77-FM-18
Vol. III, Rev. 1

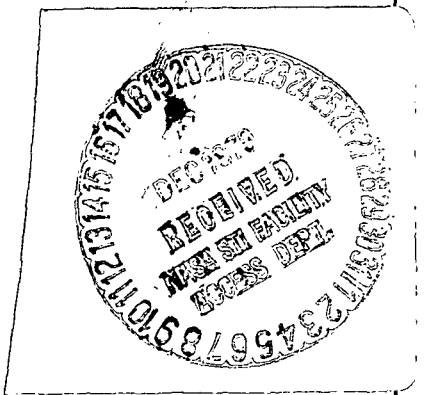
JSC-12564

Flight Design System-1
System Design Document

Processor Library
Book 1

Mission Planning and Analysis Division

October 1979



National Aeronautics and
Space Administration

Lyndon B. Johnson Space Center
Houston, Texas

77-FM-18
Vol. III, Rev. 1

JSC-12564

SHUTTLE PROGRAM

FLIGHT DESIGN SYSTEM-1
SYSTEM DESIGN

PROCESSOR LIBRARY
BOOK 1

By Software Development Branch

Approved: _____
Elric N. McHenry, Chief
Software Development Branch

Approved: _____
Ronald L. Berry, Chief
Mission Planning and Analysis Division

Mission Planning and Analysis Division

National Aeronautics and Space Administration

Lyndon B. Johnson Space Center

Houston, Texas

October 1979

PREFACE

The Flight Design System-1 (FDS-1) is a pilot project to evaluate current concepts and to determine the hardware/software capability that will be required for the operational era to support Shuttle flight planning. This software system is being implemented on a Hewlett-Packard 21MX computer with a Daconics documentation system and will provide terminal-based interactive flight planning capability.

The System Design Document (SDD) for FDS-1 is the specification for and description of this hardware/software facility. The SDD is logically organized into 10 published volumes. This organization is presented in the accompanying table. The material in the early volumes is primarily presented from the user's point of view, whereas the latter material is software-developer oriented. The SDD will be published by volumes over a period of time, and various volumes will be updated and republished during the development of FDS-1.

FDS-1 SYSTEM DESIGN DOCUMENT

Volume I	Introduction, Overview, and User Interface
*Volume II	Utility Processor Library
Volume III	Processor Library
Volume IV	System Architecture and Executive
Volume V	Data Management and Data Base Documentation Support System
Volume VI	Standards
Volume VII	Utility Support Software
Volume VIII	Build and Delivery Procedures, Software Development, Debug, and System Build Aids
Volume IX	Executive Logic Flow - Program Design Language
Volume X	Document Change Request Procedure and Submittal Form

*Volume II has been incorporated into volume III and renamed Processor Library, volume III, revision 1.

ACKNOWLEDGMENT

This document was written and prepared by a team consisting of Johnson Space Center civil service and contractor personnel. The following organizations made significant contributions to this document:

International Business Machines, Inc.
Federal Systems Division
Mission Analysis and Engineering

Lockheed Electronics Company, Inc.
Systems and Services Division

National Aeronautics and Space Administration/Johnson Space Center
Data Systems and Analysis Directorate
Mission Planning and Analysis Division
Flight Planning Branch
Software Development Branch

The Mission Planning and Analysis Division (MPAD) directed the effort and, in addition, developed software and submitted completed draft writeups of some sections. MPAD wishes to acknowledge the excellent support received from all organizations involved in preparing this document.

CONTENTS

CASKU AND QUIKU OUTPUT DISPLAY PROCESSOR (LKOUT)

xli

TABLES

Table Page

FIGURES

1.0 INTRODUCTION

This volume presents the complete program documentation for the processors in the Flight Design System-1 (FDS-1). Two types of processors exist in the processor library; utility processors and application processors. Utility processors provide a general capability that is not particularly associated with flight design, such as Data Box Display (DBDSP), Data Element Definition (DEFIN), or Conditional Execution Processors (IF, ELSE, ENDIF). Application processors provide capability that is directly related to accomplishing flight design, such as Ascent (ASENT), General Purpose Maneuver Processor (GPMP), Groundtrack (GTRAK), or Mission Plan Table Processor (MPTP).

Both types of processors are presented in this volume in alphabetical order; thus, volume II of the SDD documentation is contained in this volume and will not exist separately. The only documentation that exists in addition to this volume is the software listings and their comment cards.

Because of the magnitude of this volume, it is published in three books.

2.0 PROCESSOR LIBRARY

This volume contains the complete software documentation for all FDS-1 processors. The processors are paged alphabetically according to the processor names as shown below.

Book 1

Ascent Processor	ASENT
Data Assignment Processor	ASSGN
Attitude Tables Maintenance Processor	ATM
Basetime Initialization Processor	BASTM
Consumable Analysis for Shuttle Kits Processor	CASKU
Coasting State Vector Predictor (including AEG) Processor	COAST
Data Box Display Processor	DBDSP
Data Box Extractor Processor	DBEXT
Data Box Interpolator Processor	DBINT
Data Element Definition Processor	DEFIN
Sequence Iteration Processors (DO, ENDDO)	DO
Document Processor	DOC
Deorbit Target Module Processor	DTM

Book 2

Early Repeating Groundtrack Orbits Processor	ERGO
* Finite Burn Processor	FINBN
* Fixed Magnitude Two-Burn Processor	FM2BN
Flight Plan Display Processor	FPD
General Purpose Maneuver Processor	GPMP
Groundtrack Processor	GTRAK
Conditional Execution Processors (IF, ELSE, ENDIF)	IF

*To be supplied.

Invariant Element Ephemeris Processor	INVAR
CASKU and QUIKU Output Data Display Processor	LKOUT
Landing Opportunities Processor	LOPT
Load State Vector Processor	LSV
Launch Window Processor	LWP
* Maneuver Iterator Processor	MANIT
Matrix and Attitude Support Table Processor	MAST
Master Data Temporary Print Processor	MDTP
Mission Plan Table Processor	MPTP
* Node Definer Processor	NODE
Orbital Maneuver Processor (Rendezvous)	OMP
Book 3	
Input/Output Units Specification Processor	PHYDM
* Placement Longitude Processor	PLLON
Print State Vector Processor	PSV
Phase Table Print Processor	PTP
Quick Investigation of Consumables Kits Processor	QUIKU
Parametric Scan Processors (SCAN, ENDSC)	SCAN
Sunrise/Sunset Time Predictor Processor	SRSS
* Shuttle/SUS Burn Relative Motion Processor	SSBRM
Sun Synchronous Orbits Processor	SSYN
Station Contacts Processor	STACN
Summary Table Print Processor	STP
State Vector Units Conversion Processor	SVUCP

*To be supplied.

* State Vector Coordinate Transformation Processor	TFSV
Activity Timeline Processor	TMLNU

*To be supplied.

ASCENT PROCESSOR (ASENT)

1.0 PURPOSE

The Launch Ascent Processor, ASENT, simulates the Shuttle launch profile from lift-off to main engine cutoff (MECO), with conditions for a nominal (no failures) trajectory as well as an abort during ascent. Two state vectors are generated: an SRB state at staging, and an Orbiter state at MECO.

2.0 FUNCTIONAL DESCRIPTION

The ASENT processor simulates four guidance phases of the launch profile which can result in a nominal (no failures) trajectory or an abort-once-around (AOA) trajectory. The target plane descending node, or the time of lift-off with respect to the launch site passing under the target plane, is used as a reference for the position of the launch site relative to the target plane. Also, the launch azimuth is specified. The reference coordinate plane is coincident with the target plane, and the reference system is the Earth rotational system fixed at Greenwich.

The first guidance phase simulated is the vertical rise phase from 0 to 6 seconds, ground elapsed time (g.e.t). This 6 seconds is the length of time necessary to achieve tower clearance (i.e., to reach the point at which the base of the solid rocket booster (SRB) nozzle is above the top of the tower lightning rod). At 6 seconds, the second guidance phase, pitchover, begins. The pitch and yaw angles, measured in the cylindrical e_r, e_ϕ, e_z systems, are calculated to give a linear rate in true pitch between 6 and 16 seconds. From 16 seconds until the time for SRB staging, the gravity turn phase is simulated. During this phase, the pitch and yaw angles necessary to steer along the velocity vector are calculated. The last phase simulated, which begins at SRB staging time and lasts until MECO is reached, is the explicity guidance phase. The pitch and yaw angles for this phase are calculated by a guidance logic.

At a specified time into the flight, which is supplied by the user, the initial MECO targets are converted to a second target set. The Space Shuttle main engine (SSME) thrust level is reset to a user-specified value. If the user chooses, the OMS and RCS engines are turned on.

The convergence criteria for the processor is MECO time. When the estimated time to MECO is $\pm$0.5 seconds of the actual time to MECO, the case is considered converged.

3.0 ASSUMPTIONS AND LIMITATIONS

ASENT simulates a launch profile from lift-off to MECO, and all parameters are computed as the values at exact time of MECO. The value output as weight-to-orbit is the total Orbiter weight, which includes external tank (ET) weight because the ET is not dropped until approximately 10 seconds after MECO is reached. ASENT performs no modeling of ET propellant weight except for the

value ETPR, which is the actual ET propellant consumed. The parameter GLOW is the gross lift-off weight and contains the weight of the filled ET.

The altitude that is computed by ASENT is the distance between the radius to the vehicle and the radius of the Earth at the subvehicle point. This is not altitude as define in figure 7.3-4 in volume I.

The Patrick Reference Atmosphere Model of 1963 is the only atmosphere density model available to the ASENT processor for FDS-1.

4.0 PROCESSOR INPUT/OUTPUT

a. Processor interface table - The definition of the ASENT processor interface table parameters is provided in table 4-I.

The GLOCON array contains a set of constants that are universal to all of the FDS processors. ASENT accesses this array for the constants it requires. The default values are stored in !!GLCN.

The session constants used by ASENT are stored in the array SESCON with the default values in !SESCN. This array is also available to all FDS processors and contains some constants that are universal for FDS.

Specific constants and tolerances required by the ASENT processor are maintained in PROCON. This set of constants contains delta values for other parameters, flags for internal print analysis, and other parameters whose values will probably not change over long periods of time.

GLOW is the gross lift-off weight value for the Shuttle vehicle. The value contains the weight of the Orbiter, any payloads, all fuel loaded and the external fuel tanks.

PITCHV is the pitch angle with respect to the vertical at the end of the pitchover phase.

HDANG is the heading angle with respect to the launch reference coordinate plane at lift-off.

TGINC is the target inclination.

AZDIR is the launch azimuth specification. It is alphanumeric value "NO" for northerly launch and "SO" for southerly launch.

LATLS is the geodetic latitude of the launch site. The GLOCON array contains preset values for the Eastern Test Range (ETR) or the Western Test Range (WTR). The user may access either of these, or another value may be input. Note, however, that the value must be input in radians.

LONLS is the geographic longitude of the launch site. As with the latitude, the user may use either an ETR or WTR value from GLOCON or input another. The input value must be in radians.

RLS is the radius of the launch site. The user may use either an ETR or WTR value from GLOCON or input another. The input value must be in feet.

TARG1 is the altitude, velocity, and flightpath angle for the initial MECO target set. TARG2 (altitude, velocity, and flightpath angle) is the second target set. The initial set is changed to the second set at the time specified by the user as the time of engine failure (TAOA). In previous simulations, the trajectory for a nominal ascent is generally the same as that for an AOA trajectory through the time of engine failure. Using this as an assumption, TARG1 contains the AOA target values and TARG2 contains the nominal targets.

TTGA is the estimated time from launch to MECO (used in the fourth stage guidance logic). This variable is the MECO target time for the explicit guidance from staging until the time specified by the TAOA parameter. At TAOA, the MECO target time is changed to the value initialized by the parameter TTG.

The flag AOANOM selects whether an "AOA" or a nominal trajectory is simulated following the time of TAOA. If the value is "AOA", one of the three main engines is failed and the flags OMSFG and RCSFG, are tested. If the value is "NOM", no engine is failed and the flags are not tested.

TAOA is the time at which an AOA occurs. At this time, the throttle setting (THRT) is reset to THROTA, and the targets (TARG1) are reset to TARG2.

THROT is the initial percent of the throttle setting for all main engine thrusting. THROTA is the percent of the throttle setting that is used after TAOA.

OMSFG and RCSFG are flags to determine if the OMS and RCS engines are to be on ("ON") or off ("OF") during the "AOA" simulation.

OMSAV and RCSAV are the amounts of OMS and RCS propellant available for use during the ascent simulation.

ATMOS is a flag that indicates which atmosphere density model is to be used during the simulation. The Patrick Reference Atmosphere Model of 1963 is the only atmosphere density model available to ASENT.

TTG is the estimated time from lift-off to MECO that is used after TAOA. The explicit guidance logic iterates on this value to converge the targeting. After the first iteration, TTG is recomputed internally.

MAXQ is the maximum dynamic pressure (Q) allowed.

TLNFL is the time of launch flag. If the time of launch (TLN) is input, (TLNFL = "TL") the descending node of the target plane with respect to the launch site (TIGM) is computed internal. If TIGM is input (TLNFL = "TI"), TLN is computed internally.

TLN is the time of launch with respect to the theoretical inplane time. A negative value means lift-off prior to inplane, and a positive value means lift-off after inplane. TLN must be input in seconds.

TIGM is the descending node of the target plane with respect to the launch site meridian, defined at lift-off.

DSPLY is a flag that selects the optional Detailed Ascent Profile display.

The following sets of parameters consist of data from the May 1977 Aeordynamic Design Data Book: CAXMN, FHBAF, ISP, and THRST. CAXMN is an array of the coefficients of axial force and their Mach numbers. FHBAF contains the force and altitude of the base axial force (BAF). ISP is an array of the SRB specific impulse (ISP) and the time for each ISP value. THRST is the SRB thrust table and the corresponding time values.

TSTAG is the time for SRB staging. TSTAG must be compatible with the last SRB thrust table time value in the array THRST.

ENGNS contains the engine parameters (i.e., vacuum thrust, specific impulse, sea level thrust factors, and cant angles) for three stages. The first stage values are not used. The three main engines are defined by the third stage values.

OMSTHR and OMSISP are the vacuum thrust and specific impulse of the OMS system.

ARSRB and CDSRB are the area and coefficient of drag values used in creation of the SRB state vector defined at staging.

ARORB and CDORB are the area and coefficient of drag values used in creation of the Orbiter state vector defined at MECO.

RCSTHR and RCSISP are the vacuum thrust and ISP of the RCS system.

SRBWT contains the SRB case weight and propellant weight.

OMSUSE and RCSUSE are the total amounts of OMS and RCS propellant used during ascent. They are output quantities which may be saved by the user.

STSRB is the SRB output state at staging time. STMECO is the Orbiter output state at the solution MECO time.

SUMTAB is the output summary table data array. The information placed in this array is available for recall by the user at any later time in the session.

b. Interface table data arrays definitions - The detailed definition of the data arrays contained in the interface table is printed in table 4-II.

c. Interface table data files definitions - None.

d. Processor solicited (prompted) inputs - None.

e. Processor display and display parameter definition table - The ASENT processor generates two displays, Detailed Ascent Profile and Ascent Display. When selected, the Detailed Ascent Profile display is repeated for each iteration of the solution process. The Ascent Display is always generated showing the converged solution. The format and content of the

ASENT displays and a definition of the display parameters are provided in tables 4-III(a) through 4-III(d).

f. Processor message table - The messages that will be output to a user, either to denote error conditions or for general information, while executing the ASENT processor are given in table 4-IV. When the processor terminates, the output data elements are not produced.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-V.

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR ASENT

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
GLOCON	AWA	Free	I	180	180	!!GLCN	Global constants array master data base element
SESCON	AWA	Free	I	90	90	!!SESCN	Session constants array
PROCON	AWA	Free	I	18	18		ASENT program constants and tolerances
GLOW	AWA	Real	I	2	1		Gross lift-off weight
PITCHV	AWA	Real	I	2	1		Pitch angle with respect to vertical at end of pitchover phase
HDANG	AWA	Real	I	2	1		Heading angle with respect to launch (coordinate) plane at lift-off
TGINC	AWA	Real	I	2	1		Target inclination
AZDIR	AWA	2CH	I	1	1	"NO"	Code for north (NO) or south (SO) launch azimuth
LATLS	AWA	Real	I	2	1		Geodetic latitude of launch site, radians
LONLS	AWA	Real	I	2	1		Geographic longitude of launch site, radians

NOTES	CLASS	TYPE			USE
	AWA	Free	2CH	72CH	I = Input
	Disk	Intg	6CH	Mix	O = Output
		Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-I.- Continued

PROCESSOR ASENT

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
RLS	AWA	Real	I	2	1		Radius of launch site, feet
TARG1	AWA	Real	I	6	3		Altitude, velocity, and flightpath angle for initial targeting of trajectory
TARG2	AWA	Real	I	6	3		Altitude, velocity, and flightpath angle for second set of MECO targets
TTGA	AWA	Real	I	2	1	200.	Time; launch to MECO used for fourth stage guidance logic prior to TAOA, seconds
AOANOM	AWA	6CH	I	3	1		Code to select either nominal ("NOM") or AOA ("AOA") trajectory logic
TAOA	AWA	Real	I	2	1		Time at which targets are reset, seconds
THROT	AWA	Real	I	2	1	100.	Initial percent of throttle setting for main engine thrusting
THROTA	AWA	Real	I	2	1	100	Value that throttle setting is reset to at TAOA
OMSFG	AWA	2CH	I	1	1	"OF"	Flag for OMS engine on ("ON") or off ("OF") for AOA trajectory

NOTES	CLASS	TYPE			USE
	AWA	Free	2CH	72CH	I = Input
	Disk	Intg	6CH	Mix	O = Output
		Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-I.- Continued

PROCESSOR ASENT

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
RCSFG	AWA	2CH	I	1	1	"OF"	Flag for RCS engine on ("ON") or off ("OF") for AOA trajectory
OMSAV	AWA	Real	I	2	1		Amount of OMS propellant available for use during ascent
RCSAV	AWA	Real	I	2	1		Amount of RCS propellant available for use during ascent
ATMOS	AWA	2CH	I	1	1	"PA"	Atmosphere model flag:= "PA" Patrick Reference '63
TTG	AWA	Real	I	2	1		Time from; liftoff to MECO used by guidance after TAOA, seconds
MAXQ	AWA	Real	I	2	1		Maximum Q desired by throttling, units are force/length^2
TLNFL	AWA	2CH	I	1	1		Time of launch flag: ="TI", input TIGM; compute TLN ="TL", INPUT TLN; compute TIGM
TLN	AWA	Real	I	2	1		Time of launch WRT theoretical inplane point, seconds
TIGM	AWA	Real	I	2	1		Descending node of target plane WRT launch site

NOTES	CLASS	TYPE			USE
	AWA	Free	2CH	72CH	I = Input
	Disk	Intg	6CH	Mix	O = Output
		Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-I.- Continued

PROCESSOR ASENT

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
DSPLY	AWA	2CH	I	1	1	"NO"	Flag to select the detailed ascent profile display
CAXMN	AWA	Real	I	76	(19,2)		Coefficient of axial force and corresponding Mach number (drag coefficients)
FHBAF	AWA	Real	I	56	(14,2)		Force and corresponding altitude of base axial force, in lbf and feet
ISP	AWA	Real	I	40	(10,2)		Specific impulse and corresponding time for the SRB, each in seconds
THRST	AWA	Real	I	80	(20,2)		SRB thrust profile, in 1000 lbf and seconds
TSTAG	AWA	Real	I	2	1	123.	Time of SRB staging, seconds
ENGNS	AWA	Real	I	24	(3,4)		Vacuum thrust, specific impulse, thrust factors and cant angles for the SRB and main engines, in lbf, seconds lbf, and degrees respectively
OMSTHR	AWA	Real	I	2	1		Vacuum thrust of OMS system
OMSISP	AWA	Real	I	2	1		Specific impulse of OMS system, seconds

NOTES

CLASS
AWA
Disk

TYPE
Free 2CH 72CH
Intg 6CH Mix
Real 18CH Symb
Dubl 36CH

USE
I = Input
O = Output
I/O = Input/Output

TABLE 4-I.- Continued

PROCESSOR ASENT

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
ARSRB	AWA	Real	I	2	1		Area for the drag computation for the SRB
CDSRB	AWA	Real	I	2	1		Coefficient of drag for the SRB at staging
ARORB	AWA	Real	I	2	1		Area for the drag computation for the Shuttle
CDORB	AWA	Real	I	2	1		Coefficient of drag for the Shuttle state vector at MECO
RCSTHR	AWA	Real	I	2	1		Vacuum thrust of RCS system
RCSISP	AWA	Real	I	2	1		Specific impulse of RCS system, seconds
SRBWT	AWA	Real	I	4	2		SRB case and propellant weights
OMSUSE	AWA	Real	O	2	1		Amount of OMS fuel consumed during simulation
RCSUSE	AWA	Real	O	2	1		Amount of RCS fuel consumed during simulation
STSRB	AWA	Real	O	30	15		State vector at SRB staging

NOTES	CLASS	TYPE			USE
	AWA	Free	2CH	72CH	I = Input
	Disk	Intg	6CH	Mix	O = Output
		Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-I.- Concluded

PROCESSOR ASENT

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
STMECO	AWA	Real	0	30	15		State vector at MECO
SUMTAB	AWA	Real	0	288	(8,36)		Output summary table

NOTES	CLASS	TYPE			USE
	AWA	Free	2CH	72CH	I = Input
	Disk	Intg	6CH	Mix	O = Output
		Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-II.- INTERFACE TABLE DATA ARRAY DEFINITIONS

PROCESSOR ASENT

Array name	Index location	Default value	Definition
GLOCON	(1) . . . (180)	!!GLCN	GLOBAL constants array, master data base element See table 7.2-III in volume I for content and definitions.
SESCON	(1) . . . (90)	!SESCN	Input/output session constants array. See table 7.2-II In volume I for content and definitions.
PROCON	(1)	0	Output unit on which to write all print. = 0, All print is output at user terminal. If positive N = some system LU number, all print is output on LU = N.
	(2)		Not used.
	(3)	.5	Tolerance value for convergence.
	(5)	2690.	Aerodynamic reference area at launch, ft^2
	(7)	.17	Coefficient of drag after staging, n.d.
	(9)	6.	Time to initiate pitchover, seconds.
	(11)	16.	Time to end pitchover, seconds.
	(13)	1.0	Initial integration step size, seconds.
	(15)	0	Not used.
	(16)	0	Not used.
	(17)	8	Maximum number of iterations to be run.
	(18)	0	Not used.
TARG1	1		Altitude for initial target
	2		Velocity for initial target
	3		Flightpath angle for initial target
TARG2	1		Altitude for second target set - MECO
	2		Velocity for second target set - MECO
	3		Flightpath angle for second target set - MECO

TABLE 4-II.- Continued

PROCESSOR ASENT

Array name	Index location	Default value	Definition
CAXMN	(1,1)	.1050	Coefficient of axial force - DRAG, n.d.
	(2,1)	.1154	
	(3,1)	.1243	
	(4,1)	.1372	
	(5,1)	.1583	
	(6,1)	.2089	
	(7,1)	.2521	
	(8,1)	.2717	
	(9,1)	.2810	
	(10,1)	.2929	
	(11,1)	.2945	
	(12,1)	.2883	
	(13,1)	.2781	
	(14,1)	.2683	
	(15,1)	.2440	
	(16,1)	.2272	
	(17,1)	.2249	
	(18,1)	.2129	
	(19,1)	.2129	
	(1,2)	.0	
	(2,2)	.6	
	(3,2)	.7	
	(4,2)	.8	
	(5,2)	.9	
	(6,2)	1.0	
	(7,2)	1.1	
	(8,2)	1.2	
	(9,2)	1.3	
	(10,2)	1.4	
	(11,2)	1.45	
	(12,2)	1.6	
	(13,2)	1.8	
	(14,2)	2.0	
	(15,2)	2.4	
	(16,2)	3.0	
	(17,2)	4.0	
	(18,2)	5.0	
	(19,2)	100.	

TABLE 4-II.- Continued

PROCESSOR ASENT

Array name	Index location	Default value	Definition
FHBAF	(1,1)	0.	Force component for base axial force; f(H) - (lbf)
	(2,1)	176.	
	(3,1)	277.	
	(4,1)	282.	
	(5,1)	277.	
	(6,1)	100.	
	(7,1)	55.	
	(8,1)	-2.	
	(9,1)	-11.	
	(10,1)	-13.	
	(11,1)	-12.5	
	(12,1)	-10.	
	(13,1)	-6.5	
	(14,1)	-6.5	
	(1,2)	0.	Altitude of base axial force; f(BAF) - feet
	(2,2)	10000.	
	(3,2)	20000.	(independent variable)
	(4,2)	22500.	
	(5,2)	25000.	
	(6,2)	40200.	
	(7,2)	50000.	
	(8,2)	70000.	
	(9,2)	80000.	
	(10,2)	90000.	
	(11,2)	100000.	
	(12,2)	120000.	
	(13,2)	160000.	
	(14,2)	1.0E8	
ISP	(1,1)	261.5	ISP profile for the 1st stage engines and the SRB engines, seconds.
	(2,1)	261.1	
	(3,1)	260.0	
	(4,1)	259.9	
	(5,1)	259.6	
	(6,1)	258.6	
	(7,1)	257.2	
	(8,1)	254.5	
	(9,1)	250.0	
	(10,1)	250.0	
	(1,2)	0.	Time for ISP table (independent variable, seconds)
	(2,2)	20.	
	(3,2)	50.	

TABLE 4-II.- Continued

PROCESSOR ASENT

Array name	Index location	Default value	Definition
ISP (Cont'd)	(4,2)	70.	
	(5,2)	90.	
	(6,2)	110.	
	(7,2)	115.	
	(8,2)	120.	
	(9,2)	123.	
	(10,2)	150.	
THRST	(1,1)	2880.	1st stage engines, SRB thrust profile, 1000 lbf
	(2,1)	3060.	
	(3,1)	3220.	
	(4,1)	3000.	
	(5,1)	2750.	
	(6,1)	2280.	
	(7,1)	2370.	
	(8,1)	2520.	
	(9,1)	2560.	
	(10,1)	2130.	
	(11,1)	1840.	
	(12,1)	800.	
	(13,1)	400.	
	(14,1)	0.	
	(15,1)	0.	
	(16,1)	0.	
	(17,1)	0.	
	(18,1)	0.	
	(19,1)	0.	
	(20,1)	0.	
	(1,2)	0.	Time for 1st stage engines, SRB thrust table (independent variable), seconds
	(2,2)	6.	
	(3,2)	21.	
	(4,2)	23.	
	(5,2)	30.	
	(6,2)	48.	
	(7,2)	60.	
	(8,2)	70.	
	(9,2)	80.	
	(10,2)	100.	
	(11,2)	111.	
	(12.2)	116.	
	(13,2)	121.	
	(14,2)	123.	

TABLE 4-II.- Continued

PROCESSOR ASENT

Array name	Index location	Default value	Definition
THRST (Cont'd)	(15,2)	5000.	
	(16,2)	0.	
	(17,2)	0.	
	(18,2)	0.	
	(19,2)	0.	
	(20,2)	15.0	Number of parameter PAIRS in thrust table
ENGNS	(1,1)	0.	Vacuum thrust of 1st stage engines (SRB - not used), lbf
	(2,1)	0.	Vacuum thrust of 2nd stage engines, lbf
	(3,1)	1410000.	Vacuum thrust of 3rd stage engines (the three main engines), lbf
	(1,2)	0.	Specific impulse of 1st stage engines (SRB - not used) seconds
	(2,2)	455.2	Specific impulse of 2nd stage engines, seconds
	(3,2)	455.3	Specific impulse of 3rd stage engines, seconds
	(1,3)	489094.	1st stage engine thrust factor (vacuum - sea level), lbf
	(2,3)	0.	2nd stage engine thrust factor (vacuum - sea level), lbf
	(3,3)	285000.	3rd stage engine thrust factor (vacuum - sea level), lbf
	(1,4)	0.	1st stage engine cant angle (SRB used) degrees
	(2,4)	0.	2nd stage engine cant angle degrees
	(3,4)	12.	3rd stage engine cant angle degrees
STSRB	(1)		SRB staging state vector
	.		See figure 7.3-2 in volume I for content and definitions.
	.		
	(15)		
STMECO	(1)		MECO state vector
	.		See figure 7.3-2 in volume I for content and definitions.
	.		
	(15)		
SRBWT	(1)		SRB case weight
	(2)		SRB propellant weight
SUMTAB	(1,1)		ERROR - Error condition flag for summary table
	(1,2)		GMTLO - Greenwich mean time lift-off, seconds
	(1,3)		AZL - Launch azimuth
	(1,4)		INCL - Initial inclination
	(1,5)		LATLS - Geocentric latitude of launch site

TABLE 4-II.- Continued

PROCESSOR ASENT

Array name	Index location	Default value	Definition
SUMTAB (Cont'd)	(1,6)		LONGLS - Longitude of launch site
	(1,7)		GLOW - Gross lift-off weight
	(1,8)		PHILS - Position of launch site along $\hat{e}_\phi$
	(1,9)		RLS - Position of launch site along $\hat{e}_r$
	(1,10)		ZLS - Position of launch site along $\hat{e}_z$
	(1,11)		PHIDLS - Angular rate of launch site along $\hat{e}_\phi$
	(1,12)		RLSVEL - Velocity of launch site along $\hat{e}_r$
	(1,13)		ZLSVEL - Velocity of launch site along $\hat{e}_z$
	(1,14)		TSTAG - Time of staging-GET
	(1,15)		LATSTG - Geocentric latitude at staging time
	(1,16)		LONSTG - Longitude at staging
	(1,17)		AZSTG - Azimuth at staging
	(1,18)		RDSTG - Radius to vehicle at staging
	(1,19)		TQMAX - Time of maximum Q
	(1,20)		QMAX - Maximum Q incurred
	(1,21)		TGMAX - Time of maximum g
	(1,22)		GMAX - Maximum g incurred
	(1,23)		TMECO - Time of MECO-GET
	(1,24)		LTMECO - Geocentric latitude at MECO
	(1,25)		LNMECO - Longitude at MECO
	(1,26)		AZMECO - Azimuth at MECO
	(1,27)		INMECO - Inclination at MECO
	(1,28)		HEQ - Altitude above equatorial radius at MECO
	(1,29)		DFLS - Distance from launch site at MECO
	(1,30)		PFA - Powered flight angle computed at MECO
	(1,31)		TIGM - Target descending node
	(1,32)		OMSUSE - Total OMS propellant used during simulation
	(1,33)		RCSUSE - Total RCS propellant used during simulation
	(1,34)		WTMECO - Total inserted weight
	(1,35)		PITCHV - Pitch angle with respect to vertical at end of pitchover phase
	(1,36)		HDANG - Heading angle with respect to launch plane at lift-off

TABLE 4-III.- PROCESSOR DISPLAYS AND DISPLAY PARAMETER DEFINITION TABLE

(a) Detailed ascent profile

PROCESSOR ASENT

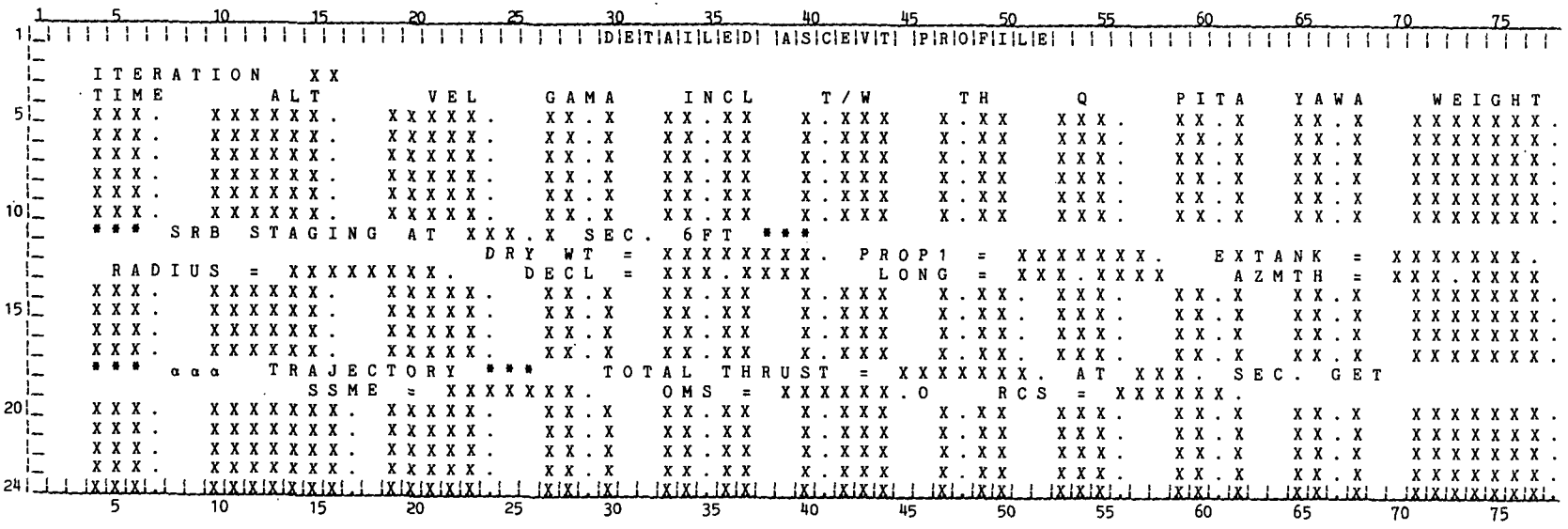


TABLE 4-III.- Continued

(b) Display parameter definition table for the detailed ascent profile display.

PROCESSOR ASENT

DETAILED ASCENT PROFILE	
Display parameter label	Parameter definition
ITERATION	Iteration number
TIME	Elapsed time from lift-off, seconds
ALT	Altitude above local surface, feet
VEL	Inertial velocity of vehicle
GAMA	Flightpath angle
INCL	Calculated value of current inclination
T/W	Thrust-to-weight ratio, n.d.
TH	Throttle setting, percent
Q	Dynamic pressure, lbs/ft^2
PITA	Steering pitch angle in coordinate plane (cylindrical $\hat{e}_r$, $\hat{e}_o$, $\hat{e}_z$, system)
YAWA	Yaw angle in coordinate plane (cylindrical $\hat{e}_r$, $\hat{e}_o$, $\hat{e}_z$ system)
WEIGHT	Total vehicle weight
DRY WT.	SRB case weight
PROP1	First stage propellant weight
EXTANK	Third stage propellant weight
RADIUS	Radius to vehicle
DECL	Declination of state
LONG	Longitude of state
AZMTH	Azimuth of state
α α α	Type of trajectory simulated
TOTAL	Total thrust of all engines, lbf
SSME	Vacuum thrust of main engines, pounds
OMS	Vacuum thrust of OMS system, pounds
RCS	Vacuum thrust of RCS system, pounds

TABLE 4-III.- Continued

(c) Ascent display

PROCESSOR ASENT

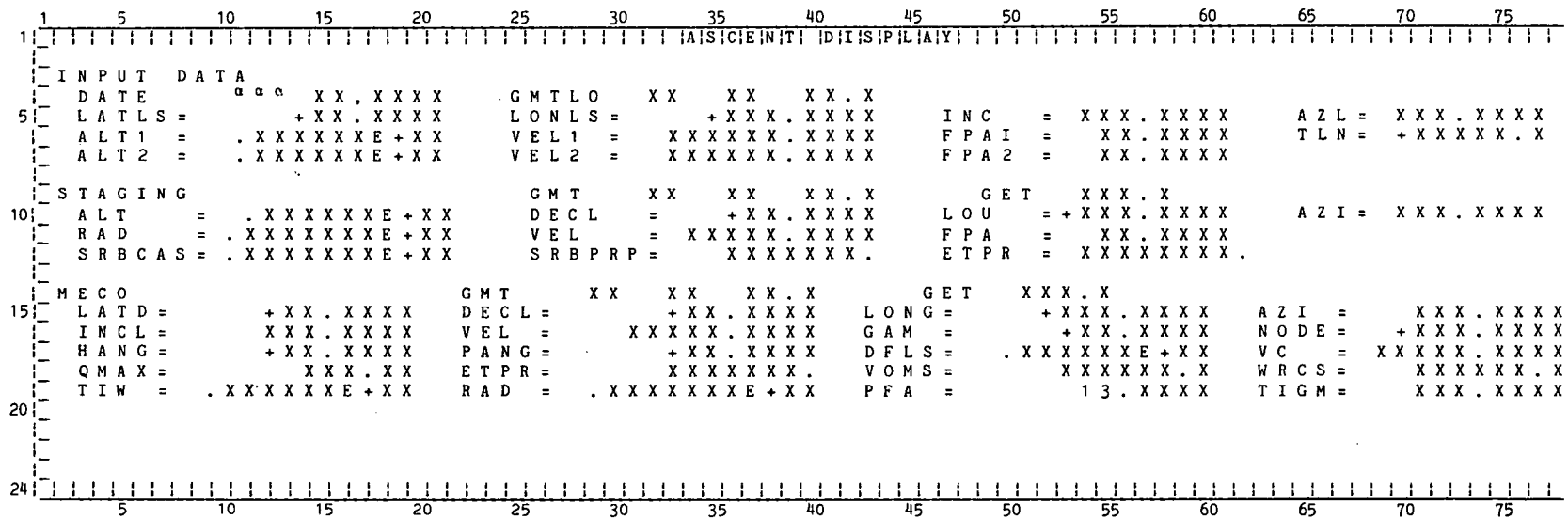


TABLE 4-III.- Continued

(d) Display parameter definition table for the ascent display.

