theta} = -\frac{\dot{B}}{B}\dot{\theta} - \left[\left(1 - \frac{A}{B}\right)\dot{\varphi}\sin\theta + \frac{A}{B}N\right]\dot{\varphi}\cos\theta - \frac{\pi}{8}\rho C_{N\alpha}(C_P - G)\frac{D^2 V V_3}{B} - \frac{T G \delta_T \sin\varphi_A}{B} \tag{30}$$

where

δ_T = thrust misalignment angle, radians

φ_T = orientation angle of jet misalignment force; measured in the $X_2^{'}$ - $X_3^{'}$ plane from the $X_2^{'}$ axis and positive in the sense of a counter clockwise rotation as seen from the positive $X_1^{'}$ axis, radians.

$\varphi_A = \theta_T + \int_{ti}^{t} N dt$ measured in the X_2 - X_3 plane from the X_2 axis and positive in the sense of a counterclockwise rotation as seen from the positive X_1 axis, radians.

These equations require a table of roll rate ($N = \dot{\phi}$) vs. time.

The thrust vector is given in the launch coordinate system and rotated to the Earth-centered system by the use of a matrix rotation. The thrust vector in the launch coordinate system is

$$T_{xL} = T \left[\cos\theta \sin\varphi - \delta_T(\cos\varphi_A \cos\varphi + \sin\varphi_A \sin\varphi \sin\theta)\right]$$

$$T_{yL} = -T \left[\sin\theta + \delta_T \sin\varphi_A \cos\theta\right]$$

$$T_{zL} = T \left[\cos\theta \cos\varphi + \delta_T (\cos\varphi_A \sin\varphi - \sin\varphi_A \cos\varphi \sin\theta)\right] \tag{31}$$

d. <u>Geocentric relationships</u>.

(1) Shape of the Earth.

Figure 3 shows a meridian section of the earth where a_E is the semi-major (equatorial) axis, θ_c is the geocentric latitude, and R_E is a radius vector from center to the surface of the earth. R_E is a function of the geocentric latitude and is given as

$$R_E = \frac{a_E}{(1 + P_E \, \mathrm{SIN}^2 \theta_c)^{1/2}}$$

where $$P_E = \frac{e_E^2}{1 - e_E^2}$$ (32)

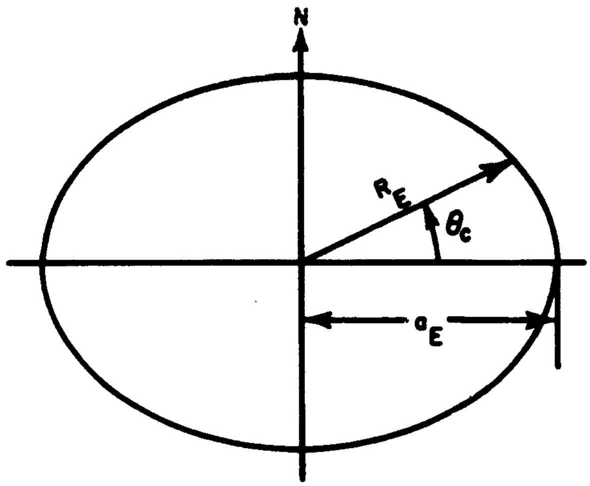


Figure 3

(2) **Geodetic sublatitude and altitude.**

Figure 4 shows the geometric relation between the geodetic and geocentric latitude and altitude.

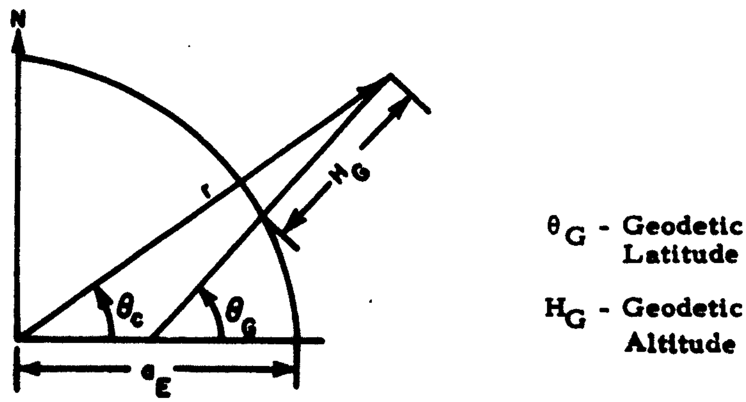


Figure 4

To convert geodetic latitude and altitude to geocentric latitude and radius: (reference 6)

$$\tan \theta_c = \left[\frac{S + H_G}{C + H_G}\right] \tan \theta_G \tag{33}$$

where

$$C \triangleq \frac{a_E}{(1 - e_E^2 \sin^2 \theta_G)^{1/2}} \qquad S \triangleq C(1 - e_E^2) \tag{34}$$

$$R = \left[(C + H_G)^2 \cos^2 \theta_G + (S + H_G)^2 \sin^2 \theta_G\right]^{1/2} \tag{35}$$

to convert geocentric latitude and radius to geodetic latitude and altitude: (reference 7)

$$\theta_G = \theta_c + \sin^{-1}\left\{\frac{a_E}{R}\left[f \sin 2\theta_c + f^2 \sin 4\theta_c\left(\frac{a_E}{R} - \frac{1}{4}\right)\right]\right\}$$

$$H_G = R - a_E\left[1 - f \sin^2 \theta_c - \frac{f^2}{2} \sin^2 2\theta_c\left(\frac{a_E}{R} - \frac{1}{4}\right)\right] \tag{36}$$

The following geocentric constants were used in the program.

Adopted Geocentric Constants (1961) (reference 8)

a_E = 20,925,647.12 ft

GM = 1.4076427×10^{16} ft^3/sec^2

= 2.316686×10^{12} NM $\left(\frac{ft}{sec}\right)^2$

g_E = 32.174 ft/sec^2

1/f = 298.30 ± 0.05

$$e_E = 0.0818133302$$

$$J = (1623.42 \pm 0.5) \times 10^{-6}$$

$$P_E = 0.00673852$$

$$\omega_E = 7.292115 \times 10^{-5} \text{ rad/sec}$$

3. DATA INPUT AND PROGRAM USAGE.

INPUT DATA

All input data are read into SPURT at one time and in one "read" block. This includes data for the subroutines (two-body and impact). Most data, except for special cases and control integers, are read in a FORTRAN 7F10.0 format. This format is for seven data words (of ten digits each) per card. The decimal point is always punched in the field. On Repeat Blocks, as many words will be read as specified by a previously read integer.

FLOW CHART OF INPUT DATA:

Parameter	Dimension	Column	Mode	Remarks
No. of stages	none	1 - 2	I	1 Card
Payload weight	pounds	3 - 15	V	

↓

Name	none	1 - 80	Hollerith	1 Card

↓

Initial latitude	degrees (+ N)	1 - 10	V	1 Card
Initial longitude	degrees (+ E)	11 - 20	V	
Initial altitude	feet	21 - 30	V	
Initial azimuth	degrees CW from N	31 - 40	V	
Initial X	feet	41 - 50	V	
Initial Y	feet	51 - 60	V	
Initial Z	feet	61 - 70	V	
Initial $\dot{X}$	feet/sec	1 - 10	V	1 Card
Initial $\dot{Y}$	feet/sec	11 - 20	V	
Initial $\dot{Z}$	feet/sec	21 - 30	V	
Initial φ	degrees	31 - 40	V	
Initial θ	degrees	41 - 50	V	
Initial $\dot{\varphi}$	deg/sec	51 - 60	V	
Initial $\dot{\theta}$	deg/sec	61 - 70	V	
Start time = 0.	seconds	1 - 10	V	1 Card
Time of last stage B.O.	seconds	11 - 20	V	

↓

(a)

(a)

Master control

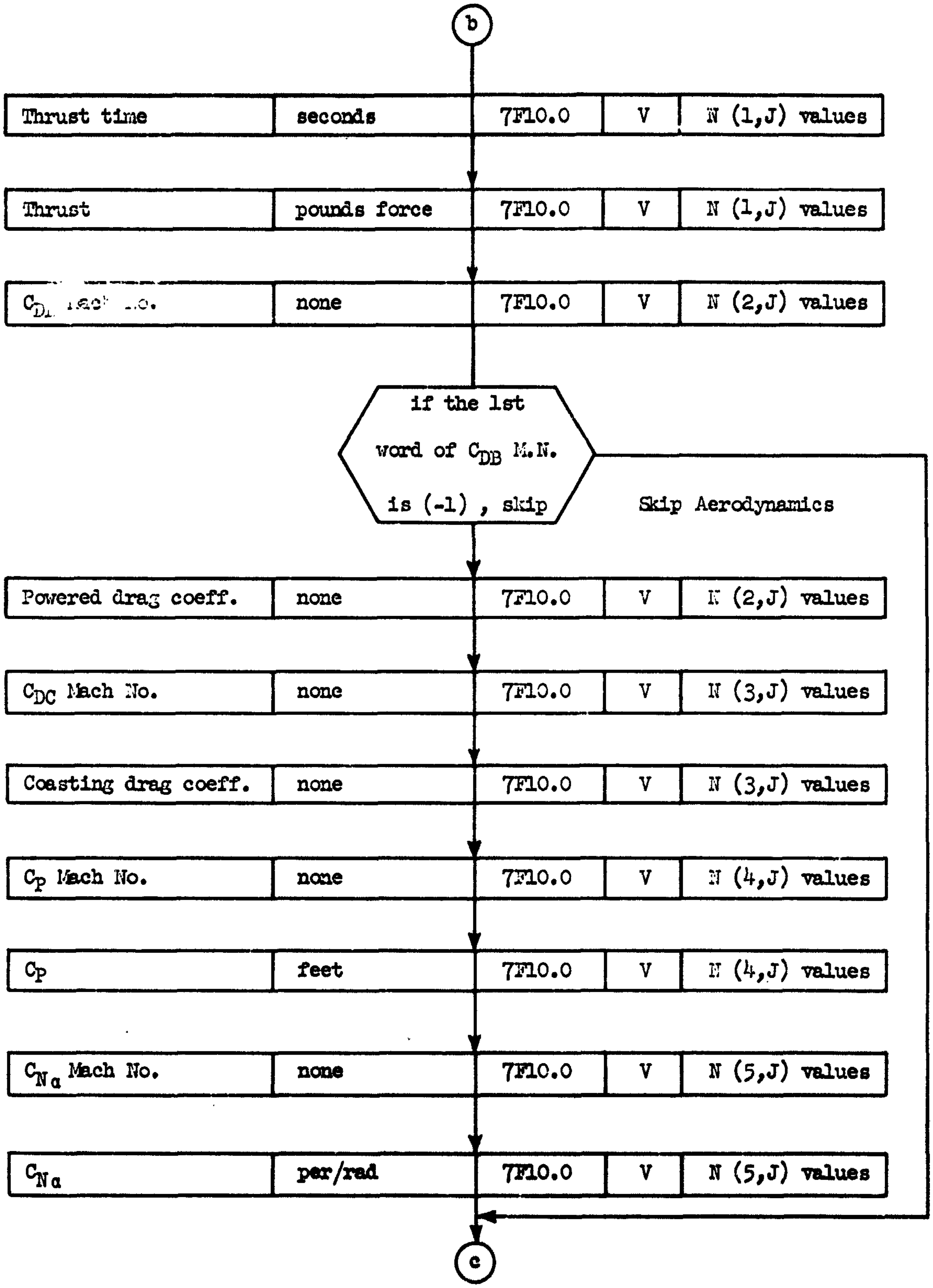
Control 1	none	1 - 2	I	metro & wind
Control 2	none	3 - 4	I	print input data
Control 3	none	5 - 6	I	write BATT tape
Control 4	none	7 - 8	I	two body sub.
Control 5	none	9 - 10	I	lines/page
Control 6	none	11 - 12	I	write plot tape
Control 7	none	13 - 14	I	right hand cord.
Control 8	none	15 - 16	I	impact sub.
Control 9	none	17 - 18	I	all this block on one card
Control 10	none	19 - 20	I	
No. of spin values	none	21 - 22	I	
Output tape No.	none	23 - 24	I	
Deg. of interpolation	none	25 - 26	I	

Spin time table	seconds	7F10.0	V	N spin values
Spin rate table	rad./sec	7F10.0	V	N spin values

A → A'

Stage number	none	1 - 2	I	N0
Size of thrust table	none	3 - 4	I	N (1,J)
Size of C_{DB} table	none	5 - 6	I	N (2,J)
Size of C_{DC} table	none	7 - 8	I	N (3,J)
Size of C_P table	none	9 - 10	I	N (4,J)
Size of $C_{N\alpha}$ table	none	11 - 12	I	N (5,J)

(b)


b
Thrust time
seconds
7F10.0
V
N (1,J) values
Thrust
pounds force
7F10.0
V
N (1,J) values
none
7F10.0
V
N (2,J) values
if the 1st
word of C_DB M.N.
is (-1) , skip
Skip Aerodynamics
Powered drag coeff.
none
7F10.0
V
N (2,J) values
C_DC Mach No.
none
7F10.0
V
N (3,J) values
Coasting drag coeff.
none
7F10.0
V
N (3,J) values
C_P Mach No.
none
7F10.0
V
N (4,J) values
C_P
feet
7F10.0
V
N (4,J) values
C_Nα Mach No.
none
7F10.0
V
N (5,J) values
C_Nα
per/rad
7F10.0
V
N (5,J) values
c

c

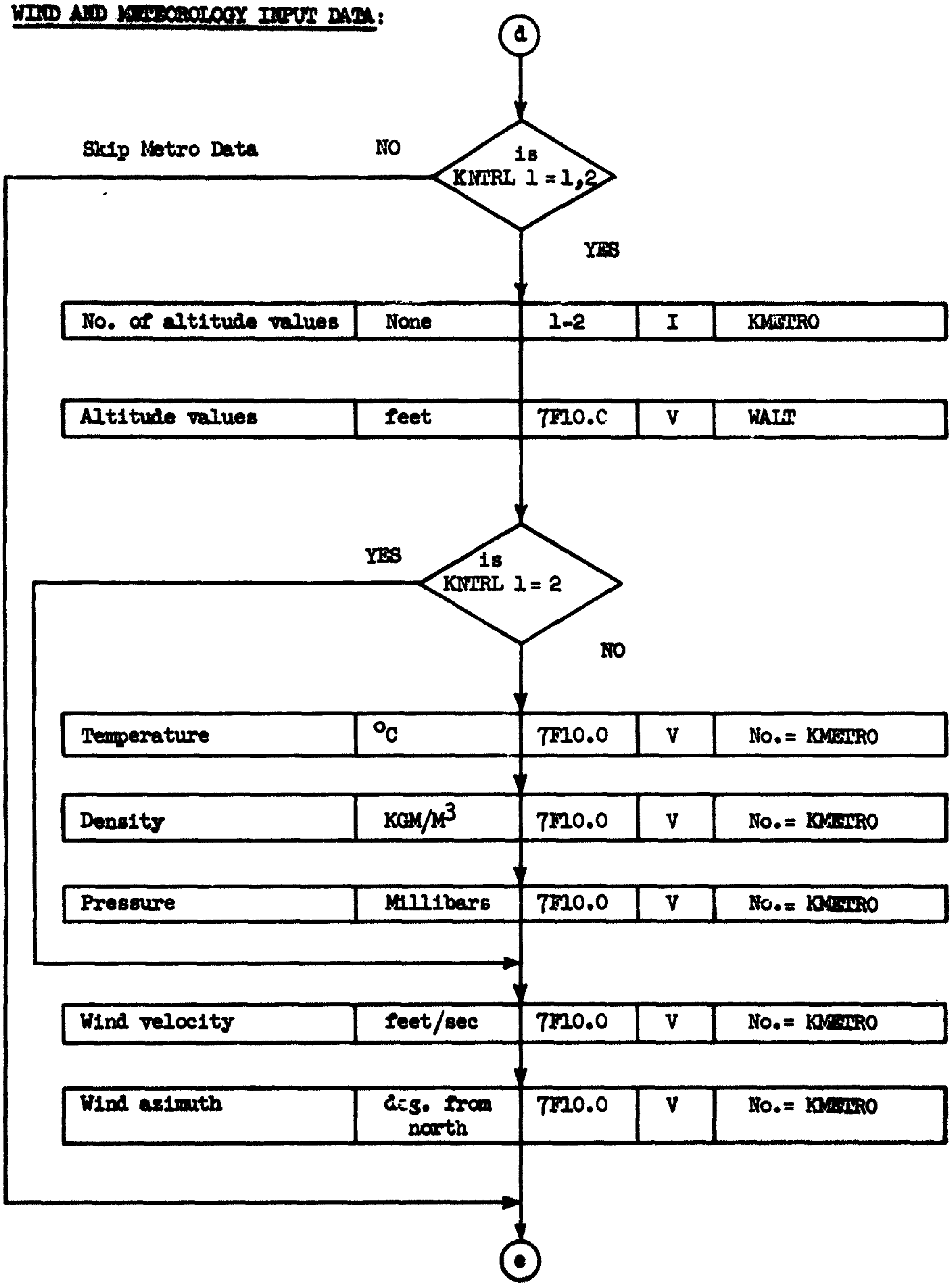
Pressure at thrust meas.	pounds/in^2	1 - 10	V	
Exit area	inch^2	11 - 20	V	
Stage diameter	feet	21 - 30	V	
Missile C.G.	feet	31 - 40	V	1 Card
Stage fuel C.G.	feet	41 - 50	V	
Fuel axial I/M	feet^2	51 - 60	V	
Fuel transverse I/M	feet^2	61 - 70	V	
Missile axial I	slug-ft^2	1 - 10	V	
Missile transverse I	slug-ft^2	11 - 20	V	
Stage weight	pounds	21 - 30	V	1 Card
Stage consumed wt	pounds	31 - 40	V	
Orientation of T.M.A.	radians	41 - 50	V	
Thrust misalign angle	radians	51 - 60	V	
- OMIT -		61 - 70	V	
Ignition time from launch	seconds	1 - 10	V	
Burnout time from launch	seconds	11 - 20	V	1 Card
TMCC from launch	seconds	21 - 31	V	

Repeat from A'
for each stage (NS)

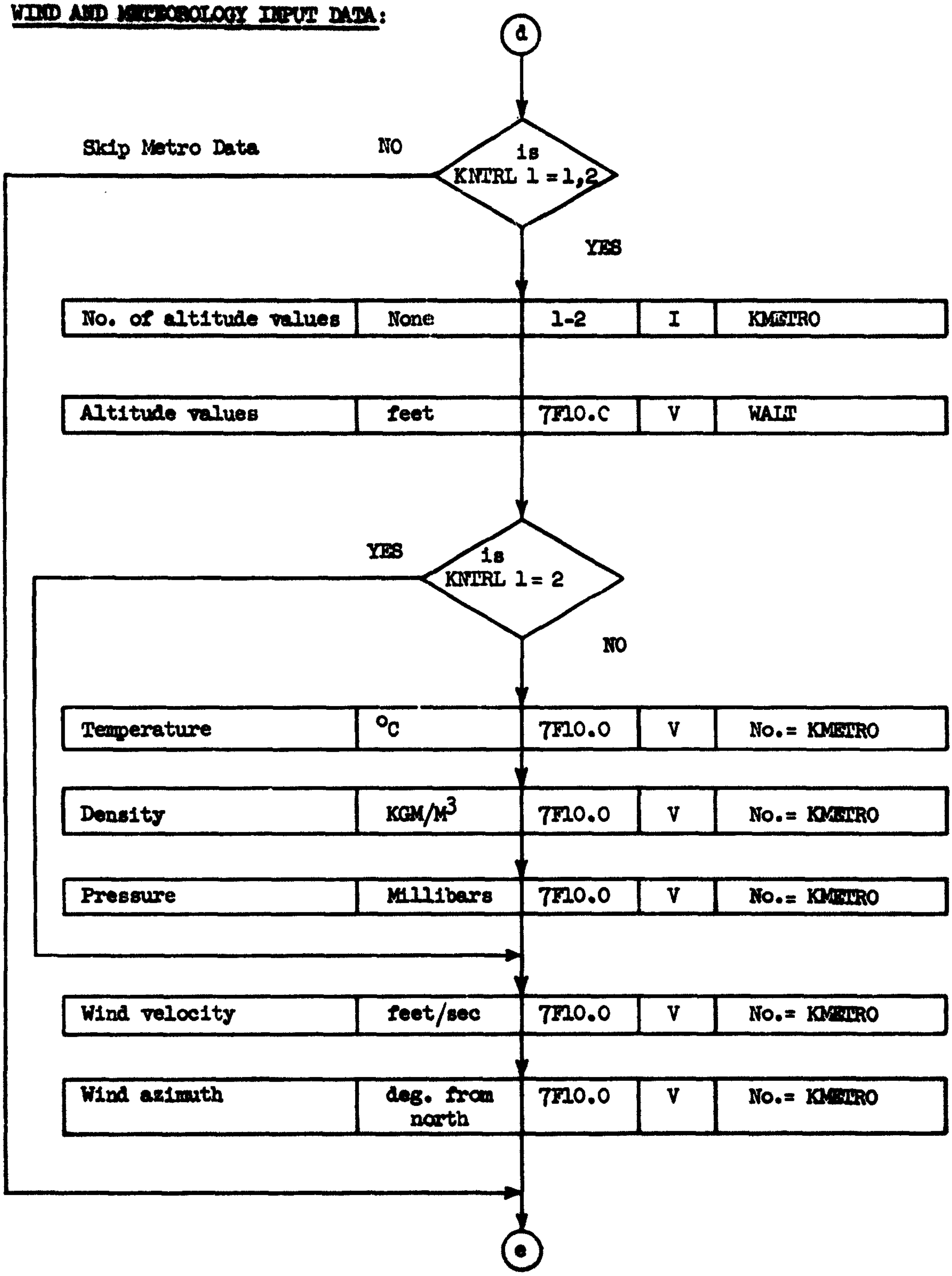
A

d

WIND AND METEOROLOGY INPUT DATA:



WIND AND METEOROLOGY INPUT DATA:



INPUT DATA FOR KEPLERIAN TRAJECTORY AND LOOK ANGLES:

(e)

Is KNTRL 4 = 1,2 — No: SKIP KEPLERIAN; YES:

No. of Time changes	None	1	I	JTB1
No. of ORBITS	None	2-11	V	TB2

Δti	seconds	1-10	V	TB3
END of Δti	seconds	11-20	V	TB4

Repeat JTB1 Times

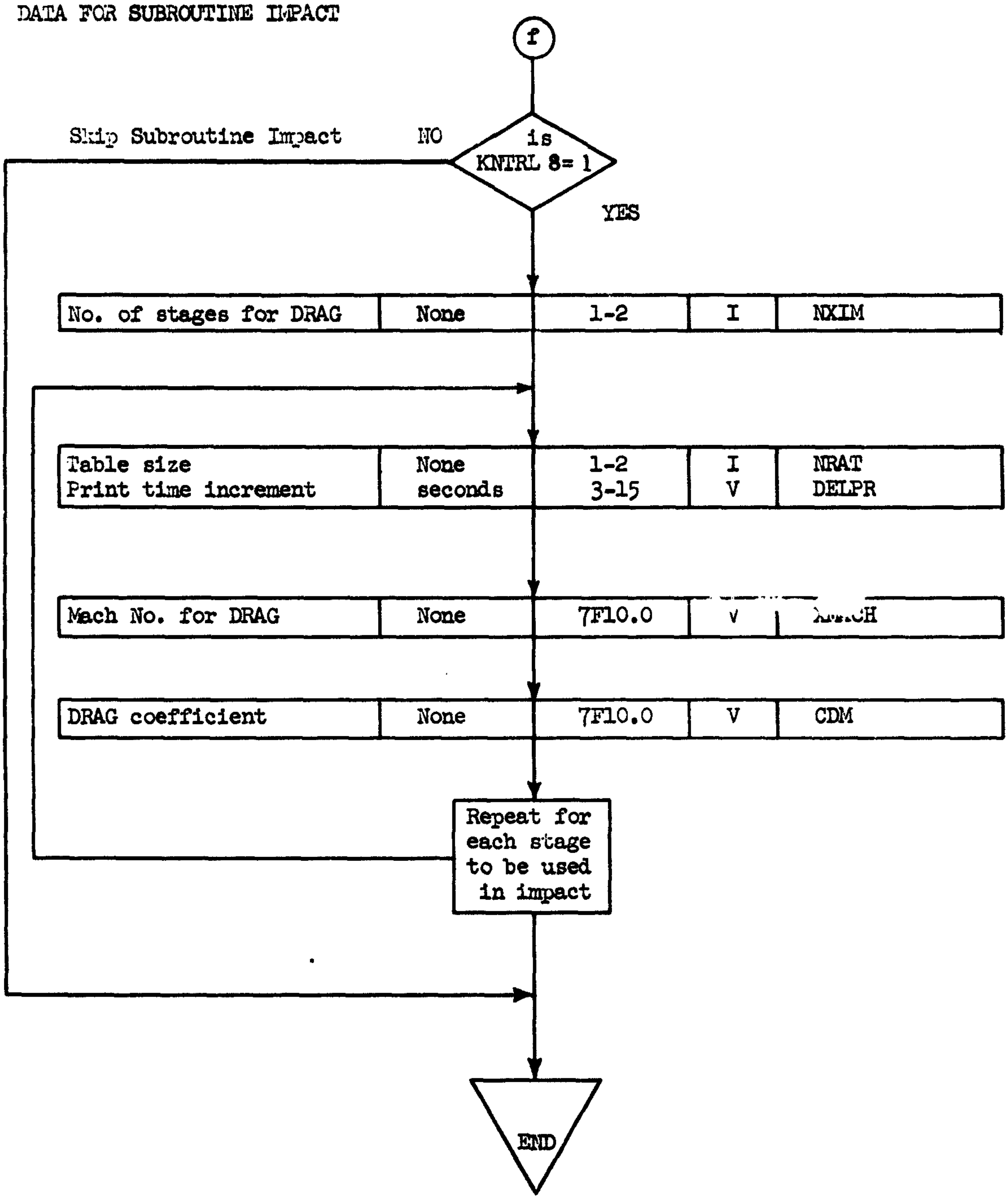
No. of Look Stations	None	1-3	I	JTB5

Station name	None	1-24	Hollerith	TB6
Station latitude	Degrees (+N)	25-34	V	TB7
Station longitude	Degrees (+E)	35-44	V	TB8
Station altitude	feet	45-54	V	TB9

Repeat JTB5 Times

(f)

DATA FOR SUBROUTINE IMPACT



I ~ INTEGER - Fixed point numbers
V ~ VARIABLE - Floating point numbers

MASTER CONTROL NUMBERS

These numbers control various options of the program as follows:

Control 1. When KNTRL(1) = 1*, the program will read in the temperature, pressure, and density and use them to compute the aerodynamic forces and moments. Otherwise, the standard 1962 atmosphere is used.

When KNTRL(1) = 1 or 2, the program will read in wind velocity and wind azimuth vs. altitude. The program will compute the path of the missile due to the winds.

Control 2. When KNTRL(2) = 1, the input data are printed out.

Control 3. When KNTRL(3) = N, the program will write tape number N for input to the BATT program.

Control 4. When KNTRL(4) = 1, the program will use the Keplerian (TWO-BOD) subroutine for coasting flight on the last stage. If KNTRL(4) = 2, it will use it on all stages.

Control 5. When KNTRL(5) = NO, the program will write NO lines of output per page.

Control 6. When KNTRL(6) = N, the program will prepare an input tape (N) for the plot program.

Control 7. When KNTRL(7) = 1, the launch coordinate system printed out is a right-hand one instead of a left-hand one.

Control 8. If KNTRL(8) = 1, the integrating unpowered flight trajectory is computed for all stages except the last one. If KNTRL(8) = 2, the trajectory is computed for all stages.

Control 9. Not used.

Control 10. Not used.

---

* When a KNTRL number is left blank, that option will be ignored.

4. VARIABLES USED IN SPURT.

INPUTS

A	*	Initial axial moment of inertia for each stage (slug-ft^2)
AALT	*	Initial geodetic altitude (ft)
AAZIM	*	Initial launch azimuth (deg)
AAZI2		Initial launch azimuth (rad)
ACODE		Integration code
ACODES		Integration code table
AE	*	Exit area of rocket for each stage (in^2)
ALAT	*	Initial geodetic latitude (deg)
ALA2		Initial geodetic latitude (rad)
ALON	*	Initial longitude (deg)
ALO2		Initial longitude (rad) +
ALT		Geocentric altitude (ft)
AN1		$\frac{\pi}{2}$ + longitude (rad)
AN2		Colatitude (rad)
AXLM		Axial moment of inertia (slug-ft^2)
AXLMB		Axial moment of inertia over transverse moment of inertia
B	*	Initial transverse moment of inertia for each stage (slug-ft^2)
BAZIM		AAZI2 - $\frac{\pi}{2}$ (rad)
BDOT		Rate of change of the transverse moment of inertia, $\dot{B}$ (slug-ft^2/sec)
BDOTB		$\dot{B}$/ B (sec^{-1})
BL		Integration data storage
BLON		$\frac{\pi}{2}$ + ALO2 (rad)
C		$\triangleq REE/\sqrt{1 - e_E^2 \sin^2 \theta_G}$ (ft)
CALAT		Cosine of the initial latitude
CBLOCK	*	Common block for integration
CDM	*	$C_D A/m$ - drag parameter used for empty stages (ft^2/slugs) +

+ Stored in common

INPUTS

CHECK		Flag for integration check
CN	*	Input table of normal force coefficient C_{N_α} (rad^{-1})
CNI		Normal force coefficient used in calculation
CNMACH	*	Input CN Mach table
CNMTAB		Inverted CN Mach table
CNTAB		Inverted CN table
COEF		Coefficient used in aerodynamic moment = C_{N_α} (Cp - CG) $\frac{D^2 V \rho}{B}$
COLAT		Initial co-latitude = $\frac{\pi}{2}$ - ALA2 (rad)
COTHC		Co-geocentric latitude = $\frac{\pi}{2}$ - THC (rad)
CP	*	Initial center of pressure table (feet from tail)
CPHI		Cosine of PHI
CPHT		Cosine of PHT
CPI		Center of pressure used in calculation
CPMACH	*	Initial Mach No. table for center of pressure
CPMTAB		Inverted CP Mach No. table
CPTAB		Inverted CP table (ft)
CTHC		Cosine of initial geocentric latitude = Cos (THC) +
CTHET		Cosine of THETA
D	*	Diameter of each stage (ft)
D1MACH	*	Input Mach No. table for burning drag coefficient
D2MACH	*	Input Mach No. table for coasting drag coefficient
DC		Direction cosines matrix
DELPR	*	Print times used in impact (sec) +
DELTH		Difference between θ_c and θ_g (rad)
DMTAB		Inverted drag Mach table
DRAG1	*	Input table of burning drag coefficient
DRAG2	*	Input table of coasting drag coefficient
DT		Print interval (sec)
DTAB		Inverted drag table

+ Stored in common

INPUTS

DUM	*	Dummy variable
ERR		Integration error
FDRAG		Drag force (lb)
FORCE		Vacuum thrust (lb)
FT		Thrust on vehicle (lb)
FTT		Total impulse (lb - sec)
GI		Center of gravity at any given time (feet from tail)
GK		Earth oblateness term a_E^2 J (ft^2)
GM		Gravitational constant for the Earth (ft^3/sec^2)
GO	*	Initial CG of the remaining missile for a given state (ft)
GOP		Difference between GO and GP (ft)
GP	*	Initial CG of the fuel for a given stage (ft)
GRAVO		Gravity constant (32.174 ft/sec^2)
GRV		Term used to compute gravity components
GRVX		Gravity component along the X axis (ft/sec^2)
GRVY		Gravity component along the Y axis (ft/sec^2)
GRVZ		Gravity component along the Z axis (ft/sec^2)
H		Velocity vector (no wind) (ft/sec)
HALFPI		$\frac{\pi}{2}$
I		Utility index
ICODE		Code used to determine integration method
II		Utility index
III		Utility index
INTN	*	Order of interpolation
IRA		Utility index
IT		Print table count
J		Utility index
JP		Lines per page count for output
JTB1	*	Two Body input (number of time changes) +
JTB5	*	Two Body input (number of look-angle stations) +
K		Utility index
KBATT		Tape number for BATT tape = KNTRL(3) +
KIX		Number of stages for subroutine impact

+ Stored in common

INPUTS

KMETRO	*	Size of Metro tables	
KNTRL(10)	*	Controls	
KPLOT		Tape number of PLOT tape = KNTRL(6)	
L		Stage count	
LSKIP		Skip aerodynamics	
N	*	Table size	
NO	*	Input card check	
N1		Thrust table size for different stages	
N2		Burning drag table size for different stages	
N3		Coasting drag table size for different stages	
N4		CP table size for different stages	
N5		CN table size for different stages	
NAME	*	Page title (up to 80 Hollerith characters)	+
NOE		Number of equations used in integration	
NOT	*	Output tape number	+
NNX	*	Table of NRAT values	+
NRAT	*	Table size for each stage in impact	
NS	*	Number of stages ($NS \leq 10$)	
NSPIN	*	Spin table size ($NSPIN \leq 100$)	
NXIM	*	Number of stages read in impact drag table	
OMEG		Spin rate of Earth = ω_E (rad/sec)	
OMEG2		Earth spin rate squared = ω_E^2 (rad^2/sec^2)	
OUT		Variable used for output	
P1		Pressure times exit area (lb)	
PAT	*	Pressure at which thrust is measured (lb/in^2)	
PHI		Euler angle used in equations (rad)	
PHID		First derivative of PHI (rad/sec)	
PHIDD		Second derivative of PHI (rad/sec^2)	
PHT	*	Orientation angle of thrust misalignment (rad)	
PI		π	
PRES		Atmospheric pressure (lb/ft^2)	
PRTIME		Time at which to print (sec)	

+ Stored in common

INPUTS

PWGT	*	Stage input weight (lb)
PWGTC	*	Stage fuel weight (lb)
PX		Earth centered position vector used in integration (ft)
PXD		First derivative of position vector (ft/sec)
PXDD		Second derivative of position vector (ft/sec^2)
PXL		Position vector in launch coordinate system (ft)
PXLD		Velocity vector in launch coordinate system (ft/sec)
PXLDD		Acceleration vector in launch coordinate system (ft/sec^2)
PXND		Velocity vector in local coordinate system (ft/sec)
PYL WGT	*	Payload weight (lb)
R		Distance from Earth center to vehicle (ft)
RA	*	Fuel axial radius of gyration squared for each stage (ft^2)
RANGE		Range at burnout of each stage (N.M.) +
RANGE1		Dummy variable used for impact (N.M.)
RAD		$\frac{\pi}{180}$
RB	*	Fuel transverse radius of gyration squared for each stage (ft^2)
RE		Radius of Earth as a function of latitude (ft)
REE		Equatorial radius of the Earth (ft)
REL		Distance from Earth center to launch pt (ft) +
RHO		Atmospheric density (slugs/ft^3)
ROT		Matrix from launch to Earth centered coordinates
ROT1		"COMMON" rotation matrix +
R1X		Dummy variable used in X integration
R2X		Dummy variable used in Y integration
R3X		Dummy variable used in Z integration
R4X		Dummy variable used in φ integration
R5X		Dummy variable used in θ integration
R2		Distance squared from Earth center to vehicle (ft^2)
S		$\triangleq C(1 - e_E^2)$ (ft)
SILAT		Sine of the initial latitude

+ Stored in common

INPUTS

SMAX		Maximum integration step size (sec)	
SMIN		Minimum integration step size (sec)	
SPHI		Sine of PHI	
SPHT		Sine of PHT	
SPI		Inverted spin table	
SPIN	*	Input spin table (rad/sec)	
SPIT		Integrated spin table (rad)	
SPT		Inverted spin time table	
SPTIME	*	Input spin time table (sec)	
SS		Integration step size (sec)	
STARTT	*	Start time or launch time (sec)	
STHC		Sine of the initial geocentric latitude	+
STHET		Sine of THETA	
STP	*	Initial PHI (deg)	
STPD	*	Initial PHI DOT (deg/sec)	
STT	*	Initial THETA (deg)	
STTD	*	Initial THETA DOT (deg/sec)	
STX	*	Initial position vector in launch coordinate system (ft)	
STXD	*	Initial velocity vector in launch coordinate system (ft/sec)	
SW		Angle between wind azimuth and X axis (rad)	
T		Thrust vector in launch coordinate system (lb)	
TABLE		Table of printout times	
TB2	*	Number of orbits	+
TB3	*	Time increment Δt (sec)	+
TB4	*	Ending time (sec)	+
TB6	*	Name of look-angle station (up to 24 Hollerith characters)	+
TB7	*	Latitude of look-angle station (deg)	+
TB8	*	Longitude of look-angle station (deg)	+
TB9	*	Altitude of look-angle station (ft)	+
TDEL	*	Thrust misalignment angle (rad)	
TFIMP		Stage drop time (sec)	+
THC		Initial geocentric latitude (rad)	

+ Stored in common

INPUTS

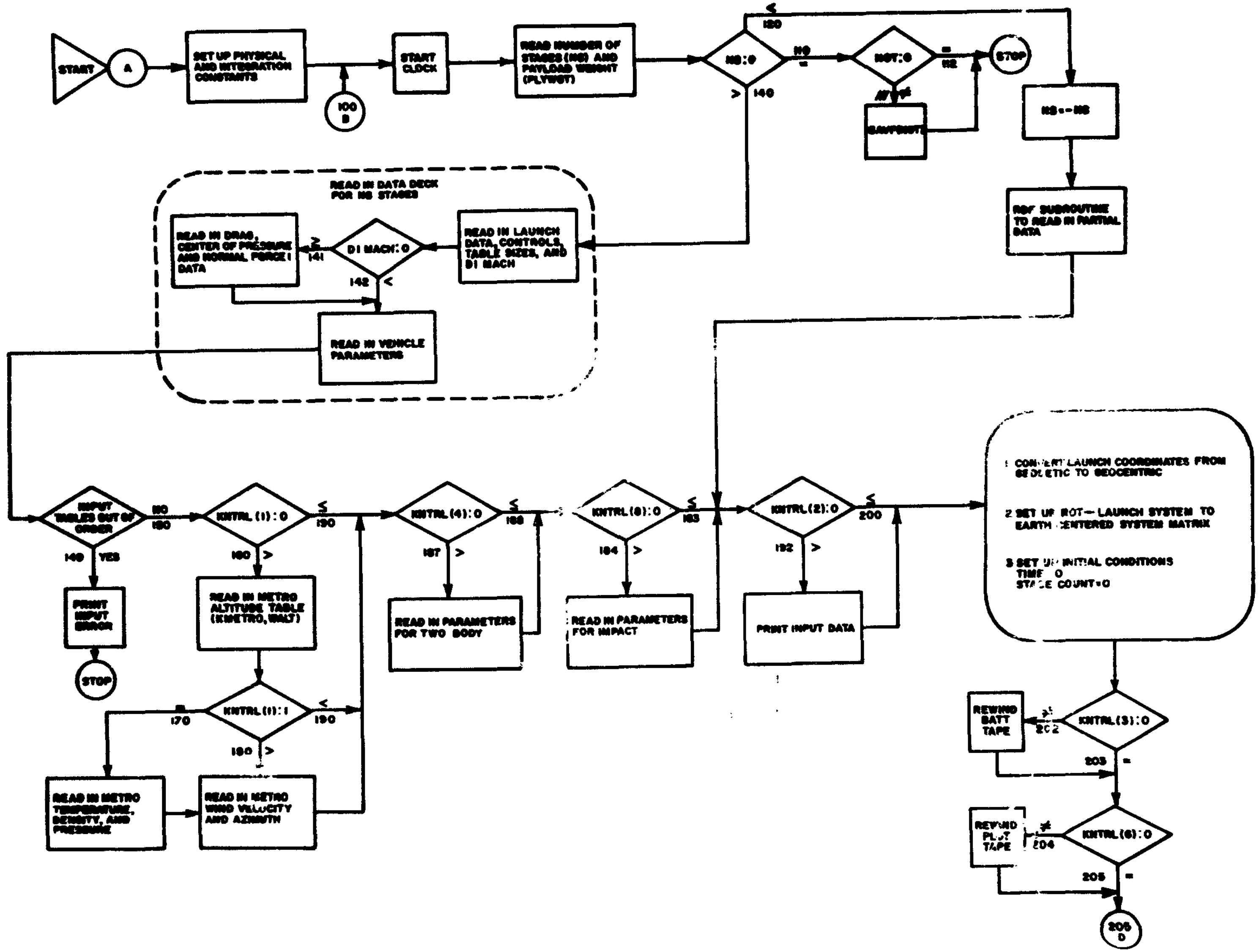
THETA		Euler angle in the horizontal plane (rad)
THETD		First derivative of THETA (rad/sec)
THETDD		Second derivative of THETA (rad/sec^2)
THRUST	*	Input thrust table (lb)
THTAB		Inverted thrust table
TIME		Time (dependent variable) (sec)
TIMET	*	Time table for thrust (sec)
TMBO	*	Burnout time for each stage (sec)
TMCC	*	Time to change coefficients for each stage (sec)
TMI	*	Ignition time for each stage (sec)
TOMEG		Twice the Earth spin rate = $2\omega_E$ (rad/sec)
TP		Intermediate variables used in output block
TP1		Intermediate variables used in output block
TP2		Intermediate variables used in output block
TPHO		Orientation angle of thrust misalignment at a given time (rad)
TRERR		Integration truncation error
TRVM		Transverse moment of inertia at any time (ft^2 - slug)
TSTOP	*	Preset time to stop powered portion of program (sec)
TT		Thrust vector in Earth centered coordinate system (lb)
TTTAB		Inverted thrust time table
TWOPI		2π
V		Velocity of vehicle (ft/sec)
VA		Velocity of sound (ft/sec)
V2		Cross wind perpendicular to the velocity vector (ft/sec)
V3		Cross wind perpendicular to the velocity vector (ft/sec)
V MACH		Mach number of vehicle
VWX		Wind components in Earth-centered system (ft/sec)
VWY		Wind components in Earth-centered system (ft/sec)
VWZ		Wind components in Earth-centered system (ft/sec)
VX		Velocity vector with wind in Earth-centered system (ft/sec)
WALT	*	Input metro altitude table (ft)