PROCESSOR ASENT

ASCENT DISPLAY	
Display parameter label	Parameter definition
	Input data
GMTLO	Greenwich mean time of lift-off, hr., min., sec.
LATLS	Geodetic latitude of launch site
LONLS	Geographic longitude of launch site
INC	Initial inclination
AZL	Azimuth of target plane at launch site latitude
ALT1	Altitude of target 1
VEL1	Velocity of target 1
FPA1	Flightpath angle of target 1
ALT2	Altitude of target 2
VEL2	Velocity of target 2
FPA2	Flightpath angle of target 2
	Staging
GMT	Greenwich mean time of staging, hr., min., sec.
GET	Ground elapsed time of staging, seconds
ALT	Vehicle altitude at staging
DECL	DECL at staging
LON	Geographic longitude at staging
AZI	Azimuth of state
RAD	Radius to vehicle
VEL	Velocity of staging
FPA	Flightpath angle at staging
SRBCAS	SRB case weight
SRBPRP	SRB propellant weight consumed by staging time
ETPR	External tank propellant weight consumed by staging time
	MECO
GMT	Greenwich mean time of MECO, hr., min., sec.
GET	Ground elapsed time of MECO, seconds
LATD	Geodetic latitude of state
DECL	Declination of state
LONG	Longitude of state
AZI	Azimuth of state
INCL	Calculated value of current inclination

TABLE 4-III.- Concluded

PROCESSOR ASENT

ASCENT DISPLAY (continued)

Display parameter label	Parameter definition
VEL	Velocity of MECO
GAM	Flightpath angle
NODE	Coordinate plane node with respect to Greenwich meridian at MECO
HANG	Heading angle with respect to coordinate plane at lift-off
PANG	Pitch angle with respect to vertical at end of pitchover
DFLS	Distance from launch site at MECO
VC	Characteristic velocity
QMAX	Maximum dynamic pressure incurred
ETPR	Weight of external tank propellant used
WOMS	Weight of OMS propellant used
WRCS	Weight of RCS propellant used
TIW	Total insertion weight of vehicle
RAD	Radius to vehicle
PFA	Powered flight angle
TIGM	Angle measured from launch site to chaser descending node at lift-off

TABLE 4-IV.- PROCESSOR MESSAGE TABLE

PROCESSOR ASENT

MSG no.	Message ID block	Message text block and explanation
1	*ASENT*	*ASNIT* "AA" IS INVALID INPUT FOR XXXXXX Meaning: User input invalid Hollerith response for parameter variable. Severity: Processor will terminate. Action required by user: User must supply valid response in interface table.
2	*ASENT*	*LNSIT* TARGET INCLINATION LESS THAN LAUNCH ISTE LATITUDE Meaning: User input values for target inclination and geocentric latitude do not correlate. The target launch inclination orbit will not pass over the launch site. Severity: The processor will terminate and control will be returned to the FDS Executive. Action required by user: The inputs should be reviewed.
3	*ASENT*	*RKSEG* ALTITUDE LESS THAN 0.0 Meaning: Vehicle altitude is less than 0.0 Severity: The processor will terminate. Action required by user: The launch trajectory has impacted the Earth; thus the input should be reviewed for consistency.
4	*ASENT*	*RKSEG* ALTITUDE GREATER THAN 1.6*TARGET ALTITUDE Meaning: Vehicle altitude has exceeded target altitude by more than the allowable range. The launch iteration has failed to converge to MECO target altitude. Severity: Processor will terminate. Action eequired by user: The inputs should be reviewed for consistency.

TABLE 4-V.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR ASENT

Processor name	Processor abstract prompt (maximum 256 characters)
ASENT	The ASENT processor simulates a Shuttle ascent profile from launch to MECO. The model integrates the profile through four guidance phases and can simulate both nominal and AOA targeting.
Parameter keyword name	**Parameter definition prompt (maximum 256 characters)**
GLOCON	GLOBAL constants input array; normally defaulted to !!GLCN
SESCON	Session constants input array; normally defaulted to !!SESCN
PROCON	ASENT constants and tolerances; input array normally defaulted
GLOW	Input vehicle gross lift-off weight
PITCHV	Input pitch angle with respect to vertical at end of pitchover guidance phase
HDANG	Input heading angle with respect to coordinate plane at lift-off
AZDIR	Launch azimuth specification: ="NO"; northerly launch ="SO"; southerly launch
TARG1	Input altitude, velocity, and flightpath angle for initial set of MECO targets
TARG2	Input altitude, velocity, and flightpath angle for second set of MECO targets
TTGA	Input guidance time, launch to MECO; used from lift-off to time TAOA

TABLE 4-V.- Continued

PROCESSOR ASENT

Processor name	Processor abstract prompt (maximum 256 characters)
Parameter keyword name	**Parameter definition prompt (maximum 256 characters)**
AOANOM	Input code to select type of trajectory to be flown after TAOA ="AOA"; AOA trajectory, one main engine is failed ="NOM"; nominal trajectory
TAOA	Time at which targets are reset
THROT	Initial percent of throttle setting for main engine thrusting
THROTA	Percent of throttle setting for engine thrusting at time TMRTLS
OMSFG	Flag for OMS engine ="ON"; OMS engine on after TAOA, if AOA targeting ="OF"; OMS engine OFF
RCSFG	Input flag for RCS engines ="ON"; RCS engines ON after TAOA, if AOA targeting ="OF"; RCS engines OFF
OMSAV	Amount of OMS propellant available for ascent
RCSAV	Amount of RCS propellant available for ascent

TABLE 4-V.- Continued

PROCESSOR ASENT

Processor name	Processor abstract prompt (maximum 256 characters)

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
ATMOS	Atmosphere model flag ="ST"; Standard '62 ="PA"; Patrick reference '63 ="VA"; Vandenburg '71
TTG	Time, lift-off to MECO
MAXQ	Maximum dynamic pressure desired by throttling
TLNFL	Time of launch flag ="TI"; input TIGM ="TL"; input TLN
TLN	Time of launch WRT analytical in-plane launch time
TIGM	Target descending node WRT launch site
DSPLY	Flag for display print ="NO"; do not display detailed print ="YE"; display detailed print
CAXMN	Coefficient of axial force and corresponding Mach number (DRAG coefficients)
FHBAF	Force and corresponding altitude of BAF

TABLE 4-V.- Continued

PROCESSOR ASENT

Processor name	Processor abstract prompt (maximum 256 characters)
Parameter keyword name	**Parameter definition prompt (maximum 256 characters)**
ISP	ISP and corresponding time for the SRB
THRST	SRB thrust profile
TSTAG	Time for SRB staging
ENGNS	Vacuum thrust, specific impulse, thrust factors, and cant angles for the SRB and main engines
OMSTHR	Vacuum thrust of OMS system
OMSISP	Specific impulse of OMS system
ARSRB	Area for the DRAG computation for the Shuttle
CDSRB	Input coefficient of DRAG for the SRB at staging
ARORB	Area for the DRAG computation for the Shuttle
CDORB	Coefficient of DRAG for the Shuttle state vector at MECO
RCSTHR	Vacuum thrust of RCS system

TABLE 4-V.- Concluded

PROCESSOR ASENT

Processor name	Processor abstract prompt (maximum 256 characters)
Parameter keyword name	**Parameter definition prompt (maximum 256 characters)**
RCSISP	Specific impulse of RCS system
SRBWT	SRB case weight and propellant weight
OMSUSE	Amount of OMS fuel consumed during simulation
RCSUSE	Amount of RCS fuel consumed during simulation
STSRB	Output state vector at SRB staging
STMECO	Output state vector at MECO
SUMTAB	Output summary table

5.0 PROCESSOR ROUTINES

The only available routine documentation is contained on the comment cards in the software listing.

DATA ASSIGNMENT PROCESSOR (ASSGN)

1.0 PURPOSE

The ASSGN utility processor computes values and stores them in an existing data element in the AWA. ASSGN supports extended FORTRAN type mixed mode expressions and functions and allows repetitive evaluations in order to compute and store multiple values.

2.0 FUNCTIONAL DESCRIPTION

Input for ASSGN is a symbolic string that is acquired via the interface table parameter keyword, EXP. The string is an expression, specifically a FORTRAN type scalar computation written in algorithmic format. The syntax of this format is discussed in the Processor Input/Output in section 4.0 of this processor.

ASSGN processing evaluates this string and stores the resulting value into the object data element. All errors detected by ASSGN are reported to the user with an appropriate message and a printed image of the symbolic string and, when possible, an indication of the part of the string in error. All errors are fatal; i.e., ASSGN processing stops but storage into the object may have occurred. See Assumptions and Limitations in section 3.0 of this processor for an example of this case.

The operations in the expression are processed according to a defined priority. The relative operational priorities are as shown in table 2-I. Syntax checking for the expression is done based on the sequence of characters and the validity of groupings that are defined in the expression.

If any element, including the object, is not defined in the AWA, an AWA access failure message is given and the processor is terminated.

During the evaluation, conversion progresses from integer to real to double precision with no regression until forced by the type requirements of a function argument, subscript, or by storage into the object. Table 2-II indicates type conversion rules for storing results in the object.

The functions that are supported by ASSGN are listed in table 2-III along with the required number of arguments and the input and output types.

3.0 ASSUMPTIONS AND LIMITATIONS

Listed as follows are the assumptions and limitations associated with the ASSGN processor.

a. All data elements must be allocated in the AWA previous to ASSGN processing.

b. Consecutive operators are not valid. An expression such as:

$$X = A_{*} -B$$

should be expressed as:

$$X = A_{*} (-B)$$

c. No more than two subscripts are supported for a data element.

d. Elements in a subscript list are truncated to integer, if necessary, after they are evaluated.

e. Elements in a function list are converted to the type required by the function, if necessary.

f. If an AWA access error occurs when storing data, the object could possibly have been modified. For example:

```
'A(I)  = 0;  I=1,9'
```

where A is dimensioned as an 8-element array would result in A being set to zero and an access failure on the ninth iteration.

g. Undefined mathematical operations such as division by zero result in a processor termination.

h. Iteration of the assignment through the specified range limits is accomplished in a nested fashion; i.e., the first limit will be cycled for each value of the second, etc. Iteration progresses from the first value to the second by incrementing or decrementing the index as appropriate.

i. A maximum of four ranges may be specified for nested iterative evaluation. Each index must be uniquely named.

j. Except in the case of a free-type object, assignment is made in units according to the object type. Assigning character string data to an object of different character type results in a blank fill or truncation, as applicable. If the object is a free data element, the assignment is made based on the attributes of the expression's result.

k. ASSGN does not support the symbolic string data type for either the object or the expression.

4.0 PROCESSOR INPUT/OUTPUT

Listed below are the inputs required and outputs for the ASSGN utility processor.

a. Processor interface table - The interface table for ASSGN is defined in table 4-I. The ASSGN interface table contains a symbolic string that is

identified by the parameter keyword EXP. (Symbolic strings are enclosed in single quotes.) The contents of the string are formatted as follows:

OBJECT = EXPRESSION [;RANGE]

The object of the assignment is an existing AWA data element that can be optionally subscripted. Any valid noncharacter expression can be used to specify the subscript index. The expression is a FORTRAN type scalar expression composed of operands and binary and unary operators. The operands can be numerical constants, numerical data elements (optionally subscripted), range indexes, function references, and parenthetical groupings. Nesting of parenthetical groupings, function references, and subscripts are supported. The data type of the operand may be any of the supported types except symbolic strings. The conversion that results when the data types of the object and expression are different is given in table II. The range of the assignment is an optional specification providing repetition of evaluation and storage.

b. Interface table data array definitions - None.

c. Interface table data file definitions - None.

d. Processor solicited (prompted) inputs - None.

e. Processor displays and display parameter definition table - None.

f. Processor message table - The messages issued by the processor are found in table 4-II.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-III.

TABLE 2-I.- OPERATIONAL PRIORITIES

Symbol or form	Relative priority	Operation
(	4	Parenthetical grouping
function [	4	Mathematical functions (table 2-III)
data element (	4	Subscripting
**	3	Exponentiation
*	2	Multiplication
/	2	Division
+	1	Addition and unary plus
-	1	Subtraction and unary minus

TABLE 2-II.- EXPRESSION-OBJECT CONVERSION

Object \ Expression	Numeric	Character	Free
Numeric	Converted as necessary	Error	No conversion (assumed correct)
Character	Error	Truncated/ padded as necessary	No conversion (assumed correct)
Free	No conversion (assumed correct)	No conversion (assumed correct)	No conversion; one word stored

TABLE 2-III.- MATHEMATICAL FUNCTIONS

Function name	Result type	Argument(s) type	Number arguments	Operation
ABS	R	R	1	Absolute value
AINT	R	R	1	Truncate fraction
ALOG	R	R	1	Natural logarithm
ALOGT	R	R	1	Base ten logarithm
AMOD	R	R	2	Modulo
ATAN	R	R	1	Inverse tangent
ATAN2	R	R	2	Inverse tangent
COS	R	R	1	Cosine
DABS	D	D	1	Absolute value
DATAN	D	D	1	Inverse tangent
DATN2	D	D	2	Inverse tangent
DBLE	D	R	1	Convert to double
DCOS	D	D	1	Cosine
DDINT	D	D	1	Truncate fraction
DEXP	D	D	1	e^x
DLOG	D	D	1	Natural logarithm
DLOGT	D	D	1	Base ten logarithm
DMOD	D	D	2	Modulo
DSIGN	D	D	1	Apply sign
DSIN	D	D	1	Sine
DSQRT	D	D	1	Square root
DTAN	D	D	1	Tangent
DTANH	D	D	1	Hyperbolic tangent

TABLE 2-III.- Concluded

Function name	Result type	Argument(s) type	Number arguments	Operation
EXP	R	R	1	e^x
FLOAT	R	I	1	Convert to real
IABS	I	I	1	Absolute value
IDINT	I	D	1	Convert to integer
IFIX	I	R	1	Convert to integer
ISIGN	I	I	2	Apply sign
MOD	I	I	2	Modulo
SIGN	R	R	2	Apply sign
SIN	R	R	1	Sine
SNGL	R	D	1	Convert to real
SQRT	R	R	1	Square root
TAN	R	R	1	Tangent
TANH	R	R	1	Hyperbolic tangent

TABLE 4-I.- INTERFACE TABLE DEFINITIONS

PROCESSOR ASSGN

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
EXP	AWA	SYMB	I	247	--	--	Symbolic string. The valid syntax for EXP is given in the Processor Input/Output section under Interface table.

NOTES	CLASS	TYPE			USE
	AWA	Free	2CH	72CH	I = Input
	Disk	Intg	6CH	Mix	O = Output
		Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-II.- PROCESSOR MESSAGE TABLE

PROCESSOR ASSGN

MSG no.	Message ID block	Message text block and explanation
1	*AS01*	EXPRESSION'S OBJECT HAS BEEN SPECIFIED AS A RANGE INDEX Meaning: The data element to which the expression is assigned appears in a range specification. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
2	*AS02*	NAME HAS ALREADY BEEN SPECIFIED AS A RANGE INDEX Meaning: An index appears in more than one range specification. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
3	*AS03*	RANGE INDEX HAS BEEN SUBSCRIPTED ELSEWHERE IN EXPRESSION Meaning: A data element name that is subscripted in the expression is later specified as a range index Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
4	*AS04*	INVALID CHARACTER INPUT Meaning: The indicated character is not a valid operand or operator. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
5	*AS05*	FUNCTION NOT SUPPORTED BY THIS PROCESSOR Meaning: The function indicated is not supported by ASSGN; the functions that are supported are listed in table III. Severity: Fatal; ASSGN processing stops; no action taken Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
6	*AS06*	INVALID SEQUENCE OF CHARACTERS Meaning: The character/token that is indicated cannot validly follow the preceding token. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.

TABLE 4-II.- Continued

PROCESSOR ASSGN

MSG no.	Message ID block	Message text block and explanation
7	*AS07*	INVALID SYNTAX TO LEFT OF = Meaning: The object of the expression is not well defined. It must be of the form: DATA ELEMENT NAME [(SUB1[,]SUB2)]. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
8	*AS08*	PARENTHESES OR BRACKETS DO NOT MATCH PROPERLY Meaning: Groupings specified by parentheses or brackets are not closed properly. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
9	*AS09*	FUNCTION LIST INCOMPLETE Meaning: More arguments are required by the function indicated than were input. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
10	*AS10*	INVALID RANGE FORMAT. MUST BE OF FORM'; NAME = INTEGER, INTEGER' Meaning: Range specification does not follow a given format. The place at which the error occurred is indicated. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
11	*AS11*	INVALID COMMA OR TOO MANY COMMAS IN LIST Meaning: The indicated comma either is used outside a subscript or function list or specifies too many elements in a subscript or function list. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.

TABLE 4-II.- Continued

PROCESSOR ASSGN

MSG no.	Message ID block	Message text block and explanation
12	*AS12*	NULL SYMBOLIC STRING Meaning: '' was entered for the parameter EXP. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
13	*AS13*	MORE THAN 4 RANGE SPECIFICATIONS INPUT Meaning: The maximum number of range specifications that can be input is four. The indicated specification is in excess of this limit. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected and ASSGN executed again.
14	*AS14*	FREE OR CHARACTER DATA ELEMENT FOUND IN AN EXPRESSION Meaning: Free or character data cannot be involved in any operation except a direct replacement. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Correct expression and execute ASSGN again.
15	*AS15*	NUMERICAL DATA ELEMENT CANNOT BE SET EQUAL TO CHARACTER DATA Meaning: A fixed-type data element can only be assigned free or fixed-type data. Character data can only be assigned to a free or character data element. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Correct expression and execute ASSGN again.
16	*AS16*	CHARACTER DATA ELEMENT CANNOT BE SET EQUAL TO NUMERICAL DATA Meaning: A character data element can only be assigned character or free data. Fixed-type data can only be assigned to a free or fixed-type data element. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Correct expression and execute ASSGN again.

TABLE 4-II.- Concluded

PROCESSOR ASSGN

MSG no.	Message ID block	Message text block and explanation
17	*AS17*	NOT A VALID ASSIGNMENT ('=' NOT INPUT) Meaning: An equal sign (=) was not found in the expression. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected using the interface table editor and ASSGN must be executed again.
18	*AS18*	OVERFLOW OR UNDERFLOW DETECTED Meaning: Overflow or underflow was detected during an arithmetic operation, function, or implicit conversion. Severity: Fatal; ASSGN processing stops; no action taken. Action required by user: Syntax of the expression must be corrected or the values of operands in the expression should be examined for either zero or near-maximum value and ASSGN executed again.

TABLE 4-III.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR ASSGN

Processor name	Processor abstract prompt (maximum 256 characters)
ASSGN	Utility processor used to evaluate a FORTRAN-type scalar computation and assign its value to a specified AWA data element name. Multiple assignments can be made using iterative control.

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
EXP	Symbolic string of the form: 'OBJECT[(SUB1[,SUB2])] = EXPRESSION[;RANGE1[;RANGE2...]]' where: OBJECT is an AWA data element name EXPRESSION is a FORTRAN-type computation RANGE is 'NAME = INTEGER, INTEGER' for iterative control (four ranges maximum)

5.0 PROCESSOR ROUTINES

The only available routine documentation is contained on the comment cards in the software listing. Additional material may be found in JSC IN 77-FM-18, vol. IV, rev. 2 dated April 1978, and in JSC IN 77-FM-18, vol. IX (to be published).

ATTITUDE TABLE MAINTENANCE PROCESSOR (ATM)

1.0 PURPOSE

The ATM processor provides the capability to initialize, modify, or list a matrix locker table, an attitude timeline phase table (ATL), a celestial target table (CTT), or an instrument definition table (IDT).

2.0 FUNCTIONAL DESCRIPTION

The type of table to be initialized or modified is specified by the TYPE parameter. If the type of table specified is a matrix locker (LO), then the user may initialize a new table, list an existing table, add an entry into an existing table, replace an entry in an existing table, or delete a specified entry from an existing table.

The initialize function must be used to enter the first entry into the table. The output data element must have been defined prior to the execution of ATM. The data element should be defined 25 words larger than 25 times the maximum number of matrices to be stored in the table.

The list function prints all entries in the input matrix locker data element.

The add, replace, and delete functions modify the input matrix locker data element and place the results in the output matrix locker data element. Both the input and output data elements must have been defined prior to the execution of ATM. The input and output locker data elements can be the same. However, care should be exercised in order to ensure that desired data is not accidentally destroyed.

The add function places the input matrix element (MATRIX) in the entry as specified by the parameter ENTRY. All subsequent entries will have an entry number one greater than its previous value.

The replace function places the input matrix element (MATRIX) in the entry as specified by the parameter ENTRY.

The delete function deletes the indicated entry. All subsequent entries have an entry number one less than the previous value.

Initialization or modification of an ATL is indicated by setting the TYPE parameter to AT. The ATL initialize, add, replace, and delete functions are functionally the same as the matrix locker functions. The ATL data is contained in a DRDE instead of an AWA data element. All of the parameters necessary to build an ATL entry are contained in the interface table.

3.0 ASSUMPTIONS AND LIMITATIONS

The initialize function must be the first operation on a new matrix locker data element, or an ATL.

The input and output matrix locker data elements must be defined before the execution of ATM.

The input matrix data element must be in the standard format as output by the MAST processor.

The IDT and CTT functions are not supported for FDS-1.

4.0 PROCESSOR INPUT/OUTPUT

a. Processor interface table - The definition of the ATM processor interface parameters is provided in table 4-I. The TYPE parameter indicates the type of table to be initialized, listed, or modified. The FUNCT parameter indicates what function is to be performed on the table. Table 4-II contains a summary of the required inputs and output for each type and function. The default value of 20 blocks for the ATL in the PROCON array provides the capability of storing 50 entries in the ATL.

b. Interface table data array definitions - The definitions of the input/output data arrays appearing in the ATM interface table is provided in table 4-III.

c. Interface table data file definitions - The format and definitions of the input/output ATL is provided in table 4-IV.

d. Processor solicited (prompted) inputs - None.

e. Processor displays and display parameter definition table - The format and content of the ATM displays and the definition of the parameters of the displays are provided in table 4-V(a) through 4-V(f).

f. Processor message table - The ATM processor error messages are provided in table 4-VI.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-VII.

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR ATM

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
TYPE	AWA	2CH	I	1	1		Type of table to be initialized, modified, or listed: "LO" = matrix locker "AT" = attitude timeline phase table
FUNCT	AWA	2CH	I	1	1		Function to be performed: "IN" = initialize table "AD" = add entry "RE" = replace entry "DE" = delete entry "LI" = list table
ENTRY	AWA	Intg	I	1	1		Entry number to be added, replaced, or deleted.
MATRIX	AWA	Free	I	25	25		Input matrix used to add or replace a matrix in the locker
INLOC	AWA	Free	I	25	25		Input matrix locker column
OUTLOC	AWA	Free	O	25	25		Output matrix locker column
ATLIN	Disk	Free	O	25	25		Input ATL
GET	AWA	Real	I	6	3		Time relative to reference time (hrs, min, sec) of ATL entry

NOTES

CLASS
AWA
Disk

TYPE
Free 2CH 72CH
Intg 6CH Mix
Real 18CH Symb
Dubl 36CH

USE
I = Input
O = Output
I/O = Input/Output

TABLE 4-I.- Continued

PROCESSOR ATM

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
ATLATT	AWA	Real	I	6	3		ATL roll, pitch, yaw attitude angles
SCMAT	AWA	6CH	I	3	3		Name of source RELMAT on REFSMMAT matrix
COMENT	AWA	18CH	I	9	9		Comments (18 characters) used to identify ATL entry
HMODE	AWA	6CH	I	3	3		Attitude hold mode "LV" = Local vertical-local horizontal "SI" = Solar inertial "IH" = Inertial hold "ROTR" = Rotation mode
AXIS	AWA	Real	I	4	2		Vehicle pitch and yaw angles from the vehicle X-axis used to define the Eigen axis for the ROTR attitude hold mode
RATE	AWA	Real	I	2	1		Vehicle attitude rate for the ROTR attitude hold mode
ATLOUT	Disk	Free	O				Name of output ATL
IMUBOD	AWA	Real	I	18	9		Elements (by column) of the IMU to body matrix
GLOCON	AWA	Free	I	180	180	!!GLCN	Global contants array

NOTES	CLASS	TYPE			USE
	AWA	Free	2CH	72CH	I = Input
	Disk	Intg	6CH	Mix	O = Output
		Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-I.- Concluded

PROCESSOR ATM

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
SESCON	AWA	Free	I	180	180	!SESCN	Session constants array
PROCON	AWA	Free	I	4	4		Processor constants array
INSIN	AWA	Free	I	250	(25,10)	DUMMY	Input instrument definition table
INSOUT	AWA	Free	O	250	(25,10)	DUMMY	Output instrument definition table
CTTIN	AWA	Real	I	1200	(3,200)	DUMMY	Input celestial target table
CTTOUT	AWA	Real	O	1200	(3,200)	DUMMY	Output celestial target table

NOTES

CLASS
AWA
Disk

TYPE
Free 2CH 72CH
Intg 6CH Mix
Real 18CH Symb
Dubl 36CH

USE
I = Input
O = Output
I/O = Input/Output

TABLE 4-II.- INPUT/OUTPUT SUMMARY

Type	LO	LO	LO	LO	LO	AT	AT	AT	AT	AT
Function	IN	AD	RE	DE	LI	IN	AD	RE	DE	LI
ENTRY		X	X	X			X	X	X	
MATRIX	X	X	X							
INLOC		X	X	X	X	X	X	X		
OUTLOC	X	X	X	X						
ATLIN							X	X	X	X
GET						X	X	X		
ATLATT						X	X	X		
SCMAT						X	X	X		
COMENT						X	X	X		
HMODE						X	X	X		
AXIS						X	X	X		
RATE						X	X	X		
ATLOUT						X	X	X		
IMUBOD						X	X	X		

TABLE 4-III.- INTERFACE TABLE DATA ARRAY DEFINITIONS

PROCESSOR ATM

Array name	Index location	Default value	Definition
MATRIX	1-18 19-21 22-24 25		Elements (by column) of input matrix. Name of input matrix. Not used. Matrix type code ("RL", "RF", or "LB").
INLOC	1-18 19-21 22-24 25		Elements (by column) of input matrix locker column. Name of input matrix locker column. Not used. Matrix type code ("RL", "RF", or "LB").
OUTLOC	1-18 19-21 22-24 25		Elements (by column) of output matrix locker column. Name of output matrix locker column. Not used. Matrix type code ("RL", "RF", or "LB").
GET	1 3 5		Time relative to reference time, hours. Time relative to reference time, hours. Time relative to reference time, hours.
ATLATT	1 3 5		ATL roll attitude. ATL pitch attitude. ATL yaw attitude.
AXIS	1 3		Vehicle pitch angle of eigenaxis. Vehicle yaw angle of eigenaxis.
IMUBOD	1-18		Elements (by column) of IMU-to-body transformation matrix
GLOCON	1-180		Global constants array
SESCON	1-90		Session constants array

TABLE 4-III.- Concluded

PROCESSOR ATM

Array name	Index location	Default value	Definition
PROCON	1	0	Delay print flag
	2	0	Output logical unit 0 = same as input unit Positive number = unit number of output
	3	20	Number of blocks for ATL
	4	20	Cartridge reference number for data files

TABLE 4-IV.- INTERFACE TABLE DATA FILE DEFINITIONS

PROCESSOR ATM

DRDE DATA FILE ATLIN/ATLOUT

Record number	Integer word allocations	Content and definition
1	1-3	Processor creating file.
	4-6	Interface table variable creating file.
	7-9	Processor last changing file.
	10-12	Interface table variable last changing file.
2-N	1-2	Greenwich mean time of entry
	3-20	Elements (by column) of transformation matrix from M50 to the body axes for IH and ROTR modes, or from LVLH or SI to the body axes for LVLH and SI modes.
	21-26	Roll, pitch, yaw attitude angles.
	27-29	Source matrix name (6 characters).
	30-37	Comments used to identify entry (18 characters).
	38-40	Attitude hold mode (6 characters).
	41-44	Vehicle pitch and yaw angles from the vehicle X-axis to define the eigenaxis.
	45-46	Eigenaxis rate.
	47-50	Not used.

TABLE 4-V.- PROCESSOR DISPLAYS AND DISPLAY PARAMETER DEFINITIONS

(a) Matrix locker aaaaaa

PROCESSOR ATM

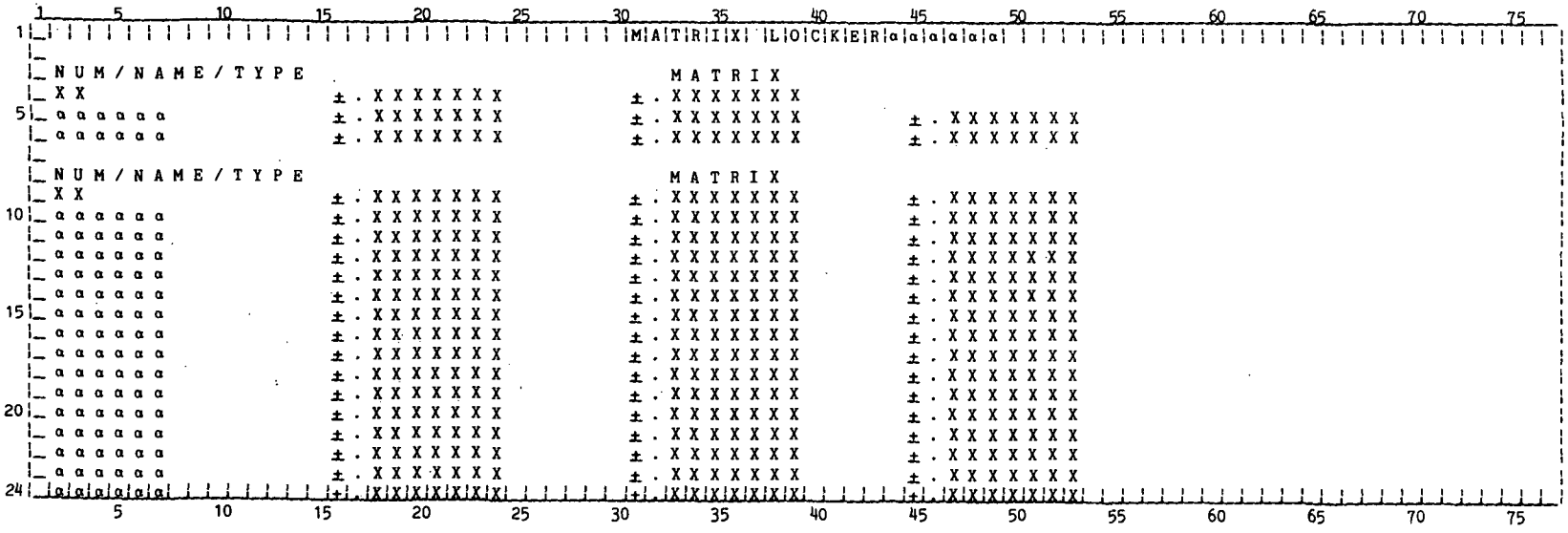


TABLE 4-V.- Continued

(b) Display parameter definition table for the matrix locker display

PROCESSOR ATM

MATRIX LOCKER aaaaaa	
Display parameter label	Parameter definition
NUM	Entry number
NAME	Matrix name
TYPE	Matrix type
MATRIX	Elements of matrix

TABLE 4-V.- Continued

(c) ATTITUDE TIMELINE aaaa

PROCESSOR ATM

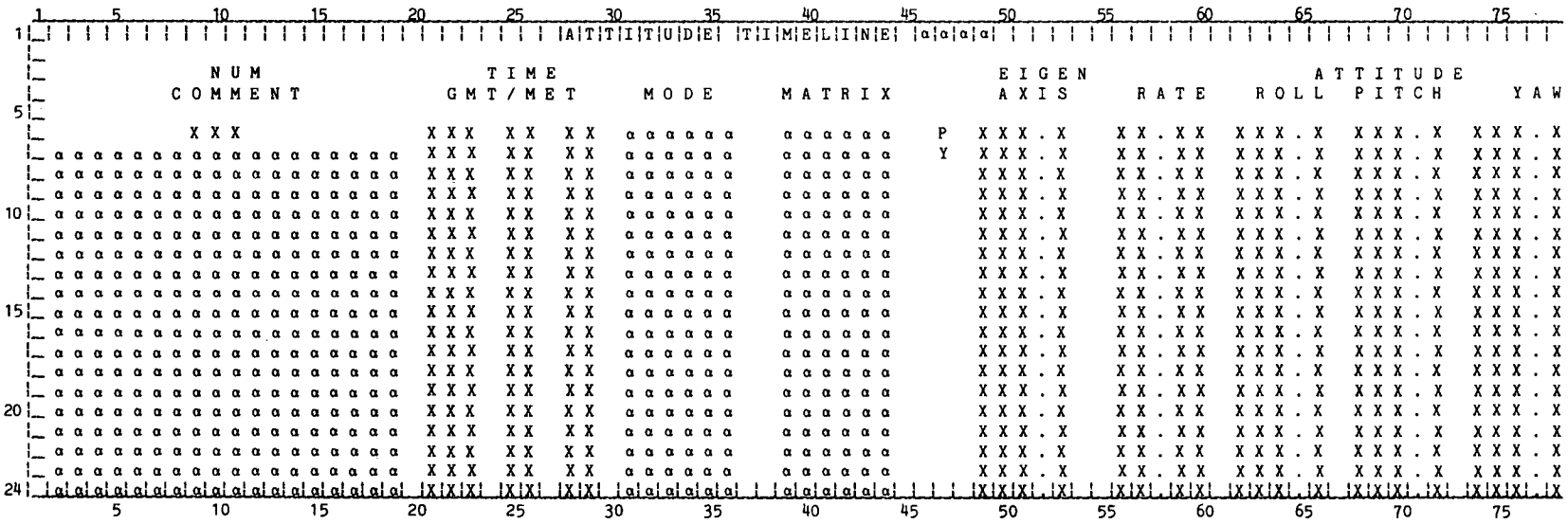


TABLE 4-V.- Continued

(d) Display parameter definition table for the attitude timeline/aaaa display

PROCESSOR ATM

ATTITUDE TIMELINE aaaa	
Display parameter label	Parameter definition
NUM	Entry number
COMMENT	Characters used to identify entry
GMT/MET	Greenwich mean time/mission elapsed time of entry
MODE	Attitude hold mode
MATRIX	Matrix name
EIGEN-Axis	Vehicle pitch, yaw angles used to define eigenaxis for ROTR mode
RATE	Vehicle rate for ROTR mode
ROLL PITCH YAW	Vehicle attitude roll, pitch, yaw with respect to attitude reference

TABLE 4-V.- Continued

(e) ATL matrices.

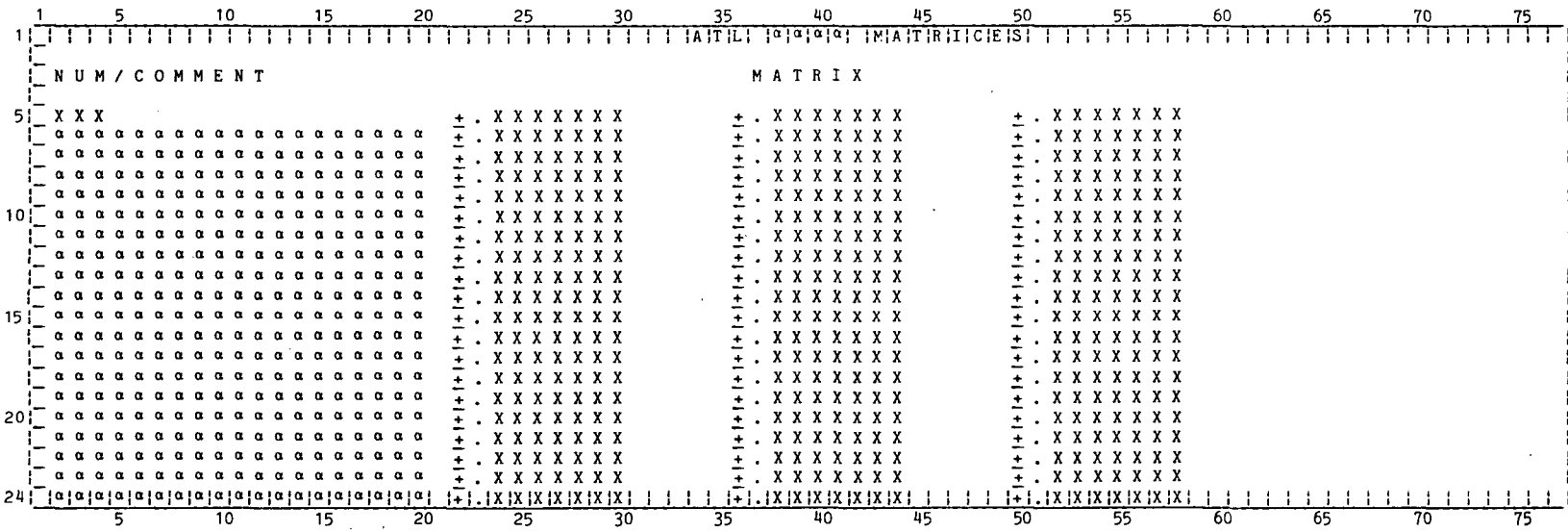


TABLE 4-V.- Concluded

(f) Display parameter definition table for the ALT matrices

PROCESSOR ATM

ALT aaaa MATRICES	
Display parameter label	Parameter definition
NUM	Entry number
COMMENT	Characters used to identify entry
MATRIX	Elements of ATL matrix

TABLE 4-VI.- PROCESSOR MESSAGE TABLE

PROCESSOR ATM

MSG no.	Message ID block	Message text block and explanation
1	*ATM*	INDICATED ENTRY NUMBER = XXX OUTSIDE RANGE OF TABLE Meaning: Attempt was made to add, delete, or replace an entry which was outside the matrix locker or ATL table range. Severity: Processor termination with no change to table. Action required by user: Rerun processor with corrected entry number.
2	*ATM*	ILLEGAL TYPE OR FUNCTION CODE = aa Meaning: Illegal type or function code specified. Severity: Processor termination with no change to table. Action required by user: Rerun processor with corrected function code.
3	*ATM*	ATL FILE CREATE ERROR = XXX NAME = aaaaaa Meaning: Error occurred when attempt was made to initialize ATL. Severity: Processor terminates. Action required by user: Consult SDB representative.
4	*ATM*	OPEN FILE ERROR = XXX NAME = aaaaaa Meaning: Error occurred when attempt was made to open ATL. Severity: Processor terminates. Action required by user: Check ATL name and rerun processor.
5	*ATM*	READ ATL FILE ERROR = XXX Meaning: Error occurred when attempt was made to read ATL. Severity: Processor terminates Action required by user: Consult SDB representative
6	*ATM*	WRITE ATL FILE ERROR = XXX Meaning: Error occurred when attempt was made to write ATL. Severity: Processor terminates. Action required by user: Consult SDB representative.

TABLE 4-VI.- Concluded

PROCESSOR ATM

MSG no.	Message ID block	Message text block and explanation
7	*ATM*	ILLEGAL ATTITUDE HOLD MODE = Meaning: Illegal attitude hold made specified. Severity: Processor terminates with no change to table. Action required by user: Rerun processor with corrected attitude hold mode.
8	*ATM*	SOURCE MATRIX = αααααα NOT FOUND Meaning: Specified source matrix was not found in the matrix locker. Severity: Processor terminates with no change to table. Action required by user : Rerun processor with corrected source matrix name or input locker.
9	*ATM*	ILLEGAL SOURCE MATRIX CODE = αα Meaning: Matrix entry in locker contains an illegal code. Severity: Processor terminates with no change to table. Action required by user: Matrix should be deleted from the locker. Rerun processor with a valid matrix locker.

TABLE 4-VII.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR ATM

Processor name	Processor abstract prompt (maximum 256 characters)
ATM	The Attitude Table Maintenance Processor provides the capability to initialize, modify, or list a matrix locker data element of an attitude time line (ATL) DRDE.

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
	The processor extended prompts for each interface table parameter keyword are the same as the definitions provided in table 4-I, with the following exceptions.
PROCON	Processor constants (1) Debug print flag 0 = no print 1 = debug print (2) Output logical unit 0 = output unit is the same as the input unit >0 = logical unit of output device (3) Number of blocks for ATL DRDE (4) Cartridge reference number for data files

5.0 PROCESSOR ROUTINES

The only available routine documentation is contained on the comment cards in the software listing.

BASETIME INITIALIZATION PROCESSOR (BASTM)

1.0 PURPOSE

The BASTM utility processor provides the FDS user with the capability to establish the basetime (epoch) for a given session and thereby initialize the session constants array !SESCN (which is a required input for most of the FDS-1 functional processors). It also provides the optional capability to initialize solar and lunar coefficient arrays that are required by processors using analytic solar and lunar ephemeris models.

2.0 FUNCTIONAL DESCRIPTION

All user communication with the BASTM processor is through its interface table. The input parameters are GLOCON, YEAR, MONTH, DAY, HOURS, MINS, SECS, and EPHEM. GLOCON is the input parameter through which the processor receives all of the global constants data that it requires. This input is defaulted to the master data base element !!GLCN, and under normal circumstances the user need not be concerned with it. (For documentation of the contents of !!GLCN see table 7.2-III of volume I.)

Through the input parameters YEAR, MONTH, DAY, HOURS, MIN, and SECS, the user specifies the desired base date and lift-off time for the simulation. The inputs, YEAR, MONTH, and DAY, define the reference base date (or epoch) for the simulation, and the BASTM processor uses this time (i.e., 0 hours, 0 minutes, 0 seconds Greenwich mean time (GMT) on the specified YEAR, MONTH, DAY) to establish the orientation in space of the several inertial reference axis systems supported by FDS. Specifically, the processor uses this input to compute the following data:

a. The right ascension of the Greenwich meridian

b. The transformation matrix from the mean-of-1950 to mean-of-base date reference system (precession matrix)

c. The transformation matrix from the mean-of-base date to true-of-base date reference system (nutation matrix), and

d. The Julian ephemeris date of base date

These quantities are output through the interface table parameter SESCON that is defaulted to the specially named AWA data element !SESCN. The name !SESCN is special because it is referenced as an input source in the default interface tables for most FDS functional processors. Under normal circumstances, the user need not be concerned with this data linkage and should not try to change it. (For a definition of the contents of the !SESCN array see table 7.2-II of volume I.)

The inputs HOURS, MINS, and SECS are used to define the lift-off time for the simulation (i.e., the specified HOURS, MINS, SECS are taken to be the GMT of

lift-off, where the zero (0) GMT time reference is defined by 0h0m0s GMT of the specified base date). The lift-off time is used as the zero-time reference for all ground elapsed time (GET) time tags in the simulation.

In addition to the quantities that it directly computes, the BASTM processor also initializes the other locations of the SESCON output array by transfer of appropriate values from the input array GLOCON. Specifically, it transfers the following data items:

a. Mnemonics associated with the default set of session input/output units

b. Conversion factors for the default set of session input/output units

It also initializes the "error/print-trace" flag and the "session units" flag locations in the SESCON output array.

The input parameter EPHEM is an alphanumeric flag that is used by the processor to determine when to compute the output solar and/or lunar coefficient arrays SCOEF AND LCOEF. The valid settings for the EPHEM input flag and their meanings are shown in the definition block of the BASTM default interface table (table I). When solar and/or lunar coefficients are computed, they are output through the interface table parameters SCOEF and LCOEF. These output arrays are normally defaulted to the specially named AWA data elements !SCOEF and LOCEF. These names are special because they are referenced as input sources in the default interface tables of other FDS functional processors that use the analytic solar and/or lunar ephemeris routines (the coefficient arrays are inputs to these routines). Under normal circumstances, the user need not be concerned with this data linkage and should not try to change it. (For a definition of the contents of the !SCOEF array see table 7.2-II(b) of volume I.) The input parameter is the ephemeris delta time (EDT) value that is used in the computation of the Julian ephemeris date of base date. It must be input in seconds. The default value is set at -9999.5. This causes the processor to internally compute the value of EDT. Any input value greater than -9999.0 will cause the internal computation of EDT to be bypassed and the input value used instead.

The outputs generated by the BASTM processor are the three data arrays SESCON, SCOEF, and LCOEF, which have already been described in the preceding paragraphs. In addition, the processor prints an output display on the user's terminal. The format and contents of this are documented under the subsequent paragraph titled "Processor Displays and Display Parameter Definition Table."

3.0 ASSUMPTIONS AND LIMITATIONS

a. For the input parameters YEAR, MONTH, and DAY, there are limits on the range of valid values. They are the following:

$$1901 \leq \text{YEAR} \leq 2099$$
$$1 \leq \text{MONTH} \leq 12$$
$$1 \leq \text{DAY} \leq 31$$

If an out-of-range value is input for one or more of these parameters, the BASTM processor will display an error message on the user's terminal and will terminate without generating the output arrays SESCON, SCOEF, or LCOEF. The format and content of the error message(s) output by BASTM are shown in table 4-V.

b. For the ephemeris initialization flag, EPHEM, there are three valid entries. For EPHEM(1), either "SUN" or Ø are correct, and for EPHEM(2), either "MOON" or Ø are correct.

c. The Moon ephemeris capability is not implemented, and would be selected by EPHEM(2) = "MOON".

4.0 PROCESSOR INPUT/OUTPUT

a. Processor interface table - The definition of the default interface table for the BASTM processor is provided in table 4-I.

b. Interface table data array definitions - A definition of the input/output data arrays appearing in the BASTM interface table is provided in table 4-II

c. Interface table data file definitions - None.

d. Processor solicited (prompted) inputs - None.

e. Processor displays and display parameter definition tables - When the BASTM processor executes correctly it will automatically generate an output display on the user's terminal/printout. The format and content of this display is shown in table 4-III and a definition of the display variables is provided in table 4-IV.

f. Processor message table - If the user inputs an out-of-range value for one or more of the parameters, YEAR, MONTH, or DAY, the BASTM processor will print an error message(s) on the user's terminal. If the user supplies an invalid OK not implemented option for the parameter EPHEM, an error message will be issued. The error message(s) will identify the parameter(s) and value(s) that are out-of-range or invalids. The format and content of the message(s) is given in table 4-V.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-VI.

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR BASTM

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
GLOCON	AWA	Free	I	180	180	!!GLCN	Global constants array, master data base element
YEAR	AWA	Intg	I	1	1	--	Year of base date
MONTH	AWA	Intg	I	1	1	--	Month of base date
DAY	AWA	Intg	I	1	1	--	Day of base date
HOURS	AWA	Real	I	2	1	--	Hour of base time (GMT)
MINS	AWA	Real	I	2	1	--	Minute of base time (GMT)
SECS	AWA	Real	I	2	1	--	Second of base time (GMT)
EPHEM	AWA	6CH	I	6	2	&, "Ь"	Ephemeris initialization flag: EPHEM(1) = "SUN" output SCOEF = "Ь" don't output SCOEF EPHEM(2) = "MOON" output LCOEF = "Ь" don't output LCOEF

NOTES	CLASS	TYPE			USE
	AWA	Free	2CH	72CH	I = Input
	Disk	Intg	6CH	Mix	O = Output
		Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-I.- Concluded

PROCESSOR BASTM

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
EDT	AWA	Real	I	2	1	-9999.5	Ephemeris delta time, sec. EDT ≤ -9999.0 causes internal computation of EDT EDT > -9999.0 causes input value to be used.
SESCON	AWA	Free	0	90	90	!SESCN	Session constants array
SCOEF	AWA	Real	0	44	22	!SCOEF	Solar coefficient array; output only if EPHEM(1) = "SUN"
LCOEF	AWA	Real	0	80	40	!LCOEF	Lunar coefficient array; output only if EPHEM(2) = "MOON"

NOTES	CLASS	TYPE			USE
	AWA	Free	2CH	72CH	I = Input
	Disk	Intg	6CH	Mix	O = Output
		Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-II.- INTERFACE TABLE DATA ARRAY DEFINITIONS

PROCESSOR BASTM

Array name	Index location	Default value	Definition
GLOCON	(1) . . . (180)	!!GLCN	Global constants array, master data base element; see table 7.2-III of volume I for definition of contents
SESCON	(1) . . . (90)	!SESCN	Output session constants array; see table 7.2-II of volume I for a definition of contents
SCOEF	(1) . . . (22)	!SCOEF	Output solar coefficient array; see table 7.2-IIA of volume I for a definition of contents
LCOEF	(1) . . . (40)	!LCOEF	Output lunar coefficient array; see table 7.2-IIB of volume I for a definition of contents

TABLE 4-III.- PROCESSOR DISPLAY TABLE

PROCESSOR BASTM

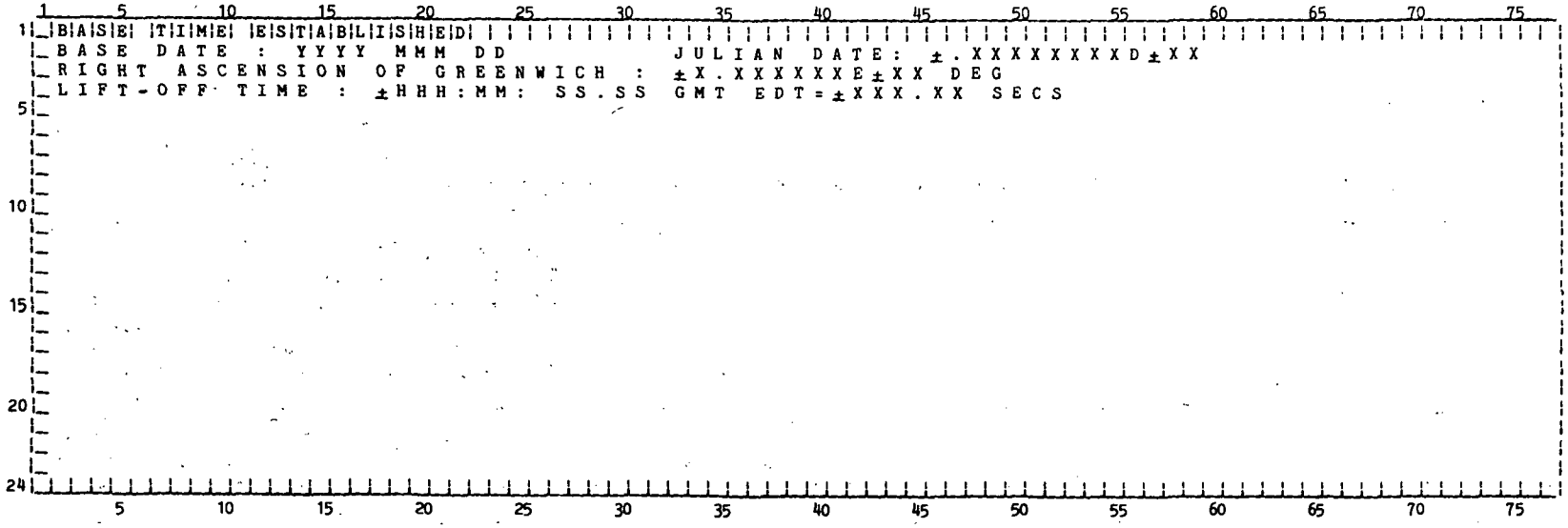


TABLE 4-IV.- DISPLAY PARAMETER DEFINITIONS TABLE

PROCESSOR BASTM

BASTM processor display	
Display parameter label	Parameter definition
Base date	The user-specified base date in the form YYYY MMM DD where: YYYY = the year (4-digit integer) MMM = month of the year (3-character abbreviation) DD = day of the month (2-digit integer)
Julian date	The computed Julian data corresponding to the specified base date in the form where: ±.xxxxxxxxD±xx = Julian day number (8-digit double precision)
Right ascension of Greenwich	The computed right ascension of the Greenwich meridian at the specified base date in the form ±x.xxxxxx E±xx where: ±x.xxxxxx E±xx = right ascension in degrees
Lift-off time	The user-specified GMT of lift-off in the form HH:MM:SS.SS where: HHH = hours (3-digit integer) MM = minutes (2-digit integer) SS.SS = seconds (4-digit real)
EDT	The value of Ephemeris delta time at base date that was used in the BASTM processor calculations in the form ±xxx.xx secs.

TABLE 4-V.- PROCESSOR MESSAGE TABLE

PROCESSOR BASTM

MSG no.	Message ID block	Message text block and explanation
1	*BASTM*	INPUT PARAMETER OUT-OF-RANGE: PPPPP = XXXX Meaning: The user specified out-of-range values for one or more input parameters; the parameter(s) and out-of-range value(s) are displayed as: PPPPP = XXXX Severity: The BASTM processor will terminate execution without generating the output arrays SESCON, SCOEF, and LCOEF. Control will be returned to the FDS executive sequence table processing (if any) will be terminated. Action required by user: Use the interface table editor to correct the out-of-range parameter values, then execute again.

TABLE 4-VI.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR BASTM

Processor name	Processor abstract prompt (maximum 256 characters)
BASTM	BASTM is the utility processor that provides the user with the capability to establish the simulation basetime, initialize the session constants array, and optionally set up coefficient arrays for analytic Sun-Moon ephemeris.

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
GLOCON	Global constants array; master data base element normally defaulted to !!GLCN
YEAR	Year of base date (integer)
MONTH	Month of base date (integer: 1 to 12)
DAY	Day of base date (integer: 1 to 31)
HOURS	Hour of base time GMT (real number)
MINS	Minute of base time GMT (real number)
SECS	Second of base time GMT (real number)
EPHEM	Ephemeris initialization flag (two-word Hollerith array) EPHEM(1) = "SUN" - Output SCOEF = "␣" - Don't output SCOEF EPHEM(2) = "MOON" - Output LCOEF = "␣" - Don't output LCOEF
SESCON	Session constants array; output normally defaulted to !SESCN

TABLE 4-VI.- Concluded

PROCESSOR BASTM

Processor name	Processor abstract prompt (maximum 256 characters)

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
SCOEF	Solar coefficient array; output only if EPHEM(1) = "SUN", and normally defaulted to !SCOEF
LCOEF	Lunar coefficient array; output only if EPHEM(2) = "MOON", and normally defaulted to !LCOEF
EDT	Ephemeris delta time in seconds (real number) EDT ≤ - 9999.0 = Compute value of EDT internally EDT > - 9999.0 = Use input value of EDT

5.0 PROCESSOR ROUTINES

5.1 ROUTINE NAME - MAIN PROGRAM BASTM

5.1.1 Purpose

BASTM serves as the main routine of the BASTM processor. The routine provides most of the computational functions of the utility processor BASTM. These computational functions are:

a. Day of year computation

b. Ephemeris delta time computation

c. Julian date computation

d. Precession and nutation matrix computation

e. Right ascension of Greenwich computation

f. Solar coefficient computations

g. Lunar coefficient computations

In addition to the computational functions, the routine BASTM performs validity checks on certain variables contained in the interface table and displays appropriate error messages.

5.1.2 Functional Description

BASTM calls the utility routine RMPAR to get the logical unit number and the user qualifier code. The FDS utility routine XPGET is called to get the data contained in the interface table. The data parameters contained in the interface table are GLOCON, YEAR, MONTH, DAY, HOURS, MINS, SECS, EPHEM, and EDT. The definitions of these parameters are given in section 5.1.5, table 5.1-IV, Routine Input/Output Variables. The BASTM utility processor subroutine CONST is called to initialize the constants that are used in the precession and nutation matrix calculations. The BASTM utility processor subroutine VALCK is called to perform validity checks on the values of the interface table variables YEAR, MONTH, and DAY. The validity check of each variable consists of determining whether or not the value of the variable is within a given range. The variables and valid range of values are

$$1901 \leq \text{YEAR} \leq 2099$$
$$1 \leq \text{MONTH} \leq 12$$
$$1 \leq \text{DAY} \leq 31$$

Error messages are displayed for variables outside the range, and execution of the processor is terminated with the error code set to the system error

condition value of -32768. The BASTM utility processor subroutine EPHMC is called to check the interface table parameter EPHEM for valid entries, and for valid entries that are implemented. Appropriate error messages are displayed when the content of EPHEM is invalid, or contains an option not implemented, and execution of the processor is terminated with the return code set to the system error condition value of -32768. When EPHEM contains valid options, a flag is set to indicate the ephemeris calculations, if any, that are to be made. The valid EPHEM values are:

```
EPHEM(1) = "SUN"  - Output SCOEF
         = "Ø"    - Don't output SCOEF
EPHEM(2) = "MOON" - Output LCOEF
         = "Ø"    - Don't output LCOEF
```

The integer day of the year is calculated from the input interface table parameters MONTH and DAY using the parameter YEAR to correct for leap year. The BASTM utility processor subroutine CDTJD is called to calculate the Julian date (JD). The interface table parameter EDT, which is the ephemeris delta time, may contain the actual value of EDT or it may contain the code value of -9999.0, indicating that the EDT is to be calculated by the BASTM processor. When the EDT value is equal to the code value, the BASTM utility processor subroutine CEDT is called to compute EDT. The Julian ephemeris date (JED) is calculated from the Julian date and the ephemeris delta time.

The precession matrix, the nutation matrix, and the right ascension of the Greenwich meridian calculations are made.

The BASTM interface table output parameter SESCON is built by transferring the required constants from the input GLOCON array and the parameters computed by the processor. The FDS utility routine XPPUT is called to output the array SESCON to the active work area (AWA). The base date, Julian date, right ascension of Greenwich, lift-off time, and ephemeris delta time are displayed.

The ephemeris calculation flag, which is set in subroutine EPHMC, is tested. If it is equal to 0, no ephemeris calculations are made. If it is equal to 1, the BASTM utility processor subroutine SCOF is called to compute the solar ephemeris coefficients. The solar ephemeris option is the only ephemeris calculation presently implemented. The FDS utility subroutine XPPUT is called to output the interface table array SCOEF to the AWA.

The FDS utility subroutine XPXIT is called to terminate the BASTM processor.

5.1.3 Assumptions and Limitations

The assumptions and limitations are discussed throughout section 5.1.4 and in the subroutine documentation. They are listed in this section.

Assumptions:

Linear relationship between Julian date and ephemeris delta time

Limitations:

a. Seven-digit single-precision real numbers on HP21MX

b. Two angles (ℓ and ℓ') are dropped in nutation

c. Four terms are used in $\delta\psi$ computation (nutation in longitude)

d. Two terms are used in $\delta\epsilon$ computation (nutation in obliquity)

e. $1900 \leq \text{YEAR} \leq 2099$

f. Option of solar ephemeris or no ephemeris calculations; lunar ephemeris calculations not implemented

5.1.4 Method

The computations for the day of the year, the precession matrix, the nutation matrix, and the right ascension of the Greenwich meridian are made in the main routine BASTM. The computations for the ephemeris delta time, the Julian date, and the solar ephemeris coefficients are made in subroutines called by the main routine BASTM. The ephemeris delta time is calculated in subroutine CEDT, which is described in section 5.2. The Julian date is calculated in the subroutine CDTJC, which is described in section 5.4. The solar ephemeris coefficients are calculated in the subroutine SCOF, which is described in section 5.6. The algorithm and calculations used in each computation are described in following sections. These sections also describe the method used in the initialization/input logic, validity check logic, and the output logic.

5.1.4.1 Initialization/Input Logic

The method, conventions, and logic involved in the initialization and input to the BASTM utility processor are described in this section. The logical unit number of the user terminal must be known internally by the utility processor program in order to provide displays on the user's terminal, and to obtain input data from the processor interface table through the AWA. The Hewlett-Packard Real-Time Executive (RTE) provides this capability to applications programs, including a FORTRAN-callable routine RMPAR. The form of the call is

```
CALL RMPAR (IPARM)
```

where the statement DIMENSION IPARM(5) must be included in the specification statements. On return from RMPAR, the logical unit number of the user terminal is contained in IPARM(1), and must be saved in a global variable for use by the processor routines as needed. The first two executable statements in the processor main routine must be

```
CALL RMPAR (IPARM)

LU = IPARM(1)
```

The input data from the interface table are obtained from the AWA by calling the FDS utility subroutine XPGET. The form of the call is

```
CALL XPGET (LU,INTBUF,INTLNG,MRBUFF,N,

        INUMS,IN(1),----IN(N))
```

A full description of the purpose, method, and use of XPGET is provided in section 10.1.1, Parameter Retrieval Routine (XPGET) (ref. 1). Briefly, a variable name is provided in the calling sequence for each parameter that is to be input. On return from XPGET, each variable has been loaded with the corresponding data contained in the AWA.

5.1.4.2 Validity Check Logic

Validity checks are performed on the input interface table parameters YEAR, MONTH, and DAY. They are checked for values within a given range by the three separate calls to the BASTM utility processor subroutine VALCK. A description of VALCK is provided in section 5.5. Because of the continual changes in the Earth's position in space, and the resultant necessary recomputation of coefficients used in defining this position, the algorithms used in the BASTM processor are considered to be valid for years in the range $1901 \leq \text{YEAR} \leq 2099$. The input variable MONTH contains a valid value when the range is $1 \leq \text{MONTH} \leq 12$. The input variable DAY contains a valid value when the range is $1 \leq \text{DAY} \leq 31$.

The input variable EPHEM is an array containing one six-character alphanumeric specification for each ephemeris calculation that is to be made. The BASTM utility processor subroutine EPHMC is called to check the input array EPHEM for valid and implemented ephemeris calculation options. A description of the subroutine EPHMC is provided in section 5.7.

When any of the parameters checked are out of range or invalid, an abnormal termination of the processor occurs. The abnormal termination procedure and error messages are discussed in section 5.1.4.10, Output Logic.

5.1.4.3 Day of Year Computation

The algorithm and code for this calculation was taken from the MDAS program, processor BASTM (ref. 2). The FORTRAN code used is as follows:

$$IDAYS = MONTH - 1$$

$$IDAYS = 31 * IDAYS - (IDAYS - IDAYS/8)/2 + DAY$$

$$- \{(IDAYS + 8)/10\} * \{[MOD(YEAR, 4) + 5]/3\}$$

where MONTH, DAY, YEAR form the date of interest and are input through the interface table. The input variable MONTH is defined as the month of base date. The input variable DAY is defined as the day of base date, and the input variable YEAR is defined as the year of base date.

The function MOD is the standard FORTRAN MOD function, which is defined as

$$a_1 - [a_1/a_2]a_2$$

where a_1/a_2 is the largest integer whose magnitude does not exceed the magnitude of a_1/a_2 and whose sign is the same as the sign of a_1/a_2 (ref. 3). It should be noted that the algorithm for IDAYS uses standard FORTRAN integer arithmetic.

During the checkout of the stand-alone processor, the computed value of IDAYS was displayed for the input MONTH, DAY, and YEAR values. For all cases, the calculated value of IDAYS agreed with the actual day number.

The day of the year, IDAYS, is used in the calendar date to Julian date calculation. This calculation is performed in subroutine CDTJD, as described in section 5.4.

5.1.4.4 Ephemeris Delta Time Computations

This section contains a discussion of the fundamental epochs and measures of time. This discussion was extracted from reference 3, section 1G.

Ephemeris time. The fundamental epoch to which the elements of the Sun, Moon, and planets are referred is

$$1900 \text{ January } 0 \text{ at } 12^h \text{ ephemeris time}$$

$$= 1900 \text{ January } 0.5 \text{ ET} = \text{JED } 241\ 5020.0 \text{ ET}$$

Ephemeris time is measured conventionally in years, months, days, and subdivisions of a day. The interval T of ephemeris time from the fundamental epoch contains:

T Julian centuries of 36525 days, each of 86400 ephemeris seconds;

d, or 10000 D, ephemeris days ($d = 36525\ T$; $D = 3.6525\ T$)

When desirable to emphasize that these relate to an interval of ephemeris time, a subscript E is added; thus

T_E, d_E, D_E

Universal time. The fundamental epoch that is used in the definition and derivation of universal time is

1900 January 0 at 12^h universal time

= 1900 January 0.5 UT = JD 241 5020.0 UT

The interval T_u of universal time from this epoch contains

T_u Julian centuries of 36525 days, each of 86400 seconds of UT;

d_u, or 10000 D_u, days of UT ($D_u = 36525\ T_u$; $D_u = 3.6525\ T_u$).

The subscript u is always used, unless the context makes it superfluous.

ET - UT. At any instant the measure of ephemeris time (epoch + T_E) is equal to the measure of universal time (epoch + T_u) + ΔT; thus

$$\Delta T = ET - UT = T_E - T_u$$

ΔT is most conveniently expressed in seconds of time.

It must be emphasized that the fundamental epochs used for ephemeris time and universal time, although denoted by the same measure, do not correspond to the same instant of time; in fact, at each epoch ΔT_0 is about -4^S (i.e., the epoch of ET is 4^S later than that of UT). The interval of time between two instants, the later one being indicated by a prime, can be expressed as

$T'_E - T_E$ of ephemeris time

or as

$$T'_u - T_u = (T'_E - T_E) - (\Delta T' - \Delta T) \text{ of universal time.}$$

The difference in the two measures involves the values of ΔT at both instants. It is only because the two fundamental epochs correspond to instants separated by ΔT_0 (and have the same measure) that it is possible to write

$$T_E = T_u + \Delta T$$

The Besselian solar year. For certain applications it is more convenient to measure time in units of tropical centuries of 36524.21988 ephemeris days; the fundamental epoch being the beginning of the Besselian (fictitious) solar year 1900.0, or 1900 January $0^d.814$ ET. In the great majority of such cases the difference in length of the century is not significant: the same symbol T is used accordingly, though always with a specific explanation. The difference between the length of the Besselian solar year and the tropical year ($0^s.148$ T) can always be neglected, and multiples of 0.01 in T thus relate to the beginning of the corresponding Besselian year (see section 2B of reference 4).

The fraction of the tropical year is denoted by τ, measured backwards or forwards from the beginning of the Besselian year. A unit difference in τ corresponds to a difference of 0.01 in T.

An interval of time measured in tropical years is denoted by t. Initial and general epochs are denoted by t_0 and t, respectively. The context will indicate the meaning to be attached to t_0 and t.

t - 1950.0 clearly implies that t is an epoch; e.g., 1960.0

1950.0 + t clearly indicates that t is an interval; e.g., 10.0.

In some contexts, the epoch t_0 is used for that of 1900.0 + T_0, and the epoch t for that of 1900.0 + T_0 + T; $t_0 = 100\ T_0$ and $t = 100\ T$ are both intervals, but are used conventionally to describe epochs.

The ephemeris delta time (EDT) is used to compute the Julian ephemeris date (JED), which is required in other calculations. The EDT may be calculated by the BASTM processor in subroutine CEDT, or the value of EDT may be supplied through the interface table input parameter EDT. Section 5.2 describes the computation of EDT.

5.1.4.5 Julian Date Computations

This section contains a discussion of the Julian date, which was extracted directly from reference 4, section 3B.

To facilitate chronological reckoning, astronomical days (beginning at Greenwich noon) are numbered consecutively from an epoch sufficiently far in the past to precede the historical period. The number assigned to a day in this continuous count is the Julian day number, which is defined to be 0 for the day starting at Greenwich mean noon on January 1, 4713 B.C., Julian proleptic calendar. The Julian day number therefore denotes the number of days that has elapsed, at Greenwich noon on the day designated, since the above epoch. The Julian date (JD) corresponding to any instant is, by a simple extension of the above concept, the Julian day number followed by the fraction of the day elapsed since the preceding noon.

Although introduced as a continuous count and measure of mean solar days, the Julian day number and the Julian date can conveniently be applied to ephemeris time, in which case the Julian date will differ from the conventional one by ΔT. The Julian day number will represent the number of ephemeris days that have elapsed, at the preceding 12^h ET, since 12^h ET, on January 1, 4713 B.C. It is not necessary, in this definition, to know to what universal time this epoch corresponds; i.e., to know ΔT at the epoch. In fact, the measure may be regard as conventional, applicable to both systems of time measurement, as in the case of calendar dates. The terminology Julian ephemeris date (JED) may be used when necessary to distinguish the Julian date in ephemeris time with the day beginning at 12^h ET from the Julian date in universal time with the day beginning at 12^h UT. Such a distinction may be essential in dating orbital elements, or in formulae for light-curves of variable stars, where the time must be given to a large number of decimal places. The fundamental epoch 1900 January 0^d 12^h ET is JED 241 5020.0.

The Julian date is used in the computation of the JED, and in the computation of the right ascension of the Greenwich meridian. Section 5.4 describes the Julian date calculation.

5.1.4.6 Precession and Nutation Matrix Computations

This section contains a discussion of the basic concepts of the motion of the Earth and derivations of the precession and nutation matrices. The discussion and derivations were extracted from reference 5.

a. Motion of the Earth. The equinox, T, is defined by the intersection of the planes of the Earth's equator and the ecliptic. In turn, the equator is defined as being normal to the Earth's pole. The fact that the equinox is in motion was discovered in the second century before the Christian era by Hipparchus (ref. 6). The primary motion of the equinox is called precession, and is due to the precession of the Earth's pole. An analogy of a

spinning top is frequently mentioned as an aid in visualizing the precession of the Earth's pole. More precisely, the precessional motion of the mean equinox is due to the combined motions of the two poles that define it (ref. 4).

The motion of the celestial pole, or the equator, is due to the gravitational action of the Sun and Moon on the equational bulge of the Earth. It consists of two components; luni-solar precession and nutation. Luni-solar precession is the smooth long-period motion of the mean pole of the equator (the word mean indicates that nutation is being neglected) around the pole of the ecliptic with an amplitude of approximately $23.^{\circ}5$, and a period of about 26 000 years. Nutation is a relatively short-period motion that carries the actual (or true) pole around the mean pole in a somewhat irregular curve with an amplitude of about 9" (" indicates seconds of arc), and a period of about 18.6 years. The motion of the ecliptic (i.e., the mean plane of the Earth's orbit) is due to the gravitational action of the planets on the Earth, and consists of a slow rotation of the ecliptic. This motion is known as planetary precession and gives a precession of the equinox of about 12" a century, and a decrease of the obliquity of the ecliptic (the angle between the ecliptic and the Earth's equator) of about 47" a century. The motions of the mean pole, together with the motion of the ecliptic, are known as general precession. General precession is discussed further in section 5.1.4.6a(1), and nutation is discussed in section 5.1.4.6a(2).

(1) General precession of the pole of the Earth - As discussed previously, the precession of the Earth consists of two components; luni-solar precession and planetary precession. It is customary to consider the two components together as a single quantity called general precession, which is described by the angles ζ_0, θ, and z. The precession angles are illustrated in figure 5.1-1.

where

$\frac{\pi}{2} - \zeta_0$ = Angle from mean equinox of epoch to the intersection of the mean equator of date and the mean equator of epoch

θ = Angle between the mean equator of date and the mean equator of epoch

$\frac{\pi}{2} + z$ = Angle from intersection of the mean equator of date with the mean equator of epoch to the mean equinox of date, measured clockwise in the mean equatorial plane

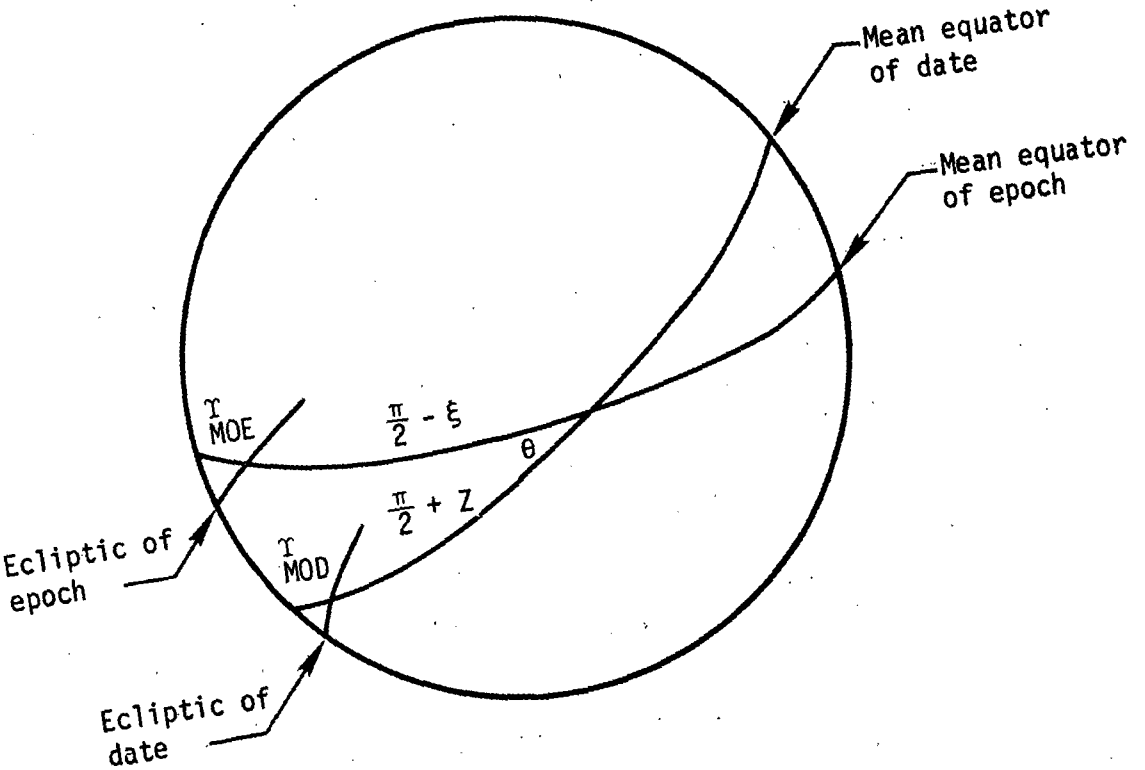


Figure 5.1-1.- Geometry for general precession terms.