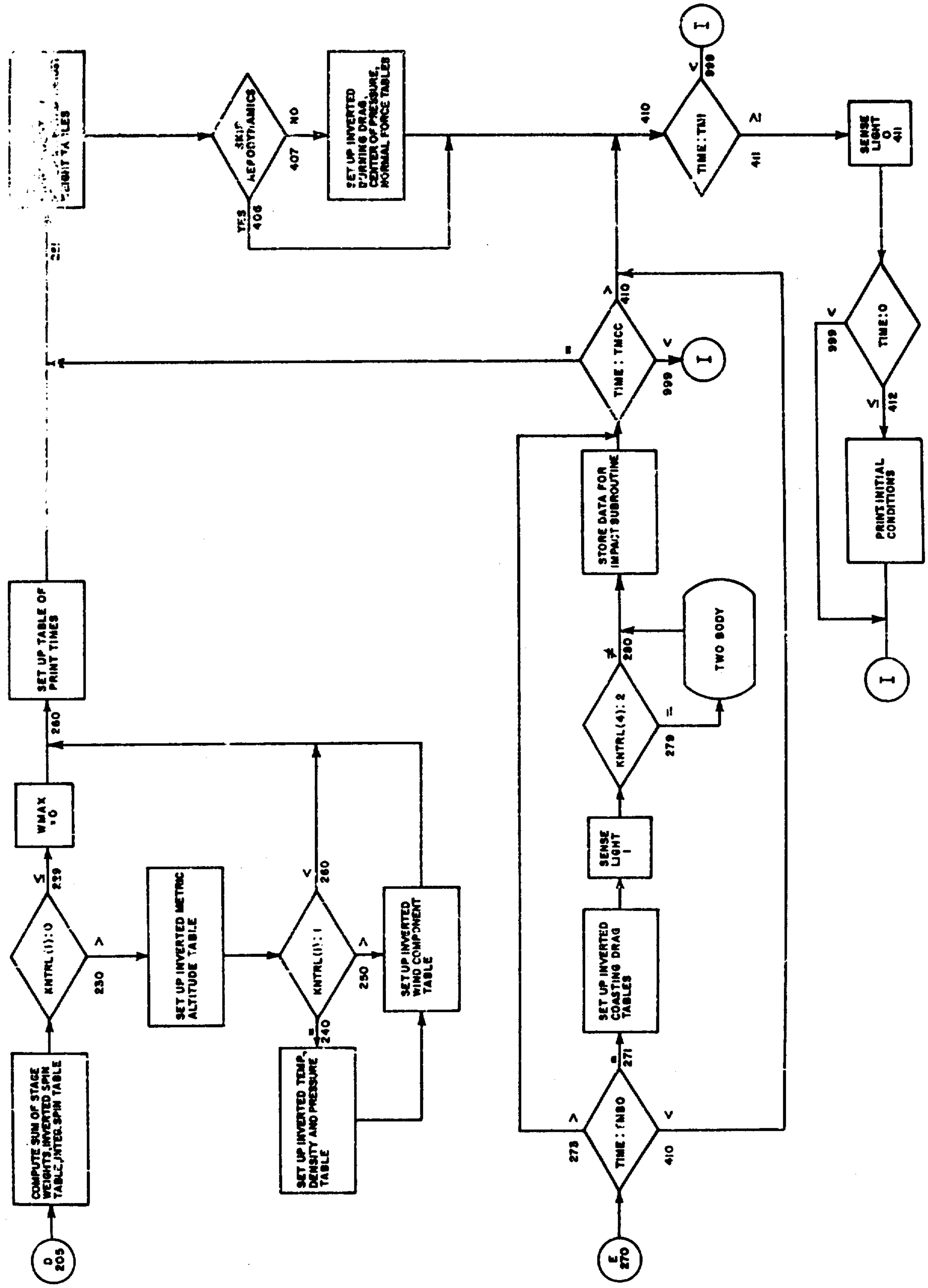
INPUTS

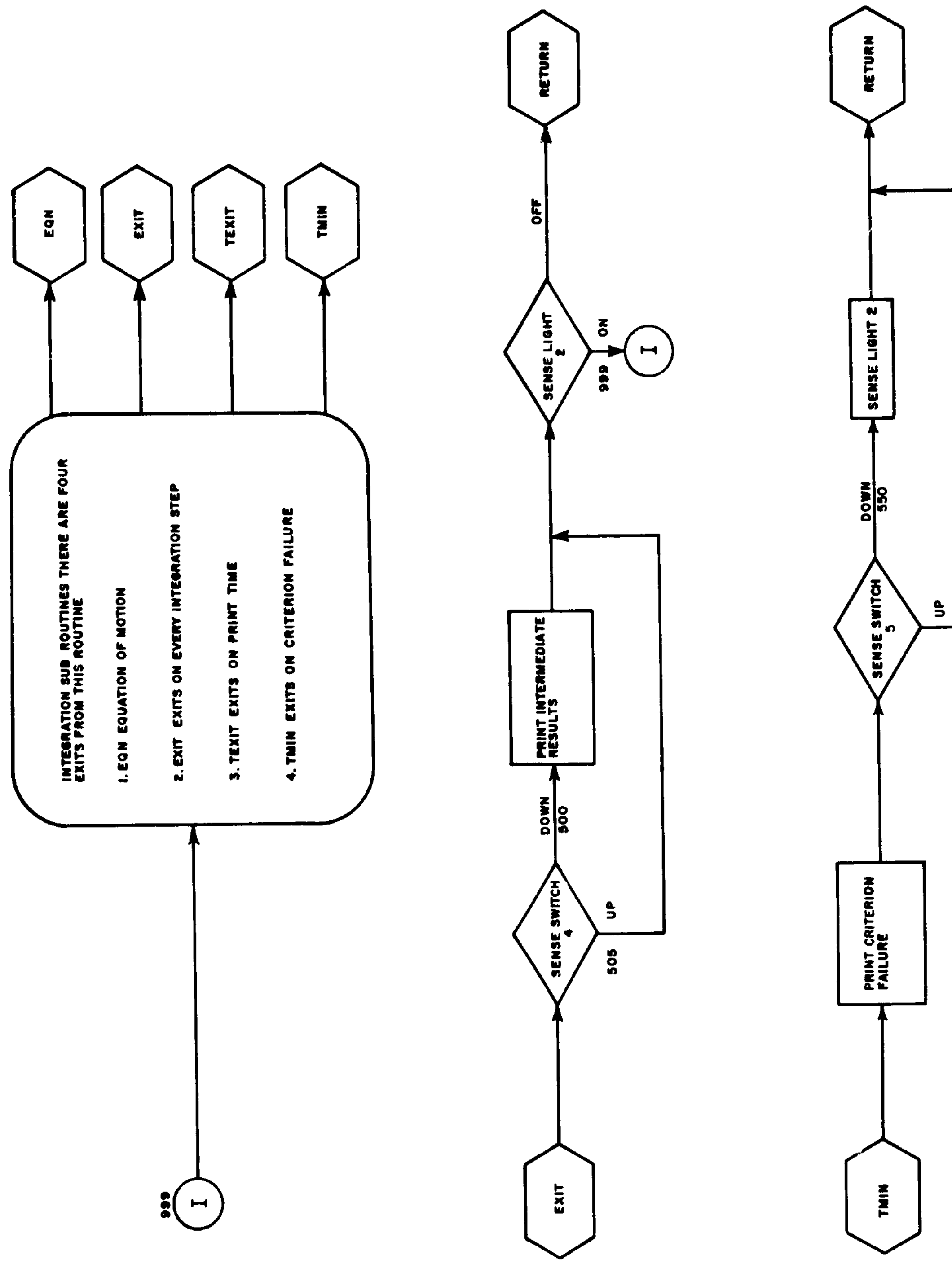
WALTU		Inverted metro altitude table (ft)
WDEN	*	Input metro atmosphere density (kgm/m^3)
WDU		Inverted metro atmosphere density (slugs/ft^3)
WGT		Mass of remaining missile for each stage (slugs)
WGTAB		Inverted mass table (slugs)
WGTC		Mass of fuel for each stage (slugs)
WGTCI		Mass expelled from ignition (slugs)
WGTI		Mass at a given time WGTI = WGT-WGTCI (slugs)
WINDA	*	Local wind azimuth (deg from North, clockwise)
WINDV	*	Local wind velocity (ft/sec)
WMAX		Maximum altitude of metro table (ft)
WPRES	*	Input metro pressure table (millibars)
WPU		Inverted metro pressure table (lb/ft^2)
WTEMP	*	Input metro temperature table (^{o}C)
WTU		Inverted speed of sound table (ft/sec)
WX		Wind component parallel to Earth-centered X axis (ft/sec)
WY		Wind component parallel to Earth-centered Y axis (ft/sec)
WZ		Wind component parallel to Earth-centered Z axis (ft/sec)
X1X		Variables used in geodetic to geocentric conversion
X1Y		Variables used in geodetic to geocentric conversion
XDIMP		Velocity vector at burnout for impact (ft/sec) +
XIMPA		Position vector at burnout for impact (ft) +
XJ		First harmonic coefficient for oblateness
XMACH		Mach number table for C_DA/m +
XSP		Spin rate at any given time (rad/sec)
XSPT		Integrated spin at any given time (rad)
XVX		Variable used in calculation of PX
Z2		Lines-per-page count for Two Body output +

+ Stored in common

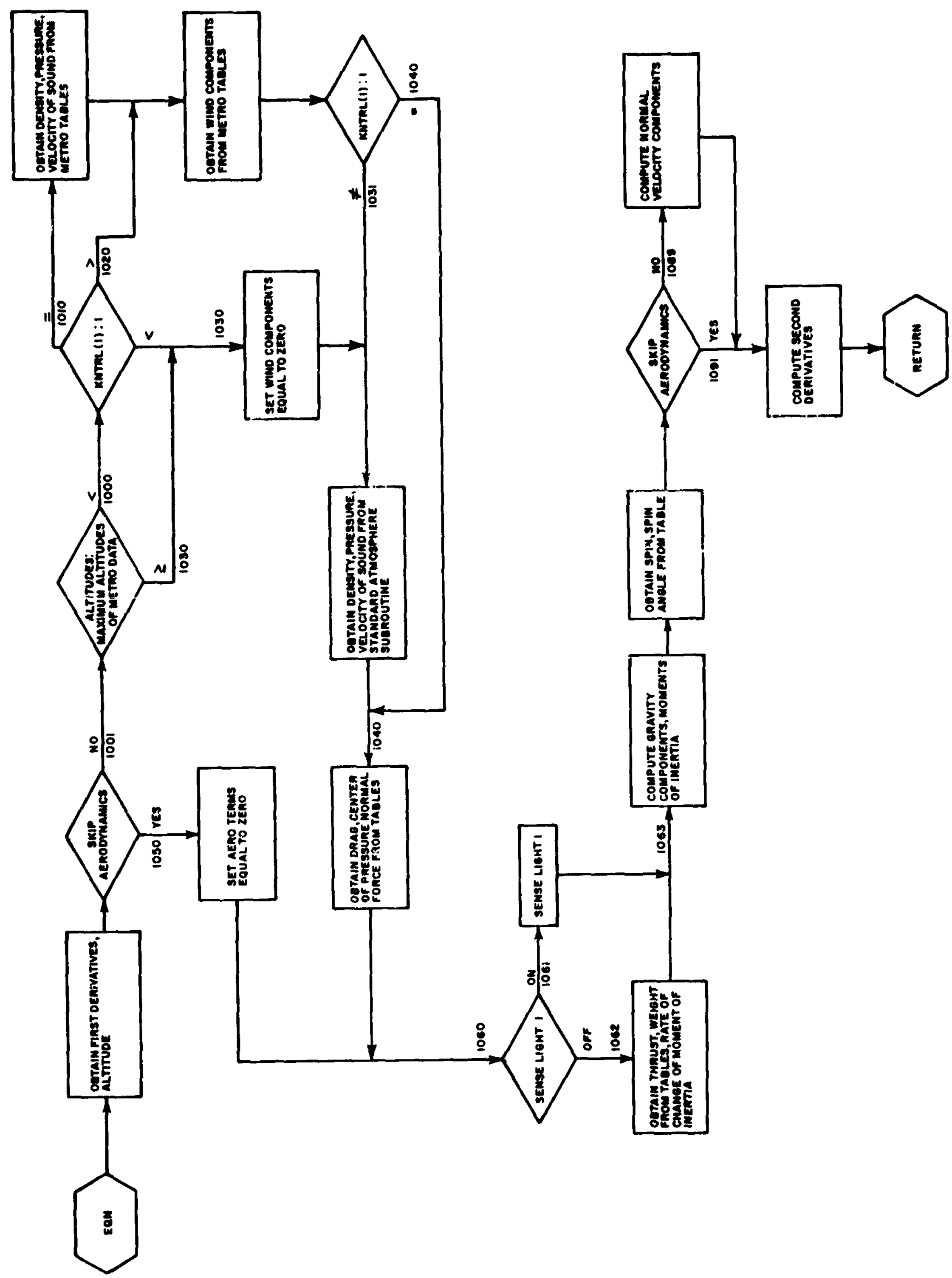
5. SPURT FLOW DIAGRAMS.

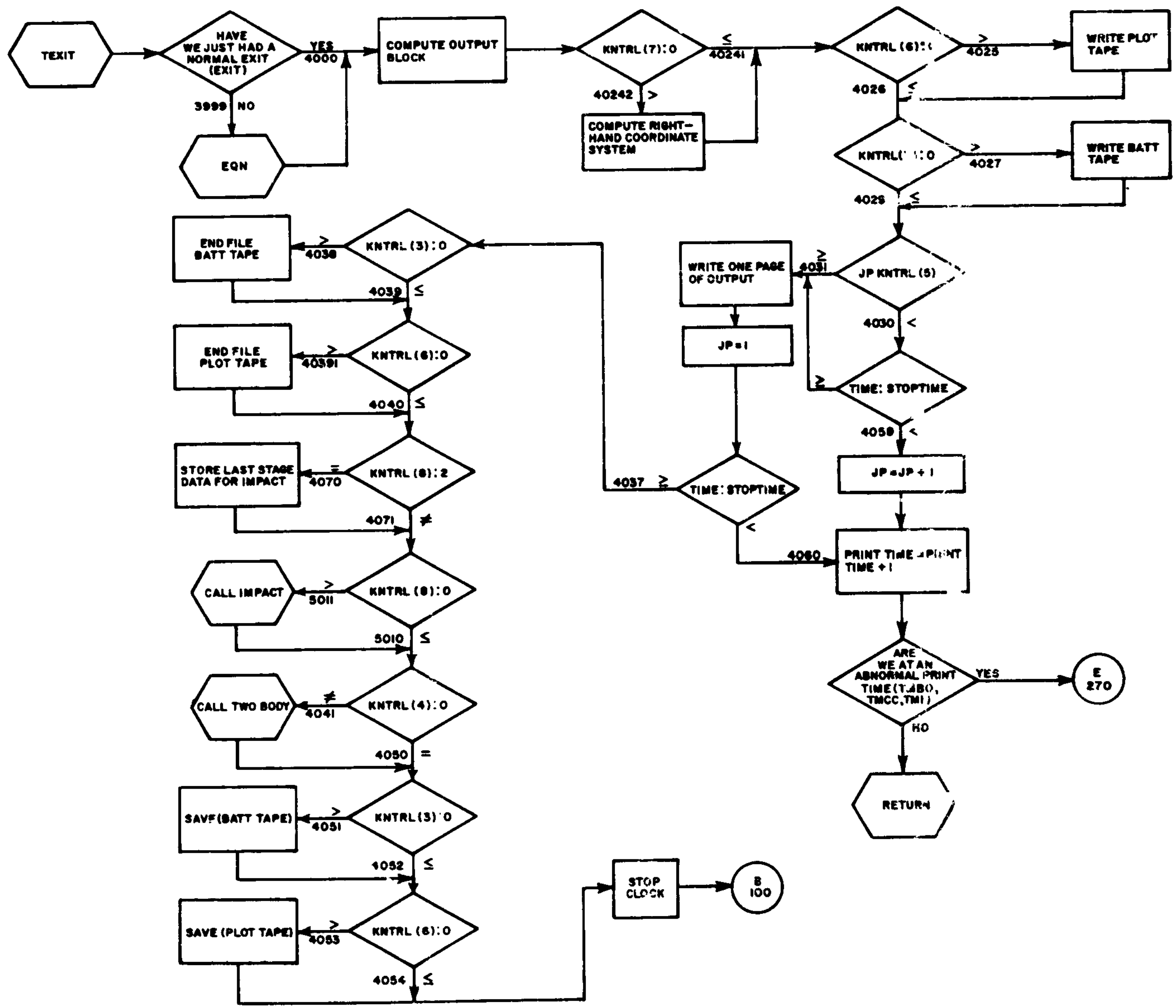

START
A
SET UP PHYSICAL AND INTEGRATION CONSTANTS
100 B
START CLOCK
READ NUMBER OF STAGES (NS) AND PAYLOAD WEIGHT (PLYWGT)
NS:0
STOP
READ IN DATA DECK FOR NS STAGES
READ IN LAUNCH DATA, CONTROLS, TABLE SIZES, AND DI MACH
DI MACH:0
READ IN DRAG, CENTER OF PRESSURE AND NORMAL FORCE DATA
READ IN VEHICLE PARAMETERS
INPUT TABLES OUT OF ORDER
YES
PRINT INPUT ERROR
STOP
KNTRL(1):0
READ IN METRO ALTITUDE TABLE (KMETRO, WALT)
KNTRL(1):1
READ IN METRO TEMPERATURE, DENSITY, AND PRESSURE
READ IN METRO WIND VELOCITY AND AZIMUTH
KNTRL(4):0
READ IN PARAMETERS FOR TWO BODY
KNTRL(8):0
READ IN PARAMETERS FOR IMPACT
KNTRL(2):0
PRINT INPUT DATA
1 CONVERT LAUNCH COORDINATES FROM GEODETIC TO GEOCENTRIC
2 SET UP ROT— LAUNCH SYSTEM TO EARTH CENTERED SYSTEM MATRIX
3 SET UP INITIAL CONDITIONS
TIME 0
STAGE COUNT=0
KNTRL(3):0
REWIND BATT TAPE
KNTRL(6):0
REWIND PLOT TAPE
206 D


D 205
COMPUTE SUM OF STAGE WEIGHTS, INVERTED SPIN TABLE, INTEG. SPIN TABLE
KNTRL (1) : 0
≤ 229
WMAX = 0
> 230
SET UP INVERTED METRIC ALTITUDE TABLE
KNTRL (1) : 1
= 240
SET UP INVERTED TEMP., DENSITY AND PRESSURE TABLE
> 250
SET UP INVERTED WIND COMPONENT TABLE
< 260
260
SET UP TABLE OF PRINT TIMES
SKIP AERODYNAMICS
YES 406
NO 407
SET UP INVERTED BURNING DRAG, CENTER OF PRESSURE, NORMAL FORCE TABLES
410
TIME : TNI
< 999
I
≥ 411
SENSE LIGHT 0 411
TIME : 0
< 999
≤ 412
PRINT INITIAL CONDITIONS
I
E 270
TIME : TNBO
> 273
= 271
< 410
SET UP INVERTED COASTING DRAG TABLES
SENSE LIGHT 1
KNTRL (4) : 2
≠ 280
= 279
TWO BODY
STORE DATA FOR IMPACT SUBROUTINE
TIME : TNCC
=
> 410
< 999
I



INTEGRATION SUB ROUTINES THERE ARE FOUR EXITS FROM THIS ROUTINE
1. EQN EQUATION OF MOTION
2. EXIT EXITS ON EVERY INTEGRATION STEP
3. TEXIT EXITS ON PRINT TIME
4. TMIN EXITS ON CRITERION FAILURE
999
I
EQN
EXIT
TEXIT
TMIN
EXIT
SENSE SWITCH 4
505
UP
DOWN 500
PRINT INTERMEDIATE RESULTS
SENSE LIGHT 2
999
ON
OFF
RETURN
TMIN
PRINT CRITERION FAILURE
SENSE SWITCH 5
UP
DOWN 550
SENSE LIGHT 2
RETURN


EGM
OBTAIN FIRST DERIVATIVES, ALTITUDE
SKIP AERODYNAMICS
NO
1001
YES
1050
SET AERO TERMS EQUAL TO ZERO
ALTITUDES: MAXIMUM ALTITUDES OF METRO DATA
<
1000
≥
1030
KNTRL (1) : 1
=
1010
>
1020
<
1030
OBTAIN DENSITY, PRESSURE, VELOCITY OF SOUND FROM METRO TABLES
OBTAIN WIND COMPONENTS FROM METRO TABLES
SET WIND COMPONENTS EQUAL TO ZERO
KNTRL(1) : 1
≠
1031
=
1040
OBTAIN DENSITY, PRESSURE, VELOCITY OF SOUND FROM STANDARD ATMOSPHERE SUBROUTINE
1040
OBTAIN DRAG, CENTER OF PRESSURE NORMAL FORCE FROM TABLES
1060
SENSE LIGHT 1
ON
1061
OFF
1062
SENSE LIGHT 1
OBTAIN THRUST, WEIGHT FROM TABLES, RATE OF CHANGE OF MOMENT OF INERTIA
1063
COMPUTE GRAVITY COMPONENTS, MOMENTS OF INERTIA
OBTAIN SPIN, SPIN ANGLE FROM TABLE
SKIP AERODYNAMICS
NO
1068
YES
1091
COMPUTE NORMAL VELOCITY COMPONENTS
COMPUTE SECOND DERIVATIVES
RETURN


TEXIT
HAVE WE JUST HAD A NORMAL EXIT (EXIT)
YES
4000
3999
NO
EQN
COMPUTE OUTPUT BLOCK
KNTRL (7):0
≤
40241
40242
>
COMPUTE RIGHT-HAND COORDINATE SYSTEM
KNTRL (6):
>
4025
WRITE PLOT TAPE
4026
≤
>
4027
WRITE BATT TAPE
4028
≤
JP KNTRL (5)
≥
4031
WRITE ONE PAGE OF OUTPUT
JP=1
4030
<
TIME: STOPTIME
≥
4059
<
JP=JP+1
4037
≥
TIME: STOPTIME
<
4060
PRINT TIME TIME + 1
ARE WE AT AN ABNORMAL PRINT TIME (T,BO, TMCC,TMI)
YES
E 270
NO
RETURN
KNTRL (3):0
>
4038
END FILE BATT TAPE
4039
≤
KNTRL (6):0
>
40391
END FILE PLOT TAPE
4040
≤
KNTRL (8):2
=
4070
STORE LAST STAGE DATA FOR IMPACT
4071
≠
KNTRL (8):0
>
5011
CALL IMPACT
5010
≤
KNTRL (4):0
≠
4041
CALL TWO BODY
4050
=
>
4051
SAVE (BATT TAPE)
4052
≤
KNTRL (6):0
>
4053
SAVE (PLOT TAPE)
4054
≤
STOP CLOCK
B 100

6. SUBROUTINES.

a. Atmosphere subroutine.

The 1962 COESA standard atmosphere is incorporated as a subroutine of SPURT.[15] This atmosphere uses the equations derived for the 1959 ARDC model atmosphere.[16] These equations are

(1) Molecular-scale temperature

$$Tm = (Tm)_b + Lm\,(H - H_b)$$

(2) Pressure - altitude

$$P = P_b \left[\frac{(Tm)_b}{(Tm)_b + Lm\,(H - H_b)}\right]^{\frac{GM_o}{R^* Lm}} \quad \text{for } Lm \neq 0$$

$$P = P_b \exp\left[\frac{-GM_o\,(H - H_b)}{R^*\,(Tm)_b}\right] \quad \text{for } Lm = 0$$

(3) Density - altitude

$$\rho = 3.236598 \times 10^{-4}\ \frac{P}{Tm}$$

(4) Speed of sound

$$Vs = 1116.4437\ \sqrt{Tm/Tm_o}$$

where b = subscript that refers to the quantity at the base of the constant-gradiant layer.

GMo/R*	=	constant of the air gas
H	=	geopotential altitude in meters
H_b	=	geopotential altitude in meters at the base of a constant gradiant layer
Lm	=	dTm/dH = the gradiant of the molecular scale temperature
P	=	pressure

P_b = pressure at the base of a particular layer
Tm = the molecular scale temperature
$(Tm)_b$ = the value of Tm at the base of a particular layer
Vs = speed of sound (ft/sec)
ρ = density of the air ($slugs/ft^3$)

The skeleton of the COESA is from reference 15 and is given in table 1.

TABLE 1.

SKELETON OF THE U.S. STANDARD ATMOSPHERE—1962

Defining temperature and molecular weights of the proposed U. S. Standard Atmosphere and computed pressures and densities, where Z = geometric altitude, h = geopotential altitude, T = kinetic temperature, M = mean molecular weight, L = gradient of molecular scale temperatures = dT_M/dh (below 79 geop. km) = dT_M/dZ (above 79 geop. km), T_M = molecular scale temperature = $(T/M)\,M_0$; and M_0 = sea-level value of M.

Z, km	h, km	T_M, K	L, K/km	M	T, K	P, mb	ρ, g/m^3
0.000	0.000	288.15	-6.5	28.966	288.15	10.1325 + 2*	1.2250 + 3*
11.019	11.000	216.65	0.0	28.966	216.65	2.2632 + 2	3.6392 + 2
20.063	20.000	216.65	1.0	28.966	216.65	5.4747 + 1	8.8033 + 1
32.162	32.000	228.65	2.8	28.966	228.65	8.6798 0	1.3225 + 1
47.350	47.000	270.65	0.0	28.966	270.65	1.1090 0	1.4275 0
52.429	52.000	270.65	-2.0	28.966	270.65	5.8997 - 1	7.5939 - 1
61.591	61.000	252.65	-4.0	28.966	252.65	1.8209 - 1	2.5108 - 1
79.994	79.000	180.65	0.0	28.966	180.65	1.0376 - 2	2.0009 - 2
90.000	88.743	180.65	3.0	28.966	180.65	1.6437 - 3	[illegible].1698 - 3
100.000	98.451	210.65	5.0	28.88	210.02	3.0070 - 4	4.9731 - 4
110.000	108.129	260.65	10.0	28.56	257.00	7.3527 - 5	9.8277 - 5
120.000	117.777	360.65	20.0	28.07	349.49	2.5209 - 5	2.4352 - 5
150.000	146.542	960.65	15.0	26.92	892.79	5.0599 - 6	1.8350 - 6
160.000	156.071	1,110.65	10.0	26.66	1,022.20	3.6929 - 6	1.1584 - 6
170.000	165.572	1,210.65	7.0	26.40	1,103.40	2.7915 - 6	8.0330 - 7
190.000	184.485	1,350.65	5.0	25.85	1,205.40	1.6845 - 6	4.3450 - 7
230.000	221.968	1,550.65	4.0	24.70	1,322.30	6.9572 - 7	1.5631 - 7
300.000	286.478	1,830.65	3.3	22.66	1,432.10	1.8828 - 7	3.5831 - 8
400.000	376.315	2,160.65	2.6	19.94	1,487.40	4.0278 - 8	6.4945 - 9
500.000	463.530	2,420.65	1.7	17.94	1,499.20	1.0949 - 8	1.5758 - 9
600.000	548.235	2,590.65	1.1	16.84	1,506.10	3.4475 - 9	4.6362 - 10
700.000	630.536	2,700.65		16.17	1,507.60	1.1908 - 9	1.5361 - 10

*Power of 10 by which preceding number must be multiplied.

This routine must be called by the CODAP symbolic language as follows:

```
        LDA    ALT
        RTJ    ATMOS
TEM     OCT
PRES    OCT
RHO     OCT
VA      OCT
        ERR
        N. R.
```

where

ALT	is	the altitude in feet
TEM	is	the temperature
PRES	is	the pressure
RHO	is	the density
VA	is	the speed of sound
ERR	is	the error return
N. R.	is	the normal return

b. Subroutine E clock and subroutine S clock.

These subroutines are incorporated to compute the time used by the computer in computing a typical trajectory. These subroutines are written in the CODAP symbolic language and are callable by FORTRAN. The S clock subroutine will initialize the computer clock to zero, and the E clock subroutine will stop the clock and print out the elapsed time in hours, minutes, and seconds.

To use the S clock subroutine:

CALL SCLOCK

To use the E clock subroutine:

CALL ECLOCK(N) where time is printed on tape number N.

c. Subroutine GEODED.

The subroutine converts geocentric latitude and radius to geodetic latitude and altitude by use of the following equations.[8]

$$\theta_G = \theta_c + \sin^{-1} \left\{ \frac{a_E}{r} \left[f \sin 2\theta_c + f^2 \sin 4\,\theta_c \left(\frac{a_E}{r} - \frac{1}{4} \right) \right] \right\}$$

$$H_G = r - a_E \left[1 - f \sin^2 \theta_c - \frac{f^2}{2} \sin^2 2\,\theta_c \left(\frac{a_E}{r} - \frac{1}{4} \right) \right]$$

where a_E = equatorial radius of the Earth

f = flattening of the Earth

r = geocentric position vector

H = geodetic altitude

θ = latitude

c = refers to geocentric

G = refers to geodetic

This subroutine is callable by FORTRAN.

To use:

```
CALL GEODED (A, B, C, D)
```

where A is the geocentric latitude (radians)

B is the geocentric position vector (feet)

C is the geodetic latitude (radians)

D is the geodetic altitude (feet)

d. Impact subroutine.

The Impact subroutine is a point mass three-degree-of-freedom trajectory subroutine. This routine is incorporated primarily to compute the trajectories of the "separated" expended stages.

The vector form of the equation of motion is the geocentric position equation derived in section 2 of this report and is

$$\frac{d^2\vec{R}}{dt} = \frac{\Sigma\vec{F}}{M} + 2\left(\frac{d\vec{R}}{dt} \times \vec{\omega}_E\right) - \vec{\omega}_E \times (\vec{\omega}_E \times \vec{R})$$

The atmosphere subroutine is used along with the oblate Earth described in section 2. The drag parameter (C_dA/M) vs. Mach number table for the expended stages are read in the main program and placed in common for use by the Impact subroutine.

The Integration routine will use the same option as the main portion of the program (powered flight) uses. When the altitude is negative, the Impact subroutine will punch a card containing the name, stage number, impact latitude, and longitude. The routine will then terminate and return to the main program. To use the Impact subroutine, control number 8 must be set equal to 1 or to 2.

VARIABLES USED IN IMPACT SUBROUTINE

AB	Not used in impact	*
AC	Not used in impact	*
ACODE	Integration code	+
ACODES	Integration code table	+
AD	Not used in impact	*
AE	Equatorial radius of the Earth	
AG	Not used in impact	*
AH	Not used in impact	*
AI	Not used in impact	*
AK	Not used in impact	*
AL	Block reserved for equivalence	
ALT	Altitude of empty stage	
AM	Not used in impact	*
AN	Not used in impact	*
AN1	Variables used for output parameters	
AN2	Variables used for output parameters	
AO	Not used in impact	*
AP	Not used in impact	*
ARG	Argument used for gravitational computation	
CBLACK	Block reserved for integration	
CDM	$C_D A/M$ - drag parameter	
CHECK	Check used in integration	*
COSRA	Cosine of the range angle	
DELPR	Print time increment	*
DM	Drag over mass parameter	
DRA	Variable used in computing drag	
DRAG	Drag parameter of empty stage	
DUMB1	Not used in impact	*
ERR	Integration error	
FMN	Mach number of empty stage	

---

* Stored in common
\+ Equivalence with AL

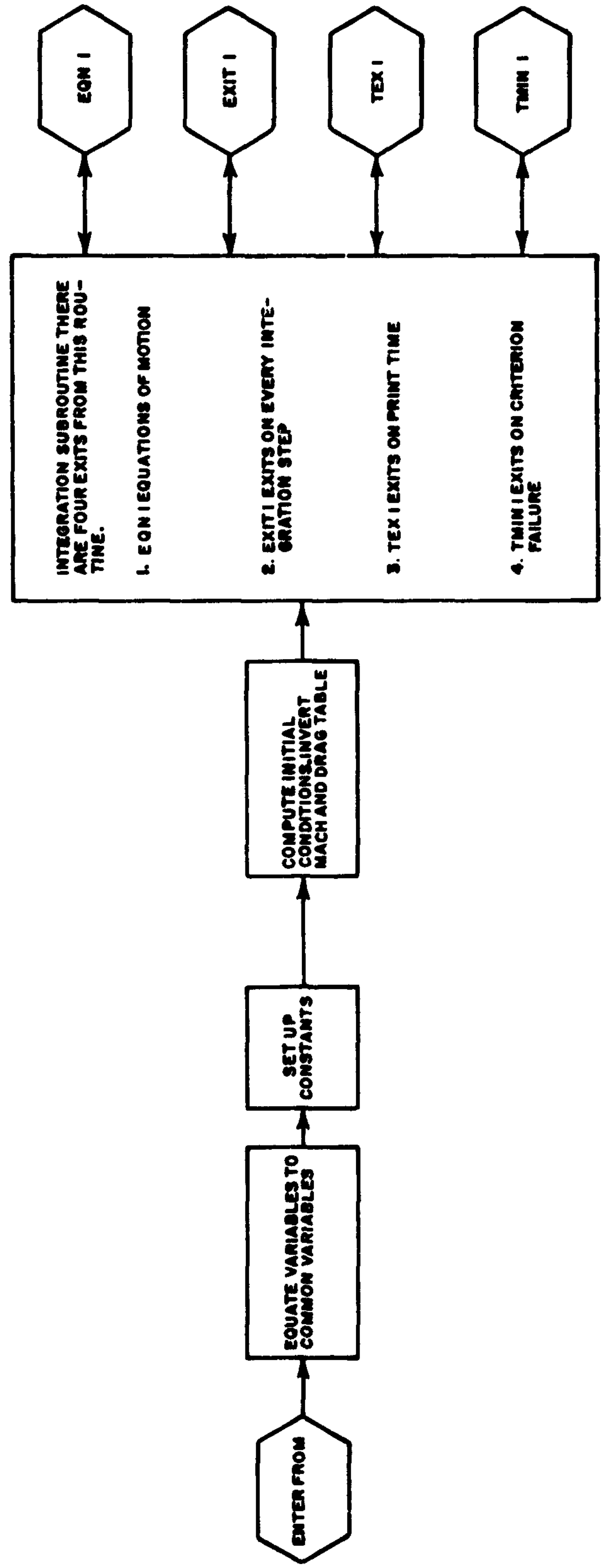
GM	Gravitational constant	
GX	X component of gravitational attraction	
GY	Y component of gravitational attraction	
GZ	Z component of gravitational attraction	
H	Variable used for first time increment	
I	Utility index	
ICODE	Code for method of integration	+
II	Utility index	
JP	Page count	
JJ	Number of stage being computed	
J6	Number of lines per page	
K	Utility index	
KAA	Not used in impact	*
KAF	Not used in impact	*
KAZ	Not used in impact	*
N	Number of values in drag table	
NAME	Name stored in common	*
NN	Integer for drag selection	*
NOE	Number of equations for integration	+
NOT	Number of output tape	*
OUT	Dimensioned output variables	
PE	Earth flattening constant	
PI	$\pi = 3.1415927$	
PRTIM1	Print time for integration routine	
R	Length of position vector	
RAD	$= \pi/180. = 1/57.2957795$	
RANGE	Great circle range from launch point	*
R2	Position vector squared	
SIT	Sine of the latitude	
SMAX	Maximum integration step size	+
SMIN	Minimum integration step size	+

---

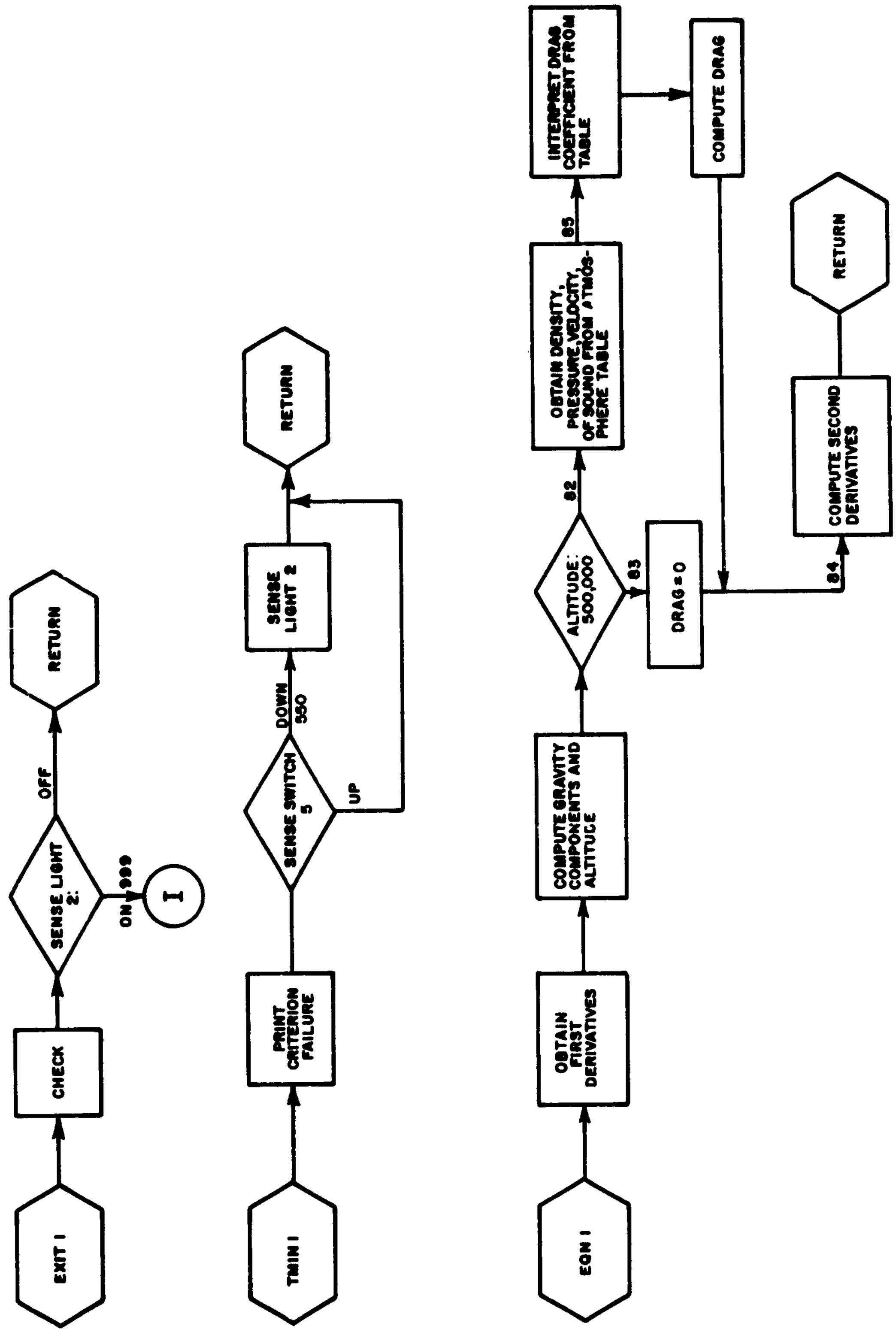
* Stored in common
\+ Equivalence with AL

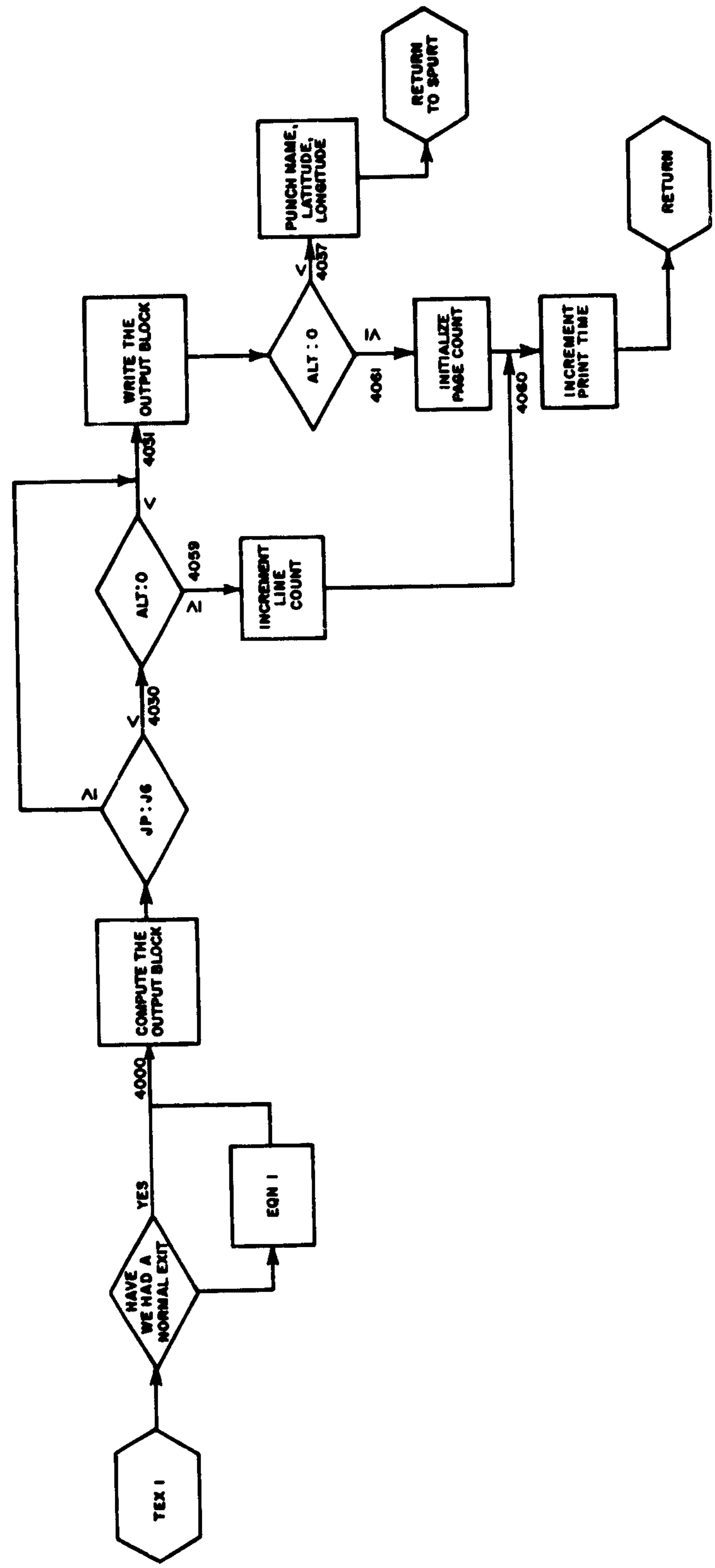
SS	Integration step size	+
T	Time, independent variable	+
TFIMP	Initial time for Impact computation	*
THET	Latitude of position vector	
TRERR	Integration truncation error	+
VA	Velocity of sound	
VD	Vector used for matrix rotation	
VEL	Velocity of empty stage	
W	Rotational velocity of the Earth	
WD	Vector used for matrix rotation	
W2	= W squared	
X	Component of position vector	+
XALT	Altitude from GEOD subroutine	
XD	Velocity vector component	+
XDD	Second derivative of the motion equation	+
XDIMP	Input velocity vector	*
XD1	Velocity used in integration routine	+
XIMPA	Input position vector	*
XK2	Oblateness constant	
XLAT	Geodetic latitude from GEOD subroutine	
XMACH	Mach number table for drag	*
XMN	Stage Mach number table	
XX	Vector for computing range	
XXD	Velocity vector	
Y	Component of position vector	+
YD	Velocity vector component	+
YDD	Second derivative of the motion equation	+
YD1	Velocity used in integration routine	+
Z	Component of position vector	+
ZD	Velocity vector component	+
ZDD	Second derivative of the motion equation	+
ZD1	Velocity used in integration routine	+
Z2	Number of lines per page	*

* Stored in common
+ Equivalence with AL



ENTER FROM
EQUATE VARIABLES TO COMMON VARIABLES
SET UP CONSTANTS
COMPUTE INITIAL CONDITIONS,INVERT MACH AND DRAG TABLE
INTEGRATION SUBROUTINE THERE ARE FOUR EXITS FROM THIS ROU- TINE.
1. E Q N I EQUATIONS OF MOTION
2. EXIT I EXITS ON EVERY INTE- GRATION STEP
3. TEX I EXITS ON PRINT TIME
4. TMIN I EXITS ON CRITERION FAILURE
EQN I
EXIT I
TEX I
TMIN I


EXIT I
CHECK
SENSE LIGHT 2:
ON 999
I
OFF
RETURN
TMIN I
PRINT CRITERION FAILURE
SENSE SWITCH 5
UP
DOWN 550
SENSE LIGHT 2
RETURN
EQN I
OBTAIN FIRST DERIVATIVES
COMPUTE GRAVITY COMPONENTS AND ALTITUDE
ALTITUDE: 500,000
82
83
DRAG = 0
OBTAIN DENSITY, PRESSURE, VELOCITY, OF SOUND FROM ATMOS-PHERE TABLE
85
INTERPRET DRAG COEFFICIENT FROM TABLE
COMPUTE DRAG
84
COMPUTE SECOND DERIVATIVES
RETURN



TEX I
HAVE WE HAD A NORMAL EXIT
YES
EQN I
4000
COMPUTE THE OUTPUT BLOCK
JP : J6
≥
<
4030
ALT : 0
≥
4059
INCREMENT LINE COUNT
<
4031
WRITE THE OUTPUT BLOCK
ALT : 0
<
4037
PUNCH NAME, LATITUDE, LONGITUDE
RETURN TO SPUIRT
≥
4061
INITIALIZE PAGE COUNT
4060
INCREMENT PRINT TIME
RETURN

e. Integration.

A. IDENTIFICATION

TITLE: Numerical Integration of Ordinary Differential Equations with Error Control

CO-OP ID: D2 CODA NMI 2

PROGRAMMER: Roger Johnson

DATE: February 27, 1961

B. PURPOSE

To integrate a set of N simultaneous first order differential equations of the form:

$$\frac{dx_i}{dt} = f_i\,(t,\ x_1,\ x_2, \ldots x_N),\ i = (1, 2, \ldots N).$$

The user has the option of either a Runge-Kutta or Adams-Moulton integration scheme. The integration step size, Δt, may be a variable step size under error control or a fixed step size. A print option causes printing of the initial conditions and various parameters before the start of the integration to help define each case. The user also has an option to break into the program at exact specified values of the independent variable t.

C. USAGE

The index registers are saved. No internal checks of arithmetic faults (exponential overflow, underflow, etc.) are made. When the print option is used, the differential equation routine removes the selection of interrupt on arithmetic faults before printing and does a clear arithmetic faults after printing. The differential equations routine should not cause an arithmetic fault unless the x_i's exceed the range of the floating point format.

1. Calling Sequence -

The calling sequence used to start or restart; if parameters are changed, the integration of a set of differential equation is

Loc	OPN	B	M	OPN	B	M
BETA	SLJ	4	ADAMS	0	P	DERIV
BETA+1	0	0	TEXIT	0	0	T
BETA+2	0	0	DATA	0	0	COMMON
BETA+3	0	0	EXIT	0	0	TMIN
BETA+4	ERROR RETURN					

2. Parameters -

The upper part of the first instruction of the calling sequence is a return jump command SLJ 4 ADAMS , to the first location of differential equation routine.

If P is unequal to zero the differential equation routine will print the initial conditions and the parameters in the DATA region using the Generalized Listable Output Routine (J5 LMSD OUTPUT). If P is zero no printing takes place.

DATA: starts a block of 3N+ 8 locations that the user sets up with the parameters and variables. The location and description of the parameters and variables are as follows:

DATA contains the integration scheme code which is a fixed point binary integer (binary point at the far right) set up by the user.

a) If code = 0: The integration scheme is the Runge-Kutta mode with a fixed Δt.

b) If code = 1: The integration scheme is the Runge-Kutta predictor-corrector mode with a variable Δt.

c) If code = 2: The integration scheme is the Adams-Moulton mode with a fixed Δt.

d) If code = 3: The integration scheme is the Adams-Moulton predictor-corrector mode with a variable Δt.

e) If the code is negative or any binary digit greater than 3, the code is out of range. If this occurs, the AC is set to the binary integer 2 and a jump is made to the error return in BETA+ 4.

DATA+1 contains the floating point number A used in the variable step mode to prevent unnecessary halfing of Δt when x_i becomes small in magnitude. If A is positive then a positive floating point number A_i must be determined for each x_i, $(i = 1, 2, \ldots N)$, and be set up by the user in the following locations:

A_1 in DATA + 2N + 8

A_2 in DATA + 2N + 9

⋮

AN in DATA + 3N + 7

If A is a negative floating point number then the absolute value of A is set in location DATA + 2N + 8 and used for all A_i $(i = 1, 2, \ldots, N)$. If A is zero then location DATA + 2N + 8 is set to zero and A is ignored in the halfing and doubling option.

DATA+2 contains a positive floating point number E used in the truncation error test in the predictor-corrector variable Δt mode. If E is set to 10^{-h} approximately h significant figure are asked for in the truncation error test. $(10^{-1} \leq E \leq 10^{-8})$ is the suggested range of E.

DATA+3 contains a positive floating point number called MINIMUM DT. If in the truncation error test, after a step Δt, one or more of the variables doesn't meet the convergence test and Δt is less than or equal to MINIMUM DT, the differential equation routine does a return jump to the users subroutine starting at TMIN. The location of TMIN is given in the calling sequence. The user may stop and do some checking or do an unconditional jump to TMIN to return to the differential equation routine. The differential equation after the return accepts the step Δt ignoring the failure of the convergence test and continues the integration.

DATA+4 contains the positive floating point number MAXIMUM DT. If in the truncation error test all the variables have converged so well that doubling is indicated but ΔT is already greater or equal to MAXIMUM DT then ΔT is not doubled and the next integration step is still ΔT.

DATA+5 contains a positive fixed point integer N, the number of differential equations to be solved.

DATA+6 contains the floating point number DELTA T. In the fixed ΔT mode the whole integration is done with the fixed integration step DELTA T. In the halfing and doubling mode the initial trial step is ΔT but if the convergence test fails, the initial step will be redone at half ΔT and this halfing will continue until the convergence test is passed. If the convergence test indicates doubling the next step made will be $2\Delta T$. If on entering the differential equation routine DELTA T is zero or negative the AC is set to the binary integer + 1 and an unconditional jump is made to the error return BETA + 4.

DATA+7 is the location of independent variable t. The user must set up an initial value of t (a floating point number) that can be negative, zero, or positive. The differential equation will advance t by some Δt during each integration step.

DATA+8 is the beginning of the N locations containing the dependent variable x_1 through x_N. The user sets them up to their initial values at the start of a problem. The differential equation integrates them with step Δt and replaces them with their new values.

x_1 is in location DATA + 8

x_2 is in location DATA + 9

. . . .

x_N is in location DATA + N + 7

DATA + N + 8 is the beginning of the N locations of the derivatives of x_i or $f_i(t, \dot{x}_1, x_2, \ldots, x_N)$. The user must code a subroutine starting at location DERIV (given in the calling sequence) to calculate the derivatives of x_i using the values of t, x_1, x_2, through x_N in their DATA locations and then store the derivatives in their locations in the DATA region. As the DERIV subroutine is entered by a return jump from the differential equations routine, an unconditional jump to location DERIV gives the return to the differential equations routine so it can continue the solution. The DATA locations of f_i, or the derivatives are:

$\frac{dx_1}{dt} = f_1$ is in location DATA + N + 8

$\frac{dx_2}{dt} = f_2$ is in location DATA + N + 9

$\frac{dx_N}{dt} = f_N$ is in location DATA + 2N + 7

DATA + 2N + 8 is the beginning of the N locations of A_i. These are the last locations used in the DATA region. A description of the A_i's has already been given under the discussion of DATA + 1.

DERIV is the beginning of a subroutine to be coded by the user. The user's DERIV subroutine function uses the variables t and x_i in the DATA regions to calculate the derivatives, f_i, and stores them in the DATA region. The location of the variables in the DATA region have been described before, but to repeat:

t is in DATA + 7

x_1 is in DATA + 8

x_2 is in DATA + 9

. . . .

x_N is in DATA + N + 7

f_1 is in DATA + N + 8

f_2 is in DATA + N + 9

. . . .

f_N is in DATA + 2N + 7

After calculating and storing the f_i's the DERIV routine does an unconditional jump to DERIV which causes a return to the differential equations routine so it can continue the solution. No printing should be done in this routine as the x_i's in the DATA region may be just preliminary estimates.

EXIT is the beginning of a subroutine coded by the user to perform printing. When t and the x_i's have been integrated a step Δt and the x_i's have satisfied the convergence conditions, the differential equation routine goes to the DERIV subroutine which calculates the derivatives of the new x_i's. The differential equation routine then executes a return jump instruction, SLJ 4 EXIT, to the users subroutine for possible printing. The user's subroutine can just bump a counter and do printing just every k steps or can print at each step. The user can take selected values of t, x_i and the derivatives of x_i from the DATA region, or calculate functions of these variables and print them or save them for future interpolation.

If the Generalized Listable Output Routine is used the user should remove the selection of interupt on arithmetic fault before printing as the output routine often causes arithmetic faults not related to computational errors. After printing, a clear arithmetic faults instruction should be given to clear possible arithmetic faults from the output routine. At the end of the EXIT subroutine the user does an unconditional jump to EXIT to return control to the differential equation routine.

TEXIT and T are set up by the user to get points on the solution of the differential equation for specified values of the independent variable t. The way it works is if the independent variable t is equal to or greater than the contents of location T, the differential equation routine will do a return jump to TEXIT with the contents of location T minus the independent variable t in the AC. In using this feature the contents of location T should be always set greater than the independent variable t. Because if the next integration step will make t larger than the contents of location T the differential equation routine does a special Runge-Kutta integration step with a value of Δt to advance t and calculate x_i and the derivatives of x_i. Δt is such that t is equal to the contents of location T. After this special Δt step the old Δt is restored and the differential equation routine does a return jump, SLJ 4 TEXIT, to the user's TEXIT subroutine and as t equals the contents of location T the AC is zero. The user's subroutine, TEXIT, may do printing, keeping in mind the suggestions given in the EXIT subroutine. It should then advance the contents of T to the next break point. To exit, an unconditional jump to TEXIT returns to the differential equation routine.

If the contents of T are less than the independent variable t because of improper updating or some other oversight, the differential equation routine still executes the instruction, SLJ 4 TEXIT, but with the AC minus to show that the break point has been passed in t. The break point features of the differential equation routine will be ignored if T or the contents of location T are set to zero, then no jump to TEXIT will ever occur. If both T and the contents of location T are nonzero, but TEXIT is zero, the differential equations routine will go to its error return with a fixed integer -1 in the AC because no subroutine can start at zero.

If, when t equals the contents of location T the formulas to compute the derivatives of x are changed or it is wished to change some of the parameters in the DATA region, the solution of the differential equation must be restarted by either going to a new calling sequence or back to the start of the old calling sequence.

If t is at the end of a case, the next case can be set up and the differential equation routine restarted with a calling sequence. Only in the user's EXIT and TEXIT subroutines can the values of the variables be trusted either for printing or restarting the solution by going to a calling sequence.

BETA + 4 is the error return for the differential equation routine. An unconditional jump is made to BETA + 4, with a fixed point binary integer in the AC which tells the type of error, if a parameter is obviously bad. To correct this, the bad parameter should be changed and the case rerun. When at location BETA + 4, if:

AC is 0: then N, the number of equations to be solved, (location DATA + 5) has been set to zero, a negative number, or is greater than 200.

AC is +1: then Δt (location DATA + 6) is either zero, negative, or not in normalized floating point form.

AC is +2: then the integration code in location DATA is either negative or greater than + 3.

AC is -1: Both T and the contents of T are nonzero, but TEXIT is zero. This means the exact t break feature is being used but the users subroutine, TEXIT, starts at location zero. This is invalid.

COMMON is a block of 14N + 5 locations starting at COMMON that the user must reserve for the differential equation routine. When command has been transferred to the users EXIT subroutine the first N location of COMMON contain the predictor values of the new x_i. The truncation error of the last step for the variable x_i is about $1/14\,(x_{p,i}-x_i)$ in magnitude. The locations of the variables $x_{p,i}$ and x_i are:

$x_{p,1}$ is in location COMMON

$x_{p,2}$ is in location COMMON + 1

. . . .

$x_{p,N}$ is in location COMMON + N-1

If the user needs the values of t and the variables x_i of the last step for interpolation they can be found in the locations:

t in COMMON + N
x_1 in COMMON + N + 1
x_2 in COMMON + N + 2

. . . .

x_N in COMMON + 2N

As already described in the writeup, the current x_i's are in the DATA locations:

x_1 in DATA + 8
x_2 in DATA + 9

. . . .

x_N in DATA + N + 7

TMIN is the first location of a subroutine coded by the user that the differential equation goes to if the convergence test has failed and Δt is less than or equal to MINIMUM DT.

Its use is discussed later in this writeup.

3. SPACE REQUIRED - is about 700 locations not including the 610 locations of the Generalized Listable Output routine.

4. TEMPORARY STORAGE - none. The two blocks of storage DATA (3N+ 8 locations) and COMMON (14 N+ 5 locations) cannot be used by the programer to store numbers.

7. ERROR STOPS - the error return is described before. It is in location BETA + 4 in the calling sequence.

9. PRINT INFORMATION - If the print option is used the "General Listable Output Routine" must be in the computer and its symbol location, OUTPUT, in 70412B. At the head of the assembly is the card

   OUTPUT EQU 70412B

   The symbolic location ADTAPE, in the differential equation routine determines the channel number of the output tape, the tape number, and 1607 number. The print option sets up words for the write BCD tape calling sequence by using ADTAPE after inserting the proper carriage control in the lower address. Therefore, to change the channel number, tape number, or 1607 number, just modify the contents of ADTAPE in the differential equation routine. At present:

   CN or channel number of the output tape is 4

   TN or tape number of the output tape is 4

   UN or 1607 number is 2

D. METHOD

To compute the change of x_i with the integration step Δt the fourth order Runge-Kutta method gives the formulas:

$$k_{1,i} = (\Delta t)\, f_i\,(t, x_1, x_2 \ldots x_n)$$

$$k_{2,i} = (\Delta t)\, f_i\,(t + ½\Delta t, x_1 + ½\, k_{1,1}, \ldots, x_n + ½\, k_{1,n})$$

$$k_{3,i} = (\Delta t)\, f_i\,(t + ½\Delta t, x_1 + ½\, k_{2,1}, \ldots, x_n + ½\, k_{2,n})$$

$$k_{4,i} = (\Delta t)\, f_i\,(t + \Delta t, x_1 + k_{3,1}, \ldots, x_n + k_{3,n})$$

$$\Delta x_i = \frac{1}{6}(k_{1,i} + 2k_{2,i} + 2k_{3,i} + k_{4,i})$$

The calculation: $x_i^c\,(t + \Delta t) = x_i(t) + \Delta x_i$ is done in double precision to help control rounding errors. All other calculations are performed in single precision. Programers at STL didn't feel the Gill or other modifications to control rounding errors to be of much value so the standard Runge-Kutta method was programed.

The Adams-Moulton method requires the derivatives of x_i three equal steps behind. To start the Adams-Moulton integration, several Runge-Kutta integrations are required. The Adams-Moulton predictor-corrector formulas are:

$$x^p_{i,\ k+1} = x_{i,k} + \frac{\Delta t}{24}(55\, f_{i,k} - 59\, f_{i,k-1} + 37 f_{i,k-2} - 9\, f_{i,k-3})$$

$$\Delta x_{i,k} = \frac{\Delta t}{24}\ (9\, f^p_{i,k+1} + 19\, f_{i,k} - 5\, f_{i,k-1} + f_{i,k-2})$$

Again the calculation $x^c_{i,k+1} = x_{i,k+1} + \Delta x_{i,k}$ is done in double precision to help control rounding errors.

Both Adams-Moulton and Runge-Kutta have error terms of the order $(\Delta t)^5$ but the Runge-Kutta step requires four references to the derivatives against just two references in an Adams-Moulton step; so for most problems the Adams-Moulton integration method is to be preferred.

In integrating a differential equation numerically, it is replaced with a difference equation and we solve this difference equation. If the integration steps used are small and an integration scheme like

Adams-Moulton or Runge-Kutta which have favorable stability properties is used, the solution of the difference equation is usually close to the solution of the differential equation. However, if some of the variables x_i given by the differential equation routine have either odd oscillations or increase very rapidly with the physical problem not suggesting such behavior, the solution is probably unstable and greatly in error. In long problems with many integration steps a slight instability will cause the solution to slowly drift from the true solution of the differential equation. An unstable solution can often be made stable by integrating with a smaller step Δt or by forcing a smaller step by asking for more accuracy in the predictor-corrector mode. The reason for this is that different step sizes change the parameters relating to the stability of the difference equation. Therefore, if two solutions with different step sizes are similar and close together, both solutions can be accepted as correct. The routine also makes it easy to do checking by running the same case over using both the Runge-Kutta and Adams-Moulton integration methods. Usually the truncation error requirements in solving a set of differential equations dictate integration steps sufficiently small to insure stability.

HALFING AND DOUBLING MODE

When doing an Adams-Moulton integration step Δt to advance the dependent variables from $x_i(t)$ to $x_i(t + \Delta t)$ for each variable, we first calculate the predicted value $x_i^p(t + \Delta t)$ for each variable. The users DERIV routine is used to calculate the predictor derivatives, $f_i(t + \Delta t, x_1^p(t + \Delta t), \ldots, x_n^p(t + \Delta t))$. Using the predicted derivatives the step is finished by calculating $x_i^c(t + \Delta t)$ the corrector. An estimate of the magnitude of the truncation error of each variable in the last step is:

$$\frac{1}{14} \left| x_i^c(t + \Delta t) - x_i^p(t + \Delta t) \right|$$

To get an estimate of the truncation error in the Runge-Kutta mode we first do a Runge-Kutta integration step of $2\Delta t$ to get the predictor, $x_i^p(t+2\Delta t)$. Then restarting with the variables $x_i(t)$ we do two Runge-Kutta integration steps each of length Δt to get the corrector, $x_i(t+2\Delta t)$. An estimate of the magnitude of the truncation error for each variable in the step from $x_i(t)$ to $x_i(t+2\Delta t)$ is:

$$\frac{1}{15}\left|x_i^c(t+2\Delta t) - x_i^p(t+2\Delta t)\right| .$$

Going from $x_i(t)$ to $x_i(t+2\Delta t)$ requires 4 references to the derivatives using the Adams-Moulton integration scheme and 12 references to the derivatives using the Runge-Kutta scheme.

The convergence test uses the parameters A_i and E described before in the writeup of the DATA region. Using the predictor, x_i^p, and corrector, x_i^c, of each variable from the last integration step (for either the Runge-Kutta or Adams-Moulton method) if the inequality:

$$\frac{\left|x_i^c - \left\|x_i^p\right|\right\|}{\max\left[A_i, \left|x_i^p\right|, \left|x_i^c\right|\right]} < E$$

is satisfied for all i (i = 1, 2, . . . , N) then we say the convergence test has been passed and the results of the last integration step will be accepted by the differential equation routine. If the inequality doesn't hold for any one of the i, then the convergence test has failed and the results of the last integration step are not accepted. If E in the above inequality is replaced by E/M, "where M is in symbolic location ADC + 2 and set to the value 20 in floating point format", and the above inequality is still satisfied for every i, we say the convergence test to double has passed. Again, if when E is replaced by E/M in the above inequality and the inequality doesn't hold for any one of the i we say the convergence test to double has failed.

If the convergence test is satisfied but the convergence test to double has failed, then the integration step Δt is left unchanged and the differential equation routine goes to users subroutine EXIT for possible printing. Upon return to the differential equation routine it integrates the variables again by the same step Δt.

If the convergence test to double is satisfied the differential routine goes to the users subroutine EXIT for possible printing of the accepted x_i^c but on return a set up may be made to make the next integration step $2\Delta t$. Even though the convergence test to double is satisfied, sometimes Δt is not doubled. For instance, if Δt is already greater than or equal to the number, MAXIMUM DT in DATA + 4, then Δt is not doubled. If Δt was halfed within the last four steps in the Runge-Kutta mode, controlled by the fixed point binary number 4 in symbolic location ADC, no doubling of Δt is done or if Δt was halfed within the last six steps in the Adams-Moulton mode, controlled by the fixed point binary number 6 in symbolic location ADC + 1, no doubling is done. The above delays are inserted to save machine time that might be wasted in halfing and doubling oscillations.

In the Runge-Kutta mode doubling can take place every integration step but in the Adams-Moulton mode sufficient delays may not exist to do the next integration step at $2\Delta t$. To integrate with the next step $2\Delta t$; first $x_i(t + \Delta t)$ becomes $x_i(t^*)$, where $t^* = t + \Delta t$ the advanced independent variable, and $\dot{x}_i\ (t-5\Delta t)$ becomes $\dot{x}_i(t^*-3(2\Delta t))$. Therefore, after any halfing or doubling operation in the Adams-Moulton mode Δt cannot be doubled until 3 integration steps of the same length are made to get the delays needed for the next integration step $2\Delta t$.

If the convergence test has failed Δt is usually halfed. The only case where Δt is not halfed, is if Δt is less than or equal to the floating point number, MINIMUM DT set up by the user in location DATA + 3. As described before in this writeup then Δt is supposed to half but it is already less than or equal to MINIMUM DT. If a return jump is made to the users subroutine TMIN and if the user returns to the differential equation routine by an unconditional jump to TMIN, then $x_i(t + \Delta t)$ is accepted and the next integration step will also be Δt.

If the convergence test has failed and Δt is greater than MINIMUM DT the $x_i(t + \Delta t)$ in the Adams-Moulton mode and $x_i(t + 2\Delta t)$ in the Runge-Kutta mode are not accepted and the differential equation routine goes back to variables $x_i(t)$ and the next integration step is $.5\Delta t$. In the Runge-Kutta mode the differential equation routine restores the old $x_i(t)$ and its derivatives it had saved in COMMON storage and takes the saved result of the first RK step $x_i(t + \Delta t)$ and makes it the new predictor. It then enters the RK predictor-corrector mode at the point it computes the two Runge-Kutta steps, now each of length $.5\Delta t$ to get the corrector.

In the Adams-Moulton mode after Δt is halfed the derivatives of $x_i(t - .5\Delta t)$ and $x_i(t - 1.5\Delta t)$ are needed for the next integration step of length $.5\Delta t$. Using $\dot{x}_1(t)$, $\dot{x}_i(t - \Delta t)$, $\dot{x}_i(t-2\Delta t)$, $\dot{x}_i(t-3\Delta t)$, and $\dot{x}_i(t-4\Delta t)$ in the five point interpolation formula of Lagrange; $\dot{x}_i(t -.5\Delta t)$ and $\dot{x}_i(t-1.5\Delta t)$ are computed. The five point formula was used as its error term is of the same order $(\Delta t)^5$ as the error terms in the Runge-Kutta and Adams-Moulton integration methods with integration steps of length Δt. After interpolation, the routine restores $x_i(t)$ and returns to the Adams-Moulton mode in the routine to the part where the Adams-Moulton step will be done over with the new step $.5\Delta t$. As referred to before, no doubling of Δt will take place after a halfing in Adams-Moulton mode until sufficient delays exist for doubling Δt. Tags were set to delay 4 steps in the Runge-Kutta mode and 6 steps in the Adams-Moulton mode to save machine time.

The five point Lagrange Interpolation Formulas came from page 118 of "Introduction to Numerical Analysis" by F. Hildebrand. In the same book can be found the formulas of Runge-Kutta on page 237 and Adams-Moulton on page 200.

SETTING PARAMETERS

In solving a set of differential equations we are interested in keeping the error at all points of the solution within certain specified limits. However, by setting the parameters A_i and E, the truncation error is only controlled for each single step in the predictor-corrector mode. The total error is a combination of the truncation and rounding errors of many steps. The truncation error per step for the variable x_i is less then $\frac{1}{14} E x_i$ if the absolute value of x_i is greater than the value of A_i or less than $\frac{1}{14} E A_i$ if A_i is larger of the two. As most problems are only a few thousand integration steps and are well behaved, the truncation error per step is suggested to be set to one fiftieth of the total error allowed.

Two examples for setting A and E are:

1) Integrate $\dot{x} = \cos t$ with truncation error per step less than 10^{-5}. As the truncation error requirement is independent of the magnitude of the variable x, set A to 1 which is equal to or greater than the absolute value of x at any time.
As $\text{MAX}\ (A, |x_i^c|, |x_i^p|) = 1.$ set $E = 14\,(10^{-5})$.

2) Integrate $\dot{x} = e^t$, where $x(o) = 1.$, with truncation error per step less than $x(10^{-5})$. As the truncation error requirement per step is some proportion of the variable x set A to zero. As $\text{MAX}\ (A, |x_i^c|, |x_i^p|) = x_i^c$ set $E = 14(10^{-5})$. If when integrating the last case to t_f seconds we set $A = e^{t_f}$; as $\text{MAX}\ (A, |x_i^c|, |x_i^p|) = A = e^{t_f}$ we would have large Δt steps at the beginning resulting in great relative errors that would be carried through the whole solution.

To avoid unnecessary halfing of Δt; each value of A_i should be set slightly less than the average magnitude of its corresponding variable x_i, but, if the solution tends to be inaccurate or unstable when some of the variables are small the value of A_i corresponding to these variables must be decreased.

When trying to see the effects of the range of a parameter to check for the total error of an integration; change the parameter at least by a factor of 10 to make sure it reruns with a different step size and change E or some of the A_i's only to isolate the effect. When running a problem the user is not too familiar with; a typical case, or the extreme cases plus a few middle cases, should be run with different values of the parameters A_i and E. Then after comparing the solutions of the same case which has been run with different parameters A_i and E, run the rest of the cases with the values of the parameters A_i and E that gave satisfactory total error bounds. Sometimes when E is decreased the greater number of integration steps can increase the total rounding error until it is so much greater than the total truncation error that the solution with the smaller E has greater total error than the solution with the larger E; however, as the variables x_i are accumulated in double precision and since the Control Data computer has a 36 bit mantissa then only when E becomes less than 10^{-8} are the total errors likely to increase when E is further decreased.

When solving some problems a part of the solution may have very small truncation errors; then Δt can become so large that the solution becomes unstable. The parameter MAXIMUM DT, in DATA + 4, is used to prevent Δt from becoming too large. As an example, when a trajectory was being calculated in a region of very thin air, integration steps of 15 seconds passed the truncation error test but the solution soon became unstable and did false large oscillations. By setting a limit on the integration step to a few seconds the rerun solution was stable.

BREAK POINTS

As described before in the writeup the contents of T and TEXIT can be set up by the user to get points of the solution of the differential equation for specified values of the independent variable t.

One use of this feature is to do printing every h seconds of t. Initially, the contents of T are set to h. Then the k-th time the differential equation routine does a return jump to the user's TEXIT subroutine the user prints the solution, where $t = kh$, then bumps location T by h and does a return jump to the differential equation routine so it can calculate to the next t break, $t = (k+1)h$.

Another use of this feature is when starting a solution, if some of the variables are initially equal to zero, a different set of parameters is required for a small value of t. At the start of the problem the contents of T are set to a value t_p, slightly greater than the initial value of t and the starting parameters are set up. When t equals t_p the users TEXIT subroutine is used to modify the parameters.

When just the contents of T or any of the convergence parameters like E, A_i, MINIMUM DT, or MAXIMUM DT are changed in the middle of a case by a user's subroutine the users normal return, which is an unconditional jump to the beginning of his subroutine may be used in returning to the differential equation routine. However, almost any other kind of change in a users subroutine requires a jump back to the beginning of the calling sequence or a new calling sequence. Such changes by the user requiring a return to a calling sequence are: if the calling sequence is to be modified, either N, code, or Δt is changed, changes of the variable t or x_i. This is necessary because parameter setups from the calling sequence are done only at the beginning of the routine and the logic of the differential equation routine depends on the code to tell where it has stored the variables and delays of the derivatives of x_i in COMMON. The delays are also assumed to be integral multiples of the present Δt in DATA + 6.

To return to examples in the use of the time interrupt feature. One way to start solutions where some of the initial values of the variables x_i are zero is to run in the fixed Δt mode with a small

Δt until the solution is started. Then, after the solution is assumed started at time t_p change to the predictor-corrector mode by modifying the code, and restart the solution at $t = t_p$ by entering a new calling sequence. At the same time TEXIT could be changed to a new TEXIT subroutine that does printing. To accurately simulate a missile in the velocity range of zero to a few hundred feet per second when it is under the influence of torques that change its direction (like crosswinds or controllers) requires small integration steps of the order .01 second or large errors in the orientation of the missile occur from the integration process.

The last use of time interrupt discussed is when there are discontinuities of the variables or their derivatives at specified times. An example is when a rocket engine burns out at tp. By setting tp in location T then the user in his TEXIT routine must modify his DERIV routine to omit the thrust term and the change the drag calculations. As such changes cause discontinuities in the derivatives of x_i the differential equation routine should be restarted by going to a new calling sequence.

Another case similar to the above is in the staging of missiles, there should be a break when a missile separates into two sections, a break at the end of free flight of the second stage, and at the time of burnout of the rocket engine; whose firing terminated the free flight of the second stage. When controllers are turned on and off they can cause discontinuities in the derivatives of the variables x_i or even in x_i if the controllers are impulse functions, so the differential equation routine should be restarted by entering a calling sequence at such times. When interpolating through nonsmooth functions with discontinuous derivatives to maintain accuracy the differential equation routine halfs Δt many times using a great deal of machine time; it may also cause an exit to the users TMIN subroutine because Δt is less than or equal to MINIMUM DT and the error requirements are not satisfied. In runs with controllers even though the errors in interpolating variables through nonsmooth functions may be small in terms of the magnitudes of the variables

x_i, as the error term driving the controller is a function of the difference of some ideal x_i and actual x_i, this error may jump several hundred percent making the rest of the controller or guidance simulation useless.

USERS INTERPOLATIONS

The user may require other breaks than time breaks. He may wish to print the output every 1000 feet of altitude or turn on or off a controller when the error, which is a function of the x_i's, becomes equal to some critical value. To do this the user could save several values of the functions at different times and for some interpolation formulas also the derivatives of the functions and do an inverse interpolation to find the time when the function of the error reached the critical point. The interpolation formula used should have an error term of the order $(\Delta t)^5$ to have about the same accuracy as the differential equation routine.

After the time of the break t (break), is known by inverse interpolation the user can interpolate for all the other variables, if he has saved enough delays, then change the method of calculating the derivatives to include the new status of the controller and restart the differential equation routine by going to a calling sequence. If the user hasn't saved enough delays of the variables to do sufficiently accurate interpolation, he could let the differential equation routine do the interpolation. The user is in his EXIT routine, the time of break is between the current t in DATA + 7 and the t that was in DATA + 7 at the time of the last entry into the EXIT routine as the break occurred between steps. The user first moves the old t and the old variable x_i from COMMON in this way:

old t from COMMON + N to t in DATA + 7

old x_1 from COMMON + N + 1 to x_1 in DATA + 8

. . . .

old x_n from COMMON + 2N to x_n in DATA + N + 7

Then set the new ΔT = t(BREAK) - t(old) in DATA + 6 and change the code to zero, by setting zero in location DATA, and go to a calling sequence to restart the differential equations routine. The first return to users EXIT will be with t = t (old) but in the second return to EXIT the variables and x_i will be advanced to the break point. The user then makes the necessary changes in his DERIV subroutine that happen at the break point, restores the old code and Δt and goes to a calling sequence to restart the differential equation.

The user shouldn't turn on the controller or ignite the rocket motor in the simulations until he has all his variables interpolated at the break time or he will have serious errors in his interpolations because of the nonsmooth functions introduced. If the user hasn't saved delays of the function that determines the break point he could calculate the old value of the function from t and the variables x_i in COMMON + N through COMMON + 2N and do a linear interpolation for the break point. Then advance t and the variables x_i to the estimated break from linear interpolation by a Runge-Kutta step as before. Then the estimated t break from linear interpolation and the new variable x_i with a new interpolation can be used to get a better estimate of the break time and this process repeated until the break point is calculated to sufficient accuracy. The user should note the process will iterate closer and closer to the break point but may not go past it so the exit from above process should be done when an estimate of the t (break) is close to the actual t (break).

TMIN SUBROUTINE

If in the predictor-corrector mode one of the variables has failed the convergence test and ΔT is less than or equal to MINIMUM DT the differential equation routine does a return jump to the users subroutine TMIN.

If the user has a problem he knows can be integrated to sufficient accuracy using a step, ΔT, he can set MINIMUM DT to ΔT and in the TMIN subroutine do an unconditional jump to TMIN to cause the return to the differential equation. The differential equation routine accepts the last step integrated with the last ΔT and also does the next step with the same ΔT. Therefore, machine time is not wasted with a ΔT much smaller than ΔT and if in other regions of the problem, where truncation error is small ΔT may be much larger than ΔT. The regions of small ΔT could be caused by slight discontinuities in curve fits of the empirical data that determine the coefficients of the differential equation or the values of A_i are too small.

If the user has a problem that isn't too familiar MINIMUM DT should be set very small. Then the TMIN subroutine should stop the problem so it can be examined. The user may find derivatives changing rapidly or which are very large because they are not calculated correctly or an unnoticed singular [illegible] is making a derivative infinite. Also the user may be changing t[illegible]iables because of program errors, or if the differential equation has impulse functions the variables were changed but the differential equation routine wasn't restarted. If the value of MINIMUM DT was only a factor of 10 smaller than the ΔT that runs most of the problem accurately, an exit to TMIN could mean, discontinuous derivatives being integrated through a point where the differential equation routine should have been restarted. Another possibility with MINIMUM DT a factor of 10 smaller than the standard DT, if in TMIN when some of the variables are small; the values of some of the A_i could be increased and the problem reran with both a smaller MINIMUM DT and the old A_i's and the old MINIMUM DT and the larger A_i's and the solution compared to see if the shorter machine time in running with larger A_i's gives sufficiently accurate solutions.

RUNNING A PROBLEM IN REVERSE

If a user wishes to run a problem backwards in time, as Δt cannot be made negative in this routine, the method described below must be used.

Given the standard set of N simultaneous differentiai equations to be integrated we have:

$$\frac{dx_i}{dt} = f_i \left[t, x_1(t), x_2(t), \ldots x_N(t)\right]$$

Let the initial value of t be t_o and its final value be t_f. To run this same set of differential equations backwards we make the transformation $T = t_f - t$, to the new independent variable T. Let the initial value of T be 0, then $t = t_f$ and the initial values of the variables x_i is $x_i(t_f)$. Then the final value of T is $t_f - t_o$ and this corresponds to $t = t_o$ and the final value of the variables x_i is $x_i(t_o)$. The derivatives of the variables x_i with respect to T are:

$$\frac{dx_i}{dT} = -f_i \left[t_f - T, x_1(t_f - T), x_2(t_f - T), \ldots x_N(t_f - T)\right]$$

as $t = t_f - T$ and $dT = -dt$.

E. TEST CASE

To help clarify the writeup, a simple test case is inserted. I am integrating the three equations:

$$\frac{dx_1}{dt} = x_2$$

$$\frac{dx_2}{dt} = -4x_1$$

$$\frac{dx_3}{dt} = 2x_3$$

Let $x_1(0) = 0$, $x_2(0) = 2$, and $x_3(0) = 1$. The solutions of the above differential equations are:

$$x_1 = \sin 2t$$

$$x_2 = 2 \cos 2t$$

$$x_3 = e^{2t}$$

Using the T break feature printing is done every second. At t equal one second.

$x_1 = \sin 2 = .909297427$ the routine gives .9092974248

$x_2 = 2 \cos 2 = -.832293674$ the routine gives .8322936859

$x_3 = e^{2t} = 7.38905610$ the routine gives 7.389056142

The last printout is $t = 5$ seconds.