The transformation from the mean celestial equator of epoch system to the mean celestial equator of date system is accomplished by the matrix P, which is made up of three rotation matrices. First, a positive rotation about the Z-axis through the angle $\pi/2 - \zeta_0$ is performed by the matrix

$$R_1 = \begin{bmatrix} \cos\left(\frac{\pi}{2} - \zeta_0\right) & \sin\left(\frac{\pi}{2} - \zeta_0\right) & 0 \\ -\sin\left(\frac{\pi}{2} - \zeta_0\right) & \cos\left(\frac{\pi}{2} - \zeta_0\right) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Next, a positive rotation above the new X-axis through the angle θ is performed by the matrix

$$R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & +\sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}$$

Finally, a negative rotation about the new Z-axis through the angle $\pi/2 + z$ is performed by the matrix

$$R_3 = \begin{bmatrix} \cos\left(\frac{\pi}{2} + z\right) & -\sin\left(\frac{\pi}{2} + z\right) & 0 \\ \sin\left(\frac{\pi}{2} + z\right) & \cos\left(\frac{\pi}{2} + z\right) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Then the matrix $P = R_3 R_2 R_1$, and

$P_{11} = \cos\zeta_0 \cos\theta \cos z - \sin\zeta \sin z$

$P_{12} = -\sin\zeta 0 \cos\theta \cos z - \cos\zeta_0 \sin z$

$P_{13} = -\sin\theta \cos z$

$P_{21} = \cos\zeta_0 \cos\theta \sin z + \sin\zeta_0 \cos z$

$P_{22} = -\sin\zeta_0 \cos\theta \sin z + \cos\zeta_0 \cos z$

$P_{23} = -\sin\theta \sin z$

$P_{31} = \cos\zeta_0 \sin\theta$

$P_{32} = -\sin\zeta_0 \sin\theta$

$P_{33} = \cos\theta$

Thus, if $\underline{r}$ is a vector in the mean celestial equator of epoch system, the same vector measured in the mean celestial equator of date system is given by $\underline{R}$, where

$$\underline{R} = P\underline{r}$$

The transpose of P, indicated by P^T, performs the transformation from the mean celestial equator of date system to the mean celestial equator of epoch system. It is noted that the matrix P is orthogonal; hence, the inverse of $P(P^{-1})$ is equal to the transpose of $P(P^T)$. The computation of the angles ζ_0, θ, and z is described in section 5.1.4.6b(1).

(2) Nutation of the pole of the Earth - Nutation is that part of the precessional motion of the pole of the Earth's equator that depends on the periodic motions of the Sun and Moon in their orbits around the Earth. The long-period motion of the mean pole is called luni-solar precession. Nutation is the circular motion of the true pole about the mean pole in a period of about 19 years, with an amplitude of about 9". The principal term depends on the longitude of the node of the Moon's orbit, and has a period of 6798 days (or 18.6 years); the amplitude of this term, 9".210, is known as the constant of nutation. In the complicated theory of the gravitational action of the Sun and Moon on the rotating nonspherical Earth, other terms arise that depend on the mean longitudes and mean anomalies of the Sun and Moon, and on their combinations with the longitude of the Moon's node. The resulting shift of the true pole, with respect to the mean pole, can be resolved into corrections to the longitude ($\delta\psi$, nutation in longitude) and to the mean obliquity ($\delta\epsilon$, nutation in obliquity). Expressions for $\delta\psi$ and $\delta\epsilon$ are generally developed in series form. The terms divide naturally into those not depending on the Moon's longitude, which can be interpolated at intervals of 10 days, and those depending on the Moon's longitude, with periods of less than about 60 days, which cannot be so interpolated. Nutation is therefore divided into long-period and short-period terms. The short-period terms, which have periods less than 35 days, are summed separately as $d\psi$ and $d\epsilon$, and are called the short-period terms of nutation in longitude and obliquity, respectively. The long-period terms of nutation in longitude and obliquity are denoted by $\Delta\psi$ and $\Delta\epsilon$, respectively. Then the total nutations in longitude and obliquity (fig. 5.1-2) are

$$\delta\psi = \Delta\psi + d\psi$$
$$\delta\epsilon = \Delta\epsilon + d\epsilon$$

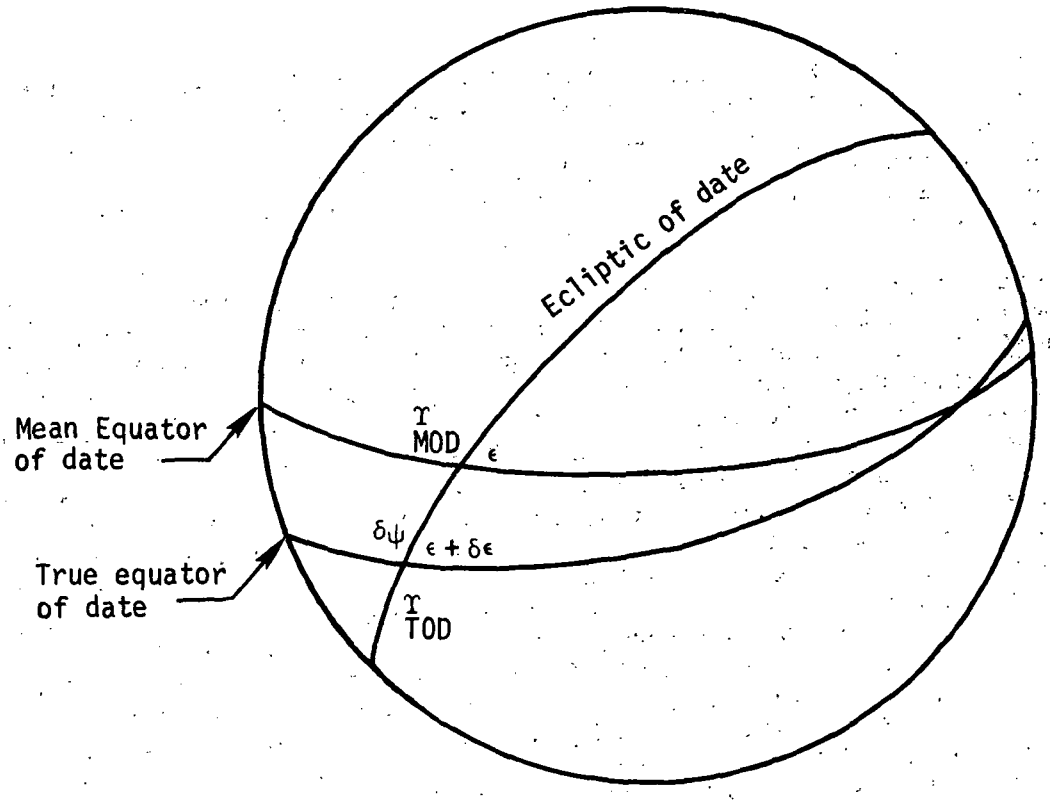


Figure 5.1-2.- Geometry for nutation terms.

The angle ϵ in figure 5.1-2 is the mean obliquity of the ecliptic defined as the angle between the ecliptic of date and the mean celestial equator of date. Angles ϵ, $\delta\psi$, and $\delta\epsilon$ are computed in section 5.1.4.6b(2).

The transformation from the mean celestial equator of date system to the true celestial equator of date system is accomplished by the matrix N, where N is the product of three rotation matrices.

$$N = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\epsilon + \delta\epsilon) & -\sin(\epsilon + \delta\epsilon) \\ 0 & \sin(\epsilon + \delta\epsilon) & \cos(\epsilon + \delta\epsilon) \end{bmatrix} \begin{bmatrix} \cos\delta\psi & -\sin\delta\psi & 0 \\ \sin\delta\psi & \cos\delta\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\epsilon & \sin\epsilon \\ 0 & -\sin\epsilon & \cos\epsilon \end{bmatrix}$$

and

$$n_{11} = \cos\delta\psi$$

$$n_{12} = -\sin\delta\psi\cos\epsilon$$

$$n_{13} = -\sin\delta\psi\sin\epsilon$$

$$n_{21} = \cos(\epsilon + \delta\epsilon)\sin\delta\psi$$

$$n_{22} = \cos(\epsilon + \delta\epsilon)\cos\delta\psi\cos\epsilon + \sin(\epsilon + \delta\epsilon)\sin\epsilon$$

$$n_{23} = \cos(\epsilon + \delta\epsilon)\cos\delta\psi\sin\epsilon - \sin(\epsilon + \delta\epsilon)\cos\epsilon$$

$$n_{31} = \sin(\epsilon + \delta\epsilon)\sin\delta\psi$$

$$n_{32} = \sin(e + \delta\epsilon)\cos\delta\psi\cos\epsilon - \cos(\epsilon + \delta\epsilon)\sin\epsilon$$

$$n_{33} = \sin(\epsilon + \delta\epsilon)\cos\delta\psi\sin\epsilon + \cos(\epsilon + \delta\epsilon)\cos\epsilon$$

The transpose of N, N^T, performs the transformation from the celestial equator true of date system to the celestial equator mean of date system.

b. Calculation of Precession and Nutation Matrices.- The expressions for the precession and nutation matrix calculations are presented in the following two sections. All computations for the precession and nutation matrices are performed in double precision on the HP21MX, and are converted to single precision when stored in the output array SESCON.

(1) Precession matrix calculations - The precession matrix describes the transformation from the mean of 1950 to the mean of base date reference system. As discussed in section 5.1.4.6.a(1), the precession matrix calculation involves the three angles ζ_0, θ, and z, which are computed from the following expressions.

$$\zeta_0 = 2304''.951665T + 0''.302165T^2 + 0''.0180T^3$$

$$\theta = 2004''.258258T - 0''.426885T^2 - 0''.0418T^3$$

$$z = 2304''.951615T + 1''.095195T^2 + 0''.01832T^3$$

where T = (JED - 2433282.4236)/36524.219879 tropical centuries.

JED is the Julian ephemeris date of the date of interest. (Refer to sections 5.1.4.4 and 5.1.4.5 for discussion of the systems of time measurement.)

2433282.4236 = JED of the beginning of the Besselian year of 1950.

36524.219879 = Number of ephemeris days in a tropical century.

Table 5.1-I provides a list of the mathematical symbols versus internal code symbols for the precession calculations.

The coefficients of the terms in the expressions for ζ_0, θ, and z are exact values, which were taken from SVDS subroutine RPNMAT (ref. 7).

(2) Nutation matrix calculations - The nutation matrix describes the transformation from the mean of base date to the true of base reference system. As discussed in section 5.1.4.6a(2), the nutation matrix calculation of the angles ϵ, $\delta\epsilon$, and $\delta\psi$. Reference 4 gives the following expression for ϵ.

$$\epsilon = 84428''.26 - 46''.845T_{1900} - 0''.0059T_{1900}^2 + 0''.0181T_{1900}^3$$

Reference 8 gives expressions for $\delta\epsilon$ and $\delta\psi$ as a polynomial series of the sines and cosines of five angles. These angles are defined by the following expressions.

$$\ell = 296^\circ 06' 16''.59 + 1325^r 198^\circ 50' 56''.79T_{1900} + 33''.09T_{1900}^2 + 0.0518T_{1900}^3$$

$$\ell' = 358^\circ 28' 33''.00 + 99^r 359^\circ 02' 59''.10T_{1900} - 0''.54T_{1900}^2 - 0''.0120T_{1900}^3$$

$$F = 11^\circ 15' 03''.20 + 1342^r 82^\circ 01' 30''.54T_{1900} - 11''.56T_{1900}^2 - 0''.0012T_{1900}^3$$

$$D = 350^\circ 44' 14''.95 + 1236^r 307^\circ 06' 51''.18T_{1900} - 5''.17T_{1900}^2 + 0''.0068T_{1900}^3$$

$$\Omega = 259^\circ 10' 59''.79 - 5^r 134^\circ 08' 31''.23T_{1900} + 7''.18T_{1900}^2 + 0''.0080T_{1900}^3$$

The series for $\delta\varepsilon$ is composed of 40 terms involving cosines of various combinations of the five angles. The series for $\delta\psi$ is composed of 69 terms involving sines of various combinations of the five angles. The number of terms used in the calculations was reduced for the following reasons.

(a) The seven-digit accuracy of single-precision numbers on the HP21MX.

(b) The FDS-1 overall design philosophy of limited accuracy to obtain increased speed and efficiency.

The series for $\delta\varepsilon$ and $\delta\psi$ were truncated by dropping all terms with coefficients less than 0.2 arc second. This results in the following series, which are expressions of the angles F, D, and Ω only. It should be noted that the omitted terms amount to only 0.333 arc second in $\delta\psi$, and 0.276 arc second in $\delta\varepsilon$. These values are the sum of the magnitudes of the omitted terms, and scarcely ever will be attained since the five angles rarely will have the correct phasing for this to occur. Table 5.1-II provides a list of the mathematical symbols versus internal code symbols for the nutation calculations.

The nutation in longitude is given by

$$\delta\psi = -17\overset{''}{.}2327 \sin(\Omega) - 1\overset{''}{.}2729 \sin(2F - 2D + 2\Omega) + 0\overset{''}{.}2088 \sin(2\Omega) - 0\overset{''}{.}2037 \sin(2F + 2\Omega)$$

The nutation in obliquity is given by

$$\delta\varepsilon = 9\overset{''}{.}2100 \cos(\Omega) + 0\overset{''}{.}5522 \cos(2F - 2D + 2\Omega)$$

where reference 8 gives the following expressions for the fundamental arguments.

$$F = 11^{\circ}15'03\overset{''}{.}20 + 1342^{r}82^{\circ}01'30\overset{''}{.}54T_{1900} - 11\overset{''}{.}56T_{1900}^{2} - 0\overset{''}{.}0012T_{1900}^{3}$$

$$D = 350^{\circ}44'14\overset{''}{.}95 + 1236^{r}307^{\circ}6'51\overset{''}{.}18T_{1900} - 5\overset{''}{.}17T_{1900}^{2} + 0.0068T_{1900}^{3}$$

$$\Omega = 259^{\circ}10'59\overset{''}{.}79 - 5^{r}134^{\circ}8'31\overset{''}{.}23T_{1900} + 7\overset{''}{.}48T_{1900}^{2} + 0.0080T_{1900}^{3}$$

T_{1900} is the time elapsed in Julian centuries from noon ephemeris time on the day before January 1, 1900, to the date of interest, and is calculated from

$$T_{1900} = (JED - 2415020)/36525.$$

The coefficients in the expressions for F, D, and Ω were converted to degrees, and were coded as follows.

$$F = 11^{\circ}.2508888889 + 483202^{\circ}.02517T_{1900} - 0^{\circ}.321111111111 \times 10^{-2}T_{1900}^{2} - 0^{\circ}.333333333333 \times 10^{-6}T_{1900}^{3}$$

$$D = 350^{\circ}.737486111 + 445267^{\circ}.114217T_{1900} - 0^{\circ}.143611111111 \times 10^{-2}T_{1900}^{2} + 0^{\circ}.188888888889 \times 10^{-5}T_{1900}^{3}$$

$$\Omega = 259^{\circ}.183275000 - 1934^{\circ}.14200833T_{1900} + 0^{\circ}.207777777778 \times 10^{-2}T_{1900}^{2} + 0^{\circ}.222222222222 \times 10^{-5}T_{1900}^{3}$$

As stated previously, in order to obtain a nutation matrix that is accurate to single precision (i.e., seven digits), all calculations are performed in double precision and the results truncated to single precision.

5.1.4.7 Right Ascension of Greenwich Computations

A discussion of the astronomical measures of time and related concepts, which was extracted from reference 4, section 3B, follows.

a. Ephemeris time. Ephemeris time is a uniform measure of time depending on the laws of dynamics for its determination. It is the independent variable in the gravitational theories of the Sun, Moon, and planets, and the argument for the fundamental ephemerides in the ephemeris.

The measure of ephemeris time has been chosen to agree as nearly as possible with that of universal time during the nineteenth century, and it is unlikely that the two measures will differ by more than a few minutes in the twentieth century. Ephemeris time is expressed accordingly in the conventional units of centuries, years, months, days, hours, minutes, and seconds. The numerical values of the ephemeris time and the universal time at the same instant differ only slightly. To avoid possible confusion, it is essential to indicate unambiguously which measure of time is being used.

The fundamental epoch from which ephemeris time is measured is the epoch that Newcomb designatd as 1900 January 0, Greenwich mean noon, but which is now properly designated as 1900 January 0, 12^h ET. The instant to which this designation is assigned is the instant near the beginning of the calen dar year A.D. 1900 when the geometric mean longitude of the Sun (referred to as the mean equinox of date) was $279^\circ\ 41'48\overset{s}{.}04$.

b. Universal time. Universal time is the precise measure of time used as the basis for all civil timekeeping. It conforms with a very close approximation to the mean diurnal motion of the Sun.

Since the introduction of Newcomb's Tables of the Sun (ref. 9), universal time has been defined as 12 hours plus the Greenwich hour angle of a point on the equator whose right ascension (measured from the mean equinox of date) is

$$R_u = 18^h\ 38^m\ 45\overset{s}{.}836 + 86\ 40184\overset{s}{.}542\ T_u + 0\overset{s}{.}0929\ T_u^2$$

where T_u is the number of Julian centuries of 36 525 days of universal time elapsed since the epoch of Greenwich mean noon (regarded as 12^h UT) on 1900 January 0. The expression for R_u is identical to that given by Newcomb for the right ascension of the fictitious mean Sun, with the exception that Newcomb used T instead of T_u and did not specify in what measure of time T was to be reckoned. Not recognizing the variable rotation of the Earth, Newcomb considered that T was measured in mean solar time applicable to orbital motions and to hour angles. Thus, Newcomb's T may now be identified with ephemeris time. The point on the equator whose right ascension is R_u is not identical with the "fictitious mean Sun" as defined by Newcomb. The right ascension of the fictitious mean Sun is

$$R_E = 18^h\ 38^m\ 45\overset{s}{.}836 + 86\ 40184\overset{s}{.}542\ T_E + 0\overset{s}{.}0929\ T_E^2$$

where T_E is the number of Julian centuries of 36 525 days of ephemeris time elapsed since the epoch of 12^h ET on 1900 January 0. R_E differs from R_u by 0.002738 ΔT where ΔT is the difference ET - UT.

The measure of universal time at time T_u, expressed in hours, minutes, and seconds, is

12^h + the Greenwich hour angle of the mean equinox of date = R_u.

The date expressed in the form of either a calendar date or a Julian date is that corresponding to the time T_u (sec. B.1, ref. 4).

The Greenwich hour angle of the mean equinox of date is Greenwich mean sidereal time by definition. At 12^h UT, the Greenwich mean sidereal time will therefore be R_u, which may now be described as "the mean sidereal time of 12^h UT." It may therefore be distinguished from the right ascension of the fictitious mean Sun.

Although universal time is no longer definable as "12^h + the Greenwich hour angle of the fictitious mean Sun," it is sufficiently close (compared with the deviation between the mean Sun and the true Sun) to justify the retention of the terms "mean solar time" and "mean solar day" in the sense in which they have been used in the past. The continued use of these descriptive terms is not to be regarded as identifying universal time with a precise measure of mean solar time. With this understanding, the danger of confusion is small. In this sense, universal time may be identified with Greenwich mean time.

c. Right ascension of Greenwich calculation. The right ascension of the Greenwich meridian (or the Greenwich apparent sidereal time (GAST)) is calculated from the following expression.

$$\text{GAST} = \text{GMST} + \text{equation of the equinoxes}$$

The Greenwich mean sidereal time (GMST) (or the right ascension of the Greenwich meridian relative to the mean equator and equinox of date) is calculated from Newcomb's expression according to the procedure used in the SVDS program, subroutine RPNMAT (ref. 5).

$$\text{Equation of the equinoxes} = \delta\psi \cos(\varepsilon + \delta\varepsilon)$$

where $\delta\psi$, $\delta\varepsilon$, and ε are given above under calculation of the nutation matrix.

Therefore,

$$\text{GAST} = \text{GMST} + \delta\psi \cos(\varepsilon + \delta\varepsilon)$$

Newcomb's expression uses universal time rather than ephemeris time. The Julian universal date (JUD) is

$$JUD = JED - EDT,$$

which corresponds to the Julian date (JD), described previously, and computed by the subroutine CDTJD. The Julian universal date of midnight (JUDM) is expressed as

$$JUDM = [JUD + 0.5] - 0.5$$

where $JUD + 0.5$ refers to the integer part of the expression. One-half day is subtracted to obtain the Julian universal day at midnight. Accordingly, the fractional part of the day past midnight is expressed as

$$\Delta t = JUD - JUDM$$

The fraction of a Julian century corresponding to the difference between the Julian universal date of midnight and the fundamental epoch ($JUDM - JUD_{1900}$) is computed by

$$TU = (JUDM - 2415020.0)/36525.0$$

The Greenwich mean sidereal time (GMST) at midnight is computed from the following expression.

$$\Omega_{GO} = 23925\overset{s}{.}836 + 8640184\overset{s}{.}542TU + 0.0929\overset{s}{.}TU^2 \text{ sec}$$

The Earth's rotational rate relative to the equinox is expressed as

$$\frac{d\Omega_{GO}}{dTU} = (8640184\overset{s}{.}542 + 0.1858\overset{s}{.}TU)/36525 \text{ sec/Julian day}$$

and

$$\dot{\omega}_{g_{ex}} = \left(86400 + \frac{d\Omega_{GO}}{dTU}\right) \text{ sec/Julian day}$$

The GMST corresponding to the given Julian universal date is

$$GMST_{JUD} = GMST_{JUDM} + \left(\dot{\omega}_{g_{ex}}\right) \Delta t$$

$$GMST_{JUD} = \Omega_{GO} + \left(\dot{\omega}_{g_{ex}}\right) \Delta t$$

$GMST_{JUD}$ is converted to seconds, and multiples of 2π are removed. The GAST, as previously defined, is computed as follows.

$$GAST = GMST + \delta\psi \cos(\varepsilon + \delta\varepsilon)$$

In order to obtain seven-digit single-precision accuracy, all calculations are performed in double precision and the results truncated to single precision. Table 5.1-III provides a list of mathematical symbols versus the internal code symbols for the right ascension of Greenwich meridian calculation.

5.1.4.8 Solar Coefficient Computations

The solar coefficient computations provide initialization data used by the analytic solar ephemeris routine. These data are a function of the selected base date. The quantities computed are

a. The eccentricity of the Earth's orbit (e)

b. The mean longitude of perigee (Γ)

c. The rate of the mean longitude of perigee ($\dot{\Gamma}$)

d. The mean anomaly (M)

e. The rate of the Earth in its orbit about the Sun ($\dot{M}$)

f. Coefficients for the series of the true anomaly as a function of the mean anomaly

g. The rotation matrix [RM] from the mean equinox and ecliptic of epoch to the true equator and Greenwich meridian of epoch

These quantities are described in section 5.6.

5.1.4.9 Lunar Coefficient Computations

The lunar coefficient computations provide initialization data used by the analytic solar ephemeris routine. These data are a function of the selected base date. This option is not implemented at the present time.

5.1.4.10 Output Logic

The output of the BASTM processor consists of the processor display, error messages (as necessary), and output to the AWA. The BASTM processor display is presented in table 4-III. The error messages are described in subroutines VALCK and EPHMC. The FDS-1 utility subroutine XPPUT is called to output the interface table array SESCON to the AWA. If the solar coefficients are computed, XPPUT is called again to output the interface table array SCOEF to the AWA.

5.1.5 Routine Input/Output Variables

The BASTM input/output variables are presented in table 5.1-IV.

5.1.6 Functional Logic Flow

The functional logic flow for BASTM is presented in figure 5.1-3.

5.1.7 Diagnostics and Debug

None.

5.1.8 Special Comments

None.

5.1.9 References

1. Flight Design System-1, System Design Document. JSC IN 77-FM-18, vol. VII, May 1977.

2. MPAD Program MDAS, Processor BASTIM, Feb. 1975.

3. FORTRAN IV Reference Manual. 2000 Series, Part No. 5951-1321. Hewlett-Packard Company, Dec. 1975.

4. Explanatory Supplement to the Astronomical Ephemeris and the American Ephemeris and Nautical Almanac. H. M. Nautical Almanac Office, London, 1961.

5. Geographic and Selenographic Coordinate Transformation Program. TRW Note No. 69-FMT-749, April 18, 1969.

6. Newcomb, S.: A Compendium of Spherical Astronomy. MacMillan Company, 1906.

7. SVDS Program Subroutine RPNMAT. June 16, 1977.

8. Woolard, Edgar W.: A Redevelopment of the Theory of Nutation. The Astronomical Journal, Feb. 1953.

9. Newcomb, S.: Tables of the Sun. A.P.A.E., vol. 6, 1895.

TABLE 5.1-I.- MATH SYMBOLS VERSUS CODE SYMBOLS

[PRECESSION CALCULATIONS]

Math symbol	Internal code symbol
JED	JED
P	XM50TD
θ	THETA
cos θ	COST
sin θ	SINT
T	T
T^2	T2
T^3	T3
z	Z
cos z	COSZ
sin z	SINZ
ζ_0	ZETAO
cos ζ_0	COSP
sin ζ_0	SINP

TABLE 5.1-II.- MATH SYMBOLS VERSUS CODE SYMBOLS

[NUTATION CALCULATIONS]

Math symbol	Internal code symbol	Math symbol	Internal code symbol
D	DLS	$\sin(\varepsilon + \delta\varepsilon)$	SINED
$\delta\psi$	DELC	F	F
$\cos \delta\psi$	COSDC	2F	DTWOF
$\sin \delta\psi$	SINDC	$2F + 2\Omega$	DTWOFL
$\delta\varepsilon$	DELEP	$2F + 2\Omega + 2D$	DANGLE
ε	EPSLN	JED	JED
$\cos \varepsilon$	COSEP	N	XMBTTD
$\sin \varepsilon$	SINEP	Ω	DOMEGL
$(\varepsilon + \delta\varepsilon)$	EPDE	T_{1900}	TE
$\cos(\varepsilon + \delta\varepsilon)$	COSED	T^2_{1900}	TE2
		T^3_{1900}	TE3

TABLE 5.1-III.- MATH SYMBOLS VERSUS CODE SYMBOLS

[RIGHT ASCENSION OF GREENWICH CALCULATIONS]

Math symbol	Internal code symbol	Math symbol	Internal code symbol
$\cos(\varepsilon + \delta\varepsilon)$	COSED	Ω_{GO}	OMGO
$\delta\psi$	DELC	$\frac{d\Omega_{GO}}{dTU}$	DOMGO
GAST	GAST	TU	TU
GMST	GMST	TU^2	TU2
JUD	JUD	$\dot{\omega}_{g_{ex}}$	WDGEX
JUDM	JUDM		

TABLE 5.1-IV.- ROUTINE INPUT/OUTPUT VARIABLES

Routine BASTM

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
GLOCON	--	Mix	I	--	IT	!!GLCN	Global constants array
YEAR	--	Intg	I	--	IT	YEAR	Year of base date
MONTH	--	Intg	I	--	IT	MONTH	Month of base date
DAY	--	Intg	I	--	IT	DAY	Day of base date
HOURS	--	Real	I	--	IT	HOURS	Hour of base time
MINS	--	Real	I	--	IT	MINS	Minute of base time
SECS	--	Real	I	--	IT	SECS	Second of base time
EPHEM	--	6CH	I	--	IT	EPHEM	Initialization flag
EDT	--	Real	I	--	IT	EDT	Ephemeris delta time
SESCON	--	Mix	O	--	IT	!SESCN	Session constants array
SCOEF	--	Real	O	--	IT	!SCOEF	Solar coefficient array
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.1-IV.- Concluded

Routine BASTM

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
COSEP	$\cos(\epsilon + \delta\epsilon)$	Dubl	O	—	C	COSEP	--
DBPI	π	Dubl	O	--	C	DBPI	--
DB2PI	2π	Dubl	O	--	C	DB2PI	--
GAST	GAST	Dubl	O	rad	C	GAST	Greenwich apparent sidereal time
NUMAT	[N]	Dubl	O	--	C	NUMAT	Nutation matrix
SINEP	$\sin\epsilon$	Dubl	O	—	C	SINEP	--
TE	T	Dubl	O	Julian centuries	C	TE	--
TE2	T^2	Dubl	O	(Julian centuries)^2	C	TE2	--
TE3	T^3	Dubl	O	(Julian centuries)^3	C	TE3	--

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

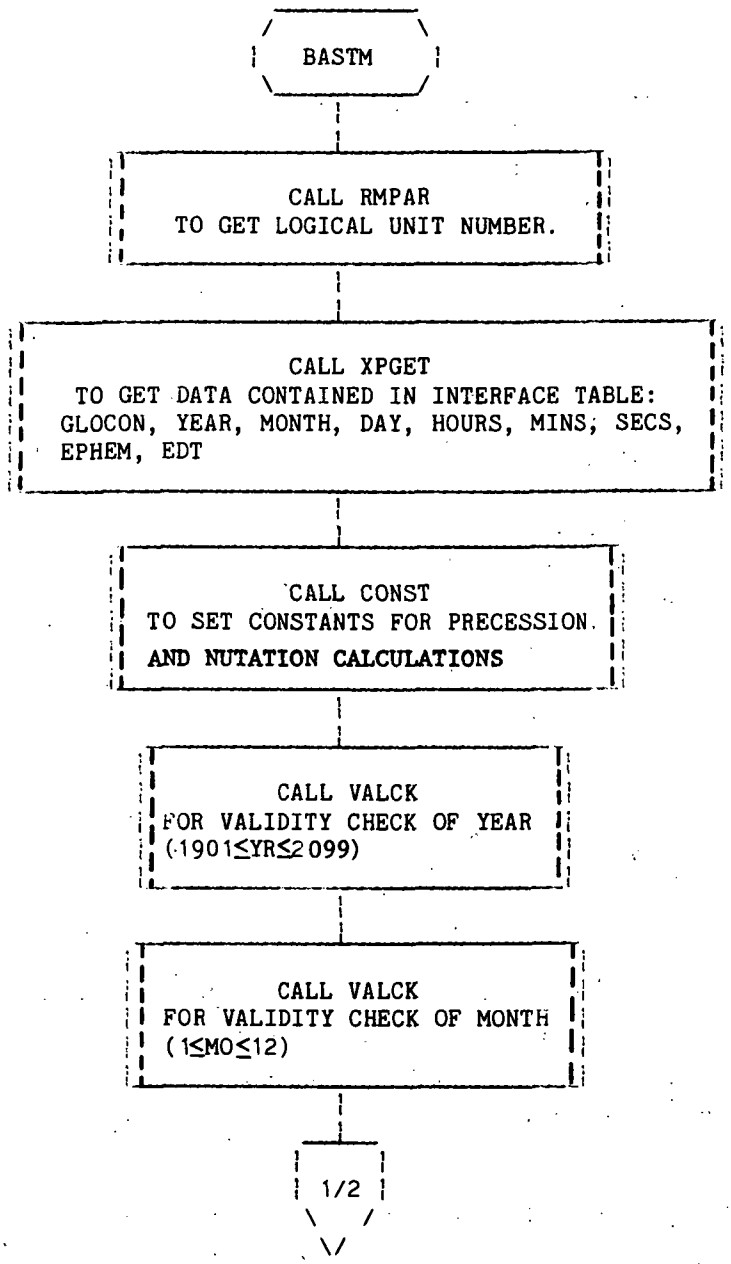


Figure 5.1-3.- BASTM functional logic flow.

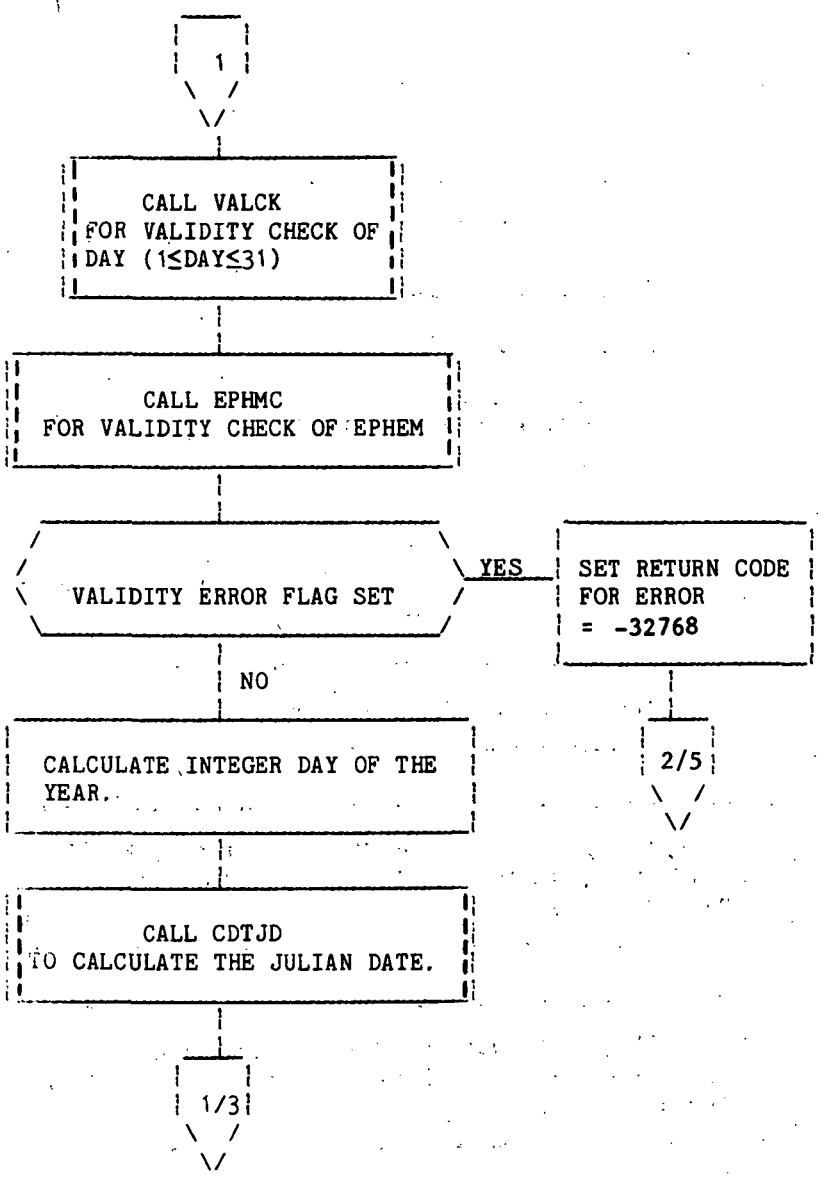


Figure 5.1-3.- Continued.

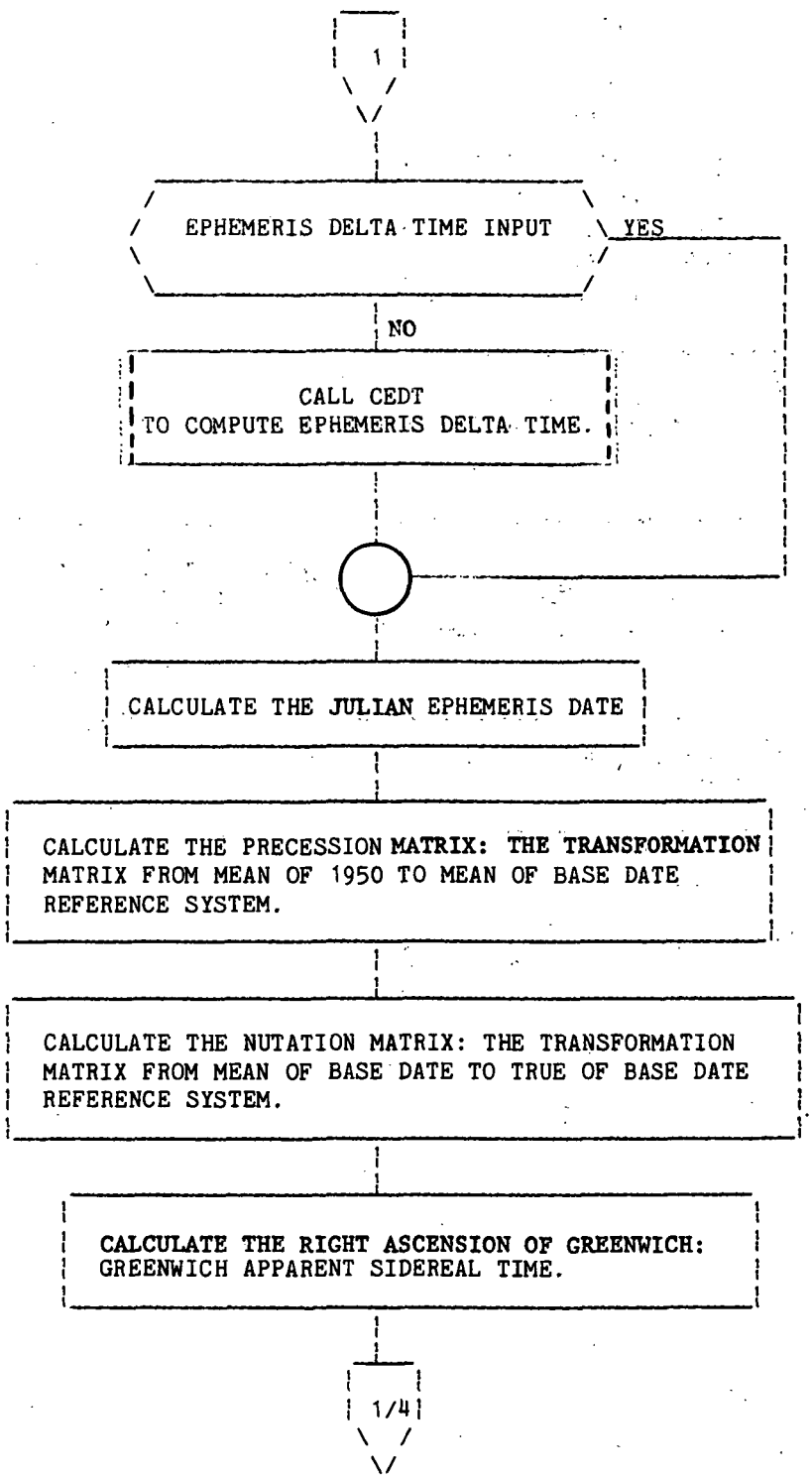


Figure 5.1-3.- Continued.