$x_1 = \sin 10 = -.544021111$ the routine gives -.544021143

$x_2 = 2 \cos 10 = -1.678143058$ the routine gives 1.678143027

$x_3 = e^{10} = 22026.4658$ the routine gives 22026.46649

TEST CASE CODING

```
         ORG     10000
OUTPUT   EQU     70412B
ADAMS    EQU     3720B
TEST     LDA     ONE
         STA     T           FIRST  T BREAK IS 1.
         STA     DATA+10     X3 (0) IS 1.
         LDA     TWO
         STA     DATA+9      X2(0) IS 2.
         ENA     0
         STA     DATA+7      +0 IS ZERO
         STA     DATA+8      X1(0) IS 1.
BETA     SLJ   4 ADAMS       CALLING SEQUENCE
         ZRO   1 DERIV       B  NOT ZERO SO PRINT
         ZRO     TEXIT       INITIAL CONDITIONS
         ZRO     T
+        ZRO     DATA
         ZRO     COMMON
+        ZRO     EXIT
         ZRO     TMIN
A2       SLS     A2          ERROR RETURN
ONE      DEC     1.          1.
TWO      DEC     2.          2.
SIX      DEC     6.          7.
M FOUR   DEC     -4.         -4.
DATA     DEC     3           CODE IS 3
         DEC     1.          A  IN DATA+1  IS PLUS
         DEC     1.D-8       E IN DATA+2 IS .000001
         DEC     3.D-4       MIN DT IN DATA+3 IS .0003
         DEC     1.          MAX DT IN DATA+4 IS 1.
```

```
              DEC      3                      N IS 3 IN DATA+5
              DEC      5.D-3                  INITIAL DT IS .005
              DEC      0,0,0,0,0,0,0          + X1, X2, X3, and DERIV X1, X2, X3
              DEC      .1,.1,1.               A1, A2, A3
COMMON        BSS      47                     14 W+5 IS 47
T             BSS      1                      T BREAK
TMIN          SLS      TMIN                   STOP IF DT TOO SMALL
COUNT         BSS      1
DERIV         SLJ      0
              LDA      DATA+9
              STA      DATA+11                DERIV X1 IS X2
              LDA      M FOUR
              FMU      DATA+8
              STA      DATA+12                DERIV X2 IS -4 X 1
              LDA      DATA+10
              FAD      DATA+10
              STA      DATA+13                2X3 IS DERIV X3
              SLJ      DERIV
EXIT          SLJ    0                        EXIT EACH DT STEP
              SLJ      EXIT
TEXIT         SLJ      0
              SLJ    4 OUTPUT                 PRINT TITLE
A3            SLS      A3
+             03       BCD                    SUBSCRIPT
              00       2021
+             03       BCD+2
              00       1029                   X
+             03       BCD+3
              00       1047                   DERIV X
+             03       BCD+4
              00       1070                   T
+             01     4 4
              00     2 10                     PRINT TITLE
```

```
        ENA       1
        STA       COUNT
        ENI    1  0
LOOP    SLJ    4  OUTPUT
A4      SLS       A4
+       04        COUNT          COUNT 1 TO 3
        00        10
+       06     1  DATA+8
        00        10030          X1
+       06     1  DATA+11
        00        10050          DERIV X1
+       06        DATA+7
        00        10070          TIME
+       01     4  4
        00     2  16             PRINT X, DERIV X, AND T
        RAO       COUNT          EVERY SEC
+       ISK    1  2              BUMP 1
        SLJ       LOOP
        LDA       T              END PRINT
        FAD       ONE
        STA       T
        FSB       SIX
        AJP    M  TEXIT
STOP    SLS       STOP           IF DATA+7 OR T IS 7 STOP
BCD     BCD       2SUBSCRIPT
        BCD       1X
        BCD       1 DERIV X
        BCD       1 T
        END       TEST
```

TEST CASE OUTPUT

CODE		A	E	MIN DT	MAX DT	N	DELTA T
3	.10	+1	.10 -9	.30 -3	.10 +1	3	.5000000000 -2

SUBSCRIPT	X		DERIV X	A	T
1		0	.2000000000 +1	.1000000000 + 0	0
2	.2000000000	+1	0	.1000000000 + 0	0
3	.1000000000	+1	.2000000000 +1	.1000000000 + 1	0

SUBSCRIPT	X		DERIV X	T
1	.9092974248	+0	-.8322924574 +0	.1000000000 + 1
2	-.8322936859	+0	-.3637190736 +1	.1000000000 + 1
3	.7389056142	+1	.1477812185 +2	.1000000000 + 1
		0		

SUBSCRIPT	X		DERIV X	T
1	-.7568025058	+0	-.1307288205 +1	.2000000000 + 1
2	-.1307287223	+1	.3027208220 +1	.2000000000 + 1
3	.5459815071	+2	.1091963721 +3	.2000000000 + 1

SUBSCRIPT	X		DERIV X	T
1	-.2794154782	+0	.1920340542 +1	.3000000000 + 1
2	.1920340590	+1	.1117662231 +1	.3000000000 + 1
3	.4034288012	+3	.8068576686 +3	.3000000000 + 1

SUBSCRIPT	X		DERIV X	T
1	.9854519960	+0	-.2909987991 +0	.4000000000 + 1
2	-.2910001235	+0	-.3957433293 +1	.4000000000 + 1
3	.2980958060	+4	.5961919977 +4	.4000000000 + 1

SUBSCRIPT	X		DERIV X	T
1	-.5440211430	+0	-.1678143115 +1	.5000000000 + 1
2	-.1678143027	+1	.2176084290 +1	.5000000000 + 1
3	.2202646649	+5	.4405293659 +5	.5000000000 + 1

f. INTERPF.

A. IDENTIFICATION

TITLE:	Divided Difference Interpolation or Extrapolation Routine
CATEGORY:	Mathematical Subroutine
PROGRAMER:	Sanford Elkin, William Silverman, and Ed Fleming
MODIFIED:	June 1961

B. PURPOSE:

Given a table of M values in floating point which define a function, and given a floating point argument X, this routine approximated the functional value Y of that argument by a polynomial interpolation or extrapolation.

C. USAGE:

1. Calling sequence:

CALL INTERPF (X, M, N, a, b)

Interpolate, find Y for X, where various X vs. Y given in table.	X	Flt argument
	M	Size of the table
	N	Degree of interpolation
	a	Table of X values
	b	Table of Y values

Where M is in fixed integer form, N is the order of interpolation or extrapolation desired — in fixed integer form.

The X's and Y's must be in floating format and the table of X values must be stored in decreasing order.

D. MATHEMATICAL METHOD: (reference 19)

$$Y = I_{0,1,\ldots,n}(x) = \frac{1}{x_n - x_0} \begin{vmatrix} I_{0,1,\ldots,n-1}(x) & x_0 - x \\ I_{1,2,\ldots,n}(x) & x_n - x \end{vmatrix}$$

$Y = I_{0,1,\ldots,n}(x)$ is equivalent to Lagrange's Formula

$$Y = \psi(x) = \frac{(x - x_1)(x - x_2)\ldots(x - x_n)}{(x_0 - x_1)(x_0 - x_2)\ldots(x_0 - x_n)} Y_0$$

$$+ \frac{(x - x_0)(x - x_2)\ldots(x - x_n)}{(x_1 - x_0)(x_1 - x_2)\ldots(x_1 - x_n)} Y_1$$

$$+ \ldots + \frac{(x-x_0)(x-x_1)\ldots(x-x_{n-1})}{(x_n-x_0)(x_n-x_1)\ldots(x_n-x_{n-1})} Y_n$$

which yield a polynomial of degree such that $(x_0) = Y_0$, $\psi(x_1) = y_1$, $\ldots$, $\psi(x_n) = Y_n$, as is easily demonstrated by an induction proof.

When x is within the limits of the table, the points $x_0, x_1, \ldots, x_n$ are spaced about it to best advantage, but if x is outside the limits of the table, the nearest $n+1$ points in the table are used in the formula.

g. RDF.

TITLE:	Read Data, Fixed Formats
ID:	RDF
Classification:	Input Routine
Programers:	W. Silverman and R. E. Mann

PURPOSE:

To load data from Hollerith cards through the 088 card reader, or from 80 character records on BCD tape. The data are on symbolic cards of the types used for the pseudo operations DEC, OCT, and BCD as described in the assembly routine.

USAGE:

1. Calling Sequence

```
            ENA    A
            ENQ    B
α           RTJ    RDF
α+1         NOP    L(LTN)
α+2         Error Return
α+3         Normal Return
```

L(LTN) = Location of Logical Tape Number

LTN (Logical Tape Number) = 1-48 for tape input
= 49 for card input

After a description of the input formats, the function of A and B will be discussed.

2. Input Formats:

Load is controlled by a symbolic operation code in columns 10, 11, 12 of the input cards (or records). There are five permissible operation codes.

1. SLJ--This operation always causes loading to be terminated. A transfer address may appear in decimal or octal

beginning in column 20. The address is assumed decimal if it is followed by a D or blank, and octal if it is followed by a B. If index modification of the transfer address is desired, column 17 or 18 may contain an index register number. Control is transferred to the (modified) transfer address, or if there is no transfer address, control is returned to $\alpha + 3$ in the calling sequence.

2. REM-- This record will be skipped. No conversion or operation results. REM cards may be used as spacers or tags in the data deck.

Conversion operations:

3. BCD--n words of binary coded decimal information beginning in column 21 are loaded into consecutive memory locations. n is in column 20, $1 \leq n \leq 7$. The information runs through column 20 + 8n.

4. OCT--Octal data beginning in column 20 and terminating with the first blank column are interpreted as octal integers, and converted to binary integers. Successive words are separated by commas and loaded into consecutive locations. Each word may consist of a + or - sign and up to 16 octal digits. If no sign appears, a + sign is assumed.

5. DEC--Decimal data beginning in column 20 and terminating with the first blank column are loaded. Successive words are separated by commas and loaded into consecutive locations. Each word may consist of a sign, + or - or none, up to 15 decimal digits with a decimal point if desired, a D or E followed by a signed or unsigned decimal scale factor, and a B followed by a signed or unsigned binary scale factor. Presence of a binary scale factor will cause the number to be loaded as a fixed point decimal number with the binary point to the right of the bit position given by the binary scaling. If a decimal scale factor or decimal point is present, the number will be loaded as a fixed point number as above if

a binary scaling is present, or as a (scaled) floating point number if no binary scaling is present. If a binary scaling is present, it must lie between -47 and 47, and the decimal scaling, if any, must lie between -28 and 28. If no decimal point or scale factor is present, the number will be loaded as an integer. A few illustrative examples follow:

1) 1.2345 will be loaded as floating point 1.2345
2) 12345E-3 will be loaded as floating point 12.345
3) 1.2345D-2 will be loaded as floating point .012345
4) 12345 will be loaded as integer 12345
5) 12345B15 will be loaded as fixed point 12345.0 with binary point to the right of bit position 15.
6) 12.345D2B13 will be loaded as fixed point 1234.5 with binary point to the right of bit position 13.
7) 12345B-3 will be loaded as fixed point 12345.0 with binary point to the right of bit position -3, that is as (rounded) integer 1543 = 12345/8 to the nearest integer.

Any data card BCD, OCT, or DEC may have an absolute numerical address in columns 1-8. The data on the card is loaded relative to that address. The address is treated as octal or decimal according to the same conventions used for the transfer address on the SLJ card.

e. g. The card

```
2000B       BCD    2ABCDEFGHIJKLMNOP
```

will be loaded so that:

```
(2000B) =   ABCDEFGH
(2001B) =   IJKLMNOP
```

A and B (the addresses in the accumulator and quotient registers upon entry to RDF) condition where the data on cards with blank address fields is loaded and the number of words actually loaded by RDF.

If A ≠ 0, the first card with a blank address field is loaded relative to A, the data going into addresses A, A+1, . . . , A+n-1, where n is the number of data words on the card. The data from the next card with a blank address field is loaded relative to A into locations A+n, A+n+1,..., A+n+m-1 where m is the number of data words on the card. And so on, each card with a blank address field being loaded relative to A.

If A = 0, the first card must have an absolute numerical address in columns 1-8. Loading proceeds relative to the last such address. Thus if the first data card has the address 2000B, the second card has a blank address field, the third has the address 3000B, and the remaining data cards have blank address fields, the data from the first cards will go into consecutive address' s starting at 3000B. If B ≠ 0, B-A+1 words will be loaded unless the load is terminated by an SLJ card or an error (see errors below).

If B = 0, an indefinite number of words will be loaded until the routine is terminated by an SLJ card or an error.

3. RDF requires 397 locations

4. RDF uses 19 common erasable storage locations

5. Errors:

   1. End of file on tape or card.

   The end of file flag (77713B) is set non-zero and if no other errors occurred, control is returned to $\alpha + 3$ in the calling sequence.

   2. Parity error on tape.

   The RTT flag (77712B) is set non-zero, and the routine continues, although the conversion of the record on which the parity error occurred may be inaccurate. When the routine is terminated, control is returned to the error return, $\alpha + 2$, in the calling sequence, with the number of records which had parity errors in the upper address of the A register, and the transfer address, if any, in the Q register.

3. Illegal punch on card.

The RTT flag is set with 1's in bits 0 through 39 corresponding to erroneous columns on the card (these may apply to the first or second half of the card, but not both). Control is returned to $\alpha + 2$ in the calling sequence, with the upper address of A set to 1.

4. Format error:

This may be caused by an illegal operation code in columns 10, 11, 12; an illegal address; an illegal character in column 17 or 18 of an SLJ card; or an illegal character in column 20 of a BCD card. Control is restored to $\alpha + 2$ in the calling sequence with bit 47 of A set to 1.

5. Data error:

This may be caused by an illegal character in a numerical field; more than 16 digits in an octal numerical field, or more than 14 in a decimal numerical field; more than one sign in a numerical field or a sign which is preceded by a number in the field; more than one decimal point in a decimal numerical field; too large a scale factor; a decimal number which when scaled does not fit the A register, or which is too large or too small for the floating point format

$$(\geq -2^{1023} \text{ or } \leq -2^{-1023}).$$

In case of a data error, the word in storage corresponding to the erroneous field is set equal to minus zero, and loading continues. At the end of the routine, control is returned to $\alpha + 2$ in the calling sequence, with the number of erroneous fields in the lower address of the A register.

6. Termination of loading:

An SLJ card, an end-of-file, or a format error automatically terminates loading. If $B \neq 0$ (in the Q register of entry), loading is terminated when B-A+1 words have been loaded. The program then hunts forward through the remaining cards (or records on tape) to the first SLJ card, or end-of-file. The tape or card reader is left set to read the next succeeding record or card. Normal return in this case is to $\alpha + 3$ in the calling sequence.

7. All conversions are performed rapidly enough so that the 1607 tape mechanism (or the 088 card reader) can be operated at full speed; 3000 records per minute on tape or 650 cards per minute through the 088.

8. Floating point numbers with large negative exponents may be accurate to only 35 bits.

h. <u>Rotate.</u>

MATRIX ROTATION

The following transformation was used as a subroutine to transform from one Cartesian coordinate system to another Cartesian coordinate system.

1. Rotation of the initial system by an angle φ counterclockwise about the Z axis.

2. Rotation of the primed system by the angle θ counterclockwise about the X' axis.

3. Rotation of the double primed system by the angle ψ counterclockwise about the Z'' axis.

$$\begin{vmatrix} X''' \\ Y''' \\ Z''' \end{vmatrix} = \begin{vmatrix} A \end{vmatrix} \begin{vmatrix} X \\ Y \\ Z \end{vmatrix}$$

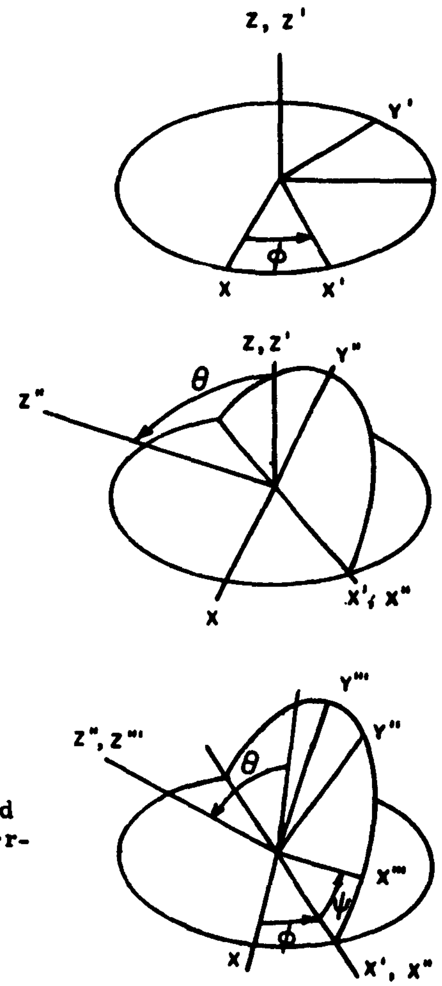


Figure 6.

$$A = \begin{vmatrix} \cos\psi\cos\varphi - \cos\theta\sin\varphi\sin\psi & -\sin\psi\cos\varphi - \cos\theta\sin\varphi\cos\psi & \sin\theta\sin\varphi \\ \cos\psi\sin\varphi + \cos\theta\cos\varphi\sin\psi & -\sin\psi\sin\varphi + \cos\theta\cos\varphi\cos\psi & -\sin\theta\cos\varphi \\ \sin\theta\sin\psi & \sin\theta\cos\psi & \cos\theta \end{vmatrix}$$

This program is callable by FORTRAN.

To use:

```
CALL ROTATE (A, B, C, D, E)
```

where

A = The angle φ

B = The angle θ

C = The angle Ψ

D = The input 3 component vector

E = The output 3 component vector

i. SETTAB

The Subroutine SETTAB is used to set up a table of print times. This table includes both even increment print times and abnormal print times such as ignition, drop stages, and burnout.

The subroutine is callable by FORTRAN. To use:

CALL SETTAB (NS, TMI, TMCC, TMBO, DT, TABLE, IX)

where

NS	=	number of stages
TMI	=	table of ignition times
TMCC	=	table of coefficient change times
TMBO	=	table of burnout time
DT	=	even time increments
TABLE	=	resulting table
IX	=	size of the table

j. TWO BOD

SYMBOLS

ENGLISH

a	-	Semi-major axis of orbit	(NM)
a_E	-	Equatorial radius of the earth	(NM)
b	-	Semi-minor axis of orbit	(NM)
C	-	Earth parameter defined by equation 3-1	(NM)
e	-	Eccentricity of the orbit	
e_E	-	Eccentricity of the earth ($e_E^2 = 2f - f^2$)	
E	-	Eccentric anomaly	(RAD)
f	-	Flattening of the earth	
GM	-	Gravitational constant of the earth	(FT^3/SEC^2)
h_G	-	Geodetic altitude	(NM)
h_x, h_y, h_z	-	Angular momentum about subscripted axis	
h	-	Total angular momentum	
h_A	-	Apogee altitude	(NM)
h_p	-	Perigee altitude	(NM)
H	-	Energy parameter defined by equation 2-2	
i	-	Inclination of orbit plane to equatorial plane	(DEG)
k	-	Constant used to obtain period ($2\pi/\sqrt{GM}$)	
M	-	Mean anomaly	
P	-	Period of the orbit	(MIN)
R	-	Geocentric radius from center of earth to vehicle	(NM)
$\dot{R}$	-	Change in R with respect to time	(NM/SEC)
R'	-	Geocentric radius used for earth's velocity	(NM)
S	-	Earth parameter defined by equation 3-2	(NM)
t_o	-	Time of entry into orbit	(SEC)

SYMBOLS

T	-	Time of perigee passage	(SEC)
t_i	-	Time at i^{th} point in orbit	(SEC)
V_E	-	Velocity of vehicle with respect to the earth	(FPS)
V_R	-	Velocity of earth's surface below vehicle	(FPS)
V_I	-	Velocity of vehicle with respect to non-rotating earth	(FPS)
x_b, y_b, z_b	-	Earth centered coordinates of the space vehicle	(Figure 3)
x, y, z	-	Geodetic coordinate system at vehicle	(Figure 4)
x', y', z'	-	Geodetic coordinate system below vehicle on surface of the earth	(Figure 4)
x'', y'', z''	-	Coordinate system in orbit plane	(Figure 2)
X, Y, Z	-	Earth centered coordinate system	(Figure 1)
x_s, y_s, z_s	-	Coordinates of the look station	(Figure 3)

GREEK

α	-	Angle between geodetic east axis and X axis of coordinate system tangent to a geodetic earth; also the aspect angle	(DEG)
β	-	Initial azimuth (positive c. w. from north)	(DEG)
γ	-	Initial flight path angle (positive up)	(DEG)
γ_I	-	Inertial flight path angle (positive up)	(DEG)
ω	-	Argument of perigee	(DEG)
ω_E	-	Rotational rate of the earth	(RAD/SEC)
Ω	-	Longitude of ascending node	(DEG)
φ	-	Longitude	(DEG)
θ_G	-	Geodetic latitude	(DEG)
θ_c	-	Geocentric latitude from vehicle	(DEG)
θ_c'	-	Geocentric latitude from earth's surface	(DEG)
μ	-	Argument of latitude	(DEG)
ν	-	True anomaly	(DEG)

SUBSCRIPTS

b	-	refers to space vehicle
c	-	refers to geocentric
E	-	refers to the earth
G	-	refers to geodetic
i	-	refers to i^{th} point in orbit
I	-	refers to nonrotating earth
o	-	refers to initial or burnout values
R	-	refers to rotational velocity radius
s	-	refers to look station
x, y, z	-	refers to X, Y, Z, coordinate system
1	-	refers to coordinate system at station parallel to equatorial plane
2	-	refers to coordinate system at look station

A dot over a variable indicates the time derivative of that variable.

A delta (Δ) in front of a variable indicates a difference in that variable.

1. INTRODUCTION.

a. Inputs.

The inputs are a position and velocity vector in a right-hand Earth-centered coordinate system with the X axis along the node of the equatorial plane and the Greenwich meridian and the Z axis along the north polar axis. The number of time increments, changes, and look-angle stations must be in COMMON. To use the subroutine, control number 4 must be set equal to 1 or 2.

b. Printout time changes.

Since some portions of the trajectory are more important than others, provisions are included for changing the time increment, with the use of a maximum of five different time increments. If the ellipse intersects the earth, the trajectory computation stops at this time. If it does not intersect the earth, the computation will continue for a prespecified number of orbits.

c. TWO-BOD computer program.

The program computes the classical orbital elements (a, b, e, P, h_A, h_P, i, ω, Ω, ν_o, E_o).

After the orbital elements are computed, a matrix-rotation is set up to transfer from the orbital plane to the equatorial plane coordinate system. Time is incremented and a new true anomaly is obtained.

The corresponding orbit plane coordinates are then transformed by a matrix rotation to the equatorial coordinate system. The latitude and longitude are obtained by taking "arc tangents" and adding the effects of a rotating earth. These values are stored along with the radius vector to the vehicle and are used to find the "look angles" at various stations.

The matrix subroutine described in rotate is used for the coordinate transformation.

2. EQUATIONS OF CELESTIAL MECHANICS.

a. Semi-major axis.

$$V_I = \sqrt{\dot{X}^2 + \dot{Y}^2 + \dot{Z}^2} \quad (1)$$

$$H = \frac{R V_I^2}{GM}, \text{ if } H \geq 2, \text{ the rocket escapes} \quad (2)$$

$$a = \frac{R}{2-H} \quad (3)$$

b. Angular momentum.

$$h_x = Y\dot{Z} - Z\dot{Y} \quad (4a)$$

$$h_y = Z\dot{X} - X\dot{Z} \quad (4b)$$

$$h_z = X\dot{Y} - Y\dot{X} \quad (4c)$$

$$h = \sqrt{h_x^2 + h_y^2 + h_z^2} \quad (4d)$$

c. Orbital elements.

$$i = \cos^{-1}(h_z/h) \quad (5)$$

$$\Omega = \tan^{-1}\left(\frac{h_x}{-h_y}\right) \quad \text{If } (-h_y) \text{ is negative, } \Omega = \Omega + 180^\circ \quad (6)$$

$$R = \frac{X\dot{X} + Y\dot{Y} + Z\dot{Z}}{R} \quad (7)$$

$$e = \left(1 - \frac{h^2}{GMa}\right)^{1/2} \quad (8)$$

$$\cos \nu_o = \frac{h^2}{RGM} - 1 \tag{9}$$

$$\sin \nu_o = \frac{h\dot{R}}{GM} \tag{10}$$

$$\nu_o = \tan^{-1}\left(\frac{\sin \nu_o}{\cos \nu_o}\right) \quad \text{If } (\cos \nu_o) \text{ is negative, } \nu_o = \nu_o + 180^\circ \tag{11}$$

$$\cos \mu = \frac{(Y\, h_x - X\, h_y)}{h} \tag{12}$$

$$\mu = \tan^{-1}\left(\frac{Z}{\cos \mu}\right) \quad \text{If } (\cos \mu) \text{ is negative } \mu = \mu + 180^\circ \tag{13}$$

$$\omega = \mu - \nu_o \tag{14}$$

$$b = a\sqrt{1 - e^2} \tag{15}$$

$$P = k\, a^{3/2} \tag{16}$$

$$h_A = a\,(1 + e) - a_E \tag{17}$$

$$h_P = a\,(1 - e) - a_E \tag{18}$$

$$E_o = 2\tan^{-1}\left(\sqrt{\frac{1 - e}{1 + e}}\ \tan \frac{\nu_o}{2}\right) \tag{19}$$

$$t_o - T = \frac{(E_o - e \sin E_o)\, P}{2\pi} \tag{20}$$

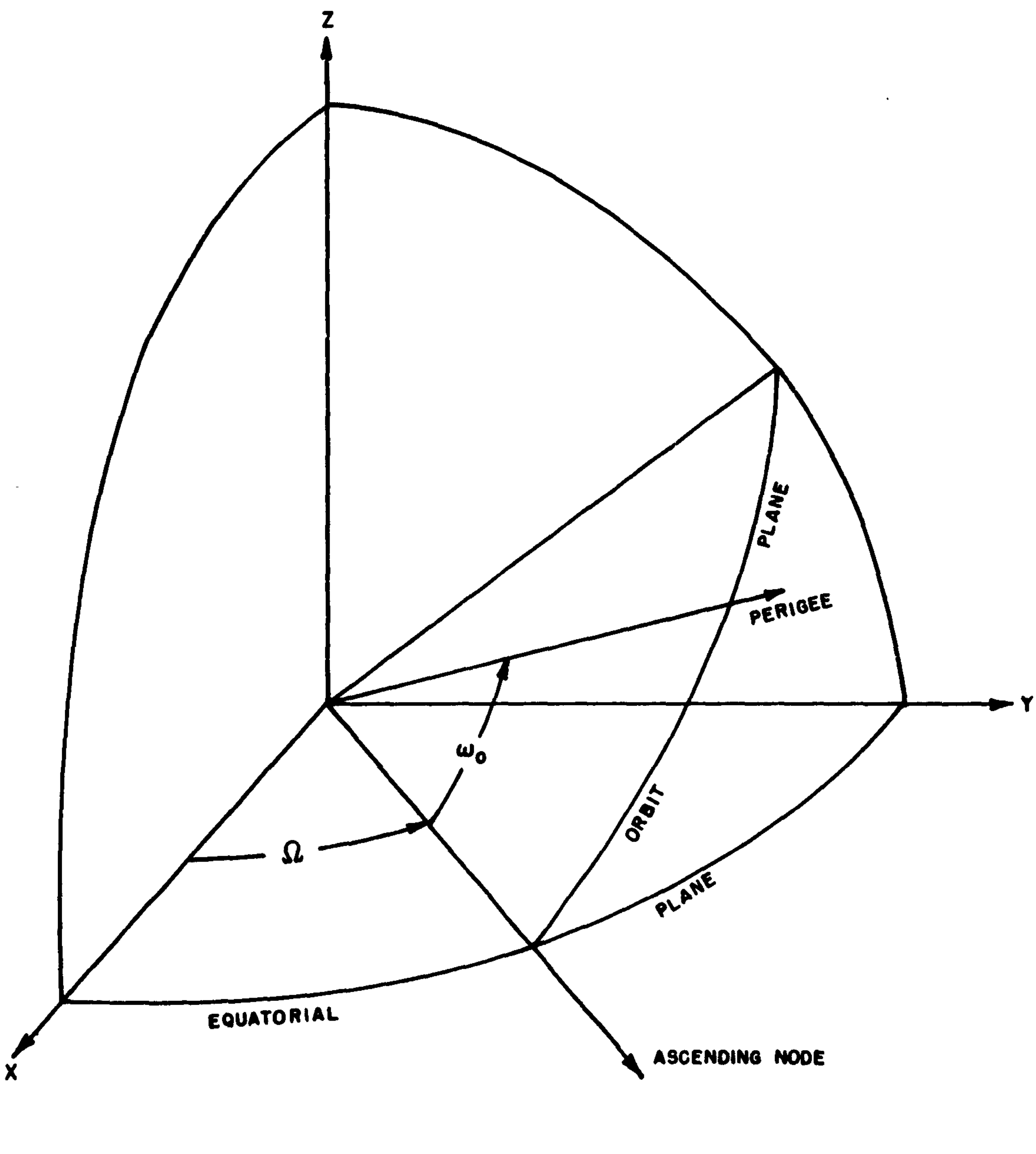

Z
Y
X
PLANE
PERIGEE
ω_o
ORBIT
Ω
PLANE
EQUATORIAL
ASCENDING NODE

Figure 1.

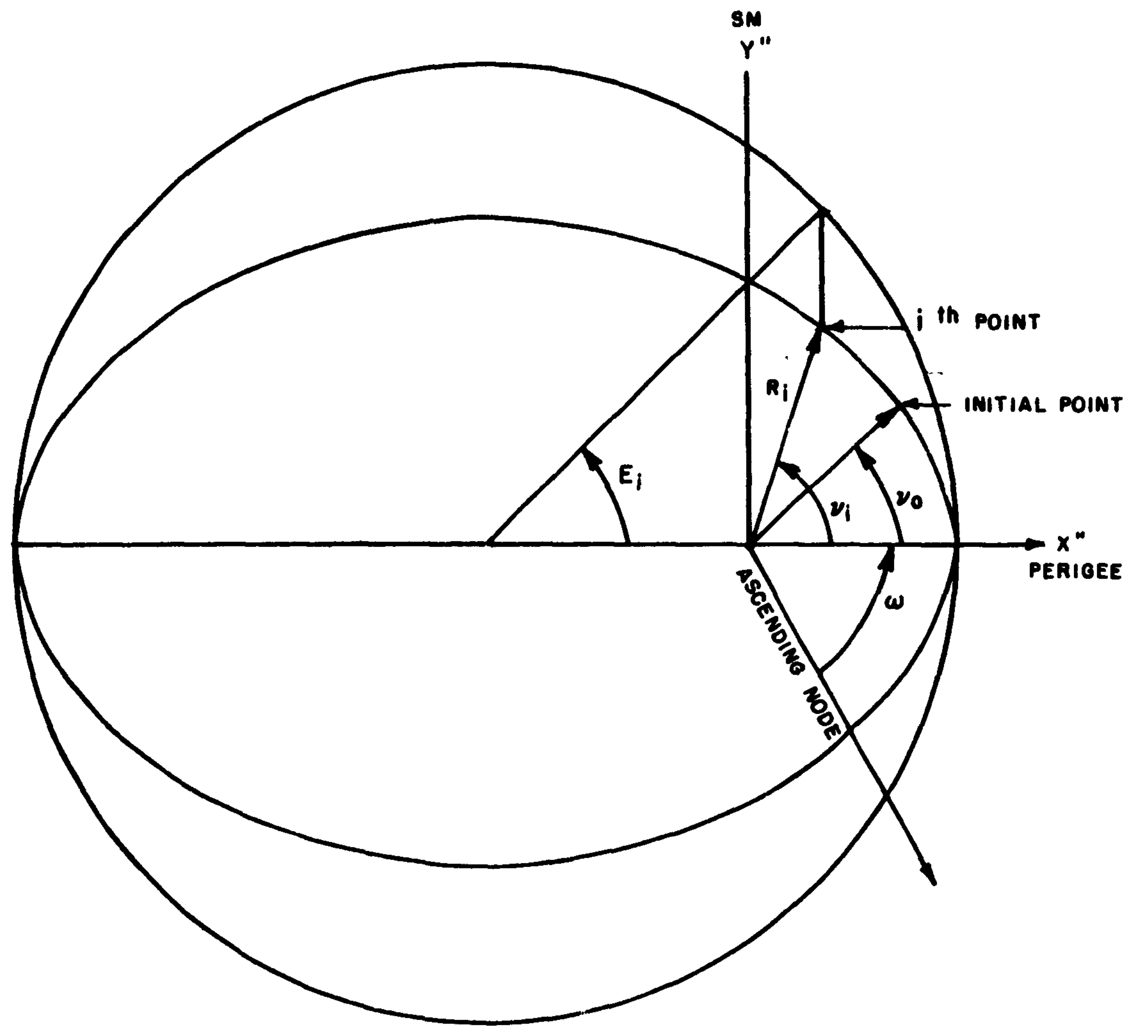

SM
Y"
j th POINT
R_i
INITIAL POINT
E_i
ν_i
ν_0
X"
PERIGEE
ω
ASCENDING NODE

Figure 2

d. Trajectory points.

$$t_{i+1} = t_i + \Delta t \tag{21}$$

$$M = \frac{2\pi}{P}(t_i - T) \tag{22}$$

$$E_2 = E_1 - \frac{(E_1 - e \sin E_1 - M)}{1 - e \cos E_1} \quad \text{Iterate until } E_2 = E_1 \tag{23}$$

$$\nu_i = 2 \tan^{-1}\left(\sqrt{\frac{1+e}{1-e}} \tan \frac{E_i}{2}\right) \tag{24}$$

$$R_i = \frac{a(1 - e^2)}{1 + e \cos \nu_i} \tag{25}$$

$$V_i = \left[GM\left(\frac{2}{R_i} - \frac{1}{a}\right)\right]^{1/2} \tag{26}$$

$$\gamma_i = \tan^{-1}\left[\frac{e \sin \nu_i}{1 + e \cos \nu_i}\right] \tag{27}$$

$$x'' = \cos \nu_i \tag{28a}$$

$$y'' = \sin \nu_i \tag{28b}$$

$$z'' = 0 \tag{28c}$$

$$\begin{vmatrix} X \\ Y \\ Z \end{vmatrix} = \begin{vmatrix} \cos\Omega & -\sin\Omega & 0 \\ \sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 \\ 0 & \cos i & -\sin i \\ 0 & \sin i & \cos i \end{vmatrix} \begin{vmatrix} \cos\omega & -\sin\omega & 0 \\ \sin\omega & \cos\omega & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} x'' \\ y'' \\ z'' \end{vmatrix} \quad (29)$$

$$\begin{aligned} X &= x''(\cos\Omega\cos\omega - \sin\Omega\cos i\sin\omega) - y''(\cos\Omega\sin\omega + \sin\Omega\cos i\cos\omega) \\ Y &= x''(\sin\Omega\cos\omega + \cos\Omega\cos i\sin\omega) - y''(\sin\Omega\sin\omega - \cos\Omega\cos i\cos\omega) \\ Z &= x''(\sin i\sin\omega) + y''(\sin i\cos\omega) \end{aligned} \quad (30)$$

$$\theta_c = \tan^{-1}\frac{Z}{\sqrt{X^2 + Y^2}} \quad (31)$$

$$\varphi_c = \tan^{-1}\left(\frac{Y}{X}\right) \quad \text{If X is negative,} \quad \varphi_c = \varphi_c + 180° \quad (32)$$

$$\varphi_c = \varphi_c - \omega_E\,(t_i - t_o) \quad (33)$$

e. Geocentric to geodetic.

$$\theta_G = \theta_c + \sin^{-1}\left\{\frac{a_E}{R_i}\left[f\sin 2\theta_c + f^2\sin 4\theta_c\left(\frac{a_E}{R_i} - \frac{1}{4}\right)\right]\right\} \quad (34)$$

$$h_G = R_i - a_E\left[1 - f\sin^2\theta_c - \frac{f^2}{2}\sin^2 2\theta_c\left(\frac{a_E}{R_i} - \frac{1}{4}\right)\right] \quad (35)$$

f. Range.

The great circle range is obtained by computing the range angle. This angle is obtained by taking the dot product of the launch vector with the radius vector at any given time.

$$\begin{aligned} \text{R.A.} &= \text{Cos}^{-1}\left\{(X_L X + Y_L Y + Z_L Z)/\sqrt{X_L^2 + Y_L^2 + Z_L^2}\ \sqrt{X^2 + Y^2 + Z^2}\right\} \\ \text{Range} &= R_E\,(\text{avg})\,(\text{R.A.}) \end{aligned} \quad (36)$$

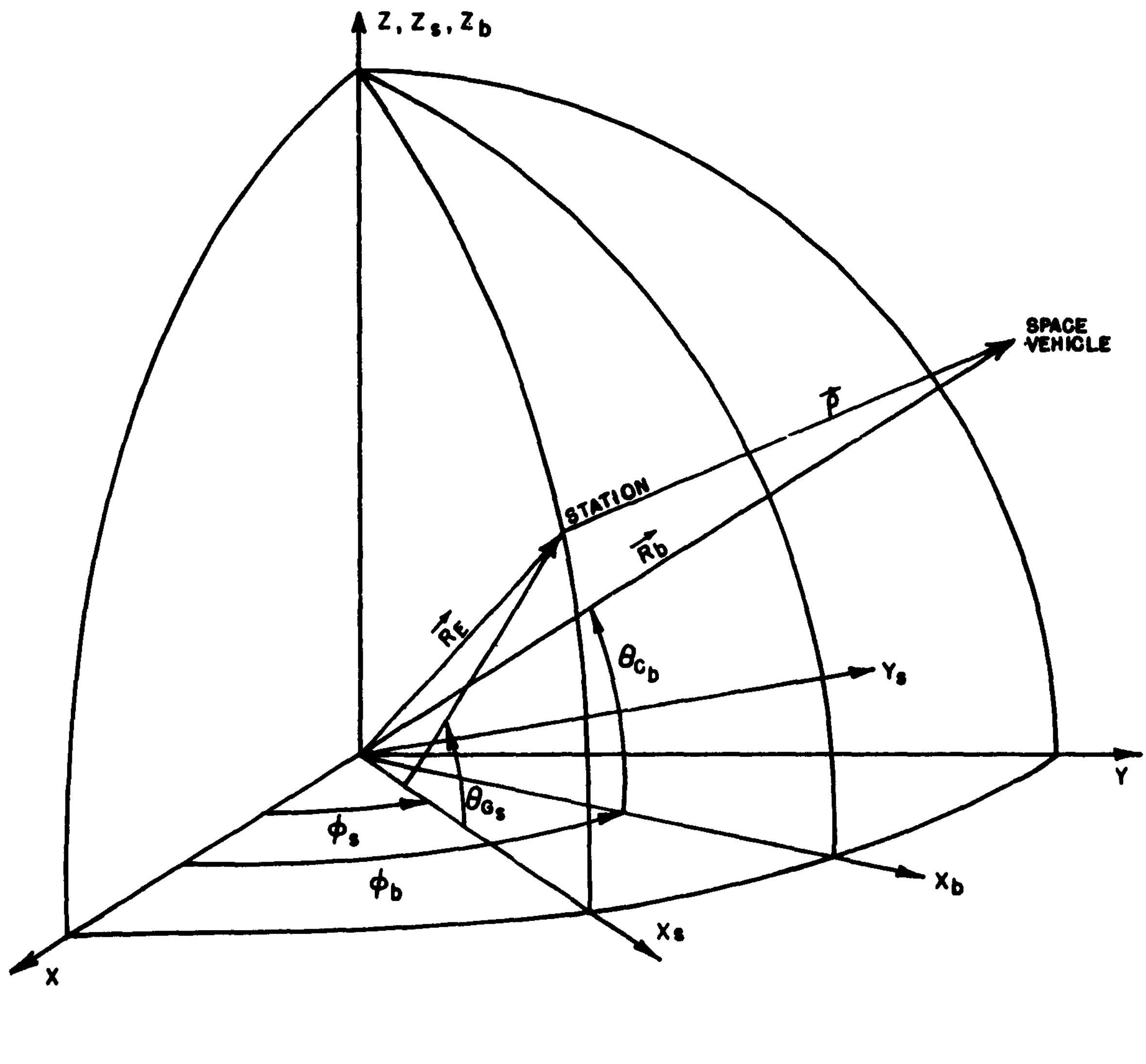


Figure 3

g. **Look angles.**

Station given station $(h_{G_s},\ \theta_{G_s},\ \varphi_s)$

$$C_s = \frac{a_E}{(1 - e_E^2 \sin^2 \theta_G)^{1/2}} \qquad S_s = C_s(1 - e_E^2) \tag{37}$$

$$\left|\vec{R}_E\right| = \left[(S_s + h_{g_s})^2 \sin^2 \theta_{G_s} + (C_s + h_{g_s})^2 \cos^2 \theta_{G_s}\right]^{1/2} \tag{38}$$

$$\theta_{c_s} = \tan^{-1}\left[\left(\frac{S_s + h_{g_s}}{C_s + h_{g_s}}\right) \tan \theta_{G_s}\right] \tag{39}$$

$$R_E \begin{cases} x_s = R_E \cos \theta_{c_s} \\ y_s = 0 \\ z_s = R_E \sin \theta_{c_s} \end{cases} \tag{40}$$

$$R_b \begin{cases} x_b = R_b \cos \theta_{c_b} \cos \Delta\varphi \\ y_b = R_b \cos \theta_{c_b} \sin \Delta\varphi \qquad \text{where } \Delta\varphi = \varphi_b - \varphi_s \\ z_b = R_b \sin \theta_{c_b} \end{cases} \tag{41}$$

$$\vec{R}_b = \vec{R}_E + \vec{\rho}$$

$$\vec{\rho} = \vec{R}_b - \vec{R}_E \tag{42}$$

$$\vec{\rho}\begin{cases} x_1 = R_b \cos\theta_{c_b} \cos\Delta\varphi - R_E \cos\theta_{c_s} \\ y_1 = R_b \cos\theta_{c_b} \sin\Delta\varphi \\ z_1 = R_b \sin\theta_{c_b} - R_E \sin\theta_{c_s} \end{cases} \quad (43)$$

$$\begin{vmatrix} x_2 \\ y_2 \\ z_2 \end{vmatrix} = \begin{vmatrix} \cos\theta_G & 0 & -\sin\theta_G \\ 0 & 1 & 0 \\ \sin\theta_G & 0 & \cos\theta_G \end{vmatrix} \begin{vmatrix} x_1 \\ y_1 \\ z_1 \end{vmatrix} \quad (44)$$

$$\text{Elevation ang.} = \tan^{-1}\left[\frac{z_2}{\sqrt{x_2^{\,2} + y_2^{\,2}}}\right] \quad (45)$$

$$\text{Azimuth ang.} = \tan^{-1}\left[y_2/x_2\right] \quad \text{If } x_2 \text{ neg, add } 180^\circ \quad (46)$$

If (azimuth) is neg, add 360°

h. Aspect angle.