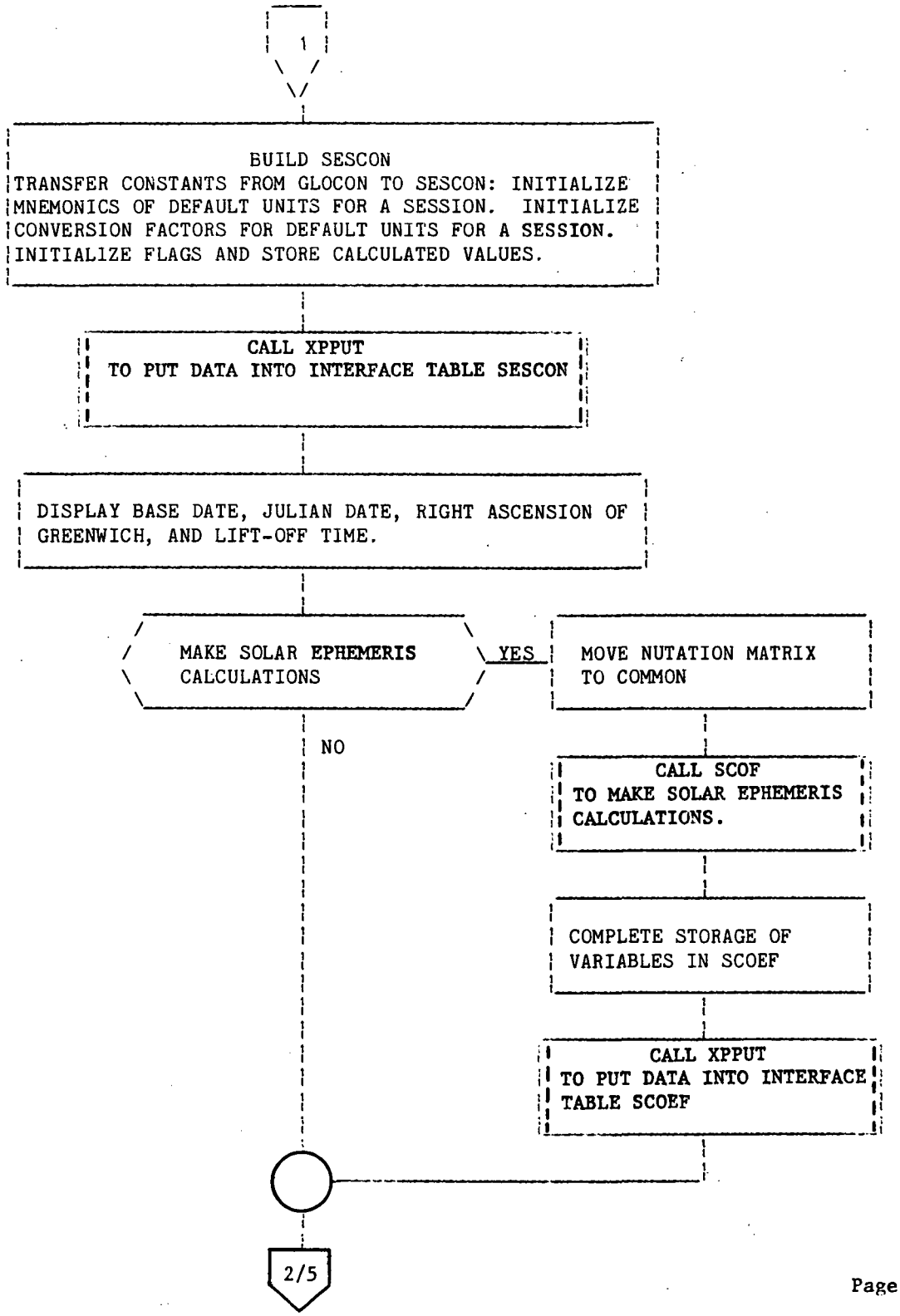


Figure 5.1-3.- Continued.

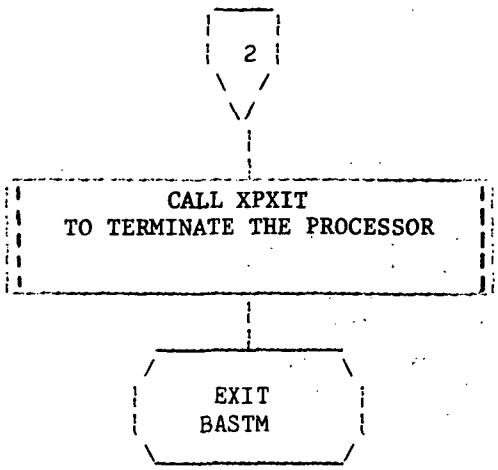


Figure 5.1-3.- Concluded.

5.2 ROUTINE NAME - CEDT

5.2.1 Purpose

The ephemeris delta time subroutine (CEDT) computes the ephemeris delta time.

5.2.2 Functional Description

CEDT converts the double-precision Julian date (DJD), which is input through the calling sequence to a relative Julian date by subtracting the Julian ephemeris date of 1200 (2 415 020.0) and converting it to single precision. With this relative Julian date, linear interpolation is performed using a table of relative Julian dates versus corresponding ephemeris delta time values. The ephemeris delta time value (EDT) results from this interpolation and is output through the calling sequence. DATA statements are used to define the values of the arrays containing the relative Julian dates and the corresponding ephemeris delta times.

5.2.3 Assumptions and Limitations

In CEDT, the relationship between the Julian date and the ephemeris delta time is assumed to be linear. The values for the Julian date and the ephemeris delta time arrays are taken from the data available in reference 1. These data are available through January 1, 1978, making it necessary to linearly extend the curve for dates of interest beyond 1978. Figure 5.2-1 presents a comparison of the results of subroutine CEDT (ref. 1) and the SVDS subroutine XDATE.

5.2.4 Method

CEDT uses the ephemeris delta time model data given in table 5.2-I for linear interpolation to the base date of interest. The Julian dates, calendar dates, and EDT values are taken from reference 1. The method used is as follows:

$$JD_{cur} = JD - 2415020.0$$

For JD_{cur} < first table value of JD($JD_{cur} < JD_{Tab1}$):

$$EDT = EDT_{Tab1} + [EDT_{Tab2} - EDT_{Tab1}][JD_{cur} - JD_{Tab1}]/[JD_{Tab2} - JD_{Tab1}]$$

For JD_{cur} > last table value ($JD_{cur} > JD_{Tab}$ last):

$$EDT = EDT_{Tab\ last-1} + \left[EDT_{Tab\ last} - EDT_{Tab\ last-1}\right]$$
$$\left[\overline{JD}_{cur} - JD_{Tab\ last-1}\right] / \left[JD_{Tab\ last} - JD_{Tab\ last-1}\right]$$

For first table value $\leq JD_{cur} \leq$ last table value ($JD_i \leq JD_{cur} \leq JD_{i+1}$):

$$EDT = EDT_{Tab\ i} + \left[EDT_{Tab\ i+1} - EDT_{Tab\ i}\right]$$
$$\left[JD_{cur} - JD_{Tab\ i}\right] / \left[JD_{Tab\ i+1} - JD_{Tab\ i}\right]$$

Table 5.2-II provides a list of mathematical symbols versus internal code symbols.

5.2.5 Routine Input/Output Variables

The CEDT input/output variables are presented in table 5.2-III. The calling sequence is

```
CALL CEDT (DJD,EDT)
```

5.2.6 Functional Logic Flow

The functional logic flow for CEDT is presented in figure 5.2-2.

5.2.7 Diagnostics and Debug

None.

5.2.8 Special Comments

None.

5.2.9 References

1. 1977 American Ephemeris and Nautical Almanac, Reduction of Time Scales, 1977.

2. What Time Is It? JSC IN 72-FM-59, March 7, 1972.

3. Chi, A. R.; and Fosque, H. S.: A Step In Time. IEEE Spectrum, 1972.

TABLE 5.2-I.- FDS EDT MODEL DATA - ROUTINE CEDT

Julian date	Calendar date	EDT (sec)	Relative Julian date
2 437 126.5	Jan. 1, 1966	36.54	24 106.5
2 440 587.5	Jan. 1, 1970	40.18	25 567.5
2 441 317.5	Jan. 1, 1972	42.22	26 297.5
2 442 048.5	Jan. 1, 1974	44.48	27 028.5
2 443 509.5	Jan. 1, 1978	48.50	28 489.5

TABLE 5.2-II.- MATH SYMBOLS VERSUS INTERNAL CODE SYMBOLS

Math symbol	Internal code symbol	Math symbol	Internal code symbol
EDT	EDT	JD_{Tab}	RJD
EDT_{Tab}	TIME	last	MAX
JD	DJD	last-1	MAXM1
JD_{cur}	JDCUR	i	I

TABLE 5.2-III.- ROUTINE INPUT/OUTPUT VARIABLES

Routine CEDT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DJD	JD	Dubl	I	days	A	DJD	Julian date
EDT	EDT	Real	O	sec	A	EDT	Ephemeris delta time

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory

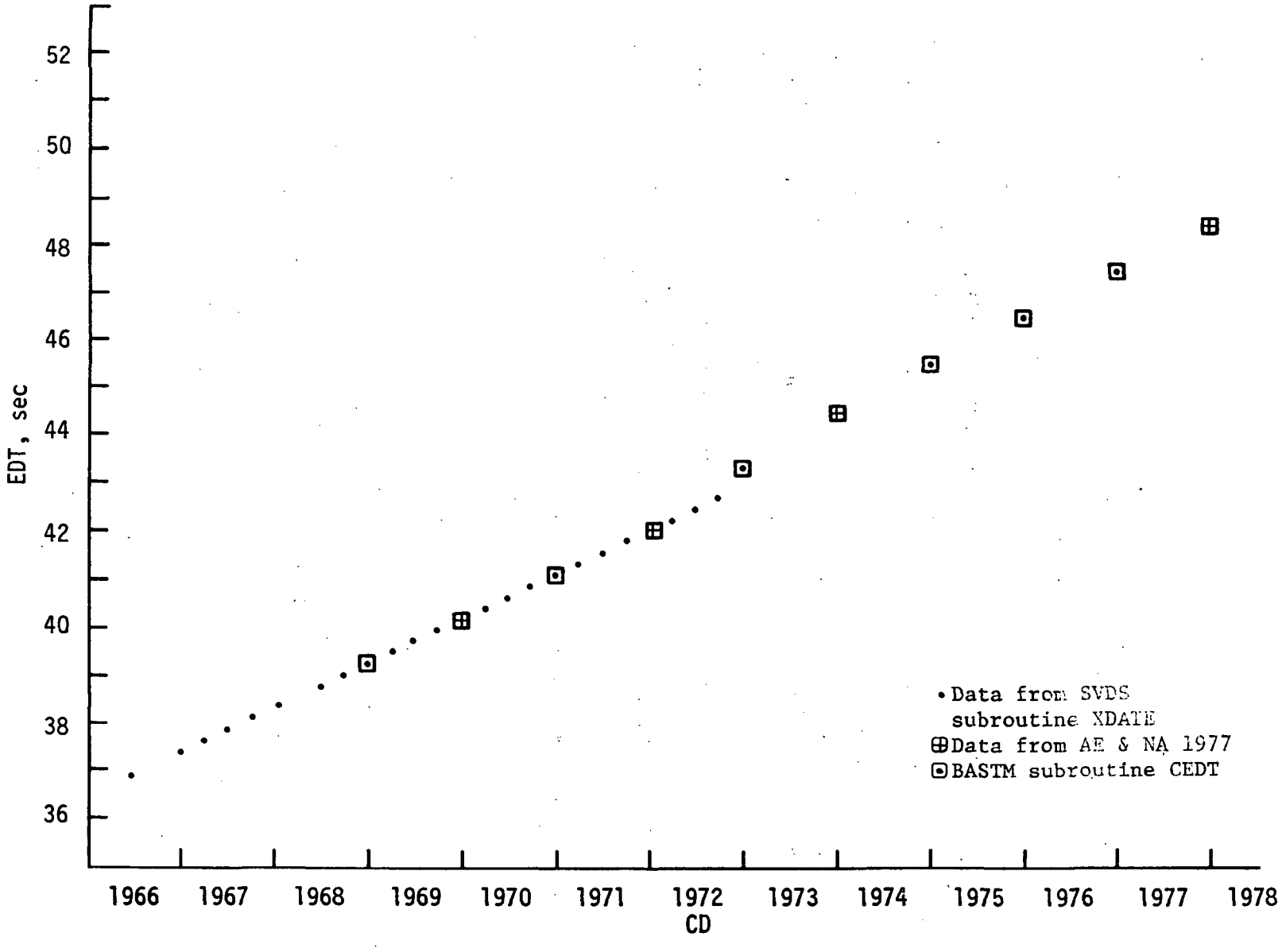


Figure 5.2-1.- Comparison of results of subroutine CEDT and SVDS subroutine XDATE.

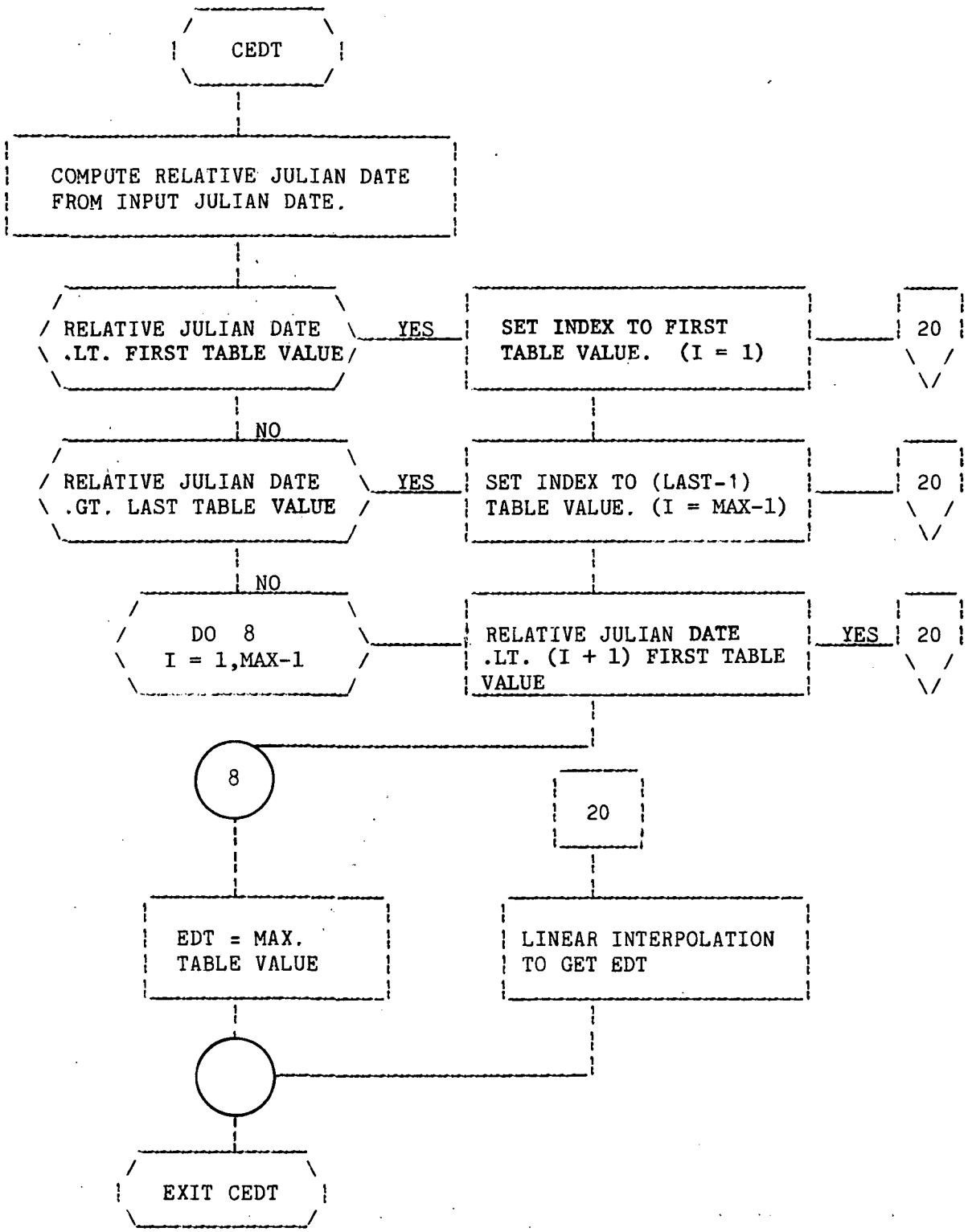


Figure 5.2-2.- CEDT functional logic flow.

5.3 ROUTINE NAME - CONST

5.3.1 Purpose

The constant initialization subroutine (CONST) initializes the constants that are used in the calculation of the precession and nutation matrices. The constants for the calculation of the time elapsed in tropical centuries from the beginning of the Besselian year 1950 to the date of interest are also initialized.

5.3.2 Functional Description

The constants that are contained in DATA statements are transferred to the output arrays using the FDS-1 system subroutine XRMOV. All coefficients are stored as double precision.

5.3.3 Assumptions and Limitations

CONST provides constants for four terms in the calculation of the nutation in longitude, $\delta\psi$, and provides for two terms in the calculation of the nutation in obliquity, $\delta\varepsilon$.

5.3.4 Method

CONST stores the coefficients for the precession matrix calculations in locations 1 through 9 of the array P, which is output through the calling sequence. P(10) is set equal to the Julian ephemeris date of 1950 and P(11) is set equal to the number of days in a tropical century. The subroutine stores the coefficients for the nutation matrix calculations in the array N, which is output through the calling sequence. DATA statements are used to define the values of coefficients.

5.3.5 Routine Input/Output Variables

The CONST input/output variables are presented in table 5.3-I. The calling sequence is

```
CALL CONST (N,P)
```

5.3.6 Functional Logic Flow

None.

5.3.7 Diagnostics and Debug

None.

5.3.8 Special Comments

The coefficients used in the precession matrix and nutation matrix calculations are collected into one routine, CONST, for ease of reference and updating (refs. 1 and 2).

5.3.9 References

1. Precession Constants: SVDS Program, Subroutine RPNMAT, June 16, 1977.

2. Nutation Constants: Woolard, Edgar W.: A Redevelopment of the Theory of Nutation. The Astronomical Journal, Volume 58, No. 1, Feb. 1953.

TABLE 5.3-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine CONST

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
P(1) - P(3)	—	Dubl	O	sec	A	P(1) - P(3)	ζ_0 for precession matrix
P(4) - P(6)	--	Dubl	O	sec	A	P(4) - P(6)	Z for precession matrix
P(7) - P(9)	--	Dubl	O	sec	A	P(7) - P(9)	θ for precession matrix
P(10)	—	Dubl	O	—	A	P(10)	Julian ephemeris date of 1950
P(11)	--	Dubl	O	days	A	P(11)	Days per tropical century
N(1) - N(4)	--	Dubl	O	sec	A	N(1) - N(4)	ϵ for nutation matrix
N(5) - N(8)	--	Dubl	O	sec	A	N(5) - N(8)	For four-term $\delta\psi$ calculations (nutation)
N(9) - N(12)	--	Dubl	O	sec	A	N(9) - N(10)	For two-term $\delta\epsilon$ calculations (nutation)
N(11) - N(14)	--	Dubl	O	deg	A	N(11) - N(14)	For F calculations (nutation)

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.3-I.- Concluded

Routine CONST

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
N(15) - N(18)	--	Dubl	O	deg	A	N(15) - N(18)	For D calculations (nutation)
N(19) - N(20)	--	Dubl	O	deg	A	N(19) - N(20)	For Ω calculations (nutation)

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

5.4 ROUTINE NAME - CDTJD

5.4.1 Purpose

The calendar date to Julian date subroutine (CDTJD) computes the Julian date corresponding to the base date of interest.

5.4.2 Functional Description

Astronomical days, beginning at Greenwich noon, are numbered consecutively from an epoch sufficiently far in the past to precede the historical period. The number assigned to a day in this continuous count is the Julian day number, which is defined to be zero for the day starting at Greenwich mean noon on January 1, 4713 B.C., Julian proleptic calendar. The Julian date (JD) corresponding to any instant is the Julian day number followed by the fraction of the day elapsed since the preceding noon.

5.4.3 Assumptions and Limitations

CDTJD is applicable within the following bounds.

a. YEAR - 1900 to 2099

b. MONTH - 1 to 12

c. DAY - 1 to 31

d. HOUR - 0 to 24

e. MINUTE - 0 to 60

f. SECONDS - 0 to 60

5.4.4 Method

CDTJD computes the number of days elapsed since January 1, 1900, to the date of interest and adds this to the number of days elapsed from Julian day zero to 1900. The fractional part of the day is computed from the hours, minutes, and seconds, and is added to the number of days to form the Julian date (refs. 1 and 2).

The number of complete days since December 31, 1899 is

$$D = 365(K - 1900) + \frac{K - 1901}{4} + \left[\begin{array}{l}\text{The number of days}\\ \text{to the first of the}\\ \text{month}\end{array}\right] + J$$

where K is the year and J is the day of the month. If the year is a leap year, and the month is greater than 2, add 1 day to D. Since the time is measured from noon GMT, 12 hours are subtracted from the days since December 31, 1899.

$$JD = D + 2\ 415\ 020.0 + \frac{L - 12}{24} + \frac{M}{1440} + \frac{S}{86400}$$

where 2 415 020.0 is the Julian date of noon, December 31, 1899.

Table 5.4-I provides a list of the mathematical symbols versus internal code symbols.

5.4.5 Routine Input/Output Variables

The CDTJD input/output variables are presented in table 5.4-II. The calling sequence is

```
CALL CDTJD (IYEAR,IDAYS,HOURS,MINS,SECS,DJL)
```

5.4.6 Functional Logic Flow

None.

5.4.7 Diagnostics and Debug

None.

5.4.8 Special Comments

None.

5.4.9 References

1. A Satellite Orbit Prediction Program (KSFAST). JSC IN 76-FM-42, July 30, 1976.

2. Explanatory Supplement to the Astronomical Ephemeris and the American Ephemeris and Nautical Almanac. H. M. Nautical Almanac Office, London, 1961.

TABLE 5.4-I.- MATH SYMBOLS VERSUS CODE SYMBOLS
[CDTJD SUBROUTINE]

Math symbol	Internal code symbol
D	J
JD	DJL
K	K
L	HOURS
M	MINS
S	SECS

TABLE 5.4-II.- ROUTINE INPUT/OUTPUT VARIABLES

Routine CDTJD

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IYEAR	--	Intg	I	--	A	YEAR	Year
IDAYS	--	Intg	I	days	A	IDAYS	Day of year
HOURS	--	Real	I	hours	A	0.0	Hours
MINS	--	Real	I	mins	A	0.0	Minutes
SECS	--	Real	I	secs	A	0.0	Seconds
DJL	--	Dubl	O	days	A	DJD	Julian date
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

5.5 ROUTINE NAME - VALCK

5.5.1 Purpose

The validity check subroutine (VALCK) checks the range of an input value and returns the results of the check.

5.5.2 Functional Description

VALCK tests the value to be checked (VAL) against an upper and lower range (RANGE1, RANG2). If the value is outside the range, the error flag (ERRFLG) is set to 1 and output through the calling sequence. When the value is outside the range, an error message is displayed identifying the out-of-range variable and gives its value. The processor error message number is 1.

5.5.3 Assumptions and Limitations

Within the subroutine VALCK, the value to be checked and the upper and lower valid ranges are defined as integers. The value to be checked is alphanumerically identified by the array TYPE, which is input through the calling sequence. TYPE may contain a maximum of six characters. Because of the word size of the HP21MX computer, only integer values may be checked by this routine.

5.5.4 Method

None.

5.5.5 Routine Input/Output Variables

Table 5.5-I contains the description of the input/output variables. The calling sequence is

```
CALL VALCK(LU,TYPE,VAL,RANG1,RANG2,ERRFLG)
```

5.5.6 Functional Logic Flow

The functional logic flow for VALCK is presented in figure 5.5-1.

5.5.7 Diagnostics and Debug

None.

5.5.8 Special Comments

None.

5.5.9 References

None.

TABLE 5.5-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine VALCK

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
LU	—	Intg	I	—	A	LU	Logical unit number of user terminal
TYPE	--	Intg	I	—	A	--	Six character identifier of VAL
VAL	—	Intg	I	—	A	--	Value to be checked
RANG1	--	Intg	I	—	A	--	Lower valid limit of value
RANG2	—	Intg	I	—	A	--	Upper valid limit of value
ERRFLG	--	Intg	O	—	A	ERRCK	Contains results of validity check

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory

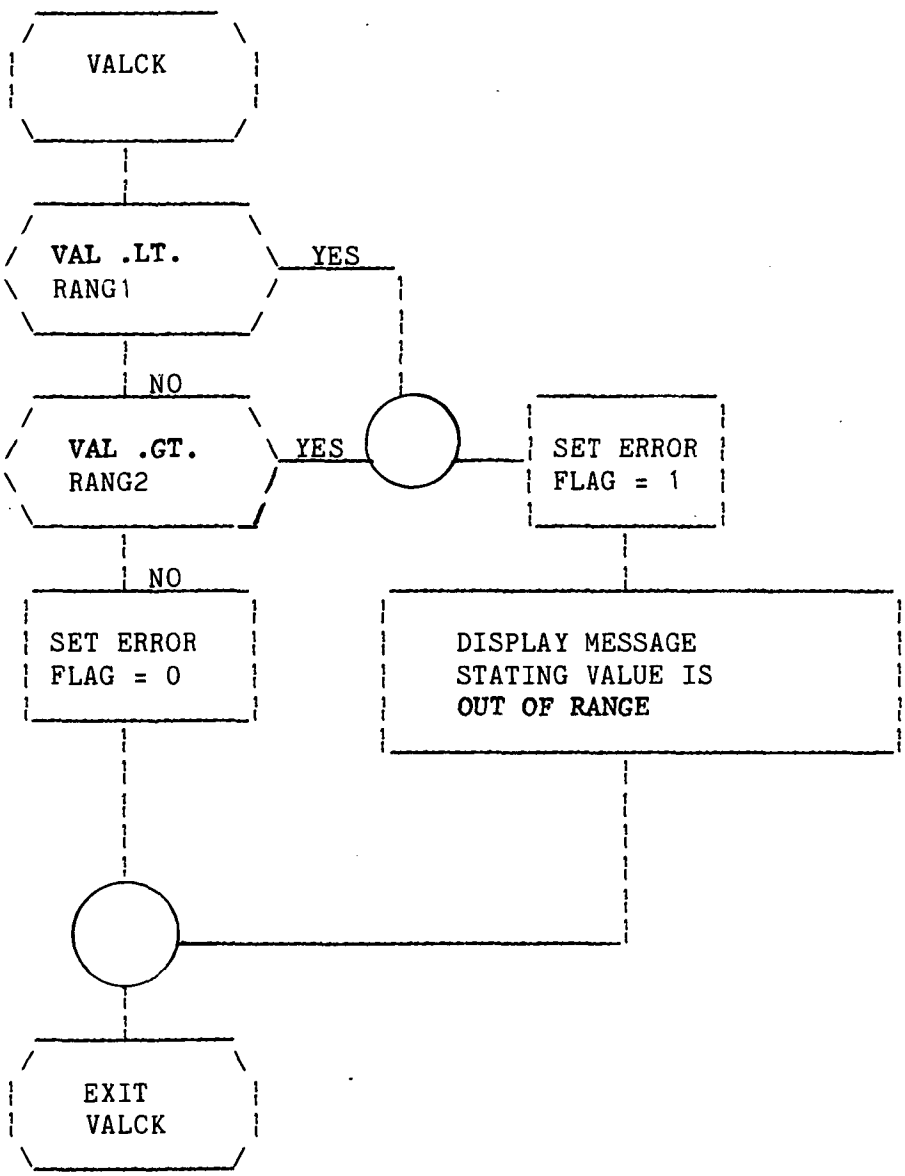


Figure 5.5-1.- VALCK functional logic flow.

5.6 ROUTINE NAME - SCOF

5.6.1 Purpose

The SCOF routine initializes the interface table array SCOEF to make the solar ephemeris coefficients available, as needed, by processors utilizing the analytical solar ephemeris routine.

5.6.2 Functional Description

The SCOF routine performs calculations necessary to initialize the interface table array SCOEF, as defined in table 1.2-II(b) of reference 1. The calculations required in SCOF are

a. The eccentricity of the Earth's orbit (e)

b. The mean Longitude of perigee (Γ)

c. The rate of the mean longitude of perigee ($\dot{\Gamma}$)

d. The mean anomaly (M)

e. The rate of the Earth in its orbit about the Sun ($\dot{M}$)

f. Coefficients for series of true anomaly as a function of mean anomaly

g. The rotation matrix RM from the mean equinox and ecliptic of epoch to the true equator and Greenwich meridian of epoch

The semimajor axis of the Earth's orbit is initially set in a DATA statement and transferred to the SCOEF array.

5.6.3 Assumptions and Limitations

None.

5.6.4 Method

The method used in subroutine SCOF to compute each of the calculated parameters (ref. 2) listed in the Functional Description section is described in this section. All constants involved in the calculations are stored as double precision, and all calculations are performed in double precision. Any quantities needed in subroutine SCOF, which have been calculated previously in either the main program BASTM or in another subroutine, are supplied by COMMON. A description of each calculation follows.

The expression to compute the eccentricity (e) of the Earth's orbit (ref. 3) is

$$e = 0.01675104 - 0.00004180T_{1900} - 0.000000126T_{1900}^2$$

Table 5.6-I provides a list of the mathematical symbols versus internal code symbols for the eccentricity calculation.

T_{1900} is the time elapsed in Julian centuries from noon ephemeris time on the day before January 1, 1900 to the date of interest, and is calculated in BASTM and supplied through COMMON.

The method used in computing the angles Γ and M, and the rates $\dot{\Gamma}$ and $\dot{M}$, is to make the calculations using the expressions as given below. The angles Γ and M are converted from degrees to radians, and multiples of 2π are removed. The rates Γ and M are converted from degrees per Julian century to radians per second.

Mean Longitude of Perigee (Γ) and Rate ($\dot{\Gamma}$) of the Mean Longitude of Perigee (ref. 3):

$$\Gamma = 281^\circ 13' 15.0'' + 6189.03''T_{1900} + 1.63''T_{1900}^2 + 0.012''T_{1900}^3$$

$$\dot{\Gamma} = 6189.03'' + 3.26''T_{1900} + 0.036''T_{1900}^2$$

Mean Anomaly (M) and Rate ($\dot{M}$) of the Earth in Its Orbit About the Sun (ref. 3):

$$M = 358^\circ 28' 33.04'' + 129596579.10''T_{1900} - 0.54''T_{1900}^2 - 0.012''T_{1900}^3$$

$$\dot{M} = 129596579.10'' - 1.08''T_{1900} - 0.036''T_{1900}^2$$

The coefficients in the expressions for Γ, $\dot{\Gamma}$, M, and $\dot{M}$ were converted to degrees and were coded as follows:

$$\Gamma = 281.220833333^\circ + 1.7191750^\circ T_{1900} + .4527777777778^\circ \times 10^{-3}T_{1900}^2$$

$$+ .333333333333 \times 10^{-5}T_{1900}^3$$

$$\dot{\Gamma} = 1.7191750^\circ + .905555555556^\circ \times 10^{-3}T_{1900} + 0.1^\circ \times 10^{-4}T_{1900}^2$$

$$M = 358.475844444^{\circ} + 35999.049750^{\circ}T_{1900} - 0.15^{\circ} \times 10^{-3}T_{1900}^2$$

$$- .333333333333^{\circ} \times 10^{-5}T_{1900}^3$$

$$\dot{M} = 35999.049750 - 0.3 \times 10^{-3}T_{1900} - 0.1^{\circ} \times 10^{-4}T_{1900}^2$$

Table 5.6-II provides a list of the mathematical symbols versus internal code symbols for the computation of Γ, $\dot{\Gamma}$, M, and $\dot{M}$.

Coefficients for True Anomaly Minus Mean Anomaly (ref. 4):

The coefficients for the calculation of true anomaly minus mean anomaly are computed and stored in the output array SCOEF. The coefficients are used in the analytic solar ephemeris routine for the calculation of θ; the true anomaly minus mean anomaly. The expression for θ is given for reference.

$$\theta = M + 2e \sin M + \frac{5}{4}e^2 \sin 2M + \frac{e^3}{12}(13 \sin 3M - 3 \sin M) + \text{----}$$

$$\theta = M + e(2 - \frac{1}{4}e^2) \sin M + \frac{5}{4}e^2 \sin 2M + \frac{13}{12}e^3 \sin 3M$$

The terms in the expression for θ, which are computed and stored in SCOEF, are

$$RJ1 = e(2 - 1/4\ e^2)$$

$$RJ2 = 5/4\ e^2$$

$$RJ3 = 13/12\ e^3$$

Table 5.6-III provides a list of the mathematical symbols versus internal code symbols for the computation of RJ1, RJ2, RJ3.

Rotation Matrix RM for Mean Equinox and Ecliptic to True Equator and Greenwich Meridian of Epoch:

$$RM = \Omega_G \quad N \quad \epsilon$$

where

$[\epsilon]$ = Rotation matrix from mean equinox and ecliptic of date to the mean equinox and equator of date

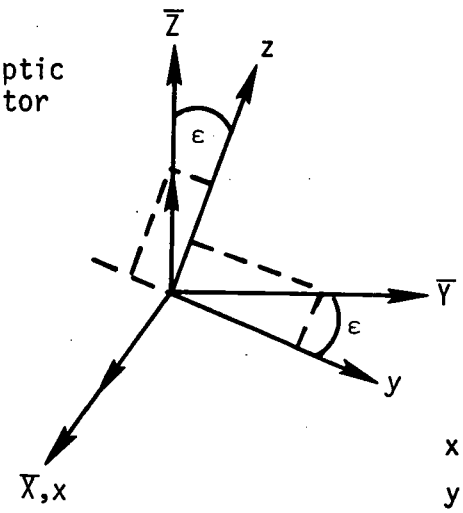
$[N]$ = Rotation matrix (nutation) from mean equinox and equator of date to the equinox and equinox of date

$[\Omega_G]$ = Rotation matrix from true equinox and equator of date to true equator and Greenwich meridian of date

The nutation matrix N is calculated in the main program BASTM, and is supplied to subroutine SCOF by the COMMON variable NUMAT.

OBLIQUITY

$\overline{X}$, $\overline{Y}$, $\overline{Z}$ = Mean equinox/ecliptic
x, y, z = Mean equinox/equator



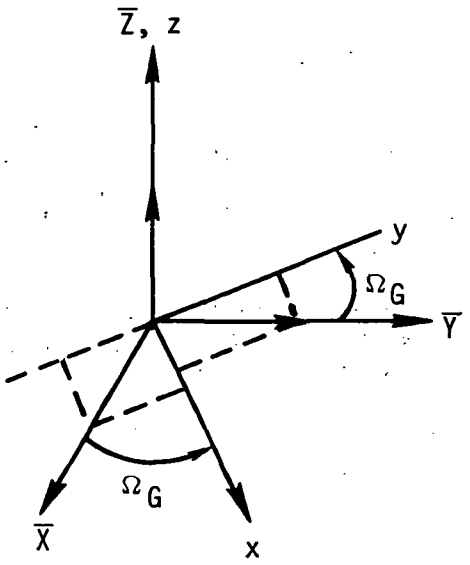
$$x = \overline{X}$$
$$y = \overline{Y} \cos \epsilon - \overline{Z} \sin \epsilon$$
$$z = \overline{Y} \sin \epsilon \; \overline{Z} \cos \epsilon$$

$$[\epsilon] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \epsilon & -\sin \epsilon \\ 0 & \sin \epsilon & \cos \epsilon \end{bmatrix}$$

where ϵ is computed in BASTM and supplied to subroutine SCOF by COMMON. ϵ is the mean obliquity of the ecliptic defined as the angle between the ecliptic of date and the mean celestial equator of date. Table 5.6-IV provides a list of mathematical symbols versus internal code symbols.

GREENWICH ROTATION

$\overline{X}$, $\overline{Y}$, $\overline{Z}$ = True equinox/equater of date
x, y, z = True equator/Greenwich of date



$$x = \overline{X} \cos \Omega_G + \overline{Y} \sin \Omega_G$$
$$y = -\overline{X} \sin \Omega_G + \overline{Y} \cos \Omega_G$$
$$z = \overline{Z}$$

$$[\Omega_G] = \begin{bmatrix} \text{Cos } \Omega_G & \text{sin } \Omega_G & 0 \\ \text{-Sin } \Omega_G & \text{cos } \Omega_G & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where Ω_G = true right ascension of Greenwich of base date (GAST). GAST is calculated in the main program BASTM and supplied to subroutine SCOF through COMMON.

Table 5.6-V provides a list of the mathematical symbols versus internal code symbols for the computation of $[\Omega_G]$.

5.6.5 Routine Input/Output Variables

The SCOF input/output variables are presented in table 5.6-VI. The calling sequence is

```
CALL SCOF (SCOEF, LU)
```

5.6.6 Functional Logic Flow

The functional logic flow for SCOF is presented in figure 5.6-1.

5.6.7 Diagnostics and Debug

None.

5.6.8 Special Comments

None.

5.6.9 References

1. Flight Design System-1, System Design Document. JSC Internal Note No. 77-FM-18, volume VI, revision 1, January 1978.

2. Trajectory Prediction Parameters for AAP, Space Station/Space Shuttle, and Interplanetary Missions. MSC Internal Note No. 70-FM-33, March 5, 1970.

3. Explanatory Supplement to the Astronomical Ephemeris and the American Ephemeris and Nautical Almanac. Prepared jointly by the Nautical Almanac Offices of the United Kingdom and the United States of America, 1961.

4. Moulton, F. R.: An Introduction to Celestial Mechanics. The MacMillan Company, 1960.

TABLE 5.6-I.- MATH SYMBOLS VERSUS CODE SYMBOLS

[ECCENTRICITY CALCULATION]

Math symbol	Internal code symbol
e	ECC
T_{1900}	TE
T^2_{1900}	TE2

TABLE 5.6-II.- MATH SYMBOLS VERSUS CODE SYMBOLS

Math symbol	Internal code symbol	Math symbol	Internal code symbol
Γ	GAMMA	T_{1900}	TE
$\dot{\Gamma}$	GAMAD	T^2_{1900}	TE2
M	M	T^3_{1900}	TE3
$\dot{M}$	MDOT		

TABLE 5.6-III.- MATH SYMBOLS VERSUS CODE SYMBOLS

Math symbol	Internal code symbol	Math symbol	Internal code symbol
e	ECC	RJ1	SCOEF(8)
e^2	ECC2	RJ2	SCOEF(9)
e^3	ECC3	RJ3	SCOEF(10)

TABLE 5.6-IV.- MATH SYMBOLS VERSUS INTERNAL CODE SYMBOLS

Math symbol	Internal code symbol
cos ϵ	COSEP
sin ϵ	SINEP
$[\epsilon]$	EPMAT

TABLE 5.6-V.- MATH SYMBOLS VERSUS INTERNAL CODE SYMBOLS

Math symbol	Internal code symbol	Math symbol	Internal code symbol
Ω_G	GAST	sin Ω_G	SINOG
cos Ω_G	COSOG	$[\Omega_G]$	OMMAT

TABLE 5.6-VI.- ROUTINE INPUT/OUTPUT VARIABLES

Routine SCOF

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
COSEP	$\cos \epsilon$	Dubl	I	--	C	COSEP	--
DPI	π	Dubl	I	--	C	DPI	--
DTWOPI	2π	Dubl	I	--	C	DTWOPI	--
GAST	Ω_G	Dubl	I	rad	C	GAST	True right ascension of Greenwich of base date
NUMAT	[N]	Real	I	--	C	NUMAT	Rotation matrix (nutation) from mean equinox and equator of date to true equinox and equinox of date.
SCOEF	--	Real	O	--	A	SCOEF	Interface table array
SINEP	$\sin \epsilon$	Dubl	I	--	C	SINEP	--
TE	T_{1900}	Dubl	I	--	C	TE	Elapsed time
TE2	T^2_{1900}	Dubl	I	--	C	TE2	Elapsed time squared
TE3	T^3_{1900}	Dubl	I	--	C	TE3	Elapsed time cubed

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

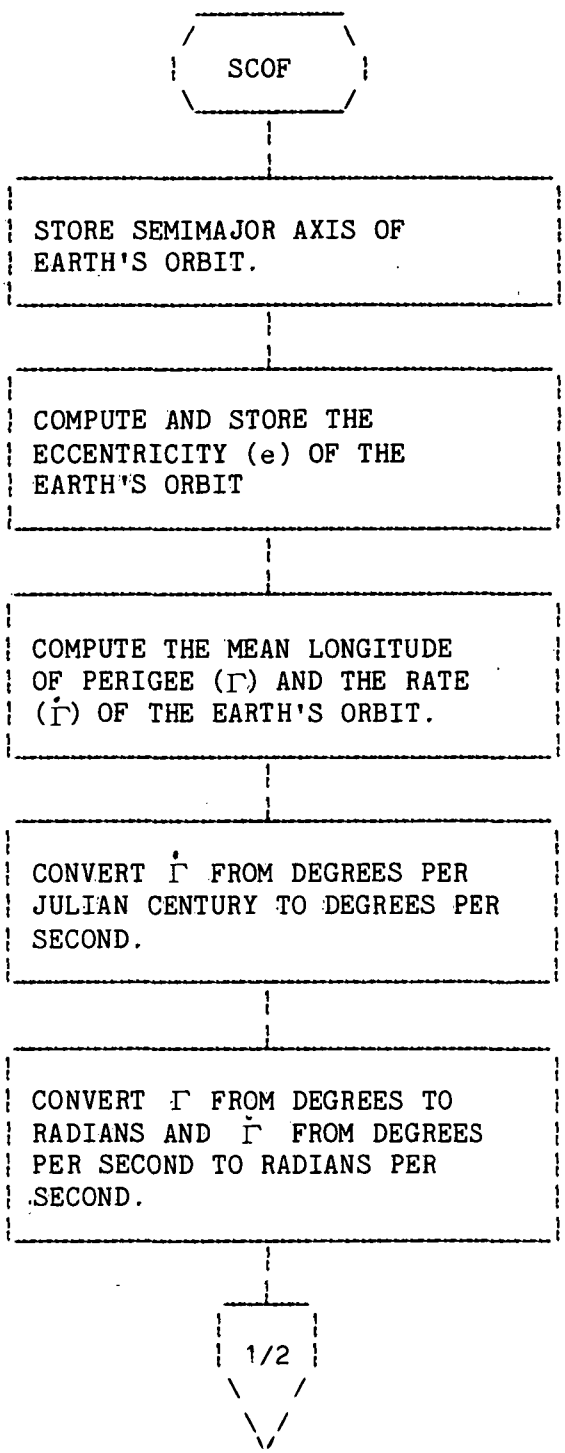


Figure 5.6-1.- SCOF functional logic flow.

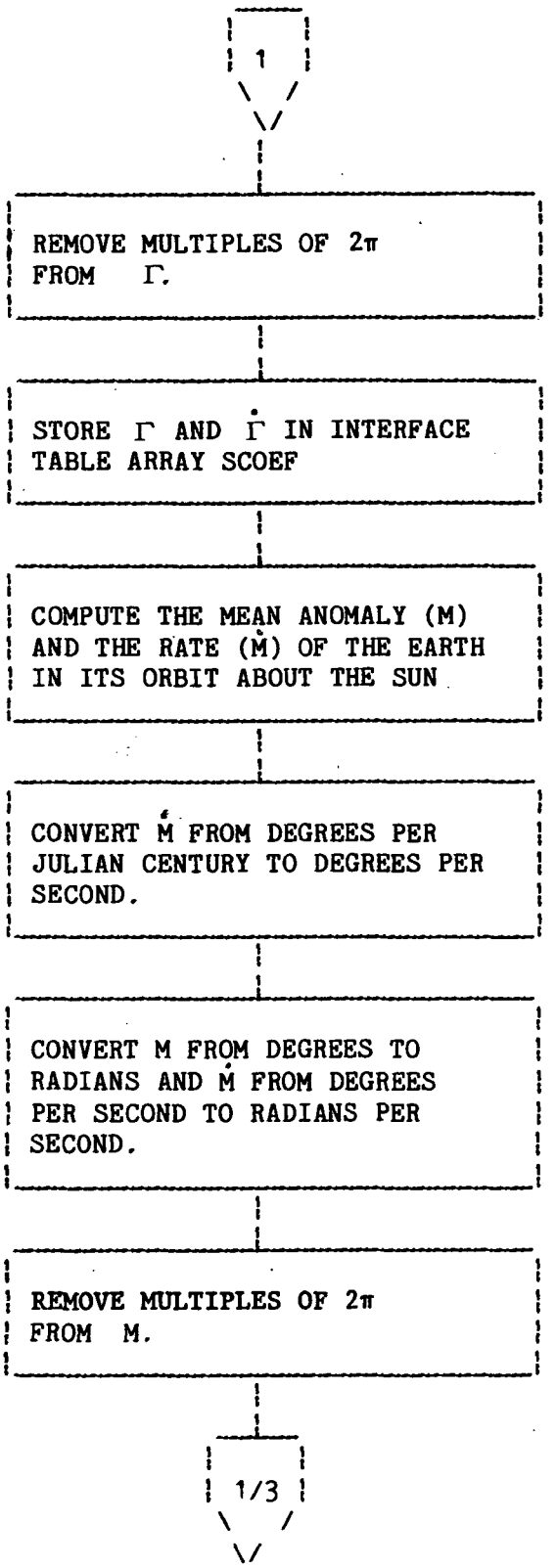


Figure 5.6-1.- Continued.

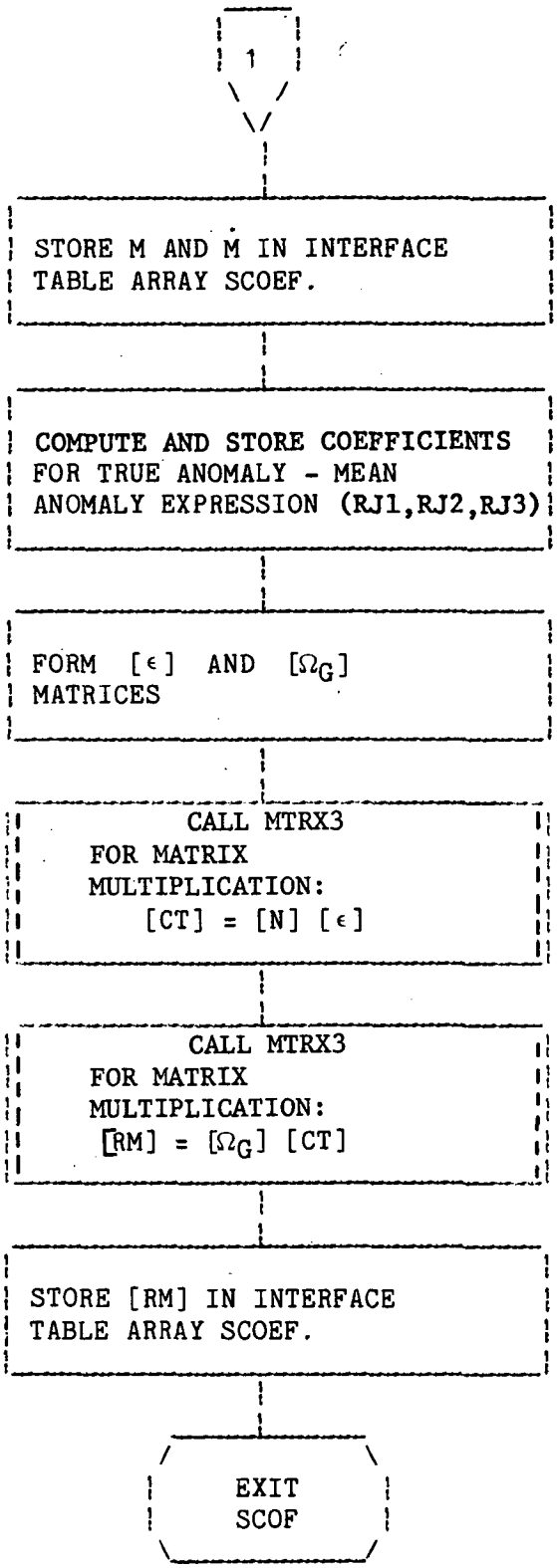


Figure 5.6-1.- Concluded.

5.7 ROUTINE NAME - EPHMC

5.7.1 Purpose

The ephemeris check subroutine (EPHMC) checks the interface table variable EPHEM for valid and implemented ephemeris calculation options and sets the flags EPFLG and ERRFLG accordingly.

5.7.2 Functional Description

EPHMC initializes the error flag to zero, which is the no error condition. It also initializes to zero the flag that indicates the type(s) of ephemeris calculations that are to be made. The interface table variable EPHEM may contain two six-character alphanumeric options indicating the ephemeris calculations, if any, that are to be made. The possible contents and meanings of EPHEM are

```
EPHEM(1) = "SUN"   - Output SCOEF
         = "Ø"     - Don't output SCOEF

EPHEM(2) = "MOON"  - Output LCOEF
         = "Ø"     - Don't output LCOEF
```

The contents of the variable EPHEM are checked for all blanks. If all blanks are found, control is returned to the main program. If the contents of EPHEM are "SUN", the flag is set to indicate that solar ephemeris calculations are to be made, and control is returned to the main program. If the contents of EPHEM are "MOON" (or any combination of options containing "MOON"), the message "OPTION NOT IMPLEMENTED" and the contents of EPHEM are displayed. The error flag is set to nonzero and control is returned to the main program. If EPHEM contains options other than blanks, SUN, or MOON, the message "INPUT PARAMETER INVALID" and the contents of EPHEM are displayed. The error flag is set to nonzero and control is returned to the main program.

5.7.3 Assumptions and Limitations

None.

5.7.4 Method

None.

5.7.5 Routine Input/Output Variables

The EPHMC input/output variables are presented in table 5.7-I.

The subroutine calling sequence is

```
CALL EPHMC (LU,EPHEM,EPFLG,ERRFLG)
```

5.7.6 Functional Logic Flow

The functional logic flow for EPHMC is presented in figure 5.7-1.

5.7.7 Diagnostics and Debug

None.

5.7.8 Special Comments

None.

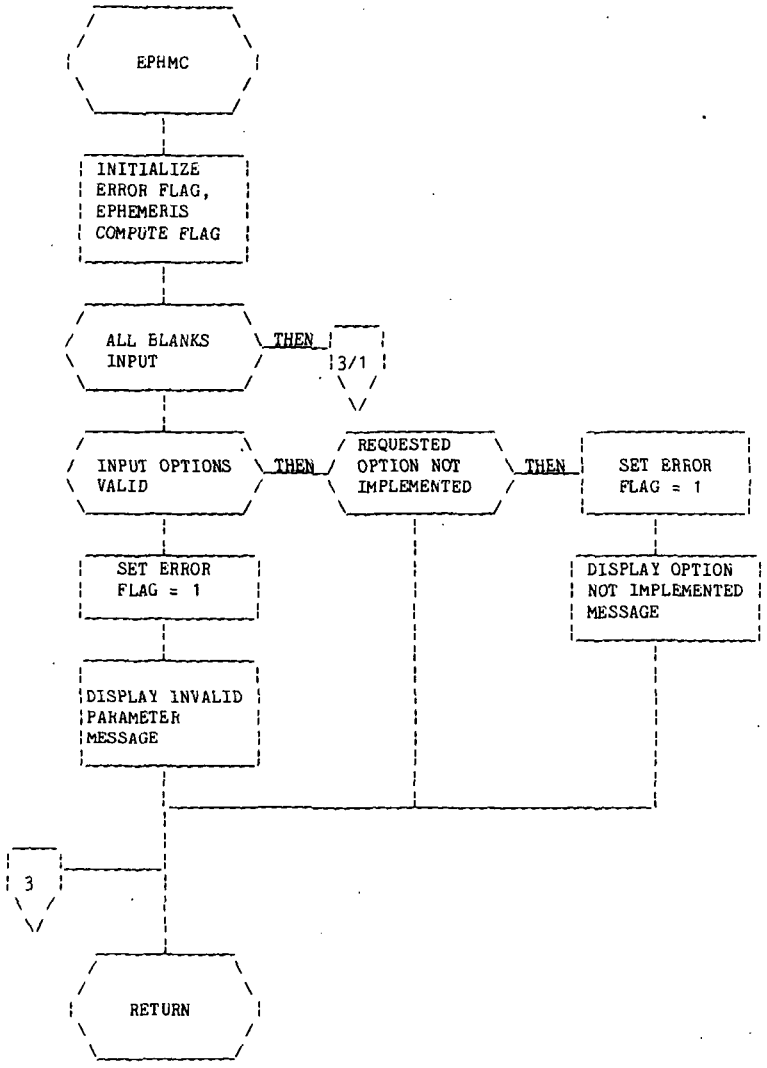
5.7.9 References

None.

TABLE 5.7-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine EPHMC

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
LU	--	Intg	I	—	A	LU	Logical unit of user's terminal
EPHEM	--	Intg	I	—	IT	EPHEM	Ephemeris options - six characters per option
EPFLG	--	Intg	O	--	A	EPFLG	Ephemeris calculation flag
ERRFLG	--	Intg	O	—	A	ERRCK	Error flag for valid option check
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory



Page 1 of 1

Figure 5.7-1.- EPHMC functional logic flow.

CONSUMABLES ANALYSIS FOR SHUTTLE KITS PROCESSOR (CASKU)

1.0 PURPOSE

The CASKU applications processor provides a complete and detailed analysis of a mission activities time line for all nonpropulsive consumables and provides systems constraint data. The consumables considered are oxygen, hydrogen, nitrogen, ammonia, waste water, potable water, and electrical energy. Constraint data are provided for the Electrical Power System (EPS) and the Environmental Control System (ECS).

2.0 FUNCTIONAL DESCRIPTION

After receiving the user provided input from the interface table, the CASKU processor will convert the data from the disk files into the proper form for internal use. The files used are products of the ABLKS and SYDUS programs and contain activity load block information for both the Orbiter and payload activities and the system source characteristics data. This information, along with the mission activities time-line data, is used to build an electrical power profile and a heat load profile for analysis.

The EPS portion of the processor evaluates the power profile to determine oxygen and hydrogen requirements. The power profile is developed by accumulating the power requirements for each activity at each time point on the mission activities time line. After the power requirements have been established at a point on the time line, they are evaluated by using a distribution network model. This network is defined by node and branch numbers representing the desired electrical power distribution. These nodes and branches are used along with the power requirements and system source characteristics to develop a set of nodal equations. The resulting matrix is then solved for the voltages at each node in the network. From the voltages the branch and source currents are then calculated. The current of each source representing a fuel cell is then used to calculate the oxygen and hydrogen usage rates. The voltages, currents, and fuel cell power are compared to known limits, and any constraints are flagged and a constraint record is written to the DRDE data file.

The ECS portion of the processor evaluates the heat load profile for cooling requirements and the cabin for atmospheric requirements. The heat load profile is developed in the same manner as the power profile. The electrical loads are based on the best dissipation for 28-volt loads, and the profile is then adjusted for resistive-type equipment using the actual voltages, as determined by the power profile, for each point on the time line. This adjusted heat load profile is then evaluated using a model of the Orbiter cooling system.

The model of the cooling system represents the cabin, the water loops, and the Freon loop. Each of the loops are modeled using algorithms representing cold plates, evaporators, heat exchangers, and other cooling system devices. The heat loads are then placed onto the specified coolant loops and evaluated for each point on the time line determining the requirements for water and ammonia. The model of the cabin is then evaluated using algorithms that take into

consideration gas leakage and a gas regulation system to determine oxygen and nitrogen requirements. Valves calculated are compared with known limits and any constraints are flagged, and a constraint record is written to the DRDE data file.

Waste water production is determined using an algorithm that uses the metabolic rate of the crew for each point on the time line.

A requirements record is written to the DRDE data file at the conclusion of the evaluation of each time-line point until all points have been evaluated. Detailed printouts of the results of either the EPS or ECS solutions, or both at any specified interval, are optional to the user.

3.0 ASSUMPTIONS AND LIMITATIONS

It is assumed that the user is familiar with the files used to support the execution of the processor.

4.0 PROCESSOR INPUT/OUTPUT

a. Processor interface table - The definition of the processor interface table parameters is provided in table 4-I.

   Specific constants required by the processor will be maintained through the default values stored in the processor constants array PROCON. Specifically, these constants are the cartridge reference numbers for the systems files and DRDE files, the number of blocks required by the output data file, and the device used to print the processor displays. The user may specify the device desired by entering the device number into PROCON(4).

   The user specifies the time reference for the displays by entering "GE" for ground elapsed time or "GM" for Greenwich mean time into TIMREF. Values for LHOUR, LYEAR, and LDAY are defaulted to corresponding values in the !SESCON array for use with this processor; therefore, the !SESCON array must be pres ent in the AWA before the processor is executed. This array is established by first executing the BASTM processor.

   The names of the time line DRDE file and the output DRDE file are specified by entering the four character names into the parameters TIMFIL and DATAFL.

   The support disk file names for the Orbiter activity data, payload activity data, and systems characteristics data are entered into the parameters ORBACT, PAYACT, and SYSFIL, respectively. When no payload activity data file is used, blanks must be entered for the name.

   A YE or NO entry is required for the ECSPRT and EPSPRT parameters, which specifies the desire for detailed displays for the Environmental Control System (ECS) or Electrical Power System (EPS) solutions at each point on the mission activities time line. The parameter EVALIN also requires a YE or NO entry to specify that an interval of time for the displays is being

specified. A NO entry will default to displays at each point in the time line if either ECSPRT or EPSPRT are YE.

INTST and INTSTP parameters are used to give the start and stop times of the interval if one is specified by EVALIN. Four zeros should be entered if the entry for EVALIN was NO. The values entered for INTST and INTSTP may be in either GET or GMT. For GET, the first position should be zero for time at or after lift-off time, or a number less than zero for times prior to lift-off (i.e., 0.,10.,30.,25., or -1.,0.,10.,5.). The second, third, and fourth positions are the hour, minute, and second for the interval to start or stop. For GMT the first position will be the Julian calendar day of the year (i.e., June 9, 1978 = 160th day of the year). The other positions are the same as for GET.

The user, through the parameter CONTNU, specifies the desire to continue processing when an activity block number cannot be found in the data arrays.

b. Interface table data array definitions - The definition of the CASKU processor constants data array is provided in table 4-II.

c. Interface table data file definitions - The definitions of the input and output DRDE files used by the CASKU processor are provided in table 4-III.

d. Processor solicited (prompted) inputs - None.

e. Processor displays and display parameter definitions - CASKU generates two displays. The format of the environmental network status is shown in table 4-IV(a), and a definition of the display parameters is provided in table 4-IV(b). The format of the distribution network solution is shown in table 4-IV(c), and a definition of the display parameters is provided in table 4-IV(d).

f. Processor message table - The CASKU error messages are provided in table 4-V.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-VI.

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR CASKU

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
TIMREF	AWA	2CH	I	1	1		Time reference base "GE" ground elapsed time (GET) "GM" Greenwich mean time (GMT)
DATAFL	Disk	Intg	O	—	—		Name of the DRDE output file to be created
TIMFIL	Disk	Intg	I	—	—		Name of the user's TIMELINE file
ORBACT	AWA	6CH	I	3	1		Name of the Orbiter activity data file
PAYACT	AWA	6CH	I	3	1		Name of the PAYLOAD activity data file
SYSFIL	AWA	6CH	I	3	1		Name of the SYSTEMS CHARACTERISTICS data file
PROCON	AWA	Intg	I	4	4	16, 20, 450, 0	LU of files, number of blocks in the DRDE output file, and the output device for displays
ECSPRT	AWA	2CH	I	1	1		User's request for detailed ECS network displays "YE" or "NO"
EPSPRT	AWA	2CH	I	1	1		User's request for detailed EPS distribution displays "YE" or "NO"

NOTES

CLASS: Symb, AWA, Disk

TYPE: Free 2CH 72CH; Intg 6CH Mix; Real 18CH; Dubl 36CH

USE: I = Input; O = Output; I/O = Input/Output

TABLE 4-I.- Concluded

PROCESSOR CASKU

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
EVALIN	AWA	2CH	I	1	1		User's desire for an evaluation interval to be established "YE" or "NO"
INTST	AWA	Free	I	8	4		Array containing either a + or - sign on the Julian calendar days and the hours, minutes, and seconds that the evaluation interval starts
INTSTP	AWA	Free	I	8	4		Array containing either a + or - sign on the Julian calendar days and the hours, minutes, and seconds that the evaluation interval stops
LYEAR	AWA	Intg	I	1	1	!SESCN (3)	Launch year
LDAY	AWA	Intg	I	1	1	!SESCN (6)	Launch day
LHOUR	AWA	Real	I	2	1	!SESCN (11)	Launch hour
CONTNU	AWA	2CH	I	1	1		User's desire to continue processing if an activity cannot be found in the activity data

NOTES

CLASS
Symb
AWA
Disk

TYPE
Free 2CH 72CH
Intg 6CH Mix
Real 18CH
Dubl 36CH

USE
I = Input
O = Output
I/O = Input/Output

TABLE 4-II.- INTERFACE TABLE DATA ARRAY DEFINITIONS

PROCESSOR CASKU

Array name	Index location	Default value	Definition
PROCON	1	16	Cartridge reference no. for systems resident disk files.
	2	20	Cartridge reference no. for DRDE files
	3	450	Number of blocks to allocate for the output DRDE file.
	4	0	Output device, =0 defaults to the LU in use.

TABLE 4-III.- INTERFACE TABLE DATA FILE DEFINITIONS

PROCESSOR CASKU

DRDE DATA FILE /XXXXC

(User's time-line file)

Record number	Integer word allocations	Content and definition
1	1 - 3	"TMLNU" - Name of the FDS-1 processor that created the file
	4 - 6	TIMFIL - Interface table variable name through which the file was created
	7 - 9	"TMLNU" - Name of the FDS-1 processor that last updated the file
	10 - 12	TIMFIL - Interface table variable name through which the file was changed
2 - n	1	MACT - Activity number
	2 - 3	STIME - Activity start time
	4 - 5	ETIME - Activity stop time

TABLE 4-III.- Continued

PROCESSOR CASKU

DRDE DATA FILE /XXXXC

(Output file)

Record number	Integer word allocations	Content and definition
1	1 - 3	"CASKU" - Name of the FDS-1 processor that created the file
	4 - 6	DATAFL - Interface table variable name through which the file was created
	7 - 9	"CASKU" - Name of the FDS-1 processor that last updated the file
	10 - 12	DATAFL - Interface table variable name through which the file was changed
2	1 - 5	JTIME - Year, Julian day, hour, minute, and second that the file was generated
	6 - 8	TIMFIL - User's time-line file name
	9 - 11	ORBACT - Orbiter activity data file name
	12 - 14	PAYACT - Payload activity data file name
	15 - 17	SYSFIL - System characteristics data file name
	18 - 20	PROGNM - Program name
3	1 - 2	RTIME - Time that a system constraint limit was exceeded
	3 - 4	KIND - Name of the system that caused the constraint
	5	NODE - Number of the system
	6 - 7	TEMP - The temperature that exceeded the system limit
	8 - 9	TLMT - The limit that was exceeded

TABLE 4-III.- Continued

PROCESSOR CASKU

DRDE DATA FILE /XXXXC

(Output file)

Record number	Integer word allocations	Content and definition
4	1 - 2	RTIME - Time that a system constraint limit was exceeded
	3 - 4	KIND - Name of the system that caused the constraint
	5	NODE - Number of the system
	6 - 7	VAL - The value that exceeded the system limit
	8 - 9	VALLMT - The limit that was exceeded
5	1 - 2	TIME - The time of the activity for which the data was computed
	3 - 14	QTY - Quantity values for O2, H2, N2, NH3, waste H20, and potable H20
	15 - 22	PWR - Power values for connected load bus load, source power, and payload
	23 - 34	SRCVLT - Source power voltages
	35 - 46	SRCCUR - Source power currents
	47 - 58	HEAT - Heat values for total, radiator ASC evaporator, topping evaporator ammonia boiler, and cooling inlet
	59 - 66	PRS - Pressure values for O2 and N2 in two cabins
	67 - 68	PTEMP - Plate temperature
	69 - 76	FTEMP - Cooling system temperatures
	77 - 78	PEAK - Statistical peak power value
	79 - 80	PMIN - Statistical minimum power value
	81 - 82	THMCAP - System cooling capacity
	83 - 84	TPMRGN - System cooling margin

TABLE 4-III.- Concluded

PROCESSOR CASKU

DRDE DATA FILE ?CAS1C

(Scratch file)

Record number	Integer word allocations	Content and definition
1	1 - 2	BUF - The start or stop time of an activity JBUF - Activity number corresponding to the time in BUF

TABLE 4-IV.- PROCESSOR DISPLAY AND DISPLAY PARAMETER DEFINITIONS TABLE

(a) Environmental network status display

PROCESSOR CASKU

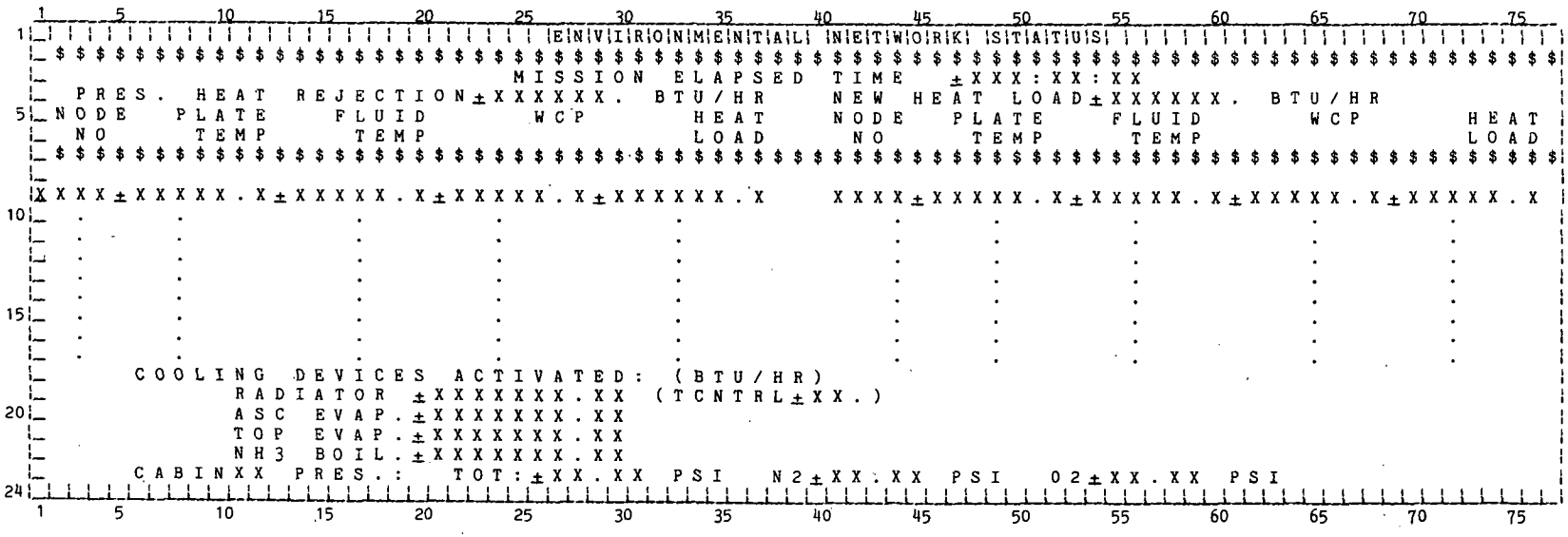


TABLE 4-IV.- Continued

(a) Concluded

PROCESSOR CASKU

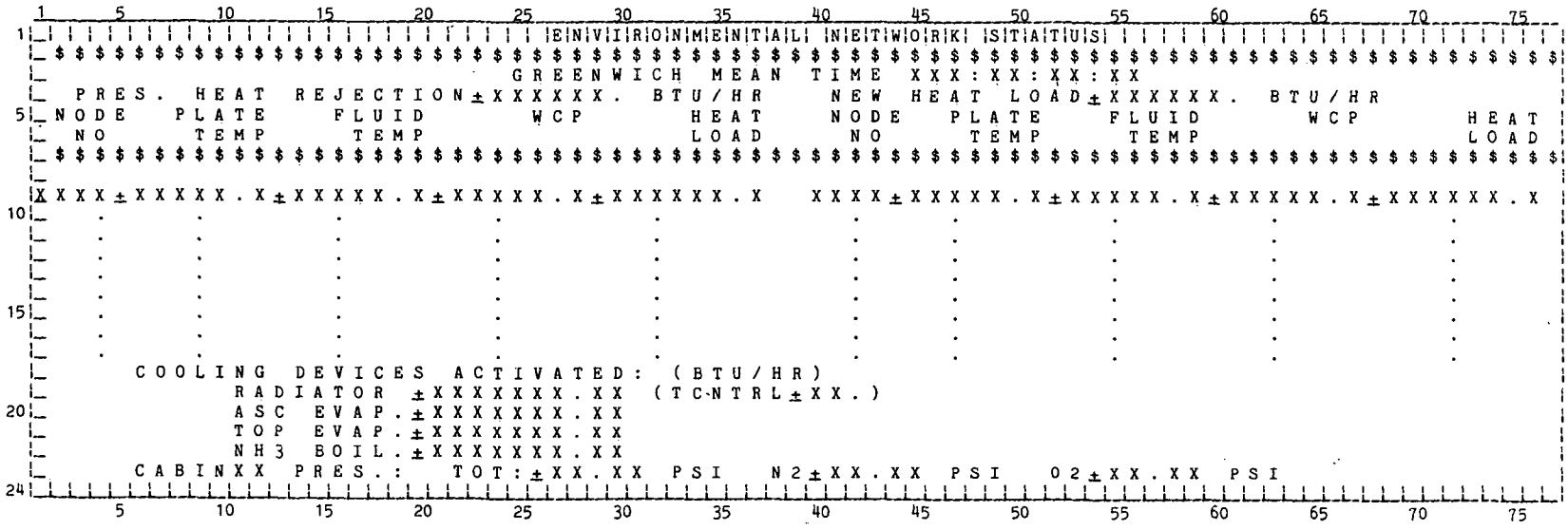


TABLE 4-IV.- Continued

(b) Display parameter definition table for the environmental network status display

PROCESSOR CASKU

ENVIRONMENTAL NETWORK STATUS	
Display parameter label	Parameter definition
MISSION ELAPSED TIME	GET of the table in ± hours, minutes, and seconds (± XXX:XX:XX) or GMT of the table in Julian calendar days, hours, minutes, and seconds (XXX:XX:XX:XX)
PRES HEAT REJECTION	The present rate of heat rejection (Btu/hr)
NEW HEAT LOAD	New heat load on the system at the time of the display (Btu/hr)
NODE NO.	System position number
PLATE TEMP	Plate temperature at the time of the display for the node number (°F)
FLUID TEMP	Fluid temperature at the time of the display for the node number (°F)
WCP	Fluid heat capacity (Btu)
HEAT LOAD	Total heat load at the time of the display for the node number (Btu/hr)
RADIATOR	Heat load for the radiator at the time of the display (Btu/hr)
TCNTRL	Radiator temperature control switch setting at the time of the display (°F)
ASC EVAP	Ascent evaporator heat load at the time of the display (Btu/hr)

TABLE 4-IV.- Continued

(b) Concluded

PROCESSOR CASKU

ENVIRONMENTAL NETWORK STATUS	
Display parameter label	Parameter definition
TOP EVAP	Topping evaporator heat load at the time of the display (Btu/hr)
NH3 BOIL	Ammonia boiler heat load at the time of the display (Btu/hr)
CABIN	Cabin number
TOT	Total cabin pressure at the time of the display (psi)
N2	Partial pressure of nitrogen at the time of the display (psi)
O2	Partial pressure of oxygen at the time of the display (psi)