The aspect angle is defined as the angle between the spin axis of the space vehicle and the vector from the look station to the vehicle. This program assumes the vehicle remains fixed in space. The angle is obtained by the following equations:

$$\alpha = \cos^{-1}\left[(\dot{X}x_1 + \dot{Y}y_1 + \dot{Z}z_1)/\sqrt{\dot{X}^2 + \dot{Y}^2 + \dot{Z}^2}\,/\,\sqrt{x_1^2 + y_1^2 + z_1^2}\right] \quad (47)$$

VARIABLES USED IN TWO BODY SUBROUTINE

ALT	Geocentric radius to vehicle
AMU	Argument of latitude
ANS	Output vector
ANT	Aspect angle
APOGEE	Apogee altitude of the orbit
ARA	Average radius of the Earth
AX	X Component of inertial burnout position vector
AXDOT	X Component of inertial burnout velocity vector
AY	Y Component of inertial burnout position vector
AYDOT	Y Component of inertial burnout velocity vector
AZ	Z Component of inertial burnout position vector
AZDOT	Z Component of inertial burnout velocity vector
AZIM	Azimuth of look vector
A1	Burnout position vector
A2	Burnout velocity vector
A1A	Not used in two body *
A1B	Not used in two body *
C	$\triangleq$ Ree/ $\sqrt{1 - e^2 \sin \theta_c}$
CAPO	Longitude of ascending node
CLAL	Cosine of the launch latitude *
CLOL	Cosine of the launch longitude
CMU	Cosine of MU
CSLANT	Cosine of inclination angle
CT	Cosine of geocentric latitude
CTRVA	Cosine of true anomaly
DELTAT	Stored time increment *
DPHI	Difference in longitude
DUMTB	Not used in two body *
E	Eccentricity of the Earth

---

* Stored in COMMON

ELEV	Elevation of look angle vector
ENDT	End of time increment *
ER	Size of allowable error
ETA	Anomaly = period/2π
EX	Orbital eccentricity
EXANOM	Eccentric anomaly
EXIMP	Impact eccentric anomaly
E1	Used to compute eccentric anomaly
E2	Used to compute eccentric anomaly
F	Flattening of the Earth
FMIN	Output - time in minutes
FMINF	Function for obtaining FMIN
FPA	Flight path angle
FX	Output vector from Rotate
GM	Earth gravitational attraction constant
GM1	Earth gravitational attraction constant
H	Angular momentum parameter
HH	Angular momentum squared
HOUR	Output time in hours
HOURH	Function for obtaining HOUR
HX	X Component of angular momentum
HY	Y Component of angular momentum
HZ	Z Component of angular momentum
H1	Angular momentum
I	Utility index
IT	Utility index
J	Utility index
K	Utility index
KBATT	Number of Batt output tape *
MEX	Utility index
MM	Utility index
NAME	Name stored in common *

---

* Stored in COMMON

NAT	Number of output tape *
NOST	Number of look angle stations *
NOT	Number of time changes *
N1C	Not used in two body *
ORBT	Number of orbits of orbital vehicle *
PERIGEE	Perigee altitude of the orbit
PERIOD	Time to make one revolution of orbit
PHI	Stored longitude
PHIX	Used to compute longitude
PI	$= \pi = 3.141592654$
PN	Page count
POP	Earth flattening constant
PRALT	Geodetic latitude of the vehicle
PRLAT	Geodetic altitude of the vehicle
PRLON	Longitude of the vehicle
PRTIME	Time from launch
R	Length of position vector n. m.
RAD	$= \pi /180. = 1/57.2957795$
RANGE	Great circle range on surface of Earth
RDOT	Rate of change along R with respect to time
RE	Radius of Earth = $f(\theta c)$
REE	Equatorial radius of Earth - n. m.
RF	Length of position vector (ft)
RLAU	Radius to launch point *
RS	Radius of look station
S	$\triangleq C (1 - e^2)$
SEC	Output time in seconds
SECF	Function for obtaining output time in seconds
SHT	Look station altitude
SLAL	Sine of the launch latitude *
SLANT	Orbital inclination angle
SLOL	Sine of the launch longitude
SLR	Distance from tracking station to vehicle

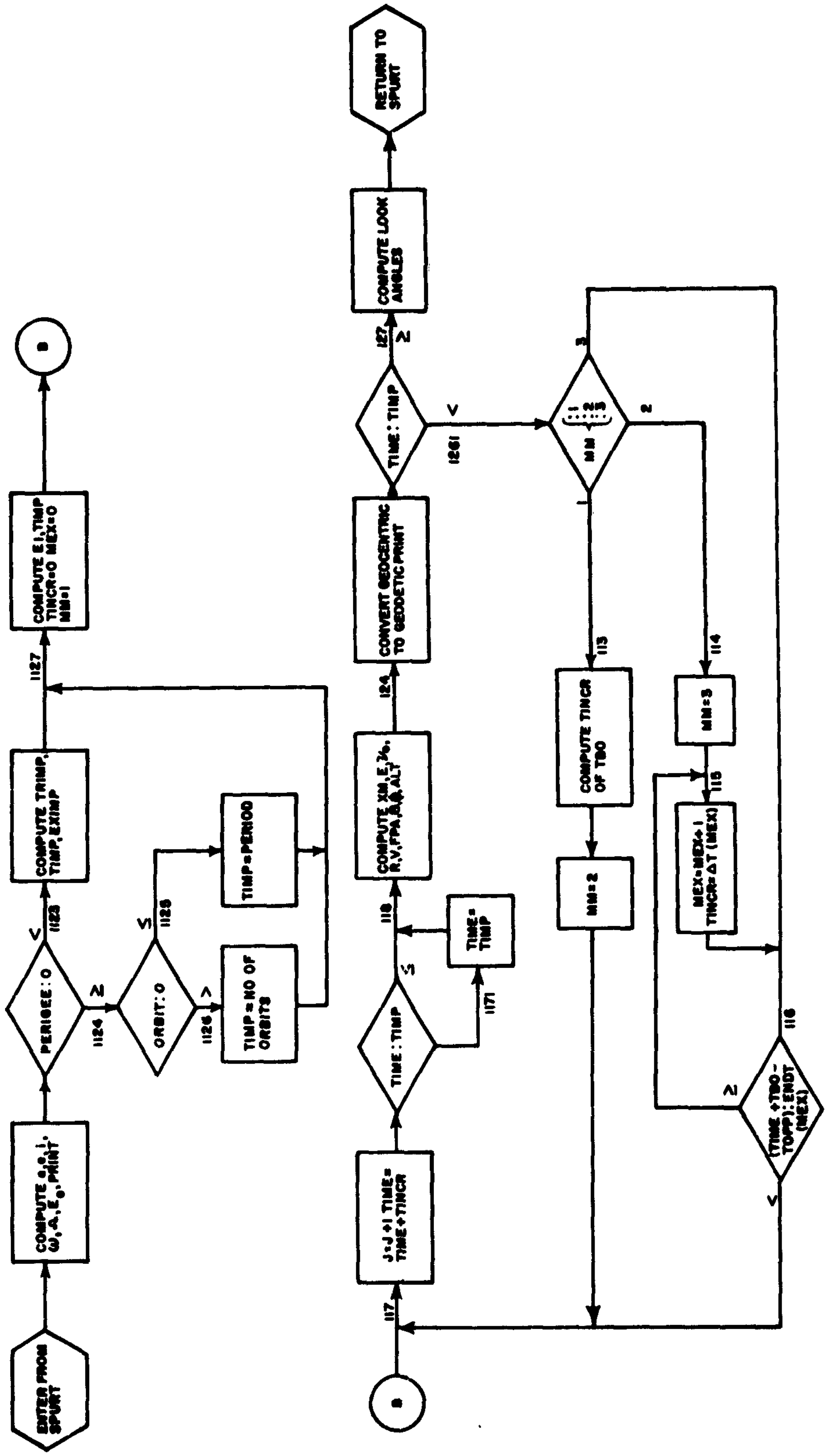
* Stored in COMMON

SMAJ	Semi major axis
SMIN	Semi minor axis
SMON	Argument of perigee
SPH	Look station longitude
SSLANT	Sine of inclination angle
ST	Sine of theta
STH	Look station latitude
STHC	Look station geocentric latitude
STRVA	Sine of the true anomaly
SX	Orbital plane coordinate system vector
S1	Variables used to compute output variables
S2	Variables used to compute output variables
S3	Variables used to compute output variables
S4	Variables used to compute output variables
S5	Variables used to compute output variables
SP6	Used to calculate aspect angle
TAG	Not used in two body *
THETA	Geocentric latitude
TIME	Independent variable
TIMP	Time to impact
TINCR	Time increment
TOPP	Time of perigee passage
TRANOM	Initial true anomaly
TRIMP	Impact point true anomaly
TRUNU	True anomaly
VELOC	Velocity at any point of the orbit
VI	Initial inertial velocity
WE	Rotational velocity of the Earth
X	X Component of look vector
XAZ	Azimuth of look vector
XEL	Elevation of look vector
XIM	X Component of position vector for range computations

* Stored in COMMON

XLAV	X Component of launch vector
XLLO	Launch longitude *
XM	Mean anomaly
XP	Used in matrix rotation
XSHT	Stored tracking station altitude *
XSPH	Stored tracking station longitude *
XSTH	Stored tracking station latitude *
XX	$= \sqrt{\frac{1 - e}{1 + e}}$, storage variable
XX1	Sine of station geodetic latitude
XX2	Cos of Station geodetic Latitude
XX3	Used in station radius calculation
XX4	Used in station radius calculation
Y	Y Component of look vector
YIM	Y Component of position vector for range computation
YLAU	Y Component of launch vector
YY	Storage variable
Z	Z Component of look vector
ZIM	Z Component of position vector for range computation
ZIZ	Range angle
ZLAU	Z Component of launch vector
ZP	Used in matrix rotation
Z2	Number of lines per page *

* Stored in COMMON


ENTER FROM SPURT
COMPUTE a,e,i, ω,Ω,E₀, PRINT
PERIGEE : 0
COMPUTE TRIMP, TIMP, EXIMP
COMPUTE E1,TIMP TINCR=0 MEX=0 MM=1
ORBIT : 0
TIMP=NO OF ORBITS
TIMP=PERIOD
B
J=J+1 TIME=TIME+TINCR
TIME : TIMP
TIME=TIMP
CONVERT GEOCENTRIC TO GEODETIC PRINT
TIME : TIMP
COMPUTE LOOK ANGLES
RETURN TO SPURT
MM
COMPUTE TINCR OF TBO
MM=2
MM=3
MEX=MEX+1 TINCR=ΔT (MEX)
(TIME+TBO-TOPP):ENDT (MEX)
1123
1124
1125
1126
1127
117
1171
118
124
1261
127
113
114
115
116

k. WOTF, SAUF.

A. IDENTIFICATION

TITLE:	WRITE ON TAPE
CO-OP ID:	WOTF
CATEGORY:	General Tape Handler
Programer:	J. W. Vise
Date:	August 4, 1961

B. PURPOSE:

This routine is used to write a record of arbitrary length on a magnetic tape in either binary or BCD mode

C. USAGE:

1. Calling Sequence

CALL WOTF (a, b, n, m)

n = 1-48, MT
m = 0, BINARY
m = 1, BCD
a = First LOCN
b = Last LOCN

CLASSIFICATION: Utility Routine

TITLE: SAVE TAPE

ID: SAVF

PURPOSE: To inform the operator that a tape is to be saved and to rewind the tape with interlock. E. O. F. and 9sssssssENDsssss 0FssssssTAPEssss is written on the tape. (Here s represents blanks).

USAGE:

1. Calling Sequence:

This routine is used with the following FORTRAN calling sequences:

```
CALL SAVF (N), or
DUMMY = SAVF (N)
```

where N is the number, 1-48, of the logical tape to be saved.

7. PROGRAM LISTING

```
C
C                  *** SPURT ***
C                  SEPTEMBER 19, 1962  MOD OF AUG. 21, 1962
C                  IF KNTRL (1)=1, USE METRO DATA
C                  IF KNTRL (1)=2, USE WINDS ONLY
C                  IF KNTRL (2)=1, PRINT INPUT DATA
C                  IF KNTRL (3)=N, WRITE OUTPUT FOR BATT ON TAPE N
C                  IF KNTRL (4)=1, USE TWOBODY FOR LAST STAGE ONLY
C                  IF KNTRL (4)=2, USE TWOBODY FOR ALL STAGES
C                  IF KNTRL (5)=NO NO IS NUMBER OF LINES ON OUTPUT PAGE
C                  IF KNTRL (6)=N, PLOT TAPE IS WRITTEN ON TAPE N
C                  IF KNTRL (7)=1, A RIGHT HAND COORDINATE SYSTEM
C                                  IS PRINTED OUT
C                  IF KNTRL (8)=1, USE IMPACT FOR N-1 STAGES
C                  IF KNTRL (8)=2, USE IMPACT FOR ALL STAGES
C                  SENSE SWITCH 4 PRINTS INTERMEDIATE RESULTS
C                  SENSE SWITCH 5 REINITIALIZES ON CRITERION FAILURE
C                  A NEG NO. IN FIRST POSITION OF BURNING DRAG-MACH NO.
C                  TABLE IGNORES AERODYNAMICS FOR THAT STAGE
C
C                  *** DIMENSION BLOCK ***
C
       DIMENSION NAME(10),KNTRL(10),TMCC(10),SPTIME(100),SPIN(100),TIMET
      1(21,10),THRUST(21,10),D1MACH(21,10),DRAG1(21,10),D2MACH(21,10),DRA
      2G2(21,10),CPMACH(21,10),CP(21,10),CNMACH(21,10),CN(21,10),FTT(10),
      3AE(10),PAT(10),D(10),G0(10),GP(10),R(10),A(10),B(10),PWGT(10),PWGT
      4C(10),TMI(10),TMBO(10),WTEMP(96),WDEN(96),WPRES(96),WINDV(96),WIND
      5A(96),WGT(10),WGTC(10),WALTU(98),DC(3,3),ACODES(10),PXLDD(3)
      6,SPT(100),SPI(100),WALT(96),WTU(96),WDU(96),WPU(96),
      7STX(3),STXD(3),PX(3),PXD(3),WX(98),WY(98),WZ(98),ROT(3,3),RA(10),
      8RB(10),TTTAB(21),THTAB(21),DMTAB(21),DTAB(21),CPMTAB(21),CPTAB(21)
      9CNMTAB(21),CNTAB(21),WGTAB(21),SPIT(100),GOP(10),N(6,10)
       DIMENSION ROT1(3,3),T(3),TT(3),TDEL(10),PHT(10),PXDD(3)
      1,OUT(25,50),PXL(3),PXLD(3),PXND(3),XVX(3),BL(38),TABLE(500),CBLOCK
      2(150),VX(3),TB3(5),TB4(5),TB6(3,50),TB7(50),TB8(50),TB9(50)
      3XMACH(10,10),CDM(10,10),NNX(10),XIMPA(3,10),XDIMP(3,10),TFIMP(10
      4),RANGE(10),DELPR(10)
C
C                  *** COMMON BLOCK ***
C
       COMMON ROT1,NAME,KBATT,JTB1,TB2,TB3,TB4,JTB5,TB6,TB7,TB8,TB9,Z2
      1,STHC,CTHC,ALO2,REL,XMACH,CDM,NNX,NOT,RANGE,XIMPA,XDIMP,TFIMP,DELP
      2R
```

```
C
C                    *** EQUIVALENCE BLOCK ***
C
      EQUIVALENCE (BL(1),ICODE),(BL(2),ACODE),(BL(3),TRERR),(BL(4),SMIN
     1),(BL(5),SMAX),(BL(6),NOE),(BL(7),SS),(BL(8),TIME),(BL(9),PX  (1))
     2(BL(12),PHI),(BL(13),THETA),(BL(14),R1X),(BL(15),R2X),(BL(16),R3X)
     3,(BL(17),R4X),(BL(18),R5X),(BL(19),PXD(1)),(BL(22),PHID),(BL(23),T
     4HETD),(BL(24),PXDD(1)),(BL(27),PHIDD),(BL(28),THETDD),(BL(29),ACOD
     5ES(1))
      RADF(A)=SQRTF(A(1)*A(1)+A(2)*A(2)+A(3)*A(3))
 ADAMS   EQU       6000B
 ATMOS   EQU       7300B
 SQRTF   EQU       7660B
C
C                    *** FORMAT BLOCK ***
C
 1    FORMAT(10A8)
 2    FORMAT(35I2)
 3    FORMAT(I2,7F13.0)
 5    FORMAT(7F10.0)
 6    FORMAT(6I2)
 7    FORMAT(17H1  INPUT TABLE   10A8)
 8    FORMAT(5X5F15.4,4X10I2)
 9    FORMAT(12H1 STAGE NO. I2,/11X4HTIME,5X6HTHRUST,5X8HMACH NO.,5X6HCD
     1 (B),3X8HMACH NO.,5X6HCD (C),3X8HMACH NO.,6X2HCP,6X8HMACH NO.,4X8H
     2CN/ALPHA/)
 10   FORMAT(14H0   SPIN TABLE//(8F10.3))
 11   FORMAT(23H1  INPUT ERROR IN STAGEI2)
 12   FORMAT(5X10F11.3)
 13   FORMAT(1H0, 9X15HSTAGE WEIGHT = F26.2,4H LBS 6X20HSTAGE FUEL WEIGH
     1T = F30.2,4H LBS//10X16HIGNITION TIME = F25.2,4H SEC 6X15HBURNOUT
     2TIME = F35.2,4H SEC//10X13HMISSILE CG = F29.4,3H FT 6X16HSTAGE FUE
     3L CG = F35.4,3H FT//10X12HEXIT AREA = F26.2,7H SQ IN. 6X34HPRESSUR
     4E AT THRUST MEASUREMENT =  F9.4,11H LBS/SQ IN.//10X28HLONGITUDINAL
     5 I OF MISSILE = F7.2,10H FT2 SLUGS6X26HTRANSVERSE I OF MISSILE = F
     618.2,10H FT2 SLUGS//10X24HFUEL LONGITUDINAL I/M = F17.4,4H FT26X22
     7HFUEL TRANSVERSE I/M = F28.4,4H FT2/)
 14   FORMAT(10X28HTHRUST MISALIGNMENT ANGLE = F13.2,4H RAD 6X43HORIENTA
     1TION ANGLE OF THRUST MISALIGNMENT = F7.2,4H RAD//10X11HDIAMETER =
     2F31.4,3H FT6X30HTIME TO CHANGE COEFFICIENTS = F20.2,4H SEC)
 15   FORMAT (21H1  INTEGRATION ERROR 016)
 16   FORMAT(21H0   ***CRT FAILURE ATF9.4,7H SEC***)
```

```
17    FORMAT (1H0F10.4,9E12.3/(5X10F11.3))
18    FORMAT(2H1 10A8)
20    FORMAT(/19X23HLOCAL FLIGHT PARAMETERS/3X4HTIME4X8HLATITUDE4X9HLONG
     1ITUDE3X8HALTITUDE2X8HVELOCITY2X7HAZIMUTH3X3HFPA/4X3HSEC7X3HDEG10X3
     2HDEG7X4HFEET4X6HFT/SEC6X3HDEG5X3HDEG/)
21    FORMAT(/83H   TIME     RANGE     DEFLECTION   ALTITUDE    VELOCITY
     1  PHI    THETA                  /83H     SEC    N.MILES        N.MILES
     2  N.MILES     FT/SEC      DEG.    DEG.                      /)
22    FORMAT(/3X4HTIME5X6HTHRUST5X6HWEIGHT4X12HTOTAL ACCEL.4X12HDYNAMIC
     1PRES4X5H DRAG7X5H MACH/4X3HSEC7X3HLBS8X3HLBS5X12HACCEL./GRAV.6X8HL
     2BS/SQFT7X3HLBS9X3HNO./)
23    FORMAT(/4X4HTIME9X1HX12X1HY12X1HZ10X5HX-DOT8X5HY-DOT8X5HZ-DOT/4X3H
     1SEC9X4HFEET9X4HFEET9X4HFEET8X6HFT/SEC7X6HFT/SEC7X6HFT/SEC/)
24    FORMAT(F8.2,2F12.4,F9.0,F9.0,2F9.2)
25    FORMAT(F8.2,F10.3,2F12.3,F11.1,2F8.3)
26    FORMAT(F8.2,2F12.2,2F12.3,2F13.2)
27    FORMAT(F8.2,6F13.0)
30    FORMAT(I1,F10.6)
31    FORMAT(2F10.0)
32    FORMAT(I3)
33    FORMAT(3A8,3F10.0)
         ORG        10050B
C
C                 *** CONSTANTS ***
C
99    PI=3.141592654
      TWOPI=2.*PI
      HALFPI=0.5*PI
      RAD=PI/180.
      REE=20925647.
      GM=1.40764E16
      XJ=.0016234
      GRAV0=32.174
      OMEG=7.292111E-5
      TOMEG=2.*OMEG
      OMEG2=OMEG*OMEG
      GK= XJ*REE*REE
      ICODE=3
      TRERR=1.E-3
      ACODE=1.0
      ACODES(1)=1.0
```

```
      ACODES(2)=1.0
      ACODES(3)=1.0
      ACODES(4)=0.1
      ACODES(5)=0.1
      ACODES(6)=1.0
      ACODES(7)=1.0
      ACODES(8)=1.0
      ACODES(9)=0.01
      ACODES(10)=0.01
      DT=1.
      SMIN=.01
      SMAX=1.
      NOE=10
      SS=1./16.
C
C                         *** INPUT BLOCK ***
C
 100  CALL SCLOCK
      KIX = 0
      READ 3,NS,PYLWGT
      IF(NS)120,110,140
 110  IF(NOT) 111,112,111
 111  CALL SAVF(NOT)
 112  STOP
 120  NS=-NS
         ENA        0
         ENQ        0
         RTJ        RDF
+        NOP        =49
+        RTJ        ERROR.
 130  GO TO 183
 140  READ 1,(NAME(I),I=1,10)
      READ 5,ALAT,ALON,AALT,AAZIM,(STX(I),I=1,3),(STXD(I),I=1,3),STP,STT
     1,STPD,STTD,STARTT,TSTOP
      READ 2,(KNTRL(I),I=1,10),NSPIN,NOT,INTN
      IF(NOT) 113,114,113
 113  REWIND NOT
 114  READ 5,(SPTIME(I),I=1,NSPIN)
      READ 5,(SPIN(I),I=1,NSPIN)
      DO 150 J=1,NS
      READ 6,NO,(N(I,J),I=1,5)
      N1=N(1,J)
```

```
      N2=N(2,J)
      N3=N(3,J)
      N4=N(4,J)
      N5=N(5,J)
      READ 5,(TIMET(I,J),I=1,N1)
      READ 5,(THRUST(I,J),I=1,N1)
      READ 5,(D1MACH(I,J),I=1,N2)
      IF (D1MACH(1,J)) 142,141,141
141   READ 5,(DRAG1(I,J),I=1,N2)
      READ 5,(D2MACH(I,J),I=1,N3)
      READ 5,(DRAG2(I,J),I=1,N3)
      READ 5,(CPMACH(I,J),I=1,N4)
      READ 5,(CP(I,J),I=1,N4)
      READ 5,(CNMACH(I,J),I=1,N5)
      READ 5,(CN(I,J),I=1,N5)
142   READ 5,PAT(J),AE(J),D(J),G0(J),GP(J),RA(J),RB(J),A(J),B(J),PWGT(J)
     1,PWGTC(J),PHT(J),TDEL(J),DUM,TMI(J),TMB0(J),TMCC(J)
      IF(NO-J)149,150,149
149   PRINT 11,J
      STOP
150   CONTINUE
      IF(KNTRL(1)) 190,190,160
160   READ 2, KMETRO
      READ 5,(WALT(I),I=1,KMETRO)
      IF(KNTRL(1)-1)190,170,180
170   READ 5,(WTEMP(I),I=1,KMETRO)
      READ 5,(WDEN(I),I=1,KMETRO)
      READ 5,(WPRES(I),I=1,KMETRO)
180   READ 5,(WINDV(I),I=1,KMETRO)
      READ 5,(WINDA(I),I=1,KMETRO)
190   IF(KNTRL(4)) 188,188,187
187   READ 30, JTB1,TB2
      READ 31, (TB3(I),TB4(I),I=1,JTB1)
      READ 32, JTB5
      DO 189 I=1,JTB5
189   READ 33,(TB6(II,I),II=1,3),TB7(I),TB8(I),TB9(I)
188   IF(KNTRL(8)) 183,183,184
184   READ 6,NXIM
      DO 185 J=1,NXIM
      READ 3,NRAT,DELPR(J)
      NNX(J)= NRAT
      READ 5,(XMACH(I,J),I=1,NRAT)
```

```
      READ 5,( CDM(I,J),I=1,NRAT)
  185 CONTINUE
  183 IF (KNTRL(2)) 200,200,192
  192 WRITE OUTPUT TAPE NOT,7,(NAME(I),I=1,10)
      WRITE OUTPUT TAPE NOT,8,PYLWGT,ALAT,ALON,AALT,AAZIM,(KNTRL(I),I=1,
     110)
      WRITE OUTPUT TAPE NOT,10,(SPTIME(I),SPIN(I),SPTIME(I+25),SPIN(I+25
     1),SPTIME(I+50),SPIN(I+50),SPTIME(I+75),SPIN(I+75),I=1,25)
      DO 191 J=1,NS
      WRITE OUTPUT TAPE NOT,9,J
      WRITE OUTPUT TAPE NOT,12,(TIMET(I,J),THRUST(I,J),D1MACH(I,J),DRAG1
     1(I,J),D2MACH(I,J),DRAG2(I,J),CPMACH(I,J),CP(I,J),CNMACH(I,J),CN(I,
     2J),I=1,21)
      WRITE OUTPUT TAPE NOT,13,PWGT(J),PWGTC(J),TMI(J),TMBO(J),GO(J),
     2GP(J),AE(J),PAT(J),A(J),B(J),RA(J),RB(J)
  191 WRITE OUTPUT TAPE NOT,14,TDEL(J),PHT(J),D(J),TMCC(J)
C
C                *** GEODETIC TO GEOCENTRIC ***
C
  200 ALA2=ALAT*RAD
      ALO2=ALON*RAD
      AAZI2=AAZIM*RAD
      COLAT=HALFPI-ALA2
      BLON=ALO2+HALFPI
      BAZIM=AAZI2-HALFPI
      CALAT=COSF(ALA2)
      SILAT=SINF(ALA2)
      C=REE/SQRTF(1.-.006693421*SILAT*SILAT)
      S=.99330658*C
      X1X=(C+AALT)*CALAT
      X1Y=(S+AALT)*SILAT
      REL=SQRTF(X1X*X1X+X1Y*X1Y)
      THC=ATANF(X1Y/X1X)
      STHC = SINF (THC)
      COTHC=HALFPI-THC
      CTHC = COSF (THC)
      CALL ROTATE(BAZIM,DELTH,0.,STX,XVX)
      XVX(3)=XVX(3)+REL
      CALL ROTATE(0.,-COTHC,-BLON,XVX,PX)
      DELTH=ALA2-THC
      CALL ROTATE (BAZIM,-COLAT,-BLON,STXD,PXD)
      DO 201 I=1,3
```

```
      DO 201 J=1,3
  201 ROT(I,J)=ROT1(I,J)
      Z2=KNTRL(5)
      PHI=STP*RAD
      THETA=STT*RAD
      R1X=PXD(1)
      R2X=PXD(2)
      R3X=PXD(3)
      R4X=STPD*RAD
      R5X=STTD*RAD
      LSKIP=0
      JP=1
      L=0
      TIME = STARTT
      KBATT = 0
      KPLOT = 0
      IF (KNTRL(3)) 202,203,202
  202 KBATT = KNTRL(3)
      REWIND KBATT
  203 IF (KNTRL (6)) 204,205,204
  204 KPLOT = KNTRL (6)
      REWIND KPLOT
C
C                 *** SET UP INPUT TABLES ***
C
  205 WGT(NS+1)=PYLWGT/GRAVO
      DO 210 J=1,NS
      K=NS-J+1
      WGT(K)=PWGT(K)/GRAVO+WGT(K+1)
      WGTC(J)=PWGTC(J)/GRAVO
  210 GOP(J)=GO(J)-GP(J)
      DO 220 I=1,NSPIN
      K=NSPIN-I+1
      SPT(K)=SPTIME(I)
  220 SPI(K)=SPIN(I)
      SPIT(NSPIN)=0.
      DO 221 I=2,NSPIN
      K=NSPIN-I+1
  221 SPIT(K)=SPIT(K+1)+(SPI(K)+SPI(K+1))*(SPT(K)-SPT(K+1))/2.
      IF(KNTRL(1))229,229,230
  229 WMAX = 0.
      GO TO 260
```

```
230   DO 251 I=1,KMETRO
      K = KMETRO-I+1
      WALTU(K)=WALT(I)
      IF(KNTRL(1)-1)260,240,250
240   WTU(K)=1116.4*SQRTF((WTEMP(I)+273.16)/288.16)
      WDU(K)=WDEN(I)*.00194032
      WPU(K)=WPRES(I)*2.088544
250   SW=WINDA(I)*RAD-BAZIM+HALFPI
      VWX=COSF(SW)
      VWY=-SINF(SW)
      WX(K)=WINDV(I)*(ROT(1,1)*VWX+ROT(1,2)*VWY)
      WY(K)=WINDV(I)*(ROT(2,1)*VWX+ROT(2,2)*VWY)
251   WZ(K)=WINDV(I)*(ROT(3,1)*VWX+ROT(3,2)*VWY)
      WMAX = WALTU(1)
260   CALL SETTAB(NS,TMI,TMCC,TMBO,DT,TABLE,IT)
      GO TO 281
270   IF(TIME-TMBO(L))410,271,273
271   DO 272 I=1,N3
      K=N3-I+1
      DMTAB(K)=D2MACH(I,L)
272   DTAB(K)=DRAG2(I,L)*D(L)*D(L)*PI/8.
      DMTAB(1)=10.E10
      DTAB(1)=DTAB(2)
      BDOT=0.
      FT=0.
      WGTCI=WGTC(L)
      SENSE LIGHT 1
      IF (KNTRL(4)-2) 280,279,280
279   CALL TWOBOD (TIME,PX,PXD)
280   KIX=KIX+1
      DO 274 I=1,3
      XIMPA(I,KIX)=PX(I)
274   XDIMP(I,KIX)=PXD(I)
      TFIMP(KIX)=TIME
      RANGE(KIX)=RANGE1
273   IF (TIME-TMCC(L)) 999,281,410
281   L=L+1
      WGTCI=0.
      GI=GO(L)
      N1=N(1,L)
      N2=N(2,L)+1
      N3=N(3,L)+1
```

```
      N4=N(4,L)+1
      N5=N(5,L)+1
      P1=PAT(L)*AE(L)
      DO 400 I=1,N1
      K=N1-I+1
      TTTAB(K)=TIMET(I,L)+TMI(L)
400   THTAB(K)=THRUST(I,L)+P1
      WGTAB(N1)=0.
      DO 404 I=2,N1
      K=N1-I+1
404   WGTAB(K)=WGTAB(K+1)+(THTAB(K)+THTAB(K+1))*(TTTAB(K)-TTTAB(K+1))*0.
     15
      FTT(L)=WGTAB(1)
      DO 405 I=1,N1
405   WGTAB(I)=WGTAB(I)*WGTC(L)/FTT(L)
      IF(D1MACH(1,L))406,407,407
406   LSKIP=1
      GO TO 410
407   LSKIP=0
      DO 401 I=1,N2
      K=N2-I+1
      DMTAB(K)=D1MACH(I,L)
401   DTAB(K)=DRAG1(I,L)*D(L)*D(L)*PI/8.
      DO 402 I=1,N4
      K=N4-I+1
      CPMTAB(K)=CPMACH(I,L)
402   CPTAB(K)=CP(I,L)
      DO 403 I=1,N5
      K=N5-I+1
      CNMTAB(K)=CNMACH(I,L)
403   CNTAB(K)=CN(I,L)*PI/8.
      DMTAB(1)=10.E10
      DTAB(1)=DTAB(2)
      CPMTAB(1)=10.E10
      CPTAB(1)=CPTAB(2)
      CNMTAB(1)=10.E10
      CNTAB(1)=CNTAB(2)
410   IF(TIME-TMI(L))999,411,411
411   SENSE LIGHT 0
      IF(TIME)412,412,999
412   CHECK=PX(1)
         RTJ        TEXIT
```

```
(999)   RTJ     ADAMS                INTEGRATION
        ZRO     EQN
        ZRO     TEXIT                               CALLING
        ZRO     PRTIME
        ZRO     BL                                          SEQUENCE
        ZRO     CBLOCK
        ZRO     EXIT
        ZRO     TMIN
        STA     ERR
      PRINT 15, ERR
      STOP
C
C                 *** EQUATIONS OF MOTION ***
C
 EQN    SLJ     **
      PXD(1)=R1X
      PXD(2)=R2X
      PXD(3)=R3X
      PHID=R4X
      THETD=R5X
      R2=PX(1)*PX(1)+PX(2)*PX(2)+PX(3)*PX(3)
      R=SQRTF(R2)
      RE=REE/SQRTF(1.+.00673852*PX(3)/R*PX(3)/R)
      ALT=R-RE
      IF(LSKIP)1050,1001,1050
1001  IF(ALT-WMAX)1000,1030,1030
1000  IF(KNTRL(1)-1)1030,1010,1020
1010  RHO = INTERPF(ALT,KMETRO,INTN,WALTU,WDU)
      VA  = INTERPF(ALT,KMETRO,INTN,WALTU,WTU)
      PRES= INTERPF(ALT,KMETRO,INTN,WALTU,WPU)
1020  VWX = INTERPF(ALT,KMETRO,INTN,WALTU,WX)
      VWY = INTERPF(ALT,KMETRO,INTN,WALTU,WY)
      VWZ = INTERPF(ALT,KMETRO,INTN,WALTU,WZ)
      IF (KNTRL(1)-1) 1031,1040,1031
1030  VWX = 0.
      VWY = 0.
      VWZ = 0.
(1031)  LDA     ALT                  ATMOSPHERE
        RTJ     ATMOS
 TEM    OCT                                    CALL
 PRES   OCT
 RHO    OCT                                         ROUTINE
```

```
VA       OCT
      GO TO 1050
1040  VX(1) = PXD(1)-VWX
      VX(2) = PXD(2)-VWY
      VX(3) = PXD(3)-VWZ
      V = RADF(VX)
      VMACH=V/VA
      FDRAG=INTERPF(VMACH,N2,INTN,DMTAB,DTAB)*V*V*RHO
      CPI=INTERPF(VMACH,N4,INTN,CPMTAB,CPTAB)
      CNI=INTERPF(VMACH,N5,INTN,CNMTAB,CNTAB)
      GO TO 1060
1050  FDRAG=0.
      CPI=0.
      CNI=0.
      VMACH=0.
      RHO=0.
1060  IF(SENSE LIGHT 1)1061,1062
1061  SENSE LIGHT 1
      GO TO 1063
1062  FORCE=INTERPF(TIME,N1,INTN,TTTAB,THTAB)
      WGTCI=INTERPF(TIME,N1,INTN,TTTAB,WGTAB)
      FT=FORCE-PRES*AE(L)/144.
      BDOT=-FORCE*WGTC(L)/FTT(L)*(WGT(L)*GOP(L)/WGTI*WGT(L)*GOP(L)/WGTI+
     1RB(L))
1063  GRV=GM/(R2*R)*(1.+3.*GK/R2-15.*GK* PX(3)/R2* PX(3)/R2)
      WGTI=WGT(L)-WGTCI
      GI=GO(L)+GOP(L)*WGTCI/WGTI
      GRVX=-GRV*PX(1)
      GRVY=-GRV*PX(2)
      GRVZ=-(GRV+6./R*GK/R2*GM/R2)*PX(3)
      IF(TIME-TMI(L))1082,1080,1082
1080  AXLM=A(L)
      TRVM=B(L)
      GO TO 1083
1082  AXLM=A(L)-RA(L)*WGTCI
      TRVM=B(L)-WGT(L)*(GI-GO(L))* GOP(L)-WGTCI*RB(L)
1083  BDOTB=BDOT/TRVM
      AXLMB=AXLM/TRVM
      COEF=CNI*(CPI-GI)*D(L)*D(L)*V*RHO/TRVM
      XSP=INTERPF(TIME,NSPIN,INTN,SPT,SPI)
      XSPT=MODF(INTERPF(TIME,NSPIN,INTN,SPT,SPIT),TWOPI)
      SPHI=SINF(PHI)
```

```
      CPHI=COSF(PHI)
      STHET=SINF(THETA)
      CTHET=COSF(THETA)
      TPHO=PHT(L)+XSPT
      CPHT=COSF(TPHO)
      SPHT=SINF(TPHO)
      IF(LSKIP)1091,1089,1091
1089  DO 1090 I=1,3
      T(I)=0.
      DO 1090 J=1,3
1090  T(I)=ROT(J,I)*VX (J)+T(I)
      V2=-T(1)*CPHI+T(3)*SPHI
      V3=-T(1)*SPHI*STHET-T(2)*CTHET-T(3)*CPHI*STHET
1091  T(1)=FT*(CTHET*SPHI-TDEL(L)*(CPHT*CPHI+SPHT*SPHI*STHET))
      T(2)=-FT*(STHET+TDEL(L)*SPHT*CTHET)
      T(3)=FT*(CTHET*CPHI+TDEL(L)*(CPHT*SPHI-SPHT*CPHI*STHET))
      DO 1100 I=1,3
      TT(I)=0.
      DO 1100 J=1,3
1100  TT(I)=ROT(I,J)*T(J)+TT(I)
      H = RADF(PXD)
1101  PXDD(1)=(TT(1)-FDRAG*PXD(1)/H)/WGTI+GRVX+TOMEG*PXD(2)+OMEG2*PX(1)
      PXDD(2)=(TT(2)-FDRAG*PXD(2)/H)/WGTI+GRVY-TOMEG*PXD(1)+OMEG2*PX(2)
      PXDD(3)=(TT(3)-FDRAG*PXD(3)/H)/WGTI+GRVZ
      PHIDD=-BDOTB*PHID+(((2.-AXLMB)*STHET*PHID+AXLMB*XSP)*THETD+COEF*V2
     1+FT*GI*TDEL(L)*CPHT/TRVM)/CTHET
      THETDD=-BDOTB*THETD-((1.-AXLMB)*STHET*PHID+AXLMB*XSP)*PHID*CTHET-F
     1T*GI*TDEL(L)*SPHT/TRVM-COEF*V3
      GO TO EQN
 EXIT    SLJ       **
 502  CHECK=PX(1)
      IF(SENSE SWITCH 4) 500,505
 500  PRINT 17, TIME,(PX(I),I=1,3),(PXD(I),I=1,3),(PXDD(I),I=1,3),PHI,TH
     1ETA,PHID,THETD,PHIDD,THETDD,ALT,V,VMACH,FT,GRVX,GRVY,GRVZ,WGTI,FDR
     2AG,CPI,CNI,AXLM,TRVM,BDOT
 505  IF(SENSE LIGHT 2) 999,EXIT
 TMIN    SLJ       **
      PRINT 16,TIME
      IF (SENSE SWITCH 5) 550, TMIN
 550  SENSE LIGHT 2
      GO TO TMIN
 TEXIT   SLJ       **
```

```
      IF(PX(1)-CHECK)4000,3999,4000
 (3999) RTJ       EQN
C
C                        *** OUTPUT BLOCK ***
C
4000  DO 4001 I=1,3
      PXLD(I)=0.
      DO 4001 J=1,3
4001  PXLD(I)=ROT(J,I)*PXD(J)+PXLD(I)
      CALL ROTATE(BLON,COTHC,0.,PX,XVX)
      XVX(3)=XVX(3)-REL
      CALL ROTATE(0.,-DELTH,-BAZIM,XVX,PXL)
      OUT(1,JP)=TIME
      TP1=RADF(PX)
      TP=ASINF(PX(3)/TP1)
      CALL GEODED(TP,TP1,TP2,OUT(4,JP))
      OUT(2,JP)=TP2/RAD
      OUT(3,JP)=ATANF(PX(2)/PX(1))/RAD
      IF(PX(1)) 4010,4011,4011
4010  OUT(3,JP)=OUT(3,JP)+180.
4011  AN1=OUT(3,JP)*RAD+HALFPI
      AN2=HALFPI-TP2
      CALL ROTATE(AN1,AN2,0.,PXD,PXND)
      OUT(5,JP)=RADF(PXND)
      IF (TIME) 4015,4015,4016
4015  OUT(6,JP)=AAZIM
      OUT(7,JP)=(HALFPI-PHI)/RAD
      GO TO 4019
4016  OUT(6,JP)=ATANF(PXND(1)/PXND(2))/RAD
      IF(PXND(2))4020,4021,4021
4020  OUT(6,JP)=OUT(6,JP)+180.
4021  OUT(7,JP)=ASINF(PXND(3)/OUT(5,JP))/RAD
4019  IF (TIME-12.)  4023,4023,4022
4022  AN1 = SINF((THC+TP)/2.)
      OUT(8,JP)=REE/6076.1033*ACOSF(STHC*PX(3)/TP1+CTHC*COSF(TP)*COSF(
     1ALO2-OUT(3,JP)*RAD))/SQRTF(1.-.00673852*AN1*AN1)
      GO TO 4024
4023  OUT(8,JP)=PXL(1)/6076.1033
4024  OUT(9,JP)=PXL(2)/6076.1033
      RANGE1=OUT(8,JP)
      OUT(10,JP)=OUT(4,JP)/6076.1033
      OUT(11,JP)=RADF(PXLD)
```

```
      OUT(12,JP)=PHI/RAD
      OUT(13,JP)=THETA/RAD
      OUT(14,JP)=FT
      OUT(15,JP)=WGTI*GRAVO
      OUT(16,JP)=RADF(PXDD)/GRAVO
      OUT(17,JP)=.5*RHO*OUT(5,JP)*OUT(5,JP)
      OUT(18,JP)=FDRAG
      OUT(19,JP)=VMACH
      OUT(20,JP)=PXL(1)
      OUT(21,JP)=-PXL(2)
      OUT(22,JP)=PXL(3)
      OUT(23,JP)=PXLD(1)
      OUT(24,JP)=-PXLD(2)
      OUT(25,JP)=PXLD(3)
      IF (KNTRL(7)) 40241,40241,40242
40242 OUT(21,JP)=-OUT(21,JP)
      OUT(24,JP)=-OUT(24,JP)
40241 IF (KNTRL (6)) 4026,4026,4025
4025  CALL WOTF (OUT(1,JP),OUT(25,JP),KPLOT,0)
4026  IF (KNTRL(3)) 4029,4029,4027
4027  AN1 = HALFPI-PHI
      AN2 = HALFPI+THETA
      CALL ROTATE (HALFPI,AN1,AN2,T,TT)
      DO 4028 I=1,3
      PXLDD(I)=0.
      DO 4028  II=1,3
      DC (I,II)=0.
      PXLDD(I) = ROT(II,I)*PXDD(II)+PXLDD(I)
      DO 4028 III= 1,3
4028  DC(I,II)=ROT(I,III)*ROT1(III,II)+DC(I,II)
      WRITE TAPE KBATT,(OUT(I,JP),I=1,25),(PXLDD(I),I=1,3),
     1((DC(I,II),I=1,3),II=1,3),(PX(I),I=1,3),(PXD(I),I=1,3)
4029  IF (JP-KNTRL(5)) 4030,4031,4031
4030  IF(TIME-TSTOP)4059,4031,4031
4031  WRITE OUTPUT TAPE NOT,18,(NAME(I),I=1,10)
      WRITE OUTPUT TAPE NOT,20
      WRITE OUTPUT TAPE NOT,24,((OUT(I,II),I=1,7),II=1,JP)
      WRITE OUTPUT TAPE NOT,18,(NAME(I),I=1,10)
      WRITE OUTPUT TAPE NOT,21
      WRITE OUTPUT TAPE NOT,25,(OUT(1,II),(OUT(I,II),I=8,13),II=1,JP)
      WRITE OUTPUT TAPE NOT,18,(NAME(I),I=1,10)
      WRITE OUTPUT TAPE NOT,22
```

```
      WRITE OUTPUT TAPE NOT,26,(OUT(1,II),(OUT(I,II),I=14,19),II=1,JP)
      WRITE OUTPUT TAPE NOT,18,(NAME(I),I=1,10)
      WRITE OUTPUT TAPE NOT,23
      WRITE OUTPUT TAPE NOT,27,(OUT(1,II),(OUT(I,II),I=20,25),II=1,JP)
      JP=1
4036  IF(TIME-TSTOP) 4060,4037,4037
4037  IF (KNTRL(3)) 4039,4039,4038
4038  END FILE KBATT
4039  IF (KNTRL(6)) 4040,4040,40391
40391 END FILE KPLOT
4040  IF(KNTRL(8)-2) 4071,4070,4071
4070  KIX = KIX + 1
      DO 4073 I = 1,3
      XIMPA(I,KIX) = PX(I)
4073  XDIMP(I,KIX) = PXD(I)
      TFIMP(KIX) = TIME
      RANGE(KIX) =  RANGE1
4071  IF(KNTRL(8)) 5010,5010, 5011
5011  DO 5009 I=1,KIX
      IRA=I
      CALL IMPACT (IRA)
5009  CONTINUE
5010  IF(KNTRL(4))4041,4050,4041
4041  CALL TWOBOD (TIME,PX,PXD)
4050  IF(KNTRL(3)) 4052,4052,4051
4051  CALL SAVF (KBATT)
4052  IF (KNTRL(6)) 4054,4054,4053
4053  CALL SAVF (KPLOT)
4054  CALL ECLOCK(NOT)
      GO TO 100
4059  JP=JP+1
4060  IT=IT-1
      PRTIME=TABLE(IT)
      IF(MODF(TABLE(IT+1),DT))270,TEXIT,270
 RDF     LIB       RDF
      END
C
C
```

```
      SUBROUTINE SETTAB (NS,TMI,TMCC,TMBO,DT,TABLE,IX)
C             THIS SUBROUTINE SETS UP THE PRINT TIME TABLE
R     DIMENSION TMI(10),TMCC(10),TMBO(10),TABLE(500)
      NSS=NS-1
      IX=(TMBO(NS)+1.)/DT
      TABLE(IX)=0.
      DO 1 I=2,IX
      K=IX-I+1
 1    TABLE(K)=TABLE(K+1)+DT
      DO 2 I=2,NS
      U=TMI(I)
         RTJ        INS
 2    TMI(I)=U
      DO 3 I=1,NS
      U=TMBO(I)
         RTJ        INS
 3    TMBO(I)=U
      DO 4 I=1,NSS
      U=TMCC(I)
         RTJ        INS
 4    TMCC(I)=U
 5    RETURN
      DUM=MODF(DUM,1.)
 INS     SLJ        **
         SIL      6 INS3
         LDA        U
         LIL      6 IX
+        THS      6 TABLE
         INI      6 -1
         INI      6 1
         FSB      6 TABLE
         AJP      1 INS1
         LDA      6 TABLE
         LDQ        DT
         RTJ        MODF
+        RTJ        ERROR.
+        AJP      N *+3
         LDA      6 TABLE
         FAD        INS3+1
         STA      6 TABLE
         STA        U
+        LIL      6 INS3
```

```
         SLJ        INS
INS1     SIU      6 *+3
         LIL      6 IX
         LDA      6 TABLE
         INI      6 1
         STA      6 TABLE
+        ISK      6 **
         SLJ        INS2
         LIU      6 *-1
         LDA        U
         STA      6 TABLE
         RAO        IX
         LIL      6 INS3
         SLJ        INS
INS2     INI      6 -3
         SLJ        INS1+1
INS3     ZRO
         DEC        .0001
         END
```

```
      SUBROUTINE ECLOCK(ITAPE)
C             THIS PROGRAM COMPUTES THE MACHINE TIME FOR ONE TRAJECTORY RUN
      DIMENSION X(3)
+        EXF      0 02000B               STOP THE CLOCK
         LDA       0
         SCM       MASK1
         FAD       ZERO                  FLOAT THE CLOCK BINARY COUNT
         FDV       F216M                 CONVERT TO HOURS
         STA       X+1
         ENA       X+1
         RTJ       INTF                  GET WHOLE NUM OF HOURS
+        NOP
+        STA       X                     INTEGER HOURS
         LDA       X+1
         FSB       X
         FMU       F60                   GET NUM OF MINUTES
         STA       X+2
         ENA       X+2
         RTJ       INTF
+        NOP
+        STA       X+1                   INTEGER MINUTES
         LDA       X+2
         FSB       X+1
         FMU       F60                   GET NUM OF SECONDS
         STA       X+2                   INTEGER AND FRACTIONAL SECONDS
         NOP
      WRITE OUTPUT TAPE ITAPE,2,(X(I), I=1,3)
 2    FORMAT(16H0ELAPSED TIME = F3.1,7H HOURS F4.1,10H  MINUTES F6.3,8H
     1SECONDS)
      RETURN
 MASK1   OCT       2044000000000000
 F60     DEC       60.
 F216M   DEC       216000.
 ZERO    DEC       0
 INTF    LIB       INTF
      END
```

```
      SUBROUTINE SCLOCK
+        EXF      0 02000B               STOP THE CLOCK
         ENA        0
         STA        0B
-        EXF      0 01000B               START THE CLOCK AFTER SETTING=0
      RETURN
         END
```

```
      SUBROUTINE TWOBOD (TBO,A1,A2)
C             THIS SUBROUTINE COMPUTES A KEPLERIAN TRAJECTORY ALONG WITH
C             LOOK ANGLES FROM VARIOUS STATIONS
      DIMENSION NAME(10),ANS(5),PRTIME(2000),THETA(2000), PHI(2000),
     1ALT(2000),DELTAT(5),ENDT(5),TAG(3,50),SX(3),FX(3),XSTH(50)
     2,DUMTB(9),A1(3),A2(3),XSPH(50),XSHT(50),A1A(10,10),A1B(10,10),N1C(
     310)
      COMMON DUMTB,NAME,KBATT,NOT,ORBT,DELTAT,ENDT,NOST,TAG,XSTH,
     1XSPH,XSHT,Z2,SLAL,CLAL,XLLO,RLAU,A1A,A1B,N1C,NAT
      SECF(X)=MODF(X,60.)
      FMINF(X)=MODF(INTF(X/60.),60.)
      HOURF(X)=INTF(X/3600.)
      REE=3443.9255
      WE=7.292115E-5
      E=.08181333
      GM=1.4076427E+16
      GM1=GM/6076.1033
      PI=3.141592654
      RAD=PI/180.0
      F=1.0/298.3
      ER=0.000005
      POP=E*E/(1.0-E*E)
      SLOL=SINF(XLLO)
      CLOL=COSF(XLLO)
      XLAU=CLAL*CLOL
      YLAU=CLAL*SLOL
      ZLAU=SLAL
      AX=A1(1)
      AY=A1(2)
      AZ=A1(3)
      AXDOT=A2(1)-AY*WE
      AYDOT=A2(2)+AX*WE
      AZDOT=A2(3)
      RF=SQRTF(AX*AX+AY*AY+AZ*AZ)
      R=RF/6076.1033
      VI=SQRTF(AXDOT*AXDOT+AYDOT*AYDOT+AZDOT*AZDOT)
  102 H=R*VI*VI/GM1
      IF(H-2.0)104,110,110
  110 WRITE OUTPUT TAPE NAT,200,(NAME(I),I=1,9)
      WRITE OUTPUT TAPE NAT,202
      GO TO 190
  104 SMAJ=R/(2.0-H)
```

```
      HX=AY*AZDOT-AZ*AYDOT
      HY=AZ*AXDOT-AX*AZDOT
      HZ=AX*AYDOT-AY*AXDOT
      HH=HX*HX+HY*HY+HZ*HZ
      H1=SQRTF(HH)
      SLANT =ACOSF(HZ/H1)
      CAPO=ATANF(-HX/HY)
      IF(HY)106,106,105
  105 CAPO=PI+CAPO
  106 RDOT=(AX*AXDOT+AY*AYDOT+AZ*AZDOT)/RF
      EX=SQRTF(1.0-H1*H1/(SMAJ*GM*6076.1033)
      CTRUA=H1/RF*H1/GM-1.0
      STRUA=H1*RDOT/GM
      TRANOM=ATANF(STRUA/CTRUA)
      IF(CTRUA)111,112,112
  111 TRANOM=PI+TRANOM
  112 CMU=(AY*HX-AX*HY)/H1
      AMU=ATANF(AZ/CMU)
      IF(CMU)1120,1121,1121
 1120 AMU=PI+AMU
 1121 SMOM=AMU-TRANOM
      SMIN=SMAJ*SQRTF(1.0-EX*EX)
      PERIOD=4.184624E-4*SMAJ**1.5
      ETA=PERIOD*60./(2.*PI)
      APOGEE=SMAJ*(1.0+EX)-REE
      PERGEE=SMAJ*(1.0-EX)-REE
      XX=SQRTF((1.0-EX)/(1.0+EX))
      EXANOM=2.0*ATANF(XX*TANF(TRANOM/2.0))
      TOPP=(EXANOM-EX*SINF(EXANOM))*ETA
      WRITE OUTPUT TAPE NAT,200,(NAME(I),I=1,9)
      ANS(1)=SLANT/RAD
      ANS(2)=SMOM/RAD
      ANS(3)=CAPO/RAD
      ANS(4)=TRANOM/RAD
      ANS(5)=EXANOM/RAD
      WRITE OUTPUT TAPE NAT,201,SMAJ,SMIN,EX,PERIOD,APOGEE,PERGEE,(ANS(I
     1),I=1,5)
C
C          TRAJECTORY POINTS
      IF(PERGEE)1123,1124,1124
 1123 TRIMP=ACOSF(((1.-EX*EX)*SMAJ/REE-1.)/EX)
      EXIMP=2.*PI-2.*ATANF(XX*TANF(TRIMP/2.))
```

```
      TIMP=(EXIMP-EX*SINF(EXIMP))*ETA
      GO TO 1127
 1124 IF(ORBT)1125,1125,1126
 1125 TIMP=PERIOD*60.
      GO TO 1127
 1126 TIMP=ORBT*PERIOD*60.
 1127 SSLANT=SINF(SLANT)
      CSLANT=COSF(SLANT)
      E1=EXANOM
      TIME=TOPP
      PN=-1.0
      J=0
      TINCR=0.0
      MEX=0
      MM=1
      GO TO 117
  113 TINCR=60.*(INTF(TBO/60.)+1.)-TBO
      MM=2
      GO TO 117
  114 MM=3
  115 MEX=MEX+1
      TINCR=DELTAT(MEX)
  116 IF(TIME+TBO-TOPP-ENDT(MEX)+.01)117,115,115
  117 J=J+1
      TIME=TIME+TINCR
      IF(TIME-TIMP)118,118,1171
 1171 TIME=TIMP
  118 XM=TIME/ETA
  119 E2=E1-(E1-EX*SINF(E1)-XM)/(1.0-EX*COSF(E1))
      IF(ABSF(E2-E1)-ER) 121,121,120
  120 E1=E2
      GO TO 119
  121 TRUNU=2.0*ATANF(TANF(E2/2.0)/XX)
      R=SMAJ*(1.0-EX*EX)/(1.0+EX*COSF(TRUNU))
      VELOC=SQRTF(GM1*(2./R-1./SMAJ))
      FPA=ATANF(EX*SINF(TRUNU)/(1.0+EX*COSF(TRUNU)))/RAD
      SX(1)=COSF(TRUNU)
      SX(2)=SINF(TRUNU)
      SX(3)=0.0
      CALL ROTATE(-SMOM,-SLANT,-CAPO,SX,FX)
      THETA(J)=ATANF(FX(3)/SQRTF(FX(1)*FX(1)+FX(2)*FX(2)))
      PHIX=ATANF(FX(2)/FX(1))
```

```
      IF(FX(1))122,123,123
122   PHIX=PI+PHIX
123   PHIX=PHIX-WE*(TIME-TOPP)
      PHI(J)=MODF(PHIX,2.*PI)
      STH=SINF(THETA(J))
      ALT(J)=R
C
C     GEOCENTRIC TO GEODETIC (TRAJECTORY POINTS)
124   S1=STH
      S2=SINF(2.0*THETA(J))
      S3=SINF(4.0*THETA(J))
      S4=F*S2+F*F*S3*(REE/R-0.25)
      S5=1.0-F*S1*S1-0.5*F*F*S2*S2*(REE/R-0.25)
      PRLAT=THETA(J)+ASINF(REE*S4/R)
      PRALT=R-REE*S5
      PRLAT=PRLAT/RAD
      PRLON=PHI(J)/RAD
      PRTIME(J)=TIME+TBO-TOPP+.01
      XIM=COSF(THETA(J))*COSF(PHI(J))
      YIM=COSF(THETA(J))*SINF(PHI(J))
      ZIM=SINF(THETA(J))
      ARA=(RLAU/6076.1033+REE*S5)/2.
      ZIZ=XLAU*XIM+YLAU*YIM+ZLAU*ZIM
      RANGE=ARA*ACOSF(ZIZ)
      IF(KBATT)1242,1242,1241
1241  WRITE TAPE KBATT,(FX(I),I=1,3),PRTIME(J),PRLAT,PRLON,PRALT,VELOC,
     1FPA,R
1242  SEC=SECF(PRTIME(J))
      FMIN=FMINF(PRTIME(J))
      HOUR=HOURF(PRTIME(J))
      PN=PN+1.0
      IF(MODF(PN,Z2)) 126,125,126
 125  WRITE OUTPUT TAPE NAT,200,(NAME(I),I=1,9)
      WRITE OUTPUT TAPE NAT,205
 126  WRITE OUTPUT TAPE NAT,206,HOUR,FMIN,SEC,PRALT,PRLAT,PRLON,VELOC,
     1FPA,RANGE
      IF(TIME-TIMP)1261,127,127
1261  GO TO(113,114,116),MM
C
C          LOOK ANGLE
127   DO 135 IT=1,NOST
      STH = XSTH(IT)*RAD
```

```
      SPH = XSPH(IT)*RAD
      SHT = XSHT(IT)/6076.1033
C
C     GEODETIC TO GEOCENTRIC (STATION)
      XX1=SINF(STH)
      XX2=COSF(STH)
      C=REE/SQRTF(1.0-E*E*XX1*XX1)
      S=C-C*E*E
      XX3=S+SHT
      XX4=C+SHT
      STHC=ATANF(XX1/XX2*XX3/XX4)
      RS=SQRTF(XX1*XX1*XX3*XX3+XX2*XX2*XX4*XX4)
      PN=-1.0
      K=1
131   IF(PRTIME(K)-TBO+TOPP-TIMP)1311,135,135
1311  DPHI=PHI(K)-SPH
      ST=SINF(THETA(K))
      CT=COSF(THETA(K))
      X=ALT(K)*CT*COSF(DPHI)-RS*COSF(STHC)
      Y=ALT(K)*CT*SINF(DPHI)
      Z=ALT(K)*ST-RS*SINF(STHC)
      XP=X*XX1-Z*XX2
      ZP=X*XX2+Z*XX1
      ELEV=ATANF(ZP/SQRTF(XP*XP+Y*Y))
      AZIM=ATANF(-Y/XP)
      IF(XP)132,132,130
130   AZIM=PI+AZIM
132   IF(AZIM)1321,1322,1322
1321  AZIM=AZIM+2.*PI
1322  XAZ=AZIM/RAD
      XEL=ELEV/RAD
      SEC=SECF(PRTIME(K))
      FMIN=FMINF(PRTIME(K))
      HOUR=HOURF(PRTIME(K))
      RE=REE/SQRTF(1.+POP*ST*ST)
      PRALT=ALT(K)-RE
      PRLAT=THETA(K)/RAD
      PRLON=PHI(K)/RAD
      SLR=SQRTF(X*X+Y*Y+Z*Z)
      SP6=SPH+WE*(PRTIME(K)-PRTIME(1))
      XX=X*COSF(SP6)-Y*SINF(SP6)
      YY=X*SINF(SP6)+Y*COSF(SP6)
```

```
      ANT=ACOSF((XX*AXDOT+YY*AYDOT+Z*AZDOT)/(SQRTF(X*X+Y*Y+Z*Z)*SQRTF(AX
     1DOT*AXDOT+AYDOT*AYDOT+AZDOT*AZDOT)))/RAD
      PN=PN+1.0
      IF(MODF(PN,Z2)) 134,133,134
133   WRITE OUTPUT TAPE NAT,200,(NAME(I),I=1,9)
      WRITE OUTPUT TAPE NAT,209,(TAG(I,IT),I=1,3),XSTH(IT),XSPH(IT),XSHT
     1(IT)
134   WRITE OUTPUT TAPE NAT,210,HOUR,FMIN,SEC,PRALT,PRLAT,PRLON,
     1XAZ,XEL,SLR,ANT
      K=K+1
      GO TO 131
135   CONTINUE
190   IF(KBATT)192,192,191
191   END FILE KBATT
192   RETURN
200   FORMAT(1H19A8)
201   FORMAT(39H0              ORBITAL ELEMENTS            ///17H   SEMIMAJO
     1R AXISE28.8,5H N.M.//17H   SEMIMINOR AXISE28.8,5H N.M.//15H   ECCE
     2NTRICITYE30.8//9H   PERIODE36.8,5H MIN.//18H   APOGEE ALTITUDEE27.
     38,5H N.M.//19H   PERIGEE ALTITUDEE26.8,5H N.M.//14H   INCLINATIONE
     431.8,5H DEG.//22H   ARGUMENT OF PERIGEEE23.8,5H DEG.//25H   LONGIT
     5UDE OF ASCENDING/23H   NODE AT BURNOUT TIMEE22.8,5H DEG.//23H   IN
     6ITIAL TRUE ANOMALYE22.8,5H DEG.//28H   INITIAL ECCENTRIC ANOMALYE1
     77.8,5H DEG.)
202   FORMAT(16H0VEHICLE ESCAPES)
205   FORMAT(39H        KEPLERIAN TRAJECTORY              /52X22HINERTIAL
     1   INERTIAL /87H   H  M  S  ALTITUDE(NM)    LATITUDE   LONGITUDE
     2 VEL.(FPS)  FLT PATH ANG    RANGE(NM)/)
206   FORMAT(1H F3.0,2F3.0,E14.5,2F12.5,F12.2,F13.5,F13.2)
209   FORMAT(13H0  STATION - 3A8/26XF12.3,6H N.LATF12.3,7H E.LONGF12.0,4
     1H FT.//20X23HTRAJECTORY (GEOCENTRIC)13X11HLOOK ANGLES7X5HSLANT5X6H
     2ASPECT/6X4HTIME4X8HALTITUDE4X8HLATITUDE3X9HLONGITUDE7X7HAZIMUTH2X9
     3HELEVATION3X5HRANGE5X5HANGLE/4X18HH  M  S    N.MILES6X3HDEG9X3HDEG
     412X3HDEG7X3HDEG8X2HNM7X3HDEG)
210   FORMAT(F5.0,2F3.0,F10.0,2F12.2,5X2F10.3,F9.0,F10.2)
SQRTF    EQU      7660B
         END
```

```
      SUBROUTINE IMPACT(JJ)
C        THIS SUBROUTINE COMPUTES THE TRAJECTORY OF THE EMPTY STAGES
C
C                   * * * DIMENSION BLOCK * * *
C
      DIMENSION XX(3),XXD(3),CDM(10,10),XMACH(10,10),AL(26),CBLACK(150),
     1NAME(10),OUT(8,42),WD(3),DRA(10),XMN(10),NN(10),DELPR(10),VD(3)
     2ACODES(6),DUMB1(3,3),AC(5),AD(5),AG(3,50),AH(50),AK(50),AI(50)
     3XIMPA(3,10),XDIMP(3,10),TFIMP(10),RANGE(10)
C
C                   * * * COMMON BLOCK * * *
C
      COMMON DUMB1,NAME,KAA,KAZ,AB,AC,AD,KAF,AG,AH,AK,AI,Z2,AM,AN,AO,
     1AP,XMACH,CDM,NN,NOT,RANGE,XIMPA,XDIMP,TFIMP,DELPR
C
C                   * * * EQUIVALENCE BLOCK * * *
C
      EQUIVALENCE (AL(1),ICODE),(AL(2),ACODE),(AL(3),TRERR),(AL(4),SMIN)
     1,(AL(5),SMAX),(AL(6),NOE),(AL(7),SS),(AL(8),T),(AL(9),XD),(AL(10),
     2YD),(AL(11),ZD),(AL(12),X),(AL(13),Y),(AL(14),Z),(AL(15),XDD),(AL(
     316),YDD),(AL(17),ZDD),(AL(18),XD1),(AL(19),YD1),(AL(20),ZD1),(AL(2
     41),ACODES)
C                   * * * FORMAT BLOCK * * *
 1    FORMAT(I2)
 2    FORMAT(7F10.0)
 3    FORMAT(21H1  INTEGRATION ERROR 016)
 4    FORMAT(19H0   CRT FAILURE AT F9.4,7H SEC***)
 5    FORMAT(2H1 10A8)
 6    FORMAT(3X4HTIME4X8HLATITUDE4X9HLONGITUDE4X8HALTITUDE4X5HRANGE/4X3H
     1SEC7X3HDEG9X3HDEG9X2HNM9X2HNM/)
 7    FORMAT(F8.2,2F12.4,F12.2,F10.2)
 8    FORMAT(3X4HTIME4X8HVELOCITY6X3HFPA6X7HAZIMUTH/4X3HSEC6X3HFPS9X3HDE
     1G8X3HDEG/)
 9    FORMAT(F8.2,F11.1,F10.2,F12.2)
 10   FORMAT (6A8,6HSTAGE=I2,2F12.6)
 11   FORMAT( 10X7HSTAGE  I2)
C                   * * * CONSTANTS * * *
      J6=Z2
      T=TFIMP(JJ)
      X=XIMPA(1,JJ)
      Y=XIMPA(2,JJ)
      Z=XIMPA(3,JJ)
```

```
      XD=XDIMP(1,JJ)
      YD=XDIMP(2,JJ)
      ZD=XDIMP(3,JJ)
      XX(1)=X
      XX(2)=Y
      XX(3)=Z
      XXD(1)=XD
      XXD(2)=YD
      XXD(3)=ZD
      JP=1
      ICODE=3
      TRERR=1.E-3
      ACODE=1.0
      ACODES(1)=1.
      ACODES(2)=1.
      ACODES(3)=1.
      ACODES(4)=1.
      ACODES(5)=1.
      ACODES(6)=1.
      DT=1.
      SMIN=.01
      SMAX=1.
      NOE=6
      SS=1./16.
      H=T-INTF(T)
      H=1.-H
      IF(H) 50,51,50
   50 PRTIM1=T+H
      GO TO 52
   51 PRTIM1=T+DELPR(JJ)
   52 AE=20925647.
      W=7.292115E-5
      W2=W*W
      XK2=AE*AE*.00162342
      GM=1.4076427E16
      PI=3.1415927
      RAD=PI/180.
      PE=.00673852
      VEL=SQRTF(XD*XD+YD*YD+ZD*ZD)
      R2=X*X+Y*Y+Z*Z
      R=SQRTF(R2)
      THET=ATANF(Z/SQRTF(X*X+Y*Y))
```

```
      SIT=SINF(THET)
      ALT=R-AE/SQRTF(1.+PE*SIT*SIT)
      N=NN(JJ)
      DO 30 I=1,N
      K=N-I+1
      DRA(K)=CDM(I,JJ)
 30   XMN(K)=XMACH(I,JJ)
C
C                 * * * INTEGRATION CALL ROUTINE * * *
C
         RTJ      TEX1
(999)    RTJ      ADAMS
         ZRO      EQN1
         ZRO      TEX1
         ZRO      PRTIM1
         ZRO      AL
         ZRO      CBLACK
         ZRO      EXIT1
         ZRO      TMIN1
         STA      ERR
      PRINT 3,ERR
C                 * * * EQUATIONS OF MOTION * * *
EQN1     SLJ      **
      XD1=XD
      YD1=YD
      ZD1=ZD
      VEL=SQRTF(XD*XD+YD*YD+ZD*ZD)
      R2=X*X+Y*Y+Z*Z
      R=SQRTF(R2)
      ARG=(1.+3.*XK2/R2-15.*XK2*Z*Z/R2/R2)/R2/R
      GX=-GM*ARG*X
      GY=-GM*ARG*Y
      GZ=-GM*(ARG+6.*XK2/R2/R2/R)*Z
      THET=ATANF(Z/SQRTF(X*X+Y*Y))
      SIT=SINF(THET)
      ALT=R-AE/SQRTF(1.+PE*SIT*SIT)
      IF(ALT) 79,81,81
 79   JP=JP-1
      GO TO 4031
 81   IF(ALT-500000.) 82,82,83
(82)     LDA      ALT
         RTJ      ATMOS
```

```
TEMP      OCT
PRES      OCT
RHO       OCT
VA        OCT
          NOP
          NOP
 85    FMN=VEL/VA
       DRAG = INTERPF(FMN,N,2,XMN,DRA)
       DM=.5*RHO*VEL*VEL*DRAG
       GO TO 84
 83    DM=0.
 84    XDD=-DM*XD/VEL+GX+2.*W*YD+W2*X
       YDD=-DM*YD/VEL+GY-2.*W*XD+W2*Y
       ZDD=-DM*ZD/VEL+GZ
       GO TO EQN1
EXIT1     SLJ        **
 502   CHECK=X
       IF(SENSE LIGHT 2) 999, EXIT1
TMIN1     SLJ        **
       PRINT 4,T
       IF(SENSE SWITCH 5)550,TMIN1
 550   SENSE LIGHT 2
       GO TO TMIN1
TEX1      SLJ        **
       IF(X-CHECK) 4000,3999,4000
(3999)    RTJ        EQN1
C     COMPUT THE OUTPUT BLOCK
 4000  OUT(1,JP)=T
       CALL GEODED(THET,R,XLAT,XALT)
       OUT(2,JP)=XLAT/RAD
       OUT(4,JP)=XALT/6076.1033
       OUT(3,JP)=ATANF(Y/X)/RAD
       IF(X) 6000,6000,6001
 6000  OUT(3,JP)=OUT(3,JP)+180.
 6001  OUT(6,JP)=VEL
       VD(1)=XD
       VD(2)=YD
       VD(3)=ZD
       AN1=OUT(3,JP)*RAD+PI/2.
       AN2=PI/2.-XLAT
       CALL ROTATE(AN1,AN2,0.,VD,WD)
       OUT(7,JP)=ATANF(WD(3)/SQRTF(WD(1)*WD(1)+WD(2)*WD(2)))/RAD
```

```
      OUT(8,JP)=ATANF(WD(1)/WD(2))/RAD
      IF(WD(2)) 6002,6002,6003
 6002 OUT(8,JP)=OUT(8,JP)+180.
 6003 COSRA=(X*XX(1)+Y*XX(2)+Z*XX(3))/SQRTF(X*X+Y*Y+Z*Z)/SQRTF(XX(1)*XX(
     11)+XX(2)*XX(2)+XX(3)*XX(3))
      OUT(5,JP) = RANGE(JJ) + 3440.*ACOSF(COSRA)
      IF (JP-J6) 4030,4031,4031
 4030 IF(ALT)4031,4059,4059
C     WRITE OUTPUT BLOCK
 4031 WRITE OUTPUT TAPE NOT,5,(NAME(I),I=1,10)
      WRITE OUTPUT TAPE NOT, 11,JJ
      WRITE OUTPUT TAPE NOT,6
      WRITE OUTPUT TAPE NOT,7,((OUT(I,II),I=1,5),II=1,JP)
      WRITE OUTPUT TAPE NOT,5,(NAME(I),I=1,10)
      WRITE OUTPUT TAPE NOT, 11,JJ
      WRITE OUTPUT TAPE NOT,8
      WRITE OUTPUT TAPE NOT,9,(OUT(1,II),(OUT(I,II),I=6,8),II=1,JP)
      IF(ALT) 4037,4061,4061
 4061 JP=1
      GO TO 4060
C    STORE IMPACT POINT FOR PUNCH AND RETURN
 4037 PUNCH 10,(NAME(I),I=1,6),JJ,OUT(2,JP), OUT(3,JP)
      RETURN
 4059 JP=JP+1
 4060 PRTIM1=PRTIM1 + DELPR(JJ)
 4064 GO TO TEX1
 ADAMS   EQU        6000B
 ATMOS   EQU        7300B
 SQRTF   EQU        7660B
RDF      LIB        RDF
      END
C
C
```

```
      SUBROUTINE GEODED(A,B,C,D)
C              THIS SUBROUTINE CONVERTS THE GEOCENTRIC COORDINATES TO
C              GEODETIC COORDINATES
      S1=SINF(A)
      S2=SINF(2.*A)
      S3=SINF(4.*A)
      S4=20925647./B-.25
      S5=S2/298.3+S3/88982.89*S4
      S6=1.-S1*S1/298.3-S2*S2*S4/177965.78
      C=A+ASINF(20925647.*S5/B)
      D=B-20925647.*S6
      RETURN
      END
```

```
      SUBROUTINE ROTATE (PHI,THETA,PSI,U,V)
C     THIS SUBROUTINE GIVES A ROTATION OF CARTESIAN COORDINATES
      DIMENSION U(3),V(3),A(3,3),BLOCK(700)
      COMMON A,BLOCK
      T1=SINF(PHI)
      T2=SINF(THETA)
      T3=SINF(PSI)
      T4=COSF(PHI)
      T5=COSF(THETA)
      T6=COSF(PSI)
      A(1,1)=T6*T4-T5*T1*T3
      A(1,2)=T6*T1+T5*T4*T3
      A(1,3)=T3*T2
      A(2,1)=-T3*T4-T5*T1*T6
      A(2,2)=-T3*T1+T5*T4*T6
      A(2,3)=T6*T2
      A(3,1)=T2*T1
      A(3,2)=-T2*T4
      A(3,3)=T5
1     DO 2 I=1,3
      V(I)=0.0
      DO 2 J=1,3
2     V(I)=A(I,J)*U(J)+V(I)
5     RETURN
         END       10050B
```

```
         ORG    6000B
         REM    SOL OF DIFF EQ        NUMERICAL INTEGRATION
ADAMS    SLJ    0                     RUNGE KUTTA STARTER FOR
         SIU  1 AD+13                 ADAMS MOULTON DIFF EQ
         SIL  2 AD+13                 EVALUATOR
         LIU  1 ADAMS                 BETA+1 IN INDEX 1
         LDA  1 1                     C(BETA+2) INAC
         ENQ    0
         STQ    AD+10
         SAL    AD+10                 COMMON TO AD+10
         ARS    24
         STQ    AD+9
         SAL    AD+9                  DATA TO AD+9
         LDA    AD+9                  DATA IN AC
         ENI  2 1                     1 TO INDEX 2
AD50     INA    1
         STA  2 AD+13                 DATA+J TO AD+13+J
+        ISK  2 7                     J IS 1 THROUGH 7
         SLJ    AD50
         LDA  7 AD+18                 N
         AJP  N AD51                  JUMP N NOT ZERO
ADERR    LIU  1 AD+13                 ERROR N IS ZERO
         LIL  2 AD+13
         LDA    ADTHR
         RAD    ADAMS
         SLJ    ADAMS
ADTHR    OCT    300000000             ERROR RETURN WROMG SETUP
AD51     AJP  P AD52
AD53     ENA    0
         SLJ    ADERR                 ERROR N -
ADNMAX   DEC    200
AD52     STA    AD+12                 N TO AD+12
         INA    -1
         STA    AD+21                 N-1 TO AD+21
         SUB    ADNMAX
         AJP  P AD53                  N TO LARGE JUMP ERROR
         LDA  7 AD+9
         STA    AD+6                  CODE TO AD+6
         AJP  P AD55                  ERROR CODE MINUS
AD56     ENA    2                     SET AC TO 2
         SLJ    ADERR                 GO TO ERROR RETURN
AD55     INA    -4
```

```
AJP     P AD56       IF CODE MORE THAN 3 ERROR
LDQ     7 AD+19      DT
ENA       1          AC IS +1
QJP     M ADERR      ERROR DT MINUS
QLS       12
QJP     P ADERR      ERROR IF DT NOT FLOATING NUM
LDA     1 0          C(BETA+1) TO AC
SAL       ADTI       T TO LOWER ADDRESS AD+I
ARS       24
SAU       ADEXT      TEXIT ADDRESS
SAU       ADEXT1
SAL       ADEXT2
LDA     1 2          C(BETA+3) IN AC
SAU       ADTMC
SAU       ADTMIN
ARS       24
SAU       ADEXIT     EXIT ADDRESS
LDA     1 -1         C(BETA) IN AC
SAU       ADY        DERIV ADDRESS
SAL       ADY1
SAU       ADY3
SAU       ADY4
SAU       ADY5
SAU       ADY6
SCL       ADMSP
STA       AD+5
LDA       AD+20      DATA+7 ADDRESS TK
SAU       ADTS
SAU       ADTXS
SAU       ADXDP
SAL       ADTXR
INA       1
SAL       ADXR       DATA+8 ADDRESS YK
SAU       ADXR4
SAU       ADXR5
SAL       ADXR3
SAL       ADXC
SAU       ADXP
SAU       ADXMK
SAU       ADXH
SAL       ADXP3
SAU       ADXP4
```

```
ADD     AD+12          DATA+W+8 ADDRESS TK
SAU     ADK2
SAU     ADK3
SAU     ADYK
SAL     ADYP
SAL     ADYR1
SAU     ADYS
SAU     ADYM
SAU     ADYAS
ADD     AD+12          DATA+2N+8 ADDRESS AI
SAL     ADA
SAU     ADA1
LDA     AD+10          COMMON, ADDRESS XP
SAL     ADXP
SAU     ADXP2
SAU     ADXP3
SAL     ADXP4
SAL     ADXPS
ADD     AD+12          COMMON+N, ADRESS TK
SAL     ADTK
SAL     ADTS
SAU     ADXS
SAL     ADXS1
SAU     ADTDT
SAU     ADTXR
SAL     ADTIRK         STANDARD COMMON
INA     1              COMMON+N+1 ADDRESS XK
SAL     ADDX
SAL     ADXM
SAL     ADXH
SAU     ADXR
SAL     ADXR1
SAL     ADXR2
SAU     ADXR3
SAL     ADXXS
SAU     ADXPS
ADD     AD+12          COMMON+2N+1
SAU     ADTC           ADDRESS LEAST SIGN PART TK
SAU     ADTC1
SAU     ADTC2          TWO CARDS ONE LINE
INA     1
ADD     AD+12          COMMON+3N+2
```

```
SAU     ADDT       ADDRESS DTK
SAU     ADDT1
SAU     ADDT2
SAL     ADTC
SAU     ADDT3
SAL     ADDT3
INA     1          COMMON+3N+3
SAU     ADDX       ADDRESS DXK
SAL     ADDX1
SAL     ADDX2
SAU     ADDX2
ADD     AD+12      COMMON+UN+3
SAL     ADK2       ADDRESS K2
SAU     ADK22
ADD     AD+12      COMMON+5N+3
SAL     ADK3       ADL'RESS K3
SAL     ADK22
ADD     AD+12      COMMON+6N+3
SAL     ADXS       ADDRESS TK-1  OR YAM-7
SAU     ADTIRK     ADDR+7
SAU     ADXS1
STA     ADDR+7
INA     1          COMMON+6N+4
SAU     ADXXS      ADDRESS XK-1
ADD     AD+12      COMMON+7N+4
SAL     ADDT       ADDRESS SAVE DT,DX
SAL     ADDT1      AND YAM-6
STA     ADDR+6
ADD     AD+12
INA     1          COMMON+8N+5
STA     ADDR+5     ADDRESS YAM-5
ADD     AD+12      COMMON+9N+5
STA     ADDR+4     ADDRESS YAM-4
SAL     ADYRH
ADD     AD+12      COMMON+10N+5
STA     ADDR+3      ADDRESS YAM-3
ADD     AD+12      COMMON+11N+5
STA     ADDR+2     ADDRESS YAM-2
SAU     ADYRH
ADD     AD+12      COMMON+12N+5
STA     ADDR+1     ADDRESS YAM-1
ADD     AD+12      COMMON+13N+5
```

```
         STA       ADDR                 ADDRESS YAM
         LIL     1 AD+12                N TO INDEX 1
         ENA       0
ADTC2    STA     1 0
         IJP     1 ADTC2
         ENA       6
         STA       AD+1                 RKH TO 6
         STA       AD+3                 AMH TO 6
         ENA       0
         STA       AD                   RKC TO 0
         STA       AD+2                 AMC TO 0
         STA       AD+4                 RKC TO 0
         LDA       AD+5
         AJP     Z AD17                 NO PRINT IF P IS ZERO
         SLJ        AD17                CARD LOU ADDED FOR NO PRINTING
ADF1     DEC       1                    SEE IF A +
ADMSP    OCT       77777777777077777    CLEAR ALL BUT P IN BETA
ADDP     LIL     1 AD+12                N TO INDEX 1
         ENA       0                    CLEAR RKD
         STA       AD+4
ADTC     LDA     1 0                    BEGIN DP LOOP
         FAD     1 0                    DT + TK LOWER IS B IN TEMP+1
ADTDT    LDQ     1 0
         STQ       ADTEMP               TK UPPER IS C IN ADTEMP
         STA       ADTEMP+1             STORE B
         ENI     2 0
         QJP     P AD70
         ENI     2 1                    TK UPPER IS - SET INDEX2,1
AD70     FAD       ADTEMP               TK UPPER IS + SET INDEX2,0
         AJP    P  AD74
         INI     2 -1
AD74     STA       ADTEMP+2             TK UPPER + DT + TK LOWER
         ENQ       0                    DO SINGLE PRECISION IF SIGN ALT
         IJP     2 ADXDP                JUMP IF SIGN ALT
         LDA       ADTEMP               C
         AJP     P AD63
         ENQ       1
         SCM       ADA7                 ABS VALUE C
AD63     STQ       ADSC                 STORE SIGN C
         ENQ       0
         LLS       12                   POWER TO MQ
         ARS       3
```

```
        SCL     ADMS
        STA     ADCC2
        LDL     AD3777          EXP TO AC
+       THS     AD1777
        SLJ     AD64            JUMP+EXP
        SCM     ADNB            EXTEND SIGN - EXP
AD65    STA     ADCE            STORE + EXP
        LDA     ADTEMP+1
        ENQ     0               B IN AC
        AJP   P AD60
        ENQ     1               1 TO MQ
        SCM     ADA7            ABS VALUE B
AD60    STQ     ADSB            STORE SIGN B
        ENQ     0
        LLS     12
        ARS     3
        SCL     ADMS            CLEAR LEAD 3 BITS
        STA     ADB             STORE CHAR P
        LDL     AD3777
+       THS     AD1777
        SLJ     AD61
        SCM     ADNB            ENTEND SIGN NEG POWER
AD62    STA     ADBE            STORE POWER B
        SUB     ADCE            POWER B - POWER C
        AJP   M AD71            IF NEG POWERS IS GREATER
AD67    LDA     ADTEMP+2        IF C SAME, OR LESS
        ENQ     0
        SLJ     ADXDP           CLEAR LESSER
AD61    INA     -2000B
        SLJ     AD62            REMOVE BIAS EXP B
AD64    INA     -2000B
        SLJ     AD65            REMOVE BIAS EXPC
AD71    SCM     ADA7            EXP (-EXP B, ALSO EXP C LARGER
        SAU     AD69            SHIFT IN ADDRESS
+       THS     AD72D
        SLJ     AD67            POWER DIFF GREATER THAN, SKIP
        ENQ     0
        LDA     ADB             CHAR B
AD69    LRS     0
        STA     ADB             UPPER BITS SCALED B
        STQ     ADL             LOWER BITS SCALED B
        LDA     ADSB
```

```
        SCM      ADSC
        AJP    N AD66       JUMP IF SIGN B DIFF FROM SIGN C
        LDA      ADCC2
        ADD      ADB
        LDQ      ADL        ADD CHAR B TO CHAR C
AD75    SCQ    2 72
        LRS      11
        ALS      11         36 BIT CHAR B + C IN AC
        STA      ADH        LEAD 36 BITS B+C
        QRS      1          0.XXX 1 LEAD BIT MQ
        LDL      ADMP       CLEAR LEAD BIT
        STA      ADHL       LOWER BITS OF B+C
        INI    2 -70        DIFF EXP OF C AND B+C IN 2
        LDA      ADCE       EXP LARGEST OPERAND
        INA    2 0
        STA      ADHE       EXP OF C+B
        LDQ      ADSC       SIGN C SAME AS SIGN C+B
        QLS      1
        QJP    Z AD68
        ENQ      77777B     IF SIGN C NEG
        STQ      ADSC       SET ADCE TO -0 FOR SCM
AD68    AJP    M AD72       JUMP EXP C-
        INA      2000B
        STA      ADTM       + EXP +BIAS, UPPER PART
        LDA      ADHE
        INA      -44B       EXP LOWER
        AJP    M AD73       LOWER EXP- JUMP
        INA      2000B      EXP+ BIAS,LEAST SIGN PART
AD76    LDQ      ADHL       LOWER POWER
        LLS      36         LOWER CHAR
        SCM      ADSC
        STA      ADTM+1     FLOATED LOWER
        LDA      ADTM       UPPER EXP
        LDQ      ADH        UPPER CHAR
        QLS      1
        LLS      36         UPPER CHAR
        SCM      ADSC       FLOATED UPPER
        LDQ      ADTM+1
        SLJ      ADXDP
AD72    INA      -6000B     EXP - UPPER B+C
        STA      ADTM       SET BIAS, CLEAR SIGN BIT OF
        LDA      ADHE       EXP AND NUMBER TO GET BIAS
```

```
         INA        -44B                 EXP LOWER PART B+C
AD73     INA        -6000B               SET BIAS
         SLJ        AD76
ADXDP    STA      1 0                    UPPER PART IN DATA
ADTC1    STQ      1 0                    LOWER PART IN COMMON
         IJP      1 ADTC                 END DOUBLE PRE LOOP
         SLJ        AD17
ADMS     OCT        7000000000000000     USE OT CLEAR LEAD 3 BIT OF CHAR
ADNB     OCT        7777777777776000     USE TO EXTEND SIGN - POWER
ADA7     OCT        7777777777777777     USE TO ALT SIGN
ADMP     OCT        3777777777777777     MAX POS NUMBER
ADTEMP   BSS        3                    FLOATED C,B, C+B
AD3777   OCT        3777                 LOGICAL MASK EXP
AD1777   OCT        1777                 CHECK SIGN EXP
AD72D    DEC        72
ADSB     BSS        1                    SIGN B
ADB      BSS        1                    CHAR B
ADBE     BSS        1                    EXP B
ADSC     BSS        1                    SIGN C
ADCC2    BSS        1
ADCE     BSS        1                    EXP C
ADL      BSS        1                    LOWER PART CHAR B
ADH      BSS        1                    UPPER CHAR B+C
ADHL     BSS        1                    LOWER CHAR B+C
ADHE     BSS        1                    EXP B+C
ADTM     BSS        2
AD66     LDA        ADCC2
         SUB        ADB
         INA        -1
         LQC        ADL
         SLJ        AD75
AD17     LIU      1 AD+13
         LIL      2 AD+13
ADY      SLJ      4 0                    CALC YK
ADEXIT   SLJ      4 0                    TO EXIT
+        SIU      1 AD+13
         SIL      2 AD+13
         LIL      1 AD+12                H TO INDEX 1
ADTS     LDA      1 0                    SAVE DATA TK, YK, IN COMMON
         STA      1 0
         IJP      1 ADTS
         LDA        AD+2                 AMC TO AC
```

```
          AJP    N AD18
          LDA      AD+6              VALUE CODE IN AC
          AJP    N AD19              JUMP CODE NOT ZERO
          LDA      ADDR+7            CODE IS ZERO
          SLJ      AD20              ADDRESS YK IS ADDR+7
AD19      INA      -1                CODE NOT ZERO
          AJP    Z AD21              JUMP CODE IS 1
          INA      -1
          AJP    N AD22              JUMP CODE IS 3
          ENA      ADDR+3            CODE IS 2
          SUB      AD
          SAU      AD23              ADDR+3-RKC
AD23      LDA      0                 C(ADDR+3-RKC) IS ADDRESS YK
AD20      SAL      ADYR              SET UP ADDRESS RK STEP
          SAU      ADYR1
          SAL      ADYK              SAVE YK ADDRESS
          ENA      0
          STA      AD+4              CLEAR RKD
          LIL    1 AD+21             N-1 TO INDEX 1
ADYK      LDA    1 0
          STA    1 0                 SAVE YK, AMC=0, RKC IS ZERO OR 2
          IJP    1 ADYK
          SLJ    4 ADTI              TO INTERUPT SUBROUTINE
+         SLJ      AD27              EXIT, RK STEP DONE
+         SLJ    4 ADRK
+         LDA      AD+6              C(CODE)
          AJP    Z ADDP              TO NEXT STEP IF CODE IS ZERO
          LDA      AD                CODE IS 2
          INA      -2                RKC-2
          AJP    Z AD24
          RAO      AD                RKC+1 TO RKC
          SLJ      ADDP
AD24      ENA      1                 RKC IS 2
          STA      AD+2              SET AMC TO 1, NEXT STEP IS AM
          SLJ      ADDP
AD21      LDA      ADDR+5            CODE IS 1
          SLJ      AD25              ADDRESS YK IS ADDR+6
AD22      ENA      ADDR+4            CODE IS 3
          SUB      AD
          SAU      AD26
AD26      LDA      0
          SLJ      AD25
```

```
AD25     SAL       ADYR          CODE 1 OR 3, AMC IS ZERO
         SAU       ADYR1         SET UP RK ADDRESS
         SAL       ADYS          ADDRESS SAVE YS
         LDA     7 AD+19
         STA       AD+11         SAVE DT
         FAD     7 AD+19
         STA     7 AD+19
         ENA     0               2DT TO DT
         STA       AD+4          RKD TO ZERO
         LIL     1 AD+21         N-1 TO INDEX 1
ADYS     LDA     1 0
         STA     1 0             SAVE YK, CODE 1 OR 3
         IJP     1 ADYS
         SLJ     4 ADRK          DO RK STEP 2DT
+        LDA       AD+11
         STA     7 AD+19         RESTORE DT
         LIL     1 AD+21         N-1 TO INDEX 1
ADXP4    LDA     1 0
         STA     1 0             X(T+2DT) TO XP
         IJP     1 ADXP4
         LIL     1 AD+12         N TO INDEX 1
ADXS     LDA     1 0             TK AND YK IN STANDARD COMMON
         STA     1 0             TO ADDRESS ADD+7 AND DATA
ADTXS    STA     1 0
         IJP     1 ADXS
         SLJ     4 ADTI          TO INTERUPT SUBROUTINE
+        SLJ       AD27          RK STEP IN INTERUPT
AD39     SLJ     4 ADRK          NO RK STEP, DO RK FIRST DT STEP
+        LDA       AD+6          EXIT FIRST RK DT STEP
         INA       -1
         AJP     Z AD28
         RAO       AD            RKC+1 TO RKC
         ENA       ADDR+4        CODE 3
         SUB       AD            ADDR+4 - RKC IS ADDRESS YK+5
         SAU       AD299
AD299    LDA       0
         SLJ       AD29
AD27     ENA       0
         STA       AD            RKC TO ZERO
         SLJ       ADDP
AD28     LDA       ADDR+4        CODE 1, ADDRESS YK+.5
AD29     SAL       ADYR
```

```
         SAU      ADYR1          SEC RK STEP CODE 1,3
         SAL      ADYM           SAVE YK+.5
         LIU    1 AD+13
ADY1     LIL    2 AD+13
         SLJ    4 0              CALC YK+.5
+        SIU    1 AD+13
         SIL    2 AD+13
         LIL    1 AD+21          N-1 TO INDEX 1
ADYM     LDA    1 0
         STA    1 0              DAVE YK+.5
ADXH     LDA    1 0
         STA    1 0
         IJP    1 ADYM           XK+-5 DATA TO COMMON
         ENA      1
         STA      AD+4           RKD IS SET 1
         LDA    7 AD+20          TK+.5
         STA    7 ADTK           TK+.6 TO COMMON
         SLJ    4 ADTI           TO INTERUPT SUP, RK SEC STEP
+        SLJ      AD30           RK STEP DONE, INTERUPT
+        SLJ    4 ADRK           NO RK STEP IN INTERUPT
+        SLJ    4 ADCV           AFTER RK STEP DT, CONVTEST
+        AJP    N AD31           JUMP CONV TEST FAILED
         RAO      AD+1           RKH+1 TO RKH
AD37     LIL    1 AD+12          N TO INDEX 1
ADXS1    LDA    1 0
         STA    1 0              RESTORE XK FROM ADDR+7
         IJP    1 ADXS1
         LDA      AD+28          DOUPLE TAG
         AJP    Z AD32           JUMP IF CONV INDICATES DOUBLE
AD34     LDA      AD+6           NO DOUBLE
         INA      -1             CODE-1 TO AC
         AJP    Z ADDP           CODE IS 1, NEXT STEP
         LDA      AD             CODE IS 3
         INA      -3             RKC-3
         AJP    Z AD33
         RAO      AD             RKC+1 TO RKC
         SLJ      ADDP           NEXT STEP
AD33     ENA      1              RKC IS 3, CHANGE TO AN STEP
         STA      AD+2           SET AMC TO 1
         SLJ      ADDP
AD32     LDA      AD+1           DOUBLE TAG ZERO
         SUB      ADC            RKH-4, CHECK LAG IN DOUBLE
```

```
         AJP     M AD34              IF NEG NO DOUBLE
         LDA     7 AD+19
         FAD     7 AD+19
         STA       AD+45             LDT
         FSB     7 AD+17             2DT-DT(MAX)
         AJP     M AD35
         AJP     N AD34              SKIP DOUBLE IF 2DT TO LARGE
AD35     LDA       AD+45             DOUBLE DT
         STA     7 AD+19             2DT TO DT
         ENA       0
         STA       AD                RKC TO ZERO
         SLJ       ADDP
AD31     LDA     7 AD+16             CONV TEST FAILED
         FSB     7 AD+19             -DT +DT(MIN)
         AJP     M AD36              SEE LAST PAGE
         LIU     1 AD+13             DT EQUAL OR LESS MIN(DT)
         LIL     2 AD+13
ADTMIN   SLJ     4
+        SIL     2 AD+13
         SIU     1 AD+13
         ENA       1
         STA       AD+28             DOUBLE TAG SET NON ZERO
         ENA       0
         STA       AD+1              RKH TO ZERO
         SLJ       AD37
AD36     ENA       0
         LDQ       AD
         STQ       AD+7
         STA       AD+4              CLEAR RKD, RKC, RCH
         STA       AD
         STA       AD+1
         LDA       AD+6
         INA       -1
         AJP    Z  AD38
         LDA       AD+7
         INA       -1
         AJP     Z AD38
         LIL     1 AD+21             N-1 TO INDEX 1
ADYRH    LDA     1 0                 YK IN ADDR+2 MOVED TO ADD+4
         STA     1 0                 MOVE YK FROM RKC 2 TO 0
         IJP     1 ADYRH
AD38     ENI     1 0                 CLEAR INDEX 1
```

```
         LDA    7 AD+19
         FMU      ADC+10
         STA    7 AD+19              .5DT TO DT (COMMON)
         LDA    7 ADXS               INDEX 1 IS ZERO
         STA    7 ADTS               RESTORE TK-1 TO TK
         LIL    1 AD+21              N-1 TO INDEX 1
ADXPS    LDA    1 0
         STA    1 0                  XK+.5 TO XP IF DT AALF
ADXXS    LDA    1 0
         STA    1 0                  XK RESTORED IF DT AALF
         IJP    1 ADXPS
         LDA      AD+6
         INA      -1
         AJP   Z  AD399
         LDA      ADDR+4
AD388    SAL      ADYR               CODE 3
         SAU      ADYR1
         SLJ      AD39               REDO RK STEP, HALF DT
AD399    LDA      ADDR+5             CODE 1 ADDRESS TK
         SLJ      AD388
AD18     LDA      ADDR               AMC NOT ZERO
         SAL      ADYAS              BEGIN AM STEP
         LIL    1 AD+21              N-1 TO INDEX 1
ADYAS    LDA    1 0
         STA    1 0                  SAVE YK IN ADDR
         IJP    1 ADYAS
ADMSET   ENA      0
         STA      AD+4               CLEAR RKD
         LDA      ADDR               ADDRESS SETUP AM STEP
         SAL      ADYR               YAM SET FOR RK STEP
         SAU      ADYR1
         SAU      ADYMP
         SAL      ADYMC              AM STEP
         LDA      ADDR+1             YAM-1
         SAU      ADYMP1
         SAL      ADYMC1
         LDA      ADDR+2             YAM-2
         SAU      ADYMP2
         SAU      ADYMC2
         LDA      ADDR+3
         SAL      ADYMP3             YAM-3
         SLJ    4 ADTI               GO TO INTERUPT SUBROUTINE
```

```
+        ENI       0          EXIT SAME LOGIC EXIT +1
+        SLJ     4 ADAMM      DO AM STEP
+        LDA       AD+6       CODE IN AC
         INA       -2
         AJP     N AD40       JUMP IF CODE 3
         LDA       ADDR       CODE 2, YAM IN AC
         LDQ       ADDR+1     SHIFT ADDRESS YAM
         STA       ADDR+1     ADDRESS YAM TO YAM-1
         LDA       ADDR+2
         STQ       ADDR+2     ADDRESS YAM-1 TO YAM-2
         LDQ       ADDR+3
         STA       ADDR+3     ADDRESS YAM-2 TO YAM-3
         STQ       ADDR       ADDRESS YAM-3 TO YAM
         SLJ       ADDP       NEXT STEP
AD40     SLJ     4 ADCV
+        AJP     Z AD43       JUMP IF YK+1 CONV
         ENA       0          CONV TEST FAILED
         STA       AD+3       SET AMH TO 0
         ENA       1
         STA       AD+2       SET AMC TO 1
         LDA     7 AD+16
         FSB     7 AD+19      DT(MIN)-DT
         AJP     M ADINT      HALF DT IF MINUS
         LIU     1 AD+13      DT TO SMALL DO NOT HALF DT
         LIL     2 AD+13
ADTMC    SLJ     4 0
+        SIU     1 AD+13
         SIL     2 AD+13
         SLJ       AD42       ACCEPT DT STEP
AD43     LDA       AD+28      DOUBLE TAG
         AJP     N AD41       IF NO DOUBLE JUMP
         LDA       AD+2
         INA       -4
         AJP     M AD41       NO DOUBLE IF AMC LESS THAN 4
         LDA       AD+3
         SUB       ADC+1
         AJP     M AD41       NO DOUBLE IF AMH LESS THAN 6
         LDA     7 AD+19
         FAD     7 AD+19      2DT
         STA       AD+45
         FSB     7 AD+17      2DT-MAX DT
         AJP     P AD44       STEP TO LARGE NO DOUBLE
```

```
AD46     LDA       AD+45        DOUBLE DT
         STA     7 AD+19        2DT TO DT
         ENA       1
         STA       AD+2         AMC TO 1
         LDA       ADDR+3       ADJ ADDRESS YK FOR DOUBLE
         LDQ       ADDR+2
         STA       ADDR+2       YAM-3 TO YAM-2
         LDA       ADDR+5
         STA       ADDR+3       YAM-5 TO YAM-3
         STQ       ADDR+5       YAM-2 TO YAM-5
         LDA       ADDR+7
         LDQ       ADDR+4
         STA       ADDR+4       YAM-7 TO YAM-4
         STQ       ADDR+7
         SLJ       ADDP
AD44     AJP     Z AD46         IF 2DT EQ DT MAX, DOUBLE
         SLJ       AD41
AD41     RAO       AD+2         DT STEP ONLY
         RAO       AD+3         BUMP AMC AND AMH BY 1
AD42     LDQ       ADDR         SHIFT ADDRESS YAM
         LDA       ADDR+1
         STQ       ADDR+1       YAM TO YAM-1
         LDQ       ADDR+2
         STA       ADDR+2       YAM-1 TO YAM-2
         LDA       ADDR+3
         STQ       ADDR+3       YAM-2 TO YAM-3
         LDQ       ADDR+4
         STA       ADDR+4       YAM-3 TO YAM-4
         LDA       ADDR+5
         STQ       ADDR+5
         LDQ       ADDR+6
         STA       ADDR+6       YAM-5 TO YAM-6
         LDA       ADDR+7
         STQ       ADDR+7       YAM-6 TO YAM-7
         STA       ADDR         YAM-7 TO YAM
         SLJ       ADDP
AD30     ENA       0            RK STEP IN INTERUPT SECRKSTEP
         STA       AD           CLEAR RKC
         LIL     1 AD+12
ADTIRK   LDA     1 0            RESTORE OLD TK,XK FROM
         STA     1 0            ADDR+7 TO STANDARD COMMON
         IJP     1 ADTIRK       DUM FIRST DT STEP + INT RK STEP
```

```
        SLJ     ADDP
ADRK    SLJ     0           SUBROUTINE RK STEP
        LDA   7 AD+19
        FMU     ADC+10      .5DT TO AD+22
        STA     AD+22
        LIL   1 AD+21       N-1 TO INDEX 1
        FAD   7 ADTK        TK+.5DT TO DATA
        STA   7 AD+20
ADYR    LDA     AD+22       LOOP TO GET XK+.5DT(YK)
        FMU   1 0           .5DT(YK)
ADXR    FAD   1 0           ADD XK
        STA   1 0           STORE YK+.5
        IJP   1 ADYR        END LOOP TO GET YK+.5DT(YK)
        LIL   2 AD+13
        LIU   1 AD+13       RESTORE INDEX
ADY4    SLJ   4 0           CALC U(T+.5DT,XK+.5DTYK)
+       SIU   1 AD+13       SAVE INDEX
        LIL   1 AD+21       N-1 TO INDEX 1
ADK2    LDA   1 0           LOOP TO GET X IN ARG K3
        STA   1 0           STORE K2
ADXR1   FMU     AD+22       .5D1(YT+.5DT, XK+.5DTYK)
        FAD   1 0           ADD YK FOR ARG K3
ADXR4   STA   1 0           STORE X, ARG K3
        IJP   1 ADK2        END LOOP TO GET X, ARG K3
        LIU   1 AD+13
ADY5    SLJ   4             CALC Y IN K3, RK STEP
+       SIU   1 AD+13
        LDA   7 AD+19
        STA     AD+23       DT TO AD+23
        FAD   7 ADTK        TK+DT TO DATA
        STA   7 AD+20
        LDA  7  AD+19       DT
        FDV     ADC+11      DIV DT BY 6
        STA     AD+24
        LIL   1 AD+21       N-1 TO INDEX 1
ADK3    LDA   1 0           LOOP TO GET X IN K4
        STA   1 0           STORE X3, RK STEP
ADXR2   FMU     AD+23       DT(K3)
        FAD   1 0           XK+K3DT IS ARG X IN K4
ADXR5   STA   1 0
        IJP   1 ADK3        END LOOP TO GET ARGX, K4
        LIU   1 AD+13
```

```
ADY6     SLJ     4                    CALC Y IN K4, RK STEP
+        SIU     1 AD+13
         SIL     2 AD+13
         LIL     1 AD+21              N-1 TO INDEX 1
ADK22    LDA     1 0                  K2, LOOP TO CALC DT
         FAD     1 0                  K2+K3
         STA       AD+27
         FAD       AD+27              2K2+2K3
ADYR1    FAD     1 0                  +K1
         FAD     1 0                  +K4
         FMU       AD+24              .1666DT(K1+2K2+3K3+K4)
         LDQ       AD+4               RKD IN MQ
ADDX1    QJP     N AD3
         STA     1 0                  RKD IS ZERO, STORE DX
         SLJ       ADXR3
AD3      STA       AD+27              RKD NOT ZERO
ADDX2    FAD     1 0                  SAVE DX THIS STEP
         STA     1 0                  CUMULATE DX
         LDA       AD+27              RESTORE DX THIS STEP
ADXR3    FAD     1 0                  ADD XK TO DX
         STA     1 0                  STORE YK+1 IN DATA
         IJP     1 ADK22              END LOOP TO CALC DT, IN RK
         LDA     7 AD+19              DT IN AC
         LDQ       AD+4               RKD IN MQ
         QJP     N ADDT3
         STA     7 ADDT3              DT TO DT COMMON IF RRD IS ZERO
         SLJ       ADRK
ADDT3    FAD       0
         STA       0
         SLJ       ADRK
ADCV     SLJ       0                  CONV SUBROUTINE
         ENA       0
         STA       AD+28              CLEAR DOUBLE TAG
         LIL     1 AD+21              N-1 TO INDEX 1
         SIU     3 AD+33              SAVE INDEX 3
         LDA     7 AD+15              E
         STA       AD+34              E TO AD+34
         FDV       ADC+2              DIV E BY DOUBLE FACTOR
         STA       AD+35              STORE DOUBLE ERROR
         LDA     7 AD+14              A IN AC
         AJP     Z ADA                JUMP IF A ZERO
         AJP     P AD4
```

```
         SCM        ADMASK          ABS VAL A IN AC
ADA      ENI      2 0               CLEAR INDEX2, AI IS A
         STA        0
         SLJ        ADA1
AD4      LIL      2 AD+21           N-1 TO INDEX 2 IF A+
ADA1     LDA      2 0               BEGIN CONV LOOP
         STA        AD+30           A TO AD+30
ADXP2    LDA      1 0               XP
         AJP      P ADXC
         SCM        ADMASK          ABS VALUE XP
ADXC     STA        AD+29           XP TO AD+29
         LDA      1 0               XC
         ENI      3 2               2 TO INDEX 3
         AJP      P AD5
         SCM        ADMASK
AD5      THS      3 AD+29
         SLJ        AD6             HALF EXIT A GREATEST
         LDA      3 AD+29           REPLACE A BY LARGER TABLE VAL
         ENQ      3 0
         QJP      N AD5             IF INDEX 3 IS ZERO, END TABLE
AD6      STA        AD+32           STORE LARGEST 4, XP, XC
         AJP      Z AD7             SKIP CONU TEST IF ALL ZERO
ADXP3    LDA      1 0               XP
         FSB      1 0               XP÷XC
         FDV        AD+32           DIV BY LARGEST OF A, XP, XC
         AJP      P AD8
         SCM        ADMASK
AD8      FSB        AD+35
         AJP      M AD7
         AJP      Z AD7             SKIP IF ZERO, CONV DOUBLE
         STA        AD+28           SET DOUBLE TAG NON ZERO, NO DOUBLE
         FSB        AD+34           -E, DOUBLE TEST FAILED
         AJP      P AD10            IF AC+ CONV TEST FAILED
AD7      IJP      2 AD9
AD9      IJP      1 ADA1            END CONV TEST LOOP
         ENA        0               RESTORE INDEX 3
AD57     LIU      3 AD+33
         SLJ        ADCV            CONV TEST PADDED
AD10     ENA        -1
         SLJ        AD57            CONV TEST FAILED
ADMASK   OCT        -1
ADAMM    SLJ        0               SUBROUTINE AM STEP
```

```
        LIL   1 AD+21     N-1 TO INDEX 1
        LDA   7 AD+19     DT
        FDV     ADC+3     DIV DT BY 24, PUT IN AD+25
        STA     AD+25
ADYMP3  LAC     ADC+4     LOOP TO CALC XP
        FMU   1 0
        STA     AD+26     -.9YAM-3
        LDA     ADC+5
ADYMP2  FMU   1 0         37 YAM-2
        FAD     AD+26
        STA     AD+26
        LDA     ADC+6
ADYMP1  FMU   1 0         -59YAM-1
        FAD     AD+26
        STA     AD+26
        LDA     ADC+7
ADYMP   FMU   1 0         55.YAM
        FAD     AD+26     (55YAM-59YAM-1+37YAM-2-9YAM-3)
ADXM    FMU     AD+25     MULT BY DT OVER 24
        FAD   1 0         ADD XK IN COMMON
ADXP    STA   1 0         XP TO DATA
        STA   1 0         YP TO COMMON
        IJP   1 ADYMP3    END LOOP TO CLAC XP
        LDA   7 AD+19     DT IN DATA
ADTK    STA   7 ADDT3     DT TO DT COMMON REGION
        FAD     0         TK FROM COMMON
        STA   7 AD+20     TTAT TO T IN DATA
        LIU   1 AD+13
        LIL   2 AD+13     RESTORE INDEX
ADY3    SLJ   4 0         CALC YP IN AM
+       SIU   1 AD+13
        SIL   2 AD+13     SAVE INDEX
        LIL   1 AD+21     N-1 TO INDEX 1
ADYMC1  LDA     ADC+8     LOOP TO CALC XC
        FML   1 0         -5 YAM-1
ADYMC2  FAD   1 0         YAM-2 -5YAM-1
        STA     AD+26
ADYMC   LDA     ADC+9
        FMU   1 0         19YAM
        FAD     AD+26
        STA     AD+26
ADYP    LDA     ADC+4
```

```
         FMU    1 0             9YP
         FAD      AD+26
         FMU      AD+25         MULT BY DT OVER 24
ADDX     STA    1 0             STORE DX
         FAD    1 0             ADD XK FROM COMMON
ADXMK    STA    1 0             STORE YK+1 IN DATA
         IJP    1 ADYMC1        END LOOP TO CALC XC
         SLJ      ADAMM         EXIT AM STEP SUBROUTINE
ADINT    LDA      ADDR          INTERPOLATE FOR HALF STEP
         SAU      ADI
         SAL      ADII          ADDRESS YAM
         LDA      ADDR+1
         SAU      ADI1
         SAU      ADI11         ADDRESS YAM-1
         LDA      ADDR+2
         SAL      ADI2
         SAL      ADI22         ADDRESS YAM-2
         LDA      ADDR+3
         SAU      ADI3
         SAU      ADI33         ADDRESS YAM-3
         SAL      ADI333
         LDA      ADDR+4
         SAL      ADI4          ADDRESS YAM-4
         SAL      ADI44
         LDA      ADDR+5
         SAU      ADI5          ADDRESS YAM-5
         LIL    1 AD+21         N-1 TO INDEX 1
ADI4     LDA      ADC+12        BEGIN INTERPOLATE LOOP
         FMU    1 0
         STA      AD+36         .0234575YAM-4
         LDA      ADC+13
ADI3     FMU    1 0             -.15625YAM-3
         STA      AD+37
ADI2     LDA      ADC+14
         FMU    1 0             .703125YAM-2
         STA      AD+38
         LDA      ADC+15
ADI1     FMU    1 0             .46875YAM-1
         STA      AD+39
ADI44    LDA      ADC+16
         FMU    1 0
         STA      AD+40         -.0390625YAM-4
```

```
         LDA        ADC+17
ADI33    FMU      1 0
         STA        AD+41          .21875YAM-3
ADI22    LDA        ADC+18
         FMU      1 0
         STA        AD+42          -.546875YAM-2
         LDA        ADC+19
ADI11    FMU      1 0
         STA        AD+43          1.09375YAM-1
ADII     LDA        ADC+20
         FMU      1 0              .2734375YAM
         FAD        AD+43
         FAD        AD+42
         FAD        AD+41
         FAD        AD+40
ADI5     STA      1 0              PUT YAM-.5 IN YAM-5
         LDA        ADC+16
ADI      FMU      1 0
         FAD        AD+39
         FAD        AD+38
         FAD        AD+37
ADI333   FAD        AD+36
         STA      1 0              PUT YAM-1.5 IN YAM-3
         IJP      1 ADI4           END INTERPOLATE LOOP
         LDA        ADDR+5         RESHUFFLE ADDRESS
         LDQ        ADDR+1         FOR HALF DT
         STA        ADDR+1         ADDRESS YAM-5 TO YAM-1
         LDA        ADDR+2
         STQ        ADDR+2         ADDRESS YAM-1 TO YAM-2
         LDQ        ADDR+4
         STA        ADDR+4         ADDRESS YAM-2 TO YAM-4
         STQ        ADDR+5         ADDRESS YAM-4 TO YAM-5
         LDA      7 AD+19
         FMU        ADC+10         .5DT TO DT
         STA      7 AD+19
         LIL      1 AD+12
ADTXR    LDA      1 0              MOVE TK, YK IN COMMON
         STA      1 0              TO DATA
         IJP      1 ADTXR
         SLJ        ADMSET
ADTI     SLJ        0              TIME INTERUPT SUBROUTINE
         ENA        0              EXIT INTERUPT IF T IS ZERO
```

```
          AJP    Z  AD11        C(T) TO A C
          LDA    7  ADTI
          AJP     N ADEXT       IF C(T) ZERO SKIP SUBROUTINE
AD11      LDA       ADONE       100000000 TO AC
          RAD       ADTI        BUMP UPPER ADDRESS BY 1
          SLJ       ADTI        GO TO EXIT+1 NO RK STEP
ADEXT     ENA       0
          AJP     N AD12
          ENA       -1          IF C(T) NOT ZERO BUT TEXIT ZERO
          SLJ       ADERR       GO TO ERROR RETURN BETA+4,AC-1
AD12      LDA     7 ADTI
          FSB     7 ADTK        C(T)-TK
          AJP     Z AD13        JUMP IF CT EQUALS TK
          AJP    M  AD13
          FSB     7 AD+19       -DT+C(T)-TK,+AND NOT ZERO
          AJP     P AD11        TO AD11, IF NO INTERUPT
AD12A     LDA       AD+4
          STA       AD+31       SAVE RKD
          AJP     Z AD14
          LIL     1 AD+12       N TO INDEX 1
ADDT      LDA     1 0
          STA     1 0           SAVE DT,DX ININTERUPT
          IJP     1 ADDT        IF RKD IS 1
AD14      LDA     7 AD+19       INTERUPT LOOP
          STA       AD+44       SAVE DT IN DATA
          ENQ       0
          STQ       AD+4        SET RKD TO 0
          LDA     7 ADTI
          FSB     7 ADTK
          STA     7 AD+19       C(T)-TK IS DT
          SLJ     4 ADRK        DO RK DE STEP
+         LDA       AD+44
          STA     7 AD+19       RESTORE OLD DT
          ENA       0           SET AC 0
          LIL     2 AD+13
ADEXT2    LIU     1 AD+13       TK ADVANCED TO C(T), GO TO TEXIT
          SLJ     4 0
+         SIU     1 AD+13
          SIL     2 AD+13
          LDA     7 ADTI
          FSB     7 AD+20       C(T)-TK IN DATA
          AJP     Z AD15
```

```
         AJP     M AD15                 IF C(T)-TK IS 0 OR - EXIT INT-KOOP
         LDA     7 ADTI
         FSB     7 ADTK                 C(T)-TK IN COMMON
         FSB     7 AD+19                C(T)-TK-DT
         AJP     P AD15                 EXIT INTERUPT IF +
         SLJ       AD14                 LOOP INTERUPT
AD13     LIU     1 AD+13
         LIL     2 AD+13                IF C(T)-TK ZERO OR -
ADEXT1   SLJ     4 0                    INTERUPT WITHOUT RK STEP
+        SIU     1 AD+13
         SIL     2 AD+13
         LDA     7 ADTI                 IS NEW PRINT TIEM WITHIN DT OF
         FSB     7 ADTK                 PRESENT TIME
         FSB     7 AD+19
         AJP     P AD11                 NO, NORMAL   EXIT +1 INTERUPT
         SLJ       AD12A                YES8 DO RK STEP WITH MODIFIED DT
ADONE    OCT       100000000
AD15     LDA       AD+31
         STA       AD+4                 RESTORE RKD
         AJP     Z AD16                 SKIP RESTORE IF RKD IS ZERO
         LIL     1 AD+12                N TO INDEX 1
ADDT1    LDA     1 0
         FAD     1 0                    DX+OLD DX TO DX
ADDT2    STA     1 0                    CUMULATE AND RESTORE DT
         IJP     1 ADDT1                ID RKD IS 1
AD16     SLJ       ADTI                 EXIT, RK STEPS DONE
ADC      DEC       4,6,32.
         DEC       24.,9.,37.,-59.,55.  20. IS DOUBLE FACTOR
         DEC       -5.,19.,.5,6.
         DEC       .0234375,-.15625
         DEC       .703125,.46875
         DEC       -.0390625,.21875
         DEC       -.546875,1.09375
         DEC       .2734375
AD       BSS       50
ADDR     BSS       9
```

```
         ORG       7300B
ATMOS    ZRO       0
         FDV       MTOFEET
***      EQU       0
         STA       COMP+1
         FAD       HFACT
         STA       COMP+2
         LDA       COMP+1
         FMU       HFACT
         FDV       COMP+2
         STA       COMP                ALTITUDE IN GEOPOTENTIAL METERS
         LDA       ATMOS
         ARS       24
         SAL       TEMPEXT
         ADD       OCTONE
         SAU       PRESEXT
         ADD       OCTONE
         SAL       RHOEXT
         ADD       OCTONE
         SAL       VSNDEXT
         ADD       OCTONE
         SAL       EXIT
         ADD       OCTONE
         SAL       EXIT2
         SIU     1 OCTONE              SAVE INDEX 1
         ENI     1 0
CAT      LDA     1 TABLE
         FSB       COMP
         AJP     Z EQUAL
         AJP     P EQUAL
+        ISK     1 22
         SLJ     0 CAT
EXIT     LIU     1 OCTONE
         SLJ     0 ***
EQUAL    LDA       COMP
         FSB     1 TABLE-1
         FMU     1 TABLE1X             LM
         FAD     1 TABLE2X
TEMPEXT  STA       COMP+1              TM
         STA       ***
         LDA     1 TABLE1X             LM
         AJP     N NONZERO
```

```
         LDA    1  TABLE-1             HB
         FSB       COMP                H
         FMU       QCONST
         FDV     1 TABLE2X             TMB
         SLJ    0  AROUND              EXP ARGUMENT IS IN A REGISTER
 NONZERO LDA     1 TABLE2X             TMB
         FDV       COMP+1
         STA       COMP+3
         ENA       COMP+3
         RTJ     0 LOGF
 QCONST  DEC       3.41647942D-02
+        FMU       QCONST
         FDV    1  TABLE1X             EXP ARGUMENT NOW IN A REGISTER
 AROUND  STA       COMP+3
         ENA       COMP+3
         RTJ    0  EXPF
 MTOFEET DEC       3.2808333333
+        FMU     1 TABLE3X             PB
 PRESEXT STA       ***                 PRESSURE
         FDV       COMP+1              TM
 RHOEXT  FMU       RHOCNT
         STA       ***                 DENSITY
         LDA       COMP+1              TM
         FDV       TABLE2X             BASE TEMPERATURE
         STA       COMP+3
         ENA       COMP+3
         RTJ     0 SQRTF
 CSZERO  DEC       1116.4437
 VSNDEXT FMU       CSZERO
         STA       ***                 VELOCITY OF SOUND
 EXIT2   LIU     1 OCTONE
         SLJ     0 ***
 RHOCNT  DEC       3.2365983D-04
 OCTONE  OCT       000000000001
 COMP    BSS       4
 HFACT   DEC       6356766.
         DEC       0.0
 TABLE   DEC       0.,11000.,20000.
         DEC       32000.,47000.,52000.
         DEC       61000.,79000.,88743.
         DEC       98451.,108129.,117777.
         DEC       146542.,156071.,165572.
```

```
         DEC      184485.,221968.,286478.
         DEC      376315.,463530.,548235.
         DEC      630536.,700000.
TABLE1X  DEC      -.0065,-.0065,0.0
         DEC      .001,.0028,0.0,-.002
         DEC      -.004,0.0,.00309,.0051663
         DEC      .0103648,.0208587,.015741,.0105252,.0074023
         DEC      .00533575,.0043404,.0036733,.00298114
         DEC      .002007,.00133655,.00133655
TABLE2X  DEC      288.15,288.15,216.65
         DEC      216.65,228.65,270.65
         DEC      270.65,252.65,180.65
         DEC      180.65,210.65,260.65
         DEC      360.65,960.65,1110.65
         DEC      1210.65,1350.65,1550.65
         DEC      1830.65,2160.65,2420.65
         DEC      2590.65,2700.65
TABLE3X  DEC      2116.2169,2116.2169,472.6792
         DEC      114.3415,18.12814,2.316195
         DEC      1.232178,.380303,.0216707
         DEC      3.43294E-3,6.28025E-4,15.356435E-5
         DEC      5.26501E-5,10.5678E-6,7.71278E-6
         DEC      5.83017E-6,3.51815E-6,14.53042E-7
         DEC      3.93231E-7,8.412236E-8,2.2867465E-8
         DEC      7.200254E-9,2.48704E-9
ALTER    BSS      100
EXPF     LIB      EXPF
LOGF     LIB      LOGF
SQRTF    LIB      SQRTF
         END
```