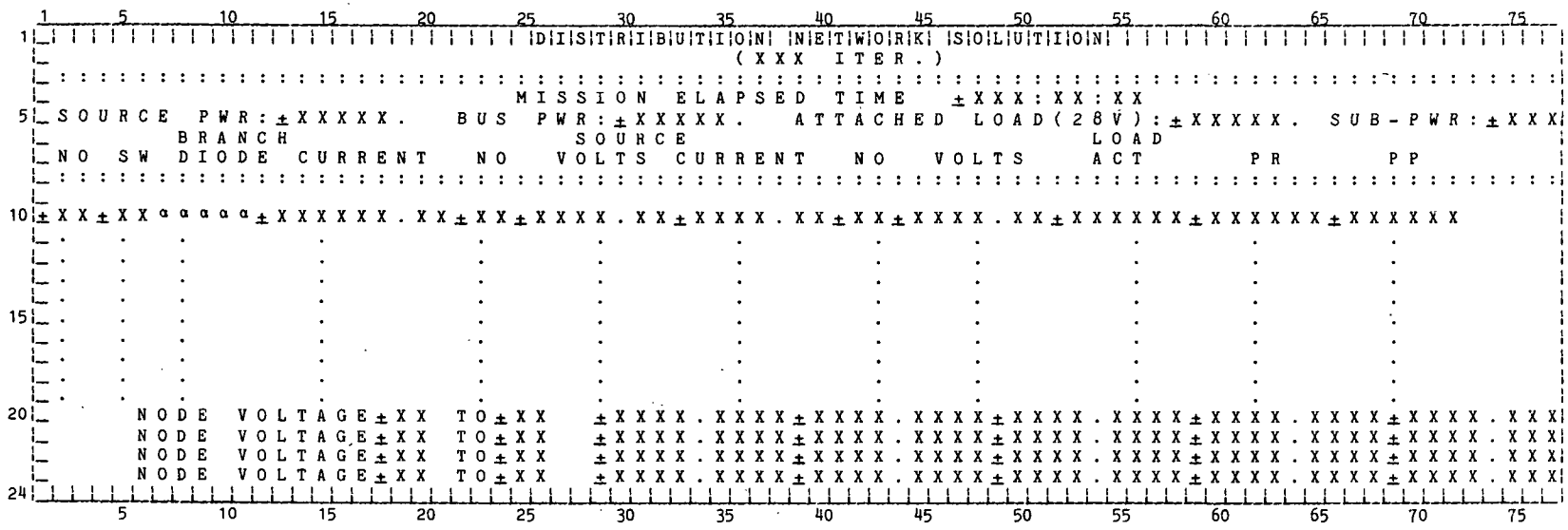


TABLE 4-IV.- Continued

(c) Distribution network solution display

PROCESSOR CASKU

```
    1   5    10   15   20   25   30   35   40   45   50   55   60   65   70   75
 1 |||||||||||||||||||||||||||DISTRIBUTION NETWORK SOLUTION|||||||||||||||||||||||
   |                                   (XXX ITER.)
   | ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
   |                        MISSION ELAPSED TIME  ±XXX:XX:XX
 5 | SOURCE PWR:±XXXXX.  BUS PWR:±XXXXX.  ATTACHED LOAD(28V):±XXXXX. SUB-PWR:±XXX
   |       BRANCH              SOURCE                    LOAD
   | NO SW DIODE CURRENT  NO  VOLTS CURRENT  NO  VOLTS   ACT     PR     PP
   | ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
   |
10 |±XX±XXααααα±XXXXXX.XX±XX±XXXX.XX±XXXX.XX±XX±XXXX.XX±XXXXXX±XXXXXX±XXXXXX
   | .  .  .      .       .     .      .      .    .       .     .      .
   | .  .  .      .       .     .      .      .    .       .     .      .
   | .  .  .      .       .     .      .      .    .       .     .      .
   | .  .  .      .       .     .      .      .    .       .     .      .
15 | .  .  .      .       .     .      .      .    .       .     .      .
   | .  .  .      .       .     .      .      .    .       .     .      .
   | .  .  .      .       .     .      .      .    .       .     .      .
   | .  .  .      .       .     .      .      .    .       .     .      .
   | .  .  .      .       .     .      .      .    .       .     .      .
20 |     NODE VOLTAGE±XX TO±XX  ±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXX
   |     NODE VOLTAGE±XX TO±XX  ±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXX
   |     NODE VOLTAGE±XX TO±XX  ±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXX
   |     NODE VOLTAGE±XX TO±XX  ±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXX
24 |_____________________________________________________________________________
        5    10   15   20   25   30   35   40   45   50   55   60   65   70   75
```

TABLE 4-IV.- Continued

(c) Concluded

PROCESSOR CASKU

```
   1   5    10   15   20   25   30   35   40   45   50   55   60   65   70   75
 1              DISTRIBUTION NETWORK SOLUTION
                          (XXX ITER.)
   ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
                   GREENWICH MEAN TIME XXX:XX:XX:XX
 5 SOURCE PWR:±XXXXXX.  BUS PWR:±XXXXXX.  ATTACHED LOAD(28V):±XXXXXX.  SUB-PWR:±XXX
          BRANCH               SOURCE                       LOAD
   NO  SW  DIODE CURRENT  NO  VOLTS CURRENT   NO  VOLTS     ACT      PR      PP
   ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

10 ±XX±XXααααα±XXXXXXX.XX±XX±XXXX.XX±XXXX.XX±XX±XXXX.XX±XXXXXX±XXXXXX±XXXXXX
   .   .  .       .      .    .       .     .    .       .      .      .
   .   .  .       .      .    .       .     .    .       .      .      .
   .   .  .       .      .    .       .     .    .       .      .      .
   .   .  .       .      .    .       .     .    .       .      .      .
15 .   .  .       .      .    .       .     .    .       .      .      .
   .   .  .       .      .    .       .     .    .       .      .      .
   .   .  .       .      .    .       .     .    .       .      .      .
   .   .  .       .      .    .       .     .    .       .      .      .
   .   .  .       .      .    .       .     .    .       .      .      .
20     NODE VOLTAGE±XX TO±XX  ±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXX
       NODE VOLTAGE±XX TO±XX  ±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXX
       NODE VOLTAGE±XX TO±XX  ±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXX
       NODE VOLTAGE±XX TO±XX  ±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXXX±XXXX.XXX
24
       5    10   15   20   25   30   35   40   45   50   55   60   65   70   75
```

TABLE 4-IV.- Continued

(d) Display parameter definition table for the distribution network solution

PROCESSOR CASKU

DISTRIBUTION NETWORK SOLUTION	
Display parameter label	Parameter definition
ITER	Number of iterations required to arrive at a solution
MISSION ELAPSED TIME	GET of the table in ± hours, minutes, and seconds (XXX:XX:XX) or GMT of the table in Julian calendar days, hours, minutes, and seconds (XXX:XX:XX:XX)
SOURCE POWER	Fuel cell power at the time of the display (watts)
BUS LOAD	Bus loads at the time of the display (watts)
ATTACHED LOAD (28V)	Equivalent 28-volt load at the time of the display (watts)
SUB-PWR	Payload systems load at the time of the display (watts)
BRANCH NO.	Node number
SW	Switch position (on or off)
DIODE	Mnemonic diode identifier
CURRENT	Current for that node at the time of the display (amps)

TABLE 4-IV.- Concluded

(d) Concluded

PROCESSOR CASKU

DISTRIBUTION NETWORK SOLUTION	
Display parameter label	Parameter definition
SOURCE NO.	Number of the source
VOLTS	Source voltage at the time of the display (volts)
CURRENT	Current for the source at the time of the display (amps)
LOAD NO.	Nomenclature number
VOLTS	Voltage reading at the time of the display (volts)
ACT	Actual power requirements at the time of the display (watts)
PR	Resistive power at the time of the display (watts)
PP	Constant power at the time of the display (watts)
NODE VOLTAGE	Node numbers and voltage summary for the time of the display (volts)

TABLE 4-V.- PROCESSOR MESSAGE TABLE

PROCESSOR CASKU

MSG no.	Message ID block	Message text block and explanation
1	*CASKU*	"COULDN'T OPEN AAAAAA XXXX." Meaning: The named file could not be opened because of the error in XXXX Severity: Terminal Action required by user: Determine cause of error, correct and reinitiate the processor
2	*CASKU*	"FILE PROBLEM XXXX IN AAAAAA." Meaning: An error, XXXX, has occurred in the named file Severity: Terminal Action required by user: Determine cause of error, correct and reinitiate the processor
3	*CASKU*	"COULDN'T FIND ACTIVITY XXXX." Meaning: The activity could not be found in the data base Severity: Warning Action required by user: None
4	*CASKU*	"TIME LINE TOO LARGE FOR ARRAY." Meaning: Array dimensions exceeded Severity: Warning Action required by user: None
5	*CASKU*	"CASKU WORKING AT XXX.XXXX." Meaning: Message to user indicating that the processor is computing data for the time shown in XXX.XXXX Severity: Normal Action required by user: None

TABLE 4-VI.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR CASKU

Processor name	Processor abstract prompt (maximum 256 characters)
CASKU	CASKU produces an output file for use by the display processor. This file contains detailed power requirements, consumption values for O_2, N_2, H_2, NH_3, and H_2O, and waste H_2O produced. Optional displays of EPS and ECS network solutions are also produced.

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
TIMREF	Time reference base to use GE for ground elapsed time (GET) and GM for Greenwich mean time (GMT)
DATAFL	Four-character name to use for output
TIMFIL	Four-character name of the time-line file
ORBACT	Six-character name of the ORBITER ACTIVITY data file
PAYACT	Six-character name of the PAYLOAD ACTIVITY data file
SYSFIL	Six-character name of the SYSTEMS CHARACTERISTICS data file
PROCON	Cartridge reference numbers for the system files, DRDE files, and the blocks to allocate for the output file
ECSPRT	Are detailed displays of the ECS network desired? YE or NO
EPSPRT	Are detailed displays of the EPS network desired? YE or NO
EVALIN	Is there an interval of time that is of particular interest? Enter YE; otherwise, the default is for the entire time line (a NO response)

TABLE 4-VI.- Concluded

PROCESSOR CASKU

Processor name	Processor abstract prompt (maximum 256 characters)
Parameter keyword name	**Parameter definition prompt (maximum 256 characters)**
INTST	Starting time of the time interval; + or -, HH, MM, SS for GET and DDD, HH, MM, SS for GMT
INTSTP	Stopping time of the time interval; + or -, HH, MM, SS for GET and DDD, HH, MM, SS for GMT
LYEAR	Launch year
LDAY	Launch day
LHOUR	Launch hour
CONTNU	Is processing desired should an activity number not be found? YE or NO

5.0 PROCESSOR ROUTINES

5.1 ROUTINE NAME - MAIN PROGRAM CASKU

5.1.1 Purpose

CASKU provides executive control, a timing loop, and sequences the evaluation models appropriately for the processor. CASKU.

5.1.2 Functional Description

Initially, the RTE service routine RMPAR is called to obtain system parameters. The logical unit number is then set to IPARM(1). IPARM(1) is then set to zero to indicate normal conditions. Processor termination flag THEEND is set to false. A call is then made to the RTE service routine LDSEG to load the segment CPRPU. CPRPU provides initialization and a preliminary contract with the interface table. If an error condition exists on the return from LDSEG (indicated by IPARM not equal to zero), an abnormal termination follows. The abnormal termination consists of setting IPARM(1) to -32768, setting THEEND to true, and calling RTE service routine LDSEG to load the segment CPRPU, which also provides the final termination routine, and finally calling XPXIT to terminate all processor input/output.

If there is no error condition from the original call to LDSEG, a large DO-WHILE loop is entered that processes the time line. This DO-WHILE loop continues while more data remains to be processed, or while the end-time value has not been read.

Initially, within this loop, a heat load and power profile is built using processor subroutine PROFL. If an error condition exists upon return from PROFL (as indicated by IPARM(1) not equal to zero), an abnormal termination follows.

If no error condition exists, a call is made to LDSEG to load and execute segment EPSU. Segment EPSU evaluates the EPS and provides a detailed print of the EPS network solution, if the user desires. If an error condition exists upon return from LDSEG, an abnormal termination follows.

If no error condition exists, a call is made to processor subroutine ADBTU. Routine ADBTU adjusts the input heat loads to account for the actual voltages determined in the electrical power system's evaluation.

A call is made to processor routine PLDAT. PLDAT outputs to file the consumable requirements and some associated systems data from the ATCS and EPS model evaluations at each time-line entry point. If an error condition exists upon return from PLDAT (IPARM(1) not equal to zero), an abnormal termination follows.

If no error is detected, a test is made to determine if all time-line entries have been processed. This is done by the variable TNEXT, which contains the value of the next time-line entry. If TNEXT is equal to a termination value of greater than 99999, control drops out of the DO-WHILE loop.

Continuing in the DO-WHILE loop, the time of the next time-line entry is compared to the end time of the evaluation. If it is greater, TNEXT is set to the end time. Now begins a DO-UNTIL loop nestled within the DO-WHILE. The loop continues until the point on the time line currently being evaluated is greater than or equal to the time of the next entry minus a tolerance of 0.01.

Within the DO-UNTIL, the delta time is computed as the difference between the next time-line entry and the present time-line entry. If the delta time is greater than a tolerance of 0.5, it is set to 0.5.

A call is then made to the RTE service routine LDSEG to load and execute the ECSU segment. Routine ECSU evaluates the environmental control system (ECS) and provides the capability to print the ECS network. If an error occurs during this segment, IPARM(1) is set to a nonzero value. If detected after the call to LDSEG, an abnormal termination follows; otherwise, the time is incremented by the delta time. A call is made to processor subroutine TKLMT. Routine TKLMT evaluates the quantity requirements against the tank capacities. If an error is returned from TKLMT (as indicated by IPARM(1) $\neq$ 0), an abnormal termination follows.

If no error condition exists, the present time is compared against the next time-line entry point minus a tolerance of 0.01. If less than 0.01, a call is made to subroutine PLDAT to output to file the consumable requirements and some other associated system data. IPARM(1) is tested for the occurrence of errors in PLDAT. If an error has occurred, abnormal termination follows; otherwise, the present time is again compared to the next time-line entry minus a tolerance. If less, control returns to the top of the DO-UNTIL loop. Otherwise, the time is set to the time of the next time-line event. If this time is less than the final time of evaluation, control returns to the top of the DO-WHILE loop.

As control leaves the DO-WHILE loop, the delta time is set to zero. A call is made to the RTE service routine LDSEG to load and execute segment ECSU to provide the final ECS detailed print. An error test is performed upon the return from LDSEG, and if an error has occurred (as indicated by IPARM(1) $\neq$ 0), an abnormal termination follows.

A call is made to PLDAT to provide the final time point for the output file. Time is set to a termination value of 99999, and a final call is made to PLDAT to provide a termination flag to the output file.

Normal termination now follows. IPARM(1) is set to zero, the end flag THEEND is set true, and a call is made to LDSEG to load and execute segment CPRPU. Routine CPRPU will truncate and close the data file. A call is then made to XPXIT to terminate all input/output and halt execution.

5.1.3 Assumptions and Limitations

None.

5.1.4 Method

None.

5.1.5 Routine Input/Output Variables

The CASKU input/output variables are presented in table 5.1-I.

5.1.6 Functional Logic Flow

A functional logic flow is not presented. The functional level PDL for CASKU is presented in figure 5.1-1.

5.1.7 Diagnostics and Debug

None.

5.1.8 Special Comments

None.

5.1.9 References

None.

TABLE 5.1-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine CASKU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
LU	--	Intg	O	--	C	--	User's logical unit number
IPARM	--	Intg	O	--	C	--	System and FDS parameters
THEEND	--	Logical	O	--	C	--	Flag denoting end of processing

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

```
1*
1 BEGIN CASKU
2     CALL RMPAR FOR SYSTEM PARAMETERS
2     SET LOGICAL UNIT NUMBER = IPARm(1)
2     SET FLAGS AND INITIALIZE VARIABLES
2     CALL LDSEG TO LOAD PREPERATION SEGMENT
2     IF(NO ERROR RETURN)
2     THEN
3        BRANCH TO -LBL100-
2     ELSE
3        EXIT TO -EX8000-
2     ENDIF
2*
2 -LBL100-
3     DOWHILE(MORE RECORDS REMAIN IN THE TIMELINE FILE)
4        PERFORM PROFL
4        IF(NO ERROR RETURN)
4        THEN
5           BRANCH TO -LBL200-
4        ELSE
5           EXIT TO -EX8000-
4        ENDIF
4*
4 -LBL200-
4        CALL LDSEG FOR ELECTRICAL POWER SEGMENT
4        IF(NO ERROR RETURN)
4        THEN
5           BRANCH TO -LBL300-
4        ELSE
5           EXIT TO -EX8000-
4        ENDIF
4*
4 -LBL300-
4        PERFORM ADBTU
4        PERFORM PLDAT
4        IF(NO ERROR RETURN)
4        THEN
5           BRANCH TO -LBL400-
4        ELSE
5           EXIT TO -EX8000-
4        ENDIF
4*
4 -LBL400-
4        IF(TIMENEXT > ENDTIME)
4        THEN
5           TIMENEXT = ENDTIME
4        ENDIF
5        DOUNTIL(TIME < TIMENEXT - TOLERANCE)
6           COMPUTE DELTA TIME
6           IF(DELTA TIME > TOLERANCE)
6           THEN
7              DELTA TIME = TOLERANCE
6           ENDIF
```

Figure 5.1-1.- CASKU functional level PDL.

```
6           CALL LDSEG FOR ENVIRONMENTAL CONTROL SEGMENT
6           IF(NO ERROR RETURN)
6           THEN
7              BRANCH TO -LBL600-
6           ELSE
7              EXIT TO -EX8000-
6           ENDIF
6*
6 -LBL600-
6           SET TIME = TIME + DELTA TIME
6           PERFORM TKLMT
6           IF(NO ERROR RETURN)
6           THEN
7              BRANCH TO -LBL650-
6           ELSE
7              EXIT TO -EX8000-
6           ENDIF
6*
6 -LBL650-
6           IF(TIME < TIMENEXT - TOLERANCE)
6           THEN
7              CALL PLDAT
6           ENDIF
6           IF(NO ERROR RETURN)
6           THEN
7              IF(TIME < TIMENEXT - TOLERANCE)
7              THEN
8                 CONTINUE DOUNTIL
7              ELSE
8                 SET TIME = TIMENEXT
8                 IF(TIME < ENDTIME)
8                 THEN
9                    CONTINUE DOWHILE
8                 ELSE
9                    BRANCH TO -LBL800-
8                 ENDIF
7              ENDIF
6           ELSE
7              EXIT TO -EX8000-
6           ENDIF
5        ENDDO
4     ENDDO
3*
2 -LBL800-
2     SET DELTA TIME = ZERO
2     CALL LDSEG FOR ENVIRONMENTAL CONTROL SEGMENT
2     IF(NO ERROR RETURN)
2     THEN
3        PERFORM PLDAT
2     ELSE
3        EXIT TO -EX8000-
2     ENDIF
2     SET TIME = TERMINATION VALUE
2     PERFORM PLDAT
2     EXIT TO -EX9000-
```

Figure 5.1-1.- Continued.

```
2*
2 -EX3000-
2     SET IPARM(1) TO ABNORMAL TERMINATION VALUE
2     SET END FLAG = TRUE
2     CALL LDSEG FOR PREPERATION SEGMENT
2     EXIT TO -EX9050-
2*
2 -EX9000-
2     SET IPARM(1) TO NORMAL TERMINATION VALUE
2     SET END FLAG = TRUE
2     CALL LDSEG FOR PREPERATION SEGMENT
2*
2 -EX9050-
2     CALL XPXIT FOR TERMINATION
1 END CASKU
1*
```

Figure 5.1-1.- Concluded.

5.2 ROUTINE NAME - CPRPU

5.2.1 Purpose

The purpose of the preparation and termination segment (CPRPU) is to input the interface table variables, read in and store the Orbiter activity data and the payload activity data into internal arrays, read in and process the user's time-line file, create and store the sorted scratch file, read in and store the systems characteristics data, create the output data file, and, finally, to provide the final termination routine for the CASKU processor.

5.2.2 Functional Description

Routine CPRPU commences with a test on the logical common variable THEEND. If it is not termination processing, initialization follows.

The user identification variable is set to IPARM(2), and a record number variable is set to one. INTBUF is initialized and a call is made to the FDS-1 utility routine XPGET to retrieve interface table parameters.

A call is made to processor routine ACTFL to open and read initialization data from the activity data file. If an error condition is detected upon return from ACTFL (indicated by IPARM(1) $\neq$ 0), an abnormal termination follows. An abnormal termination consists of setting IPARM(1) to -32768 as an error indicator. Closing any files that may be open (through the use of RTE file manager routine CLOSE), purging any scratch file that may have been created (through the use of the RTE file manager routine PURGE), a call is made to the RTE service routine RETRN to establish the return linkage, and a call is made to CASKU to return control to the calling segment.

If no error is returned from ACTFL, the number of activity blocks in the payload data base is set to zero.

If the payload activity file name is not blank, a call is made to processor routine ACTFL to open and read initialization data from the activity data file. If an error condition is detected upon return from ACTFL, an abnormal termination follows.

Some processor flags and counters are set, as is the scratch file name and its attributes. The scratch file is then created through a call to the RTE file manager routine CREAT. If CREAT returns an error message indicating a duplicate file name (IERR = -2), the existing file is purged through the use of the RTE file manager routine PURGE. If an error message is returned from the PURGE call, or if CREAT returned an error other than -2, an error message is displayed and an abnormal termination follows. If the duplicate file was purged successfully, control loops back to attempt to create the file again.

If the scratch file was created successfully, the time-line file is opened via the RTE file manager routine OPEN. If an error is returned from the open call, an abnormal termination follows. The opened file is positioned at record two

via the RTE file manager routine POSNT. If the POSNT call returns an error indication, a message is printed and an abnormal termination follows.

A DO-WHILE loop now begins, which reads and processes the time-line records. The loop continues while there are still records to be processed.

The loop begins with a call to the RTE file manager routine READF to read a record. If an error occurs during the read, an error message is printed and an abnormal termination follows. If the activity block number read in from the time-line file is greater than 9998, it is considered an indicator of the end of the time-line file, and execution drops out of the DO-WHILE loop.

Otherwise, execution continues within the loop. Flags are set for an orbital activity. A test is made to see if the activity is actually a payload activity and, if it is, the flags are reset.

A DO-LOOP now follows. It is indexed from one to the number of activity block numbers to be scanned. A search is made to locate the activity in the data arrays. If the activity is not found, a warning message is displayed and execution returns to the top of the DO-WHILE loop.

If the activity was found, counters are set for the next DO-LOOP, and the counters are incremented. A DO-UNTIL loop is initiated and continues until the counter surpasses its limit, or until the activity does not equal the activity array number.

Within the DO-UNTIL, a test is made to see if the end time is less than the start time plus the delta start time. If so, execution returns to the top of the DO-UNTIL. Otherwise, the counters are incremented and start time is set into the activity array. The activity number is set into the activity number array. If the ASCII characters representing the activity type in the LTR array equal a test value, the flow of execution returns to the top of the DO-UNTIL.

Otherwise, the counters are incremented, and the activity numbers are set into the activity number array. If the LTR array value at its current index is equal to certain test values, then the start time is set into the time array. If the time array at this point is less than the end time, control returns to the top of the DO-UNTIL.

Otherwise, the time line is set equal to the end time. Control then returns to the top of the DO-UNTIL. This ends the code within the DO-UNTIL.

Following the DO-UNTIL in the execution sequence is a DO-LOOP that is incremented from one to the value of the counter. This DO-LOOP writes the activity numbers and times determined for the time-line point to the scratch file through the use of the RTE file manager routine WRITF. An error condition is tested for upon the return from WRITF, and if an error is detected, an abnormal termination follows.

At the conclusion of this DO-LOOP, control is returned to the top of the DO-WHILE loop.

At the end of the DO-WHILE loop, a final record is written to the file, with termination values to act as flags. The RTE file manager routine WRITF is used, and if an error state is detected upon return from WRITF, an error message is displayed, and an abnormal termination follows.

Now the activity number array and the delta start and delta stop time arrays are zeroed out to zero out the common area for the next information group.

The time-line file is closed through the use of the RTE file manager routine CLOSE. If an activity was not found, or if the scratch file is larger than the internal arrays, print a warning message. If the user wishes to continue, as specified by a "YE" response to the interface table variable CONTNU, execution will continue normally. Otherwise, an abnormal termination will follow.

As normal execution proceeds, the scratch file is rewound through the use of the RTE file manager routine RWNDF. If an error is detected upon return from RWDNF, an error message is displayed and an abnormal termination follows.

If the file is rewound properly, a loop is entered that will read a record of the scratch file (using the RTE file manager command READF), test for a possible error return (print an error message and terminate abnormally, if found), and enter the record into the internal arrays. This continues for all the entries in the scratch file.

The internal arrays are sorted using a bubble sort.

The scratch file is repositioned to the beginning by a call to the RTE file manager routine RWNDF. If an error results from the rewind, an error message is printed and the routine abnormally terminates. Otherwise, a DO-LOOP is entered that executes until all the records have been written to the scratch file. The RTE file manager routine WRITF is used. If an error occurs while writing to the scratch file, an error message is displayed and the routine is abnormally terminated.

Two internal arrays, JACT and TIM are set to zero to allow for an equivalenced use of the space in common.

The scratch file is rewound as initialization for the main program. The RTE file manager routine RWNDF is used. If an error occurs on the rewind, an error message is printed and an abnormal termination follows.

The Greenwich mean time flag is set depending on the user's response to the interface table variable TIMREF. The number of days in the user-specified year of launch is determined through the use of the FORTRAN library routine MOD to determine leap years.

If the user has indicated a desire to specify an evaluation interval for the processor displays (as indicated by a "YE" response to the interface table variable EVALIN), then the calling arguments are set and a call is made to the processor routine RGMET to convert the "GMT" or "GET" to its decimal equivalent. If the Greenwich mean time flag is true, time is set at the time on the time line currently being evaluated minus the reference time. Calling arguments are

again set, and a call is again made to RGMET to convert the stop time to decimal. If the Greenwich mean time flag is true, the end time is set to the end time minus the reference time.

If the user did not wish the evaluation interval option, default values are given to the present time and the end time so as to include the entire time line.

The systems data file is now opened through a call to the RTE file manager routine OPEN. The systems data file should be a type-3 file (variable length records of any data type). If it is not a type-3 file (as indicated by the error return parameter from the OPEN call), an error message is printed and an abnormal termination follows.

A DO-LOOP is now entered, which is indexed from 1 to 900 in increments of 128. Within the DO-LOOP, the data are read from the systems data file into IECPS (a systems characteristics data array). A test for error conditions is made after each call to READF, and if an error is detected an error message is displayed and an abnormal termination follows.

A call is then made to the RTE file manager routine READF upon completion of the DO-LOOP to obtain power source information. If an error occurred in the call to READF, an error message is printed and an abnormal termination follows.

A DO-LOOP is entered, which is indexed from 1 to 378, incremented by 128. More of the systems data file is read into the IEPS1 array using the RTE file manager routine READF. If an error condition is present upon return from READF, an error message is displayed and an abnormal termination follows.

Another DO-LOOP is entered, which is indexed from 1 to 975, in increments of 128. Data are input into the systems characteristics data array IECS1 through the use of the RTE file manager routine READF. If an error condition is present upon return from READF, an error message is displayed to the user and an abnormal termination follows.

A final call is made to the RTE file manager routine READF to input the data file into the common array RMISC. If an error condition is present upon return from READF, an error message is displayed and an abnormal termination follows.

The systems data file is closed through a call to the RTE file manager routine CLOSE.

A DO-LOOP is entered, which converts to degrees Rankin fluid temperatures at each node in the ECS network, and temperatures at each node in the ATCS model.

ECS cooling parameters are set for freon flow, desired Orbiter cabin temperature, desired output temperature of the radiator, and certain control values. The intial water quantity is set.

Two cabin regulator flags (one that specifies the status of the cabin relief valve and one that indicates which type of gas is flowing through the regulators) are initialized. The weight of the oxygen and nitrogen in the

cabin's atmosphere is calculated as a function of the pressure, volume, and temperature.

Certain system index pointers and flags are now initialized. An array (VOLTS) containing initial voltage values is set to 28.

A call is now made to the RTE Executive to establish the current time.

An output disk resident data element (DRDE) file is now created. First, certain output file characteristics are set. A call is made to the FDS-1 utility routine XPATR to obtain the user's specified file name. A call is made to XPPUT, which is an FDS-1 utility routine that inserts the output file characteristics into the interface table.

The output file is then created through the use of the RTE file manager routine CREAT. If the error return parameter IERR returns a minus 2, a file by the same name already exists. An attempt is made to purge the existing file through the use of the RTE file manager routine PURGE. If the file cannot be purged, an error message is printed and an abnormal termination follows. If the file is purged, an attempt is made to create it again. If the file cannot be created for any other reason than a duplicate file name, an error message is printed and an abnormal termination follows.

Once the file is created, a three-word Hollerith array is set to the program name. The file names are then set into header records, which are written using two calls to the RTE file manager routine WRITF. Error conditions are checked for, and if an error exists an error message is displayed and an abnormal termination follows. Otherwise, a normal termination follows. The normal termination consists of setting IPARM(1) to zero, calling the RTE service routine RETRN to set the return linkage, and a call is made to the calling routine CASKU.

The above functional description applies to the termination flag THEEND set to false, implying that the processor should be initialized. The following description applies to the termination case.

A call is made to the RTE file manager routine LOCF to determine the number of blocks in the data file. If an error is returned from the LOCF call, an abnormal termination follows.

The number of blocks to be truncated is determined, and a call is made to the RTE file manager routine CLOSE to close and truncate the file. If an error occurs during the call to CLOSE, an abnormal termination follows.

If the file has closed properly, the new file characteristics are set and a call is made to the FDS-1 utility routine to insert these new file characteristics into the interface table. The scratch file is purged through a call to the RTE file manager routine PURGE. The ORBITER ACTIVITY file and the PAYLOAD ACTIVITY file are closed with calls to the RTE file manager routine CLOSE. A normal termination then follows.

5.2.3 Assumptions and Limitations

None.

5.2.4 Method

The TIM and JACT arrays are sorted in ascending order with a bubble sort. A bubble sort consists of progressing through an array linearly until an entry is found to be less than its predecessor. The entry is then exchanged, or "bubbled" upwards until it is greater than or equal to its predecessor. This continues until the array is sorted.

5.2.5 Routine Input/Output Variables

The CPRPU input/output variables are presented in table 5.2-I.

5.2.6 Functional Logic Flow

A functional logic flow is not presented. The functional level PDL for CPRPU is presented in figure 5.2-1.

5.2.7 Diagnostics and Debug

The routine CPRPU offers diagnostics and debug in the form of error checking for all file input/output, and displaying error messages in the event of an error.

Specifically, a message is printed if a file could not be opened. A message is also printed for miscellaneous file problems (these include the inability to create, read, write, rewind, position, and in some cases, purge a file).

Error messages are also printed when an activity cannot be located, or when the time line is too large for the internal arrays.

5.2.8 Special Comments

None.

5.2.9 References

None.

TABLE 5.2-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine CPRPU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ACTNMS	--	Intg	I	--	C	--	Array containing computed activity locations.
ATMLK	--	Real	O	--	C	--	Atmospheric leak rate of cabin (lb/hr/psi).
BSLQTY	--	Real	O	--	C	--	Baseline usable tank capacity for each consumable.
CONTNU	--	Intg	I	--	IT	--	Flag indicating user's desire to continue if an activity cannot be found, or if the internal arrays containing the data file become full.
DATAFL	--	Intg	I	--	IT	--	Name and characteristics of the output data file.
DIODE	--	Real	O	--	C	--	DIODE characteristics for each branch of the EPS model.
DUMPRT	--	Real	O	--	C	--	Rate at which water flows through the overboard vent lines.
ESCPRT	--	Intg	I	--	IT	--	Flag for detailed printouts of the ECS evaluation.

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.2-I.- Continued

Routine CPRPU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ENDTIM	--	Real	O	--	IT	--	End time of evaluation.
EPSPRT	--	Intg	I	--	IT	--	Flag for detailed printout of EPS evaluation
FTEMP	--	Real	O	--	C		Fluid temperature at each node in the ECS network.
FRNFLO	--	Real	O	--	C	--	Total freon flow for maximum flow conditions.
H2O	--	Real	O	--	C	--	Available potable water quantity.
H2OINT	--	Real	O	--	C	--	Initial quantity of potable water.
H2OLO	--	Real	O	--	C	--	Minimum allowable quantity of potable water.
H2OHI	--	Real	O	--	C	--	Maximum allowable quantity of potable water.
HREF	--	Real	I/O	--	IT	--	Launch hour.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.2-I.- Continued

Routine CPRPU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ICLMT	--	Intg	O	--	C	--	Current limit to each branch in the EPS model.
ICRCT	--	Intg	O	--	C	--	Node connections of each branch in the EPS model.
IDREF	--	Intg	I/O	--	C	--	Launch day.
IDUMP	--	Logical	O	--	C	--	Indicates that a water dump is in progress.
IFILE	--	Intg	I	--	C	--	Information needed by the file manager to access files.
IGMT	--	Logical	O	--	C	--	Defines time reference. True = GMT False = GET
IN202	--	Intg	O	--	C	--	Indicates which type of gas is flowing through the cabin regulators. 1 = oxygen 2 = nitrogen
INTST	--	Real	I	--	IT	--	Start time of evaluation interval.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.2-I.- Continued

Routine CPRPU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
INTSTP	--	Real	I	--	IT	--	Stop time of evaluation interval.
IOACT	--	Intg	I	--	C	--	Used to access the ORBITER ACTIVITY file.
IPACT	--	Intg	I	--	C	--	Used to access the PAYLOAD ACTIVITY file.
IPARM	--	Intg	I	--	C		System and FDS-1 parameters.
IPLMT	--	Intg	O	--	C	--	Defines power constraints by specifying nodes, branches, and a power limit.
IPRES	--	Intg	O	--	C	--	Defines the location of the present value in the PTEMP and FTEMP arrays.
IPREV	--	Intg	O	--	C	--	Defines the location of the previous value in the temperature arrays PTEMP and FTEMP.
ISBNCH	--	Intg	O	--	C	--	List of the branches that are used in determining the dissipated power.
ISNOD	--	Intg	O	--	C	--	List of nodes associated with branches in ISBNCH.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.2-I.- Continued

Routine CPRPU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ITLMT	--	Intg	O	--	C	--	Temperature limitation data for the ATCS model.
ITLN	--	Intg	I	--	C	--	Used to access the time-line array file.
IVNT	--	Intg	O	--	C	--	Specifies the status of the cabin relief valve for each cabin. 1 = closed 2 = open
IYEAR	--	Intg	I	--	IT	--	Launch year.
LTR	--	Intg	I	--	C	--	ASCII characters representing the activity type.
LDLOC	--	Intg	O	--	C	--	Correlation between cool codes and the equivalent node locations in the ATCS model.
LOAD	--	Intg	O	--	C	--	Correlation of each load to its branch location.
LNODES	--	Intg	O	--	C	--	EPS model used to adjust the heat loads to account for the actual EPS power requirements.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.2-I.- Continued

Routine CPRPU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
LU	—	Intg	I	—	C	—	Logical unit number of the user.
MCP	—	Real	O	—	C	—	Heat capacitance of each node in the ATCS model.
NCRCT	—	Intg	—	—	—	—	Number of branches in the EPS model.
NLDS	—	Intg	O	—	C	—	Number of loads in the EPS model.
NNODE	—	Intg	O	—	C	—	Number of nodes in the EPS model.
NREF	—	Intg	O	—	C	—	Reference node in the EPS model.
NSB	—	Intg	O	—	C	—	Number of branches and nodes to be used in user-specified power calculations.
NTLMT	—	Intg	O	—	C	—	Number of temperature limits defined in "ITLMT".
NVI	—	Intg	O	—	C	—	Number of data points in each source's "VI" curve.
O2RNG	—	Real	O	—	C	—	Range of allowable O_2 pressure in each cabin.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.2-I.- Continued

Routine CPRPU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ORBACT	--	6CH	I	--	IT	--	ORBITER ACTIVITY data file name.
PAYACT	--	6CH	I	--	IT	--	PAYLOAD ACTIVITY data file name.
PN2	--	Real	O	--	C		Nitrogen pressure in each cabin.
PO2	--	Real	O	--	C	--	Oxygen pressure in each cabin.
PROCON	--	Intg	I	--	IT	--	Processor constants, cartridge reference for the files, number of blocks in the output file, and the output units for the display.
PTEMP	--	Real	O	--	C	--	Temperature of each node in the ATCS model.
QTYKIT	--	Real	O	--	C	--	Usable tank capacity per kit for each consumable.
RADCAP	--	Real	O	--	C	--	Radiator cooling capacity (btu/hr).
RBNCH	--	Real	O	--	C	--	Resistance of each branch in the EPS model.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.2-I.- Continued

Routine CPRPU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TRAD	—	Real	O	—	C	—	Desired output temperature of radiator during normal operation.
REGFLO	—	Real	O	—	C	—	Flow rate through each cabin regulator.
REGRNG	—	Real	O	—	C	—	Pressure range of the cabin regulators.
SQMET	—	Real	O	—	C	—	Metabolic rate of the Shuttle crew.
SRCTYP	—	Real	O	—	C	—	Pressure source information.
SRCVI	—	Real	O	—	C	—	Defines the "VI" characteristics of each power source.
SYSFIL	—	6CH	I	—	IT	—	Name of the SYSTEMS CHARACTERISTICS file.
SWTCH	—	Intg	O	—	C	—	Specifies the switch status in each branch of the EPS model.
TCAB	—	Real	O	—	C	—	Desired temperature of the Orbiter cabin.

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.2-I.- Continued

Routine CPRPU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TCNTRL	--	Real	O	--	C	--	Control valves used for maintaining the output temperature of the radiator.
TEVAP	--	Real	O	--	C	--	Desired output temperature of radiator when potable water is being dumped through the flash evaporators.
THEEND	--	Logical	I	--	C	--	Flag denoting the termination of processing when "TRUE".
TIME	--	Real	O	--	C	--	Point on the time line currently being evaluated.
TIMFIL	--	Intg	I	--	IT	--	Name and characteristics of the user's time-line file.
TIMREF	--	2CH	I	--	IT	--	Hollerith indicator of time reference for displays (either "GMT" or "GET")
UA	--	Real	O	--	C	--	Heat transfer coefficients for each node in the ATCS model.
VLMT	--	Real	O	--	C	--	Voltage limits for each node in the EPS model.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.2-I.- Concluded

Routine CPRPU

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
VNTFLO	--	Real	O	—	C	--	Flow rate of each cabin relief valve.
VNTRNG	--	Real	O	—	C	--	Pressure range of each cabin relief valve.
VOL	--	Real	O	—	C	--	Volume of each cabin.
WCP	--	Real	O	—	C	--	Heat capacitance of the fluid at each node of the ATCS model.
WTN2	--	Real	O	—	C	--	Weight of nitrogen in each cabin's atmosphere.
WTO2	--	Real	O	—	C	--	Weight of oxygen in each cabin's atmosphere.