8. SPURT SAMPLE PRINTOUT DATA.

SPURT SAMPLE PRINTOUT DATA

1. INPUT DATA - Launch Parameters and Spin Table
2. INPUT DATA - Stage Weight and Aerodynamic Parameters -- 1 page per Stage
3. SPURT OUTPUT DATA
4. SPURT OUTPUT DATA
5. SPURT OUTPUT DATA
6. SPURT OUTPUT DATA
7. TWO-BODY OUTPUT DATA - Orbital Elements
8. TWO-BODY OUTPUT DATA - Keplerian Trajectory
9. TWO-BODY OUTPUT DATA - Look-Angles
10. IMPACT OUTPUT DATA - Position Information
11. IMPACT OUTPUT DATA - Velocity Information

~~INPUT TABLE 39 LB PL NEW DATA JANUARY 8, 1962 40 SEC COAST 85 DEG LAU~~

35.0000 28.5136 -80.5765 .0000 105.0000 0 1 0 240 0 0 0 0 0

SPIN TABLE

TIME (Sec)	SPIN RATE (Rad/Sec)	TIME (Sec)	SPIN RATE (Rad/Sec)	TIME (Sec)	SPIN RATE (Rad/Sec)	TIME (Sec)	SPIN RATE (Rad/Sec)
.000	.000	.000	.000	.000	.000	.000	.000
.900	20.100	.000	.000	.000	.000	.000	.000
2.500	14.770	.000	.000	.000	.000	.000	.000
5.000	5.280	.000	.000	.000	.000	.000	.000
7.500	5.780	.000	.000	.000	.000	.000	.000
10.000	7.950	.000	.000	.000	.000	.000	.000
12.500	8.150	.000	.000	.000	.000	.000	.000
15.000	8.150	.000	.000	.000	.000	.000	.000
30.000	14.700	.000	.000	.000	.000	.000	.000
40.000	14.800	.000	.000	.000	.000	.000	.000
50.000	14.700	.000	.000	.000	.000	.000	.000
60.000	14.700	.000	.000	.000	.000	.000	.000
70.000	14.650	.000	.000	.000	.000	.000	.000
170.000	15.500	.000	.000	.000	.000	.000	.000
999.000	16.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000

1st Line - Name - 80 Hollerith Characters.

2nd Line - Payload Weight, Latitude, Longitude, Launch Coordinate System Origin Altitude, Launch Azimuth, and Control Numbers.

Spin Table Columns

STAGE NO. 1

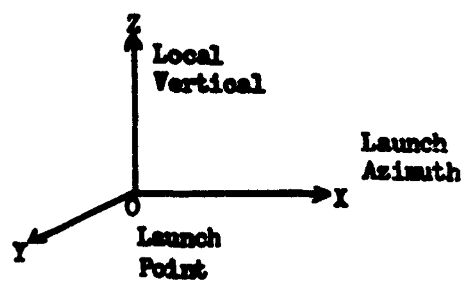
TIME	THRUST	MACH NO.	CD (B)	MACH NO.	CD (C)	MACH NO.	CP	MACH NO.	CN/ALPHA
.000	.000	.000	.368	.000	.890	.000	10.100	.000	8.022
.250	67500.000	.250	.396	2.000	.885	.500	9.760	.500	9.430
.500	58500.000	.500	.428	3.000	.637	.750	9.420	.900	10.020
1.000	57000.000	.750	.523	4.000	.542	.900	9.100	1.000	10.630
2.000	54500.000	1.000	1.146	5.000	.500	1.000	8.760	1.500	9.490
3.000	52500.000	1.200	1.146	99.000	.580	1.100	9.500	2.000	8.480
4.000	51500.000	1.500	.970	.000	.000	1.200	10.840	3.000	7.030
6.000	51500.000	2.000	.770	.000	.000	1.500	12.260	4.000	6.220
12.000	54000.000	3.000	.570	.000	.000	2.000	14.040	5.000	5.480
16.000	54000.000	4.000	.457	.000	.000	2.500	17.010	.000	.000
20.000	53000.000	5.000	.418	.000	.000	3.000	18.880	.000	.000
26.000	54500.000	.000	.000	.000	.000	3.500	19.760	.000	.000
27.000	53500.000	.000	.000	.000	.000	4.000	20.480	.000	.000
28.000	47000.000	.000	.000	.000	.000	5.000	22.500	.000	.000
31.000	16500.000	.000	.000	.000	.000	.000	.000	.000	.000
33.000	8500.000	.000	.000	.000	.000	.000	.000	.000	.000
35.000	4500.000	.000	.000	.000	.000	.000	.000	.000	.000
37.094	1500.000	.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000	.000	.000

STAGE WEIGHT =	9443.20 LBS	STAGE FUEL WEIGHT =	7467.00 LBS
IGNITION TIME =	.00 SEC	BURNOUT TIME =	37.09 SEC
MISSILE CG =	16.6750 FT	STAGE FUEL CG =	11.3100 FT
EXIT AREA =	504.00 SQ IN.	PRESSURE AT THRUST MEASUREMENT =	14.6900 LBS/SQ IN.
LONGITUDINAL I OF MISSILE =	382.00 FT2 SLUGS	TRANSVERSE I OF MISSILE =	45277.00 FT2 SLUGS
FUEL LONGITUDINAL I/M =	.9660 FT2	FUEL TRANSVERSE I/M =	20.6870 FT2
THRUST MISALIGNMENT ANGLE =	.00 RAD	ORIENTATION ANGLE OF THRUST MISALIGNMENT =	.00 RAD
DIAMETER =	2.5850 FT	TIME TO CHANGE COEFFICIENTS =	86.00 SEC

DEFINITIONS

DEFINITIONS: All trajectories are with respect to an oblate earth.

RANGE COORDINATE SYSTEM: A left hand orthogonal cartesian coordinate system with the XY plane tangent to an oblate earth at the launch point. The Z axis is positive upward along the local vertical and the X axis is positive along the launch azimuth.



TIME (Sec): Relates to the first motion of the vehicle off the launch site.

LATITUDE (Deg): Angle between the normal to the reference spheroid passing through the vehicle and the equatorial plane. Positive in the northern hemisphere.

LONGITUDE (Deg): Angular distance measured from the foot of the Greenwich meridian to the vehicle sub-point meridian. West of Greenwich is negative and East is positive.

ALTITUDE (Feet):	Distance from vehicle to the surface of a geodetic earth below the vehicle.
VELOCITY (fps):	Velocity of vehicle with respect to a point on a rotating earth directly below the vehicle.
AZIMUTH (Deg):	Angle from North the velocity vector makes with the local meridian. Measured positive C. W. from North.
F. P. A. (Deg):	Flight path angle. The angle the velocity vector makes with the local horizon; positive upward.
RANGE (N.M.):	Arc distance from launch to the point under the vehicle measured along the surface of the earth.
DEFLECTION (N.M.):	Distance the vehicle has deviated from the launch axis in the XY plane.
PHI (Deg):	Euler angle between the longitudinal axis of the vehicle and the Z axis in the range coordinate system
THETA (Deg):	Euler angle between the longitudinal axis of the vehicle and the XZ plane measured in the rotated X'Y' plane.
THRUST (Lbs):	Thrust of the rocket motor with thrust increase due to pressure decrease included.
WEIGHT (Lbs):	Weight of the remaining portion of the vehicle. Given in terms of sea level pounds.

TOTAL ACCEL (g's):	Absolute total acceleration of the vehicle normalized by the gravitational constant g_o.
DYNAMIC PRESSURE (Lb/Ft^2):	The classical dynamic pressure on the vehicle given by ½ of the density times the velocity squared.
DRAG (Lbs):	Aerodynamic drag on the vehicle.
MACH NO:	Mach number of the vehicle.
X, Y, Z (Ft):	Position vector of vehicle in range coordinate system
X-DOT, Y-DOT, Z-DOT (fps):	Velocity vector of vehicle in range coordinate system
ORBITAL ELEMENTS:	(Ref 3) The parameters of a classical Keplerian ellipse based on an inverse square gravitational field.
SEMI MAJOR AXIS (a):	One half of the longest diameter of the Keplerian ellipse.
SEMI MINOR AXIS (b):	One half of the smallest diameter of the Keplerian ellipse.
ECCENTRICITY (e):	A measure of the flattening of the Keplerian ellipse.
PERIOD (p):	The time that a space vehicle takes to make one complete orbit.
APOGEE ALTITUDE (h_A):	The distance to the highest point on the ellipse from the surface of the Earth.
PERIGEE ALTITUDE (hp):	The radius of the point on the ellipse closest to the Earth, minus the radius of the Earth.

INCLINATION (i):	The angle between the orbit plane and the equatorial plane.
ARGUMENT OF PERIGEE (ω):	The angular distance measured in the orbit plane from the line of nodes to the line of apsides.
LONGITUDE OF THE ASCENDING NODE AT BURNOUT TIME (Ω):	The angular distance measured at burnout time from Greenwich eastward in the equatorial plane to the point of intersection of the orbit plane where the vehicle crosses from south to north.
INITIAL TRUE ANOMALY (V_o):	The angle measured at burnout time at the center of the Earth between the line of apsides and the radius vector to the vehicle measured from perigee in the direction of motion.
INITIAL ECCENTRIC ANOMALY (E_o):	The angle at the center of the ellipse between the line of apsides and radius vector of the auxiliary circle through a point which has a projection of the ellipse corresponding to the initial true anomaly.
H. M. S.:	Time in hours, minutes, and seconds of the vehicle from launch.
INERTIAL VELOCITY (fps):	The velocity of the vehicle with respect to a coordinate system fixed to, but not rotating with, the Earth.

INERTIAL FLIGHT PATH ANGLE (Deg): The angle at any given time the inertial velocity vector makes with respect to the perpendicular to the radius vector at that time.

LOOK ANGLES: The direction to position a tracking antenna at a given station.

AZIMUTH (Deg): The angle that the projection in the horizontal plane of the vector pointing to the vehicle makes with the North direction.

ELEVATION (Deg): The angle from the horizon to the vector pointing to the vehicle.

SLANT RANGE (N. M.): The distance from the tracking station to the vehicle.

ASPECT ANGLE (Deg): Angle between the vehicle spin axis and a vector from a given station to the vehicle.

32 LB PAYLOAD NOVEMBER 15, 1962

LOCAL FLIGHT PARAMETERS

TIME SEC	LATITUDE DEG	LONGITUDE DEG	ALTITUDE FEET	VELOCITY FT/SEC	AZIMUTH DEG	FPA DEG
.00	28.5136	-80.5765	10		105.00	76.00
1.00	28.5136	-80.5765	50	96	105.16	71.32
2.00	28.5136	-80.5763	190	200	105.15	71.46
3.00	28.5135	-80.5761	428	302	105.08	71.14
4.00	28.5134	-80.5757	760	402	104.88	70.27
5.00	28.5133	-80.5752	1184	504	104.58	68.80
6.00	28.5132	-80.5746	1699	608	104.41	67.15
7.00	28.5130	-80.5738	2305	716	104.50	66.00
8.00	28.5128	-80.5729	3007	827	104.58	65.17
9.00	28.5125	-80.5717	3806	941	104.53	64.17
10.00	28.5122	-80.5704	4702	1057	104.59	63.44
11.00	28.5118	-80.5689	5695	1172	104.58	62.64
12.00	28.5114	-80.5672	6784	1288	104.60	61.98
13.00	28.5110	-80.5652	7970	1407	104.64	61.35
14.00	28.5105	-80.5631	9254	1528	104.63	60.72
15.00	28.5100	-80.5607	10637	1652	104.65	60.17
16.00	28.5094	-80.5581	12122	1780	104.68	59.65
17.00	28.5087	-80.5553	13709	1910	104.68	59.14
18.00	28.5080	-80.5522	15401	2043	104.69	58.67
19.00	28.5072	-80.5489	17200	2180	104.71	58.23
20.00	28.5064	-80.5453	19107	2320	104.72	57.80
21.00	28.5055	-80.5415	21127	2465	104.72	57.39
22.00	28.5045	-80.5373	23262	2616	104.74	57.01
23.00	28.5035	-80.5329	25518	2775	104.76	56.66
24.00	28.5024	-80.5282	27900	2941	104.77	56.31
25.00	28.5012	-80.5231	30414	3117	104.77	55.98
26.00	28.5000	-80.5177	33068	3301	104.79	55.67
27.00	28.4986	-80.5119	35868	3492	104.81	55.38
28.00	28.4972	-80.5057	38813	3676	104.81	55.10
29.00	28.4957	-80.4993	41889	3831	104.82	54.83
30.00	28.4941	-80.4925	45065	3946	104.84	54.57
31.00	28.4925	-80.4856	48308	4020	104.85	54.32
32.00	28.4908	-80.4785	51588	4066	104.86	54.06
33.00	28.4891	-80.4712	54888	4097	104.87	53.81

32 LB PAYLOAD NOVEMBER 15, 1962

TIME SEC	RANGE N.MILES	DEFLECTION N.MILES	ALTITUDE N.MILES	VELOCITY FT/SEC	PHI DEG.	THETA DEG.
.00	.000	-.000	.002	.0	14.000	-.000
1.00	.002	-.000	.008	95.7	14.017	-.000
2.00	.010	-.000	.031	200.1	14.313	-.016
3.00	.023	-.000	.070	301.6	15.621	-.114
4.00	.042	-.000	.125	401.9	18.652	-.362
5.00	.068	.000	.195	503.7	22.575	-.598
6.00	.103	.000	.280	608.0	24.220	-.358
7.00	.146	.001	.379	715.5	23.001	.096
8.00	.199	.001	.495	826.8	24.978	-.358
9.00	.261	.002	.626	941.0	25.991	-.131
10.00	.333	.002	.774	1056.7	26.609	-.300
11.00	.417	.003	.937	1172.0	26.950	.005
12.00	.511	.004	1.117	1288.3	28.654	-.403
13.00	.616	.004	1.312	1407.1	28.276	-.103
14.00	.731	.005	1.523	1528.3	29.124	-.049
15.00	.860	.006	1.751	1652.4	30.344	-.391
16.00	1.002	.007	1.995	1779.5	30.160	-.143
17.00	1.156	.008	2.256	1909.7	30.546	.029
18.00	1.324	.009	2.535	2042.9	31.679	-.292
19.00	1.505	.010	2.831	2179.6	31.998	-.326
20.00	1.701	.011	3.145	2319.8	31.893	.012
21.00	1.911	.012	3.477	2464.8	32.506	-.017
22.00	2.137	.013	3.828	2616.2	33.333	-.347
23.00	2.380	.014	4.200	2774.7	33.466	-.266
24.00	2.639	.015	4.592	2941.3	33.439	.087
25.00	2.916	.016	5.006	3116.7	34.087	-.096
26.00	3.212	.018	5.442	3300.8	34.637	-.395
27.00	3.527	.019	5.903	3491.9	34.564	-.066
28.00	3.863	.020	6.388	3675.8	34.799	.109
29.00	4.217	.022	6.894	3830.6	35.444	-.348
30.00	4.585	.023	7.417	3945.8	35.478	-.155
31.00	4.966	.024	7.951	4019.8	35.631	.149
32.00	5.354	.025	8.490	4065.7	36.208	-.429
33.00	5.748	.027	9.033	4097.5	36.127	.145

32 LB PAYLOAD NOVEMBER 15, 1962

TIME SEC	THRUST LBS	WEIGHT LBS	TOTAL ACCEL. ACCEL./GRAV.	DYNAMIC PRES LBS/SQFT	DRAG LBS	MACH NO.
.00	-.32	13412.00	1.000	.000	.00	.00
1.00	57010.29	13374.96	3.301	10.860	15.80	.09
2.00	54547.76	13127.38	3.191	47.322	69.65	.18
3.00	52610.84	12888.62	3.119	106.749	158.86	.27
4.00	51698.19	12655.75	3.132	187.785	282.50	.36
5.00	51808.49	12424.83	3.234	291.192	442.87	.45
6.00	51940.57	12193.91	3.320	417.893	652.68	.55
7.00	52510.28	11958.09	3.416	568.512	925.92	.65
8.00	53100.62	11722.28	3.542	743.312	1262.32	.75
9.00	53710.50	11486.46	3.609	940.122	2250.01	.85
10.00	54338.49	11250.64	3.670	1153.822	3866.83	.96
11.00	54982.73	11014.82	3.611	1377.406	5700.71	1.07
12.00	55641.17	10779.01	3.695	1609.652	6704.93	1.18
13.00	55895.34	10538.29	3.754	1851.366	7367.34	1.30
14.00	56160.19	10297.57	3.840	2098.764	7948.46	1.41
15.00	56433.88	10056.85	3.941	2349.222	8467.90	1.54
16.00	56714.54	9816.14	4.030	2599.334	8978.92	1.66
17.00	56750.29	9577.38	4.116	2844.754	9382.58	1.80
18.00	56789.15	9338.62	4.230	3081.406	9662.57	1.94
19.00	56829.16	9099.86	4.336	3306.185	9960.20	2.08
20.00	56868.37	8861.11	4.441	3514.101	10274.97	2.23
21.00	57404.90	8621.37	4.631	3704.676	10485.92	2.39
22.00	57937.04	8381.63	4.846	3877.729	10589.95	2.56
23.00	58463.16	8141.89	5.078	4030.255	10577.93	2.74
24.00	58981.72	7902.16	5.337	4159.142	10441.49	2.93
25.00	59491.21	7662.42	5.610	4260.411	10384.53	3.14
26.00	59990.10	7422.68	5.897	4328.212	10326.63	3.36
27.00	59226.90	7181.96	6.039	4352.484	10144.75	3.60
28.00	52947.41	6955.95	5.442	4199.541	9600.42	3.80
29.00	42980.17	6802.46	4.243	3936.209	8853.73	3.96
30.00	32990.77	6648.96	3.001	3587.477	7998.37	4.08
31.00	22979.30	6495.47	1.733	3188.064	7076.16	4.15
32.00	19113.52	6417.44	1.343	2787.837	6170.60	4.20
33.00	15228.90	6339.42	.945	2418.119	5341.95	4.23

32 LB PAYLOAD NOVEMBER 15, 1962

TIME SEC	X FEET	Y FEET	Z FEET	X-DOT FT/SEC	Y-DOT FT/SEC	Z-DOT FT/SEC
.00			10			
1.00	14		50	31		91
2.00	61		190	64		190
3.00	141		428	97		285
4.00	258		760	136	-	378
5.00	416	-	1184	182	-1	470
6.00	624	-2	1699	236	-2	560
7.00	888	-5	2306	291	-3	654
8.00	1207	-7	3007	347	-3	750
9.00	1585	-10	3806	410	-3	847
10.00	2026	-14	4702	473	-3	945
11.00	2532	-18	5695	539	-4	1041
12.00	3103	-21	6784	605	-4	1137
13.00	3744	-26	7970	675	-4	1235
14.00	4455	-31	9254	748	-5	1333
15.00	5239	-35	10637	822	-5	1433
16.00	6100	-41	12121	980	-5	1535
17.00	7040	-46	13708	980	-6	1639
18.00	8061	-52	15399	1063	-6	1744
19.00	9166	-58	17198	1148	-6	1853
20.00	10359	-64	19105	1237	-6	1962
21.00	11642	-71	21124	1329	-7	2076
22.00	13019	-78	23258	1426	-7	2194
23.00	14494	-85	25513	1527	-7	2317
24.00	16074	-92	27894	1633	-7	2446
25.00	17763	-99	30407	1746	-8	2582
26.00	19560	-107	33059	1864	-8	2724
27.00	21493	-115	35857	1987	-8	2872
28.00	23540	-123	38800	2107	-8	3012
29.00	25700	-131	41874	2210	-8	3129
30.00	27953	-139	45047	2292	-8	3212
31.00	30276	-147	48286	2350	-8	3262
32.00	32647	-155	51562	2391	-8	3288
33.00	35056	-163	54859	2425	-7	3303

32 LB PAYLOAD NOVEMBER 15, 1962

ORBITAL ELEMENTS

SEMIMAJOR AXIS	2.96906765E 003 N.M.
SEMIMINOR AXIS	2.43921400E 003 N.M.
ECCENTRICITY	5.70147890E-001
PERIOD	6.76996746E 001 MIN.
APOGEE ALTITUDE	1.21794981E 003 N.M.
PERIGEE ALTITUDE	-2.16766551E 003 N.M.
INCLINATION	3.05078462E 001 DEG.
ARGUMENT OF PERIGEE	-2.63843916E 001 DEG.
LONGITUDE OF ASCENDING NODE AT BURNOUT TIME	1.65357821E 002 DEG.
INITIAL TRUE ANOMALY	1.39466460E 002 DEG.
INITIAL ECCENTRIC ANOMALY	1.09577048E 002 DEG.

32 LB PAYLOAD NOVEMBER 15, 1962
KEPLERIAN TRAJECTORY

H	M	S	ALTITUDE(NM)	LATITUDE	LONGITUDE	INERTIAL VEL.(FPS)	INERTIAL FLT PATH ANG	RANGE(NM)
	2	33	9.48868E 001	27.99566	-78.32310	23020.94	33.17980	123.29
	3		1.50134E 002	27.67597	-76.93274	22578.52	32.83687	199.63
	3	30	2.09502E 002	27.31510	-75.45445	22108.63	32.42341	281.35
	4		2.66962E 002	26.94838	-74.03712	21658.91	31.97580	360.30
	4	30	3.22538E 002	26.57684	-72.67558	21228.33	31.49504	436.68
	5		3.76253E 002	26.20134	-71.36589	20815.97	30.98201	510.70
	5	30	4.28131E 002	25.82260	-70.10429	20421.00	30.43750	582.52
	6		4.78195E 002	25.44122	-68.88732	20042.65	29.86222	652.30
	6	30	5.26467E 002	25.05770	-67.71182	19680.26	29.25680	720.20
	7		5.72966E 002	24.67244	-66.57490	19333.20	28.62182	786.34
	7	30	6.17715E 002	24.28578	-65.47390	19000.93	27.95781	850.85
	8		6.60733E 002	23.89800	-64.40638	18682.95	27.26530	913.84
	8	30	7.02038E 002	23.50930	-63.37006	18378.81	26.54475	975.42
	9		7.41649E 002	23.11987	-62.36288	18088.12	25.79667	1035.69
	9	30	7.79583E 002	22.72984	-61.38288	17810.53	25.02151	1094.73
	10		8.15856E 002	22.33930	-60.42829	17545.70	24.21978	1152.64
	10	30	8.50484E 002	21.94833	-59.49743	17293.36	23.39193	1209.48
	11		8.83482E 002	21.55696	-58.58874	17053.26	22.53864	1265.33
	11	30	9.14864E 002	21.16523	-57.70078	16825.17	21.66031	1320.26
	12		9.44643E 002	20.77313	-56.83220	16608.90	20.75760	1374.34
	12	30	9.72833E 002	20.38065	-55.98171	16404.28	19.83114	1427.63
	13		9.99444E 002	19.98775	-55.14813	16211.17	18.88165	1480.18
	13	30	1.02449E 003	19.59440	-54.33032	16029.42	17.90985	1532.05
	14		1.04797E 003	19.20054	-53.52723	15858.94	16.91657	1583.29
	14	30	1.06992E 003	18.80610	-52.73786	15699.63	15.90267	1633.95
	15		1.09032E 003	18.41101	-51.96125	15551.42	14.86909	1684.07
	15	30	1.10919E 003	18.01518	-51.19650	15414.22	13.81683	1733.72
	16		1.12654E 003	17.61852	-50.44275	15288.00	12.74697	1782.91
	16	30	1.14238E 003	17.22093	-49.69917	15172.70	11.66065	1831.71
	17		1.15670E 003	16.82230	-48.96498	15068.29	10.55908	1880.14
	17	30	1.16953E 003	16.42253	-48.23942	14974.75	9.44354	1929.26

32 LB PAYLOAD NOVEMBER 15, 1962

STATION = WOOMERA

-31.3R0 N.LAT 136.890 E.LONG FT.

TIME H M S	TRAJECTORY (GEOCENTRIC) ALTITUDE N.MILES	LATITUDE DEG	LONGITUDE DEG	LOOK ANGLES AZIMUTH DEG	ELEVATION DEG	SLANT RANGE NM	ASPECT ANGLE DEG
2 33	95	27.84	-78.32	87.796	-74.444	6728	44.33
3	150	27.52	-76.93	89.065	-74.884	6799	43.78
3 30	210	27.17	-75.45	90.528	-75.357	6875	43.17
4	267	26.80	-74.04	92.055	-75.817	6948	42.58
4 30	323	26.44	-72.68	93.651	-76.262	7017	41.99
5	376	26.06	-71.37	95.319	-76.691	7084	41.41
5 30	428	25.69	-70.10	97.065	-77.105	7147	40.83
6	478	25.31	-68.89	98.893	-77.503	7209	40.27
6 30	526	24.93	-67.71	100.806	-77.885	7267	39.71
7	573	24.55	-66.57	102.809	-78.249	7323	39.16
7 30	618	24.16	-65.47	104.906	-78.596	7376	38.61
8	661	23.78	-64.41	107.099	-78.925	7427	38.07
8 30	702	23.39	-63.37	109.390	-79.235	7475	37.54
9	742	23.01	-62.36	111.782	-79.525	7521	37.01
9 30	780	22.62	-61.38	114.273	-79.795	7565	36.49
10	816	22.23	-60.43	116.863	-80.043	7607	35.98
10 30	850	21.84	-59.50	119.550	-80.270	7646	35.46
11	883	21.45	-58.59	122.328	-80.473	7683	34.96
11 30	915	21.06	-57.70	125.191	-80.654	7718	34.45
12	945	20.67	-56.83	128.131	-80.811	7751	33.95
12 30	973	20.28	-55.98	131.137	-80.943	7782	33.46
13	999	19.89	-55.15	134.198	-81.050	7811	32.97
13 30	1024	19.50	-54.33	137.299	-81.132	7837	32.48
14	1048	19.11	-53.53	140.425	-81.189	7862	32.00
14 30	1070	18.72	-52.74	143.561	-81.220	7885	31.52
15	1090	18.32	-51.96	146.690	-81.227	7905	31.04
15 30	1109	17.93	-51.20	149.797	-81.209	7924	30.57
16	1127	17.54	-50.44	152.866	-81.168	7941	30.09
16 30	1142	17.14	-49.70	155.883	-81.103	7956	29.63
17	1157	16.74	-48.96	158.837	-81.015	7969	29.16
17 30	1170	16.34	-48.24	161.715	-80.907	7981	28.69

MODIFIED SLV-1C SWTTS AUG 21,1962 255 LBS
STAGE 3

TIME SEC	LATITUDE DEG	LONGITUDE DEG	ALTITUDE NM	RANGE NM
1506.00	37.9374	218.6365	1111.19	1023.90
1516.00	37.9527	218.4926	1098.42	1030.77
1526.00	37.9681	218.3482	1085.74	1037.67
1536.00	37.9834	218.2031	1072.58	1044.60
1546.00	37.9988	218.0573	1059.11	1051.56
1556.00	38.0141	217.9109	1045.34	1058.55
1566.00	38.0295	217.7638	1031.27	1065.57
1576.00	38.0448	217.6159	1016.89	1072.63
1586.00	38.0601	217.4674	1002.21	1079.71
1596.00	38.0755	217.3180	987.22	1086.83
1606.00	38.0908	217.1679	971.93	1093.99
1616.00	38.1062	217.0170	956.32	1101.18
1626.00	38.1216	216.8653	940.39	1108.40
1636.00	38.1370	216.7127	924.15	1115.67
1646.00	38.1523	216.5592	907.59	1122.98
1656.00	38.1677	216.4048	890.71	1130.32
1666.00	38.1831	216.2494	873.50	1137.71
1676.00	38.1986	216.0932	855.97	1145.14
1686.00	38.2140	215.9359	838.11	1152.62
1696.00	38.2294	215.7775	819.92	1160.14
1706.00	38.2449	215.6182	801.39	1167.71
1716.00	38.2604	215.4577	782.93	1175.34
1726.00	38.2758	215.2961	763.32	1183.01
1736.00	38.2914	215.1334	743.77	1190.73
1746.00	38.3069	214.9695	723.98	1198.51
1756.00	38.3224	214.8043	703.44	1206.34
1766.00	38.3380	214.6378	683.04	1214.23
1776.00	38.3536	214.4701	662.08	1222.18
1786.00	38.3692	214.3010	640.77	1230.20
1796.00	38.3848	214.1304	619.09	1238.27
1806.00	38.4004	213.9585	597.05	1246.42
1816.00	38.4161	213.7850	574.63	1254.63
1826.00	38.4318	213.6100	551.84	1262.91
1836.00	38.4475	213.4334	528.67	1271.26
1846.00	38.4633	213.2552	505.11	1279.69

MODIFIED SLV-1C SWTTS AUG 21,1962 255 LBS
STAGE 3

TIME SEC	VELOCITY FPS	FPA DEG	AZIMUTH DEG
1506.00	9357.2	-53.82	-82.30
1516.00	9505.9	-54.42	-82.33
1526.00	9656.6	-55.01	-82.37
1536.00	9809.3	-55.59	-82.40
1546.00	9963.9	-56.14	-82.43
1556.00	10120.4	-56.69	-82.47
1566.00	10279.0	-57.21	-82.50
1576.00	10439.4	-57.73	-82.53
1586.00	10601.9	-58.23	-82.57
1596.00	10766.3	-58.72	-82.60
1606.00	10932.7	-59.19	-82.64
1616.00	11101.1	-59.66	-82.67
1626.00	11271.5	-60.11	-82.71
1636.00	11444.0	-60.55	-82.74
1646.00	11618.5	-60.98	-82.78
1656.00	11795.1	-61.40	-82.81
1666.00	11973.8	-61.81	-82.85
1676.00	12154.6	-62.20	-82.89
1686.00	12337.6	-62.59	-82.93
1696.00	12522.8	-62.97	-82.96
1706.00	12710.3	-63.34	-83.00
1716.00	12900.0	-63.70	-83.04
1726.00	13092.0	-64.06	-83.08
1736.00	13286.4	-64.40	-83.12
1746.00	13483.2	-64.74	-83.16
1756.00	13682.5	-65.07	-83.20
1766.00	13884.3	-65.39	-83.25
1776.00	14088.7	-65.70	-83.29
1786.00	14295.7	-66.01	-83.33
1796.00	14505.4	-66.31	-83.38
1806.00	14717.9	-66.60	-83.42
1816.00	14933.2	-66.89	-83.47
1826.00	15151.4	-67.17	-83.52
1836.00	15372.6	-67.44	-83.56
1846.00	15596.9	-67.71	-83.61

REFERENCES

1. Hawkins, Jerry. Approximate Trajectory Equations for Unguided Fin-Stabilized Multistage Rockets. APGC-TN-60-27, AD, June 1960
2. Harrington R. R., and O'Connor, A. D. General Computer Program for Unguided Multistage Rocket Trajectories. AFSWC-TR-59-17, Air Force Special Weapons Center, Kirtland AFB, New Mexico, July 1959
3. Brown, R., Brulle, R., and Giffif. Six-Degree-of-Freedom Flight Path Study Generalized Computer Program. WADD TR-60-781, McDonnell A/C Corporation, St Louis, Missouri, May 1961
4. Gianopulas, G. N. Generalized Powered Flight Trajectory Program for the IBM 704 Computer. JPL TR No. 32-38, California Institute of Technology, Pasadena, California, September 1960
5. Vought Astronautics, Dallas, Texas. Six-Degree-of-Freedom Three Dimensional, Trajectory and Control Analysis Routine for the IBM 704 Computer, Appendix A, AST/EIR-12435.
6. Baker, R. and Makemson, M. An Introduction to Astrodynamics. Academic Press, New York, 1960
7. Gersten, Robert. Geodetic Sub-Latitude and Altitude of a Space Vehicle. The Journal of Astronautical Sciences, Vol III, No. 11, Spring 1961
8. Dayton, A. D. An Earth Model. SWR-TM-61-5, Air Force Special Weapons Center, Kirtland AFB, New Mexico, October 1961
9. Goldstein, Herbert. Classical Mechanics, Addison Wesley, 1950
10. Kolk, Richard W. Modern Flight Dynamics, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1961
11. Thomson, William. Introduction to Space Dynamics, John Wiley & Sons, New York, 1961
12. Sterne, Theodore, E. An Introduction to Celestial Mechanics, Interscience Publishers, Inc., New York, 1960
13. Makerson, M., Baker, R., and Westrom, G. Analysis and Standardization of Astrodynamics Constants. The Journal of Astronautical Sciences, Vol III, No. 1, Spring 1961
14. Control Data Corporation, Minneapolis, Minnesota. Systems Programs Control Data Model 1604 Digital Computer, Vol 1, for Kirtland AFB, New Mexico, 1961
15. Sissenwine, Norman. Announcing the U.S. Standard Atmosphere, Astronautics, page 52, 53, August 1962
16. Geophysics Research Directorate, ARDC, USAF. The ARDC Model Atmosphere, 1959. AFCRC-TR-59-267, August 1959

REFERENCES (cont'd)

17. Christensen, C. S., and Dayton, A. D. A FORTRAN Computer Program of a Keplerian Trajectory, AFSWC-TDR-62-19, Air Force Special Weapons Center, Kirtland AFB, New Mexico, February 1962

18. Hildebrand, F. Introduction to Numerical Analysis.

19. Milne. Numerical Calculus, Princeton University Press, pp 68-78, 1949

DISTRIBUTION

No. cys	
	HEADQUARTERS USAF
2	Hq USAF (AFCOA), Wash 25, DC
2	Hq USAF (AFOOP), Wash 25, DC
2	Hq USAF (AFRDP), Wash 25, DC
2	Hq USAF (AFORQ), Wash 25, DC
2	Hq USAF (AFRDR), Wash 25, DC
1	Hq USAF (AFRDR-NU-1), Wash 25, DC
1	Hq USAF (AFCIN), Wash 25, DC
2	Hq USAF (AFDSD), Wash 25, DC
2	Hq USAF (AFTAC), Wash 25, DC
1	USAF Dep IG for Insp (AFCDI-B-3), Norton AFB, Calif
2	AFOAR, Bldg T-D, Wash 25, DC
4	AFCRL, Hanscom Fld, Bedford, Mass
1	AFOSR, Bldg T-D, Wash 25, DC
2	ARL (RRLO), Wright-Patterson AFB, Ohio
	MAJOR AIR COMMANDS
	AFSC, Andrews AFB, Wash 25, DC
2	(SCT)
1	(SCT-2)
	SAC, Offutt AFB, Nebr
1	(OA)
1	(OAWS)
1	(DICC)
2	ADC (Ops Anlys), Ent AFB, Colorado Springs, Colo
1	ATC, ATTN: S-W, Randolph AFB, Tex
2	AUL, Maxwell AFB, Ala
5	USAFIT (USAF Institute of Technology), Wright-Patterson AFB, Ohio
1	USAFE, APO 633, New York, NY
5	USAFA, United States Air Force Academy, Colo

DISTRIBUTION (cont'd)

No. cys	
	AFSC ORGANIZATIONS
5	AFSC Regional Office, 6331 Hollywood Blvd., Los Angeles 28, Calif
2	FTD (Library), Wright-Patterson AFB, Ohio
	ASD, Wright-Patterson AFB, Ohio
2	(ASAPRL, Technical Doc Library)
1	(ASOOE)
1	(ASAPR)
1	(ASTFP)
1	(ASRCP)
1	(ASAMCV)
	BSD, Norton AFB, Calif
1	(Technical Library)
1	(BST)
1	(BSQ)
1	(BSR)
	SSD, AF Unit Post Office, Los Angeles 45, Calif
2	(SSSC-TDC)
2	(SSVD)
2	ESD (ESAT), Hanscom Fld, Bedford, Mass
2	AF Msl Dev Cen, (RRR-T, Tech Library), Holloman AFB, NM
2	AFFTC (FTFT), Edwards AFB, Calif
2	AFMTC (MU-135), Patrick AFB, Fla
4	APGC (PGAPI), Eglin AFB, Fla
4	RADC (Document Library), Griffiss AFB, NY
1	AEDC (AEOI), Arnold Air Force Station, Tenn
	KIRTLAND AFB ORGANIZATIONS
	AFSWC, Kirtland AFB, NM
1	(SWEH)
50	(SWOI)
5	(SWR)

DISTRIBUTION (cont'd)

No. cys	
10	(SWV)
10	(SWTD)
2	(SWRPA)
2	(SWRPL)
2	(SWRPS)
2	(SWRPI)
5	(SWRJ)
30	(SWTTS)

OTHER AIR FORCE AGENCIES

No. cys	
	Director, USAF Project RAND, via: Air Force Liaison Office, The RAND Corporation, 1700 Main Street, Santa Monica, Calif
1	(RAND Physics Div)
1	(RAND Library)

ARMY ACTIVITIES

No. cys	
2	Chief of Research and Development, Department of the Army (Special Weapons and Air Defense Division), Wash 25, DC
2	Redstone Scientific Info Center, US Army Ordnance Missile Command, Redstone Arsenal, Ala
2	Director, Ballistic Research Laboratories (Library), Aberdeen Proving Ground, Md
2	Commandant, US Army Ordnance School, Aberdeen Proving Ground, Md
2	Commanding Officer, US Army Signal Research & Development Laboratory, ATTN: SIGRA/SL-SAT-1, Weapons Effects Section, Fort Monmouth, NJ
2	Research Analysis Corp., ATTN: Document Control Office, 6935 Arlington Road, Bethesda, Md., Wash 14, DC
2	Director, Army Research Office, Arlington Hall Sta, Arlington, Va
2	Commanding General, White Sands Missile Range (Technical Library), White Sands, NM

NAVY ACTIVITIES

No. cys	
	Chief of Naval Operations, Department of the Navy, Wash 25, DC
1	(OP-36)
1	(OP-75)

DISTRIBUTION (cont'd)

No. cys	
2	Chief of Naval Research, Department of the Navy, Wash 25, DC
1	Chief, Bureau of Naval Weapons, Department of the Navy, Wash 25, DC
2	Commanding Officer, Naval Research Laboratory, Wash 25, DC
1	Commanding Officer and Director, Navy Electronics Laboratory (Code 4223), San Diego 52, Calif
2	Commanding Officer, US Naval Schools Command, US Naval Station, Treasure Island, San Francisco, Calif
2	Commander, Naval Air Missile Test Center, Point Mugu, Calif
2	Commander, Naval Ordnance Test Station, Inyokern (Code 12), China Lake, Calif
1	Director, Special Projects, Department of the Navy, Wash 25, DC
5	Office of Naval Research, Wash 25, DC
	OTHER DOD ACTIVITIES
2	Chief, Defense Atomic Support Agency (Document Library), Wash 25, DC
2	Commander, Field Command, Defense Atomic Support Agency (FCAG3, Special Weapons Publication Distribution), Sandia Base, NM
2	Director, Weapon Systems Evaluation Group, Room 2E1006, The Pentagon, Wash 25, DC
4	Director, Advanced Research Projects Agency, Department of Defense, The Pentagon, Wash 25, DC
2	Director, Defense Research & Engineering, The Pentagon, Wash 25, DC
10	ASTIA (TIPDR), Arlington Hall Sta, Arlington 12, Va
	AEC ACTIVITIES
1	US Atomic Energy Commission (Headquarters Library), Wash 25, DC
4	Sandia Corporation (Tech Library), Sandia Base, NM
2	University of California Lawrence Radiation Laboratory (Technical Information Division, ATTN: Mr. Clovis Craig), P. O. Box 808, Livermore, Calif
1	Director, Los Alamos Scientific Laboratory (Helen Redman, Report Library), P.O. Box 1663, Los Alamos, NM

DISTRIBUTION (cont'd)

No. cys	OTHER
4	Administrator, National Aeronautics and Space Administration, 1520 H Street NW, Wash 25, DC
2	Langley Research Center (NASA), Langley Fld, Hampton, Va
1	National Bureau of Standards, Radiological Equipment Section, Wash 25, DC
1	Central Intelligence Agency (OCR/LY/ILS), Wash 25, DC
1	OTS, Department of Commerce, Wash 25, DC
1	Space Technology Labs, Inc., ATTN: Information Center, Document Procurement, P.O. Box 95001, Los Angeles 45, Calif
1	Battelle Memorial Institute, 505 King Avenue, Columbus, Ohio
1	Forestal Research Center Library, Aeronautical Sciences Bldg., Princeton University, Princeton, NJ
1	University of Illinois, Eastern Experiment Station, US Bureau of Mines, College Park, Md
2	University of Illinois, Talbot Laboratory, Room 207, Urbana, Ill
1	Massachusetts Institute of Technology, Division of Industrial Cooperation, 77 Massachusetts Avenue, Cambridge, Mass
1	Institute for Defense Analysis, Room 2B257, The Pentagon, Wash 25, DC
1	Institute of the Aerospace Sciences, Inc., 2 East 64th Street, New York 21, NY
1	Convair, A Division of General Dynamics Corporation, P.O. Box 1950, San Diego 12, Calif
10	Aero Space Corporation, El Segundo, Calif. ATTN: Mr. Carl S. Christensen
1	Official Record Copy (SWTTS, Lt Dayton)

Air Force Special Weapons Center, Kirtland AF Base, New Mexico
Rpt No. AFSWC-TDR-63-11. SPINNING UNGUIDED ROCKET TRAJECTORY DIGITAL COMPUTER PROGRAM (SPURT) 230 p. incl illus., tables, 19 refs. February 1963.

Unclassified Report

SPURT is a five-degree-of-freedom trajectory digital computer program for spinning unguided space probe vehicles. The program was written for the Control Data Corporation 1604 digital computer. SPURT will compute trajectories for a vehicle up to a maximum of ten stages and has provision for computing the trajectories of the separated stages.

This generalized program computes the trajectory over an oblate spheroidal, rotating Earth with atmosphere, in a geocentric rectangular coordinate

1. Digital computers
2. Drag
3. Gravitation
4. Kinematics
5. Rockets
6. SPURT (calculation)
7. Thrust
8. Trajectories

I. Project AEEP
II. Allen Dayton, Lt USAF
III. In ASTIA collection

Air Force Special Weapons Center, Kirtland AF Base, New Mexico
Rpt No. AFSWC-TDR-63-11. SPINNING UNGUIDED ROCKET TRAJECTORY DIGITAL COMPUTER PROGRAM (SPURT) 230 p. incl illus., tables, 19 refs. February 1963.

Unclassified Report

SPURT is a five-degree-of-freedom trajectory digital computer program for spinning unguided space probe vehicles. The program was written for the Control Data Corporation 1604 digital computer. SPURT will compute trajectories for a vehicle up to a maximum of ten stages and has provision for computing the trajectories of the separated stages.

This generalized program computes the trajectory over an oblate spheroidal, rotating Earth with atmosphere, in a geocentric rectangular coordinate

1. Digital computers
2. Drag
3. Gravitation
4. Kinematics
5. Rockets
6. SPURT (calculation)
7. Thrust
8. Trajectories

I. Project AEEP
II. Allen Dayton, Lt USAF
III. In ASTIA collection

Air Force Special Weapons Center, Kirtland AF Base, New Mexico
Rpt No. AFSWC-TDR-63-11. SPINNING UNGUIDED ROCKET TRAJECTORY DIGITAL COMPUTER PROGRAM (SPURT) 230 p. incl illus., tables, 19 refs. February 1963.

Unclassified Report

SPURT is a five-degree-of-freedom trajectory digital computer program for spinning unguided space probe vehicles. The program was written for the Control Data Corporation 1604 digital computer. SPURT will compute trajectories for a vehicle up to a maximum of ten stages and has provision for computing the trajectories of the separated stages.

This generalized program computes the trajectory over an oblate spheroidal, rotating Earth with atmosphere, in a geocentric rectangular coordinate

1. Digital computers
2. Drag
3. Gravitation
4. Kinematics
5. Rockets
6. SPURT (calculation)
7. Thrust
8. Trajectories

I. Project AEEP
II. Allen Dayton, Lt USAF
III. In ASTIA collection

Air Force Special Weapons Center, Kirtland AF Base, New Mexico
Rpt No. AFSWC-TDR-63-11. SPINNING UNGUIDED ROCKET TRAJECTORY DIGITAL COMPUTER PROGRAM (SPURT) 230 p. incl illus., tables, 19 refs. February 1963.

Unclassified Report

SPURT is a five-dgree-of-freedom trajectory digital computer program for spinning unguided space probe vehicles. The program was written for the Control Data Corporation 1604 digital computer. SPURT will compute trajectories for a vehicle up to a maximum of ten stages and has provision for computing the trajectories of the separated stages.

This generalized program computes the trajectory over an oblate spheroidal, rotating Earth with atmosphere, in a geocentric rectangular coordinate

1. Digital computers
2. Drag
3. Gravitation
4. Kinematics
5. Rockets
6. SPURT (calculation)
7. Thrust
8. Trajectories

I. Project AEEP
II. Allen Dayton, Lt USAF
III. In ASTIA collection

system. All input and output data are in geodetic coordinates.

Coasting flight trajectories are computed in two subroutines. The first is a Keplerian solution, which also computes orbital elements and "look angles" for various tracking stations. The second uses three-degree-of-freedom point mass equations solved by numerical integrations.

The program will prepare two special output tapes. One is used in plotting output data and the other is used to prepare a special tape for the Atlantic Missile Range.

system. All input and output data are in geodetic coordinates.

Coasting flight trajectories are computed in two subroutines. The first is a Keplerian solution, which also computes orbital elements and "look angles" for various tracking stations. The second uses three-degree-of-freedom point mass equations solved by numerical integrations.

The program will prepare two special output tapes. One is used in plotting output data and the other is used to prepare a special tape for the Atlantic Missile Range.

system. All input and output data are in geodetic coordinates.

Coasting flight trajectories are computed in two subroutines. The first is a Keplerian solution, which also computes orbital elements and "look angles" for various tracking stations. The second uses three-degree-of-freedom point mass equations solved by numerical integrations.

The program will prepare two special output tapes. One is used in plotting output data and the other is used to prepare a special tape for the Atlantic Missile Range.

system. All input and output data are in geodetic coordinates.

Coasting flight trajectories are computed in two subroutines. The first is a Keplerian solution, which also computes orbital elements and "look angles" for various tracking stations. The second uses three-degree-of-freedom point mass equations solved by numerical integrations.

The program will prepare two special output tapes. One is used in plotting output data and the other is used to prepare a special tape for the Atlantic Missile Range.