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

```
1*
1 BEGIN CPRPU
2     IF(END FLAG = FALSE)
2     THEN
3        BRANCH TO -LBL50-
2     ELSE
3        BRANCH TO -LBL1000-
2     ENDIF
2*
2 -LBL50-
2     SET VARIABLES AND INTERFACE TABLE BUFFER
2     CALL XPGET FOR INTERFACE TABLE PARAMETERS
2     PERFORM ACTFL FOR ORBITER ACTIVITY DATA
2     IF(NO ERROR RETURN)
2     THEN
3        BRANCH TO -LBL110-
2     ELSE
3        EXIT TO -EX8000-
2     ENDIF
2*
2 -LBL110-
2     IF(PAYLOAD FILE NAME NOT = BLANKS)
2     THEN
3        PERFORM ACTFL FOR PAYLOAD ACTIVITY DATA
3        IF(NO ERROR RETURN)
3        THEN
4           BRANCH TO -LBL200-
3        ELSE
4           EXIT TO -EX8000-
3        ENDIF
2     ELSE
3        BRANCH TO -LBL200-
2     ENDIF
2*
2 -LBL200-
2     SET COUNTERS, FLAGS, SCRATCH FILE NAME AND ATTRIBUTES
2*
2 -LBL201-
2     CALL CREAT TO CREATE A SCRATCH FILE
2     IF(ERROR)
2     THEN
3        IF(DUPLICATE FILE NAME)
3        THEN
4           CALL PURGE TO DELETE FILE
4           IF(NO ERROR)
4           THEN
5              BRANCH TO -LBL201-
4           ELSE
5              BRANCH TO -ER550-
4           ENDIF
3        ELSE
4           BRANCH TO -ER550-
3        ENDIF
2     ELSE
3        BRANCH TO -LBL205-
2     ENDIF
```

Figure 5.2-1.- CPRPU functional level PDL.

```
2*
2 -LBL205-
2     CALL OPEN TO OPEN THE TIMELINE FILE
2     IF(NO ERROR)
2     THEN
3        CALL POSNT TO POSITION FILE AT RECORD 2
3        IF(NO ERROR)
3        THEN
4           DOWHILE(MORE RECORDS REMAIN)
5              CALL READF FOR A TIMELINE RECORD
5              IF(NO ERROR)
5              THEN
6                 SET FLAGS FOR ORBITER ACTIVITY
6                 IF(ACTIVITY = PAYLOAD ACTIVITY)
6                 THEN
7                    RESET FLAGS FOR PAYLOAD ACTIVITY
7                    BRANCH TO -LBL235-
6                 ELSE
7                    BRANCH TO -LBL235-
6                 ENDIF
6*
6 -LBL235-
7                 DOFOR I = 1,COUNTER,1
8                    IF(ACTIVITY NUMBER NOT= ARRAY NUMBER)
8                    THEN
9                       CONTINUE DOFOR
8                    ELSE
9                       BRANCH TO -LBL250-
8                    ENDIF
7                 ENDDO
6 -LBL240-
6                 WRITE ACTIVITY NOT FOUND MESSAGE
6                 SET FLAG
6                 CONTINUE DOWHILE
6 -LBL250-
6                 SET COUNTERS FOR NEXT DO LOOP
6 -LBL255-
6                 INCREMENT COUNTERS
7                 DOUNTIL(COUNTER > LIMIT OR ACTIVITY NUMBER
7                         NOT= ACTIVITY ARRAY NUMBER)
8                    IF(ENDTIME < START TIME + DELTA START TIME)
8                    THEN
9                       CONTINUE DOUNTIL
8                    ELSE
9                       INCREMENT COUNTERS
9                       SET START TIME INTO TIME ARRAY
9                       SET ACTIVITY NUMBER INTO ACTIVITY NUMBER
9                          ARRAY
9                       IF(LETTER ARRAY VALUE = TEST VALUE
9                       THEN
10                         CONTINUE DOUNTIL
9                       ELSE
10                         INCREMENT COUNTERS
10                         SET ACTIVITY NUMBER INTO ACTIVITY
10                            NUMBER ARRAY
```

Figure 5.2-1.- Continued.

```
10                          IF(LETTER ARRAY VALUE = TEST VALUES)
10                          THEN
11                             SET START TIME INTO TIME ARRAY
11                             IF(TIME ARRAY < END TIME)
11                             THEN
12                                CONTINUE DOUNTIL
11                             ELSE
12                                SET END TIME INTO TIME ARRAY
11                             ENDIF
10                          ELSE
11                             SET END TIME INTO TIME ARRAY
11                             CONTINUE DOUNTIL
10                          ENDIF
9                        ENDIF
8                     ENDIF
8 -LBL256-
9                     DOFOR I = 1,COUNTER,1
10                       CALL WRITE TO OUTPUT ARRAYS TO FILE
10                       IF(NO ERROR)
10                       THEN
11                          CONTINUE DOFOR
10                       ELSE
11                          BRANCH TO -LBL550-
10                       ENDIF
9                     ENDDO
8                     CALL WRITE TO OUTPUT LAST RECORD
8                     IF(NO ERROR)
8                     THEN
9                        BRANCH TO -LBL295-
8                     ELSE
9                        BRANCH TO -LBL550-
8                     ENDIF
7                  ENDDO
6               ELSE
7                  BRANCH TO -LBL320-
6               ENDIF
5            ENDDO
4         ELSE
5            BRANCH TO -ER320-
4         ENDIF
3      ELSE
4         BRANCH TO -ER325-
3      ENDIF
2*
2 -LBL295-
3      DOFOR I = 1,256,1
4         ACTIVITY NUMBER ARRAY = ZERO
5         DOFOR J = 1,2,1
6            DELTA START TIME ARRAY = ZERO
6            DELTA STOP  TIME ARRAY = ZERO
5         ENDDO
4      ENDDO
```

Figure 5.2-1.- Continued.

```
3*
2     CALL CLOSE FOR THE TIMELINE FILE
2     IF(AN ACTIVITY WAS NOT FOUND OR THE SCRATCH FILE RECORD
2        COUNTER > 900)
2     THEN
3        IF(SCRATCH FILE COUNTER > 900)
3        THEN
4           WRITE ARRAY LIMITS EXCEEDED MESSAGE
3        ENDIF
3        IF(USER WANTS TO CONTINUE FLAG = TRUE)
3        THEN
4           BRANCH TO -LBL400-
3        ELSE
4           EXIT TO -EX8000-
3        ENDIF
2     ELSE
3        BRANCH TO -LBL400-
2     ENDIF
2*
2  -ER320-
2     WRITE FILE PROBLEM MESAGE FOR TIMELINE FILE
2     EXIT TO -EX8000-
2*
2  -ER325-
2     WRITE OPEN ERROR MESSAGE FOR TIMELINE FILE
2     EXIT TO -EX8000-
2*
2  -LBL400-
2     CALL RWNDF FOR THE SCRATCH FILE
2     IF(NO ERROR)
2     THEN
3        DOFOR I = 1,NUMBER OF RECORDS,1
4           CALL READF FOR A RECORD
4           IF(NO ERROR)
4           THEN
5              SET TIME AND ACTIVITY NUMBER INTO ARRAYS
5              CONTINUE DOFOR
4           ELSE
5              BRANCH TO -ER550-
4           ENDIF
3        ENDDO
2     ELSE
3        BRANCH TO -ER550-
2     ENDIF
2*
3     DOFOR I = 1,ARRAY COUNT,1
4        IF(TIME(ARRAY COUNT) > TIME(ARRAY COUNT - 1)
4        THEN
5           CONTINUE DOFOR
4        ELSE
5           SWITCH POSITIONS OF TIME AND ACTIVITY NUMBERS
4        ENDIF
3     ENDDO
```

Figure 5.2-1.- Continued.

```
2*
2     CALL RWNDF TO POSITION SCRATCH FILE
3     DOFOR I = 1,ARRAY COUNT,1
4        SET OUTPUT RECORD VALUES
4        CALL WRITF TO OUTPUT THE RECORD
4        IF(NO ERROR)
4        THEN
5           CONTINUE DOFOR
4        ELSE
5           BRANCH TO -ER550-
4        ENDIF
3     ENDDO
2*
2     CALL RWNDF TO POSITION SCRATCH FILE FOR MAIN PROGRAM
2     IF(ERROR)
2     THEN
3        BRANCH TO -ER550-
2     ELSE
3        BRANCH TO -LBL600-
2     ENDIF
2*
2 -ER550-
2     WRITE SCRATCH FILE ERROR MESSAGE
2     EXIT TO -EX8000-
2*
2 -LBL600-
2     SET GREENWICH MEAN TIME FLAG
2     SET DAYS IN THE YEAR VALUE
2     IF(EVALUATION INTERVAL FLAG = TRUE
2     THEN
3        SET CALLING ARGUMENTS
3        PERFORM RGMET TO CONVERT START TIME TO DECIMAL
3        IF(GREENWICH MEAN TIME FLAG = TRUE)
3        THEN
4           TIME = TIME - REFERENCE TIME
3        ENDIF
3        SET CALLING ARGUMENTS FOR STOP TIME
3        PERFORM RGMET TO CONVERT STOP TIME TO DECIMAL
3        IF(GREENWICH MEAN TIME FLAG = TRUE)
3        THEN
4           END TIME = END TIME - REFERENCE TIME
3        ENDIF
3        BRANCH TO -LBL700-
2     ELSE
3        SET TIME AND END TIME TO DEFAULT VALUES
3        BRANCH TO -LBL700-
2     ENDIF
```

Figure 5.2-1.- Continued.

```
2*
2 -LBL700-
2     CALL OPEN FOR SYSTEMS CHARACTERISTICS DATA
2     IF(NO ERROR)
2     THEN
4        DOFOR I = 1,900,128
5           CALL READF FOR DATA
5           IF(NO ERROR)
5           THEN
6              CONTINUE DOFOR
5           ELSE
6              BRANCH TO -ER740-
5           ENDIF
4        ENDDO
3*
3        CALL READF FOR SOURCE TYPE DATA
3        IF(NO ERROR)
3        THEN
4           BRANCH TO -LBL720-
3        ELSE
4           BRANCH TO -ER740-
3        ENDIF
3*
3 -LBL720-
4        DOFOR I = 1,378,128
5           CALL READF FOR DATA
5           IF(NO ERROR)
5           THEN
6              CONTINUE DOFOR
5           ELSE
6              BRANCH TO -ER740-
5           ENDIF
4        ENDDO
3*
4        DOFOR I = 1,975,128
5           CALL READF FOR DATA
5           IF(NO ERROR)
5           THEN
6              CONTINUE DOFOR
5           ELSE
6              BRANCH TO -ER740-
5           ENDIF
4        ENDDO
3*
3        CALL READF FOR MISCELLANEOUS DATA
3        IF(NO ERROR)
3        THEN
4           CALL CLOSE FOR SYSTEMS CHARACTERISTICS DATA FILE
4           IF(NO ERROR)
4           THEN
5              BRANCH TO -LBL800-
4           ELSE
5              BRANCH TO -ER740-
4           ENDIF
3        ELSE
4           BRANCH TO -ER740-
3        ENDIF
2     ELSE
3        BRANCH TO -ER745-
```

Figure 5.2-1.- Continued.

```
2     ENDIF
2*
2 -ER740-
2     WRITE FILE PROBLEM MESSAGE
2     EXIT TO -EX8000-
2*
2 -ER745-
2     WRITE OPEN ERROR MESSAGE
2     EXIT TO -EX8000-
2*
2 -LBL800-
3     DOFOR I = 1,150,1
4        SET TEMPERATURE VARIABLES
3     ENDDO
2     SET ECS COOLING PARAMETERS
2     SET INITIAL WATER QUANTITY
3     DOFOR I = 1,2,1
4        SET CABIN PRESSURE VARIABLES
3     ENDDO
2     SET PROGRAM FLAGS
3     DOFOR I = 1,45,1
4        SET VOLTAGE VALUES
3     ENDDO
2     CALL EXEC FOR CURRENT TIME
2     SET OUTPUT FILE CHARACTERISTICS
2     CALL XPATR FOR OUTPUT FILE NAME
2     CALL XPPUT TO INSERT FILE CHARACTERISTICS
2*
2 -LBL900-
2     CALL CREAT TO CREATE THE OUTPUT FILE
2     IF(ERROR)
2     THEN
3        IF(DUPLICATE FILE NAME)
3        THEN
4           CALL PURGE TO REMOVE FILE
4           IF(NO ERROR)
4           THEN
5              BRANCH TO -LBL900-
4           ELSE
5              BRANCH TO -ER940-
4           ENDIF
3        ELSE
4           BRANCH TO -ER940-
3        ENDIF
2     ELSE
3        BRANCH TO -LBL915-
2     ENDIF
2*
2 -LBL915-
2     SET PROGRAM NAME
3     DOFOR I = 1,3,1
4        SET FILE NAMES INTO HEADER RECORDS
3     ENDDO
2     CALL WRITF TO WRITE ORDE HEADER RECORD
2     IF(NO ERROR)
2     THEN
```

Figure 5.2-1.- Continued.

```
3        CALL WRITF TO WRITE FIRST DATA RECORD
3        IF(NO ERROR)
3        THEN
4           EXIT TO -EX9000-
3        ELSE
4           BRANCH TO -ER940-
3        ENDIF
2     ELSE
3        BRANCH TO -ER940-
2     ENDIF
2*
2 -ER940-
2     WRITE FILE PROBLEM MESSAGE
2     EXIT TO -EX8000-
2*
2 -LBL1000-
2     CALL LOCF FOR NUMBER OF BLOCKS USED IN THE FILE
2     IF(NO ERROR)
2     THEN
3        COMPUTE AMOUNT TO TRUNCATE
3        CALL CLOSE TO CLOSE AND TRUNCATE FILE
3        IF(NO ERROR)
3        THEN
4           SET NEW FILE CHARACTERISTICS
4           CALL XPPUT TO INSERT NEW CHARACTERISTICS
4           CALL PURGE TO REMOVE SCRATCH FILE
4           CALL CLOSE TO CLOSE ORBITER ACTIVITY FILE
4           CALL CLOSE TO CLOSE PAYLOAD ACTIVITY FILE
4           EXIT TO -EX9000-
3        ELSE
4           EXIT TO -EX8000-
3        ENDIF
2     ELSE
3        EXIT TO -EX8000-
2     ENDIF
2*
2 -EX8000-
2     SET IPARM TO ABNORMAL TERMINATION VALUE
2     CALL CLOSE FOR SYSTEMS CHARACTERISTICS FILE
2     CALL PURGE TO REMOVE SCRATCH FILE
2     CALL CLOSE FOR ORBITER ACTIVITY FILE
2     CALL CLOSE FOR PAYLOAD ACTIVITY FILE
2     EXIT TO -EX9010-
2*
2 -EX9000-
2     SET IPARM TO NORMAL RETURN VALUE
2*
2 -EX9010-
2     CALL RETRN
2     CALL MAIN PROGRAM CASKU
1 END CPRPU
1*
```

Figure 5.2-1.- Concluded.

5.3 ROUTINE NAME - ECPRT

5.3.1 Purpose

The ECS print routine (ECPRT) provides a detailed print of the ECS network at various time points in the evaluation.

5.3.2 Functional Description

The output logical unit is initially set to the user's logical unit. If the user has specified the printer as the output device, a call is made to EXEC to cause a page-eject and reset the line counter.

A heading is now written. A call is made to the processor routine GMET to convert ground elapsed decimal hours into the appropriate reference. Depending on the state of the IGMT flag, either the ground elapsed time or the Greenwich mean time is printed. A load limit summary is then printed.

A DO-LOOP is entered that proceeds through the plate and fluid temperature arrays and establishes a print buffer of these values converted to degrees Fahrenheit. The plate and fluid temperatures are then printed.

A cooling device heading is printed. The radiator control temperature is converted to degrees Fahrenheit. If the "radiator on" flag is set, the radiator heat value and temperature setting is written. If the "highload evaporator on" flag is greater than zero, the amount of heat rejected by the high-load evaporator is printed. If the "topping elevator on" flag is greater than zero, the amount of heat rejected by the topping elevator is set. If the "ammonia boiler on" flag is greater than zero, the amount of heat rejected by the ammonia boiler is printed.

The cabin pressure for each cabin is established as a sum of its oxygen and nitrogen pressure. If the pressure in either or both cabins is greater than a specified tolerance, the cabin number, total pressure, nitrogen pressure, and oxygen pressure are printed.

At this point, processing terminates for routine ECPRT and control returns to the calling routine.

5.3.3 Assumptions and Limitations

None.

5.3.4 Method

None.

5.3.5 Routine Input/Output Variables

The ECPRT input/output variables are presented in table 5.3-I.

5.3.6 Functional Logic Flow

A functional logic flow is not provided. The functional level PDL for ECPRT is presented in figure 5.3-1.

5.3.7 Diagnostics and Debug

None.

5.3.8 Special Comments

None.

5.3.9 References

None.

TABLE 5.3-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine ECPRT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
FTEMP	--	Real	I	--	C	--	Fluid temperatures at each node in the ECS model.
H201ON	--	Intg	I	--	C	--	Specifies on/off status of high-load evaporator.
H202ON	--	Intg	I	--	C	--	Specifies the on/off status of the topping evaporator.
HTH201	--	Real	I	--	C	--	Heat rejected by the high-load evaporator.
HTH202	--	Real	I	--	C	--	Heat rejected by the topping vaporator.
HTNH3	--	Real	I	--	C	--	Heat rejected by the amonia boiler.
HTRAD	--	Real	I	--	C	--	Heat rejected by the radiator.
HTTOT	--	Real	I	--	C	--	Total heat dissipated by the ATCS cooling systems.
IGMT	--	Logical	I	--	C	--	Defines the time reference (TRUE for GMT, FALSE for GET).

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.3-I.- Continued

Routine ECPRT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IPRES	--	Intg	I	--	C	--	Indicator for present value's location in the temperature arrays.
LU	--	Intg	I	--	C	--	Logical unit number of user's terminal.
NH3ON	--	Intg	I	--	C	--	Specifies the on/off status of the ammonia boiler.
PN2	--	Real	I	--	C	--	Nitrogen pressure in each cabin.
PO2	--	Real	I	--	C	--	Oxygen pressure in each cabin.
PROCON	--	Intg	I	--	C	--	Processor constants array.
PTEMP	--	Real	I	--	C	--	Temperature of each node in the ATCS model.
RADON	--	Intg	I	--	C	--	Specifies the on/off status of the radiator.
TCNTRL	--	Real	I	--	C	--	Control value used to maintain the output temperature of the radiator.
THEAT	--	Real	I	--	C	--	Total heat load on the ATCS.

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.3-I.- Concluded

Routine ECPRT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TIME	--	Real	I	--	C	--	Point on the time line presently being evaluated.
WCP	--	Real	I	--	C	--	Heat capacitance of the fluid at each node of the ATCS model.

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory

```
1*
1 BEGIN ECPRT
2     SET OUTPUT LOGICAL UNIT = LOGICAL UNIT USED
2     IF(USE PRINTER FLAG > ZERO)
2     THEN
3        SET OUTPUT LOGICAL UNIT = PRINTER FLAG
3        CALL EXEC FOR PAGE EJECT AND LINE COUNTER RESET
2     ENDIF
2*
2     WRITE HEADING
2     PERFORM GMET TO CONVERT TIME TO DAY,HR,MIN,SEC
2     IF(GREENWICH MEAN TIME FLAG = FALSE)
2     THEN
3        WRITE TIME GROUP IN GROUND ELAPSED TIME
2     ELSE
3        WRITE TIME GROUP IN GREENWICH MEAN TIME
2     ENDIF
2*
2     WRITE HEAT LOAD SUMMARY
2     SET DOFOR LIMIT
3     DOFOR I = 1,LIMIT,1
4        CONVERT PLATE AND FLUID TEMPERATURES TO DEG F
4        SET PRINT POSITIONS TO PROPER VALUES
4        WRITE PLATE AND FLUID TEMPERATURES
3     ENDDO
2*
2     WRITE COOLING DEVICES HEADING
2     CONVERT COOLING DEVICES TEMPERATURES TO DEG F
2     IF(RADIATOR ON FLAG > ZERO)
2     THEN
3        WRITE RADIATOR HEAT VALUE AND TEMPERATURE SETTING
2     ENDIF
2*
2     IF(HIGHLOAD EVAPORATOR ON FLAG > ZERO)
2     THEN
3        WRITE HIGHLOAD EVAPORATOR HEAT REJECTED
2     ENDIF
2*
2     IF(TOPPING EVAPORATOR ON FLAG > ZERO)
2     THEN
3        WRITE TOPPING EVAPORATOR HEAT REJECTED
2     ENDIF
2*
2     IF(AMMONIA BOILER ON FLAG > ZERO)
2     THEN
3        WRITE AMONNIA BOILER HEAT REJECTED
2     ENDIF
```

Figure 5.3-1.- ECPRT functional level PDL.

```
2*
3     DOFOR I = 1,2,1
4        COMPUTE TOTAL CABIN PRESSURES
4        IF(CABIN PRESSURE > TOLERANCE)
4        THEN
5           WRITE CABIN,N2,02 PRESSURES
4        ELSE
5           CONTINUE DO FOR
4        ENDIF
3     ENDDO
2*
1 END ECPRT
1*
```

Figure 5.3-1.- Concluded.

5.4 ROUTINE NAME - EPPRT

5.4.1 Purpose

The EPS print routine (EPPRT) provides a detailed print of the EPS network solutions at each point in the time line where the electrical loads change.

5.4.2 Functional Description

Initially, the logical unit number is defined as the user's logical unit number. If the user desires the output to be sent to another device (as indicated by PROCON(4)), the output logical unit number is redefined. An output heading is printed, along with the number of iterations necessary to obtain a solution. If the user's specified output device is the printer, a call is first made to the RTE EXEC to cause a page-eject and reinitialize the line counter.

A call is now made to the processor routine GMET to convert ground elapsed time (GET) decimal hours into the appropriate reference (Greenwich mean time or GET). If the time reference flag IGMT is false, the GET is printed. Otherwise, the Greenwich mean time (GMT) is printed.

A power summary is printed, which consists of the source power, bus power, power load applied to the EPS model, and the subpower.

A DO-LOOP is now entered, which executes until all the branches in the EPS model have been examined. This will establish and print the EPS parameters.

Initially, a Hollerith diode status indicator is set to blanks. If the current branch has diode information, then the Hollerith diode indicator is set to "FW". If the current diode information is less than a lower indexed diode information plus a tolerance, the mnemonic diode status indicator is set to "RV".

A DO-LOOP is now entered to determine if there is power source information associated with the current branch being examined by the outer DO-LOOP. If there is power source information, it is printed along with the diode information, switch status, branch number, electrical current information, output voltage, and the supplied current. Execution then returns to the top of the outer DO-LOOP.

If there is no power source number, a DO-LOOP is entered to determine if a load number exists for the current branch under analysis. If the branch contains a load, branch information is printed that describes the branch number, switch state, diode information, electrical current information, voltage information, actual power dissipated in the branch, loads due to resistive power equipment, and loads due to constant power equipment. Control then returns to the top of the outer DO-LOOP.

If the branch under examination is neither a load nor a power source, information is printed that describes the branch number, switch state, diode information, and electrical current information. Control then returns to the top of the outer DO-LOOP.

When all branches have been examined, execution exits the DO-LOOP. If the user's specified output device is not the printer, a blank line is printed for spacing.

A DO-LOOP is now entered, which is indexed from one to the number of nodes and is incremented by five. The node voltages are then displayed in groups of five.

At the termination of the DO-LOOP, control returns to the calling routine.

5.4.3 Assumptions and Limitations

None.

5.4.4 Method

None.

5.4.5 Routine Input/Output Variables

The EPPRT input/output variables are presented in table 5.4-I.

5.4.6 Functional Logic Flow

A functional logic flow is not presented. The functional level PDL for EPPRT is presented in figure 5.4-1.

5.4.7 Diagnostics and Debug

None.

5.4.8 Special Comments

None.

5.4.9 References

None.

TABLE 5.4-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine EPPRT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
BUSPWR	--	Real	I	--	C	--	Total power being dissipated at the local bus level of the EPS model.
CBNCH	--	Real	I	--	C	--	Electrical current through each branch of the EPS model.
DIODE	--	Real	I	--	C	--	Diode characteristics for each branch of the EPS model.
ICPWR	--	Intg	I	--	C	--	Loads due to constant power equipment.
IGMT	--	Logical	I	--	C	--	Defines the time reference (TRUE for GMT, false for GET).
IRPWR	--	Intg	I	--	C	--	Loads due to resistive power equipment.
ITER	--	Intg	I	--	A	--	Number of iterations required to obtain a solution.
LOAD	--	Intg	I	--	C	--	Correlation of each load to its branch location.
LU	-	Intg	I	--	C	--	Logical unit number of user's terminal.

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.4-I.- Continued

Routine EPPRT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
NBRSRC	--	Intg	I	—	C	--	Number of power sources in the EPS model.
NCRCT	--	Intg	I	—	C	--	Number of branches in the EPS model.
NLDS	--	Intg	I	—	C	--	Number of loads in the EPS model.
NNODE	--	Intg	I	—	C	--	Number of nodes in the EPS model.
PROCON	--	Intg	I	—	C	--	Processor constants, including output unit number for displays.
PWRLOD	--	Real	I	—	C	--	Total 28-volt equivalent load applied to the EPS model.
SBPWR	--	Real	I	—	C	--	Power calculation.
SRCCUR	—	Real	I	—	C	--	Current being supplied by each power source.
SRCPWR	--	Real	I	—	C	--	Power being supplied by each power source.

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.4-I.- Concluded

Routine EPPRT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
SRCTYP	--	Intg	I	--	C	--	Power source information containing branch locations, source type, and a relative count of each type of source.
SRCVLT	--	Real	I	--	C	--	Output voltage of each power source.
SWTCH	--	Intg	I	--	C	--	Specifies switch status in each branch of the EPS model.
TIME	--	Real	I	--	C	--	Point on time line presently being evaluated.
VLOAD	--	Real	I	--	C	--	Voltage at each load in the EPS model.
VNODE	--	Real	I	--	C	--	Voltage at each node in the EPS model.

NOTES:

TYPE

Free	Dubl	18CH	Mix
Intg	2CH	36CH	Char
Real	6CH	72CH	Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

```
1*
1 BEGIN EPPRT
2     SET OUTPUT LOGICAL UNIT = LOGICAL UNIT USED
2     IF(USE PRINTER FLAG > ZERO)
2     THEN
3        SET OUTPUT LOGICAL UNIT = PRINTER FLAG
3        CALL EXEC FOR PAGE EJECT AND LINE COUNTER RESET
3        WRITE HEADING
3        BRANCH TO -LBL75-
2     ELSE
3        WRITE HEADING
2     ENDIF
2*
2   -LBL75-
2     PERFORM GMET TO CONVERT TIME TO DAY,HR,MIN,SEC
2     IF(GREENWICH MEAN TIME FLAG = FALSE)
2     THEN
3        WRITE TIME GROUP IN GROUND ELAPSED TIME
2     ELSE
3        WRITE TIME GROUP IN GREENWICH MEAN TIME
2     ENDIF
2*
2     WRITE POWER SUMMARY
2*
3     DOUNTIL(NUMBER OF BRANCHES IS REACHED)
4        SET DIODE NAME = BLANKS
4        IF(DIODE(POSITION) >= TOLERANCE)
4        THEN
5           SET DIODE NAME = CHARACTER STRING "FW"
4        ELSE
5           BRANCH TO -LBL100-
4        ENDIF
4*
4        IF(DIODE(POSITION) > DIODE(POSITION)+ TOLERANCE)
4        THEN
5           SET DIODE NAME TO CHARACTER STRING "RV"
4        ENDIF
4*
4  -LBL100-
5        DOFOR J = 1,NUMBER OF SOURCES,1
6           IF(SOURCE TYPE(POSITION) NOT= LOOP COUNTER)
6           THEN
7              CONTINUE DOFOR
6           ELSE
7              BRANCH TO -LBL200-
6           ENDIF
5        ENDDO
4        BRANCH TO -LBL250-
4*
4  -LBL200-
4        WRITE BRANCH INFORMATION CONTAINING A POWER SOURCE
4        CONTINUE DOUNTIL
```

Figure 5.4-1.- EPPRT functional level PDL.

```
4*
4 -LBL250-
5       DOFOR J = 1,NUMBER OF LOADS,1
6          IF(LOAD(LOOP COUNTER) NOT= LOOP COUNTER)
6          THEN
7             CONTINUE DOFOR
6          ELSE
7             BRANCH TO -LBL350-
6          ENDIF
5       ENDDO
4       BRANCH TO -LBL400-
4*
4 -LBL350-
4       SET PRINT VALUES
4       WRITE BRANCH INFORMATION CONTAINING A LOAD
4       CONTINUE DOUNTIL
4*
4 -LBL400-
4       WRITE REMAINING BRANCH INFORMATION
3    ENDDO
2*
2    IF(USE PRINTER FLAG <= ZERO)
2    THEN
3       WRITE A BLANK LINE FOR SPACING
2    ELSE
3       BRANCH TO -LBL510-
2    ENDIF
2*
2 -LBL510-
3    DOFOR I = 1,NUMBER OF NODES,5
4       SET COUNTER VALUE
4       IF(COUNTER > NUMBER OF NODES)
4       THEN
5          COUNTER = NUMBER OF NODES
4       ENDIF
4       WRITE NODE INFORMATION
3    ENDDO
2*
1 END EPPRT
1*
```

Figure 5.4-1.- Concluded.

COASTING STATE VECTOR PREDICTOR (INCLUDING AEG) PROCESSOR (COAST)

1.0 PURPOSE

The COAST processor will propagate/predict a given state vector to a specified stop condition. Its purpose is to provide the FDS user with the capabilities of a trajectory propagator (the maneuver time search routine) in an independently executable processor.

2.0 FUNCTIONAL DESCRIPTION

The COAST processor advances/propagates a state vector to a user specified stop condition. Two propagation algorithms, a conic (two-body) propagation routine and the analytical ephemeris generator, are available for the user to select from. Several stop conditions (the list and their definitions are provided in table I) are available to choose from. A stop condition value (as defined in table I) to be used in conjunction with the stop condition code must also be specified by the user. Depending on the stop condition code and value combination, the COAST processor will propagate a state vector either forward or backward in time to achieve the stopping value. An elaboration of the processor capability of propagating a state vector backward in time is given in the Assumptions and Limitations section (sec. 3.0). The COAST processor has the capability of displaying the initial state vector and the final state vector to the user. This display is optional and the user must specify whether or not it will be generated. The two state vectors (initial and final) will, in all cases, be stored by the processor in a data element that the user must define.

3.0 ASSUMPTIONS AND LIMITATIONS

The logic that propagates a vector backwards in time works in the following manner. If a delta-T is input, that time must be a negative value. If a GET is input, the time must be less than the vector time. If any other stop condition is chosen, the threshold time must be less than the vector time. The initial vector will be propagated back to that time. Then the state will be advanced forward in time to find the specified stop condition. This means, for example, that the stop value for the Nth apsis must be a positive value.

4.0 PROCESSOR INPUT/OUTPUT

a. Processor interface table - All input to and output from the COAST processor is achieved through an interface table. A description of the COAST interface table is given in table 4-I. A brief description of the interface table parameters is given here.

   GLOCON is a set of constants universal to all processors that will be maintained in the master data base. COAST will access this away for the constants it needs. The default values are stored in !!GLCN. SESCON is a set of session related constants generated by the user upon execution of

the system utility processor BASTM. The default values are stored in !SESCN and are standard for all processors. PROCON is a set of constants required by COAST. These constants are mainly used for iteration tolerances and internal print analysis.

INVEC is the position/velocity input vector. This vector is updated by COAST to a specified stop condition.

SVSTOP is the state vector stop condition code. The valid codes and their definitions are given in table 4-I.

STOPVL is the value corresponding to the state vector stop condition code (SVSTOP). The definitions of STOPVL for each stop condition code is given in table 4-I.

THRT is a three real parameter array containing (in hours, minutes, and seconds) the threshold time or GET. This is the time at which the processor begins to search for the stop condition.

SVPROP is the state vector propagator selection code. Either a conic (two-body) or the Analytical Ephemeris Generator (AEG) may be chosen.

PRINT is the display print option code with which the user can specify whether or not to print the Coast State Vector Display. The display will include the initial state vector and the final state vector.

PVTAB is the position/velocity state vector phase table output. The initial state vector and the state vector at the terminal condition (final state vector) are output to the table in the TEG reference axis system and Cartesian element set.

b. Interface table data array definitions - A description of the interface table data arrays is given in table 4-II.

c. Interface table data files definitions - None.

d. Processor solicited (prompted) inputs - None.

e. Processor displays and display parameter definition table - A format of the processor display is given in table 4-III. The display parameters definitions are provided in table 4-IV. An example display is provided in table 4-V. All parameters are displayed in user defined external units except for the period that is displayed in minutes.

f. Processor message table - The messages that are generated by COAST are given in table 4-VI.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-VII.

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR COAST

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Value stored in default interface table	Definition
GLOCON	AWA	Free	I	180	180	!!GLCN	Global constants array, master data base elements
SESCON	AWA	Free	I	90	90	!SESCON	Session constants array
PROCON	AWA	Free	I	20	20		Processor constants array
INVEC	AWA	Real	I	30	15		Position/velocity input state vector
SVSTOP	AWA	6CH	I	3	1		State vector stop condition code "DLT"; Threshold time + init. state time "APS"; Nth apsidal crossing "TIM"; Update state to a time "ALT"; Search for altitude "LAT"; Search for latitude "LON"; Search for longitude "RAD"; Search for radius "APO"; Search for Nth apogee "PER"; Search for Nth perigee "OPT"; Optimum node shift point "ARG"; Argument of latitude

NOTES	CLASS	TYPE		USE
		Free	2CH 72CH	I = Input
	AWA	Intg	6CH Mix	O = Output
	Disk	Real	18CH Symb	I/O = Input/Output
		Dubl	36CH	

TABLE 4-I.- Continued

PROCESSOR COAST

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Value stored in default interface table	Definition
STOPVL	AWA	Real	I	2	1		Value of state vector stop condition
							When SVSTOP = ; → STOPVL =
							"DLT" → N/A
							"APS" → Nth crossing
							"TIM" → N/A
							"ALT" → Altitude
							"LAT" → Latitude
							"LON" → Longitude
							"RAD" → Radius
							"APO" → Nth apogee
							"PER" → Nth perigee
							"OPT" → N/A
							"ARG" → Argument of latitude
THRT	AWA	Real	I	6	3		Time increment, maneuver time, or threshold time
							When SVSTOP = ; → THRT =
							"DLT" → Time increment from state vector time in hours, minutes, and seconds
							"TIM" → Maneuver GET in hours, minutes, and seconds
							ALL OTHERS → Threshold GET to begin search for STOPVL, in hours, minutes, and seconds

NOTES

CLASS: AWA, Disk

TYPE: Free, Intg, Real, Dubl; 2CH, 6CH, 18CH, 36CH; 72CH, Mix, Symb

USE: I = Input; O = Output; I/O = Input/Output

TABLE 4-I.- Concluded

PROCESSOR COAST

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Value stored in default interface table	Definition
SVPROP	AWA	6CH	I	3	1		State vector propagator option flag "CON"; conic "AEG"; Analytical ephemeris generator
PRINT	AWA	6CH	I	3	1		Display print code "YE"; Print coast state vector display "NO"; Do not print display
PVTAB	AWA	Real	O	120	(30,2)		Position velocity phase table; contains the initial and final state vectors

NOTES	CLASS	TYPE			USE
		Free	2CH	72CH	I = Input
	AWA	Intg	6CH	Mix	O = Output
	Disk	Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-II.- INTERFACE TABLE DATA ARRAY DEFINITIONS

PROCESSOR COAST

Array name	Index location	Default value	Definition
GLOCON	(1) : (180)	!!GLCN	Global constants array, master data base element; see table 7.2-III in vol. I for definition of contents.
SESCON	(1) : (90)	!SESCN	Session constants array; see table 7.2-II(a) in vol. I for definition of contents.
PROCN	(1)	0	Output unit on which to write all print; the default value is a flag selecting the user terminal; a positive value identifies the output unit.
	(2)	0	Not used
	(3)	P	Internal print flag option = 0 No internal print = 1 Internal print
	(4)	0	Not used
	(5)	20	Maximum iterations limit
	(6)	0	Not used
	(7)	-25.0	First guess change in value of independent variable; used in subroutine ITERV
	(9)	2.0E-4	Transfer angle tolerance (radius)
	(11)	6.0E-4	Eccentricity tolerance
	(13)	2.0E-4	Covergence tolerance for travel angle (radians)
	(15)	0	Independent variable guess switch = 0 No independent variable input = 1 A guess for independent variable is input
	(16)	0	Not used
	(17)	0.0	An initial guess for the independent variable
	(19)	250.0	Radius tolerance (feet)
PVTAB			Position/velocity phase table; see figure 7.3-13 in vol. I for a definition of contents; contains 2 vectors
	(1,1) : (30,1)		(1) Initial state vector
	(1,2) : (30,2)		(2) Final state vector

TABLE 4-II.- Concluded

PROCESSOR COAST

Array name	Index location	Default value	Definition
INVEC	(1) : (15)		Position/velocity input state vector; see table 1.2-VI in vol. I for definition of contents
THRT	(1) (2) (3)		Hours component of threshold time or ΔGET Minutes component of threshold time or ΔGET Seconds component of threshold time or ΔGET

TABLE 4-III.- PROCESSOR DISPLAY TABLE

PROCESSOR COAST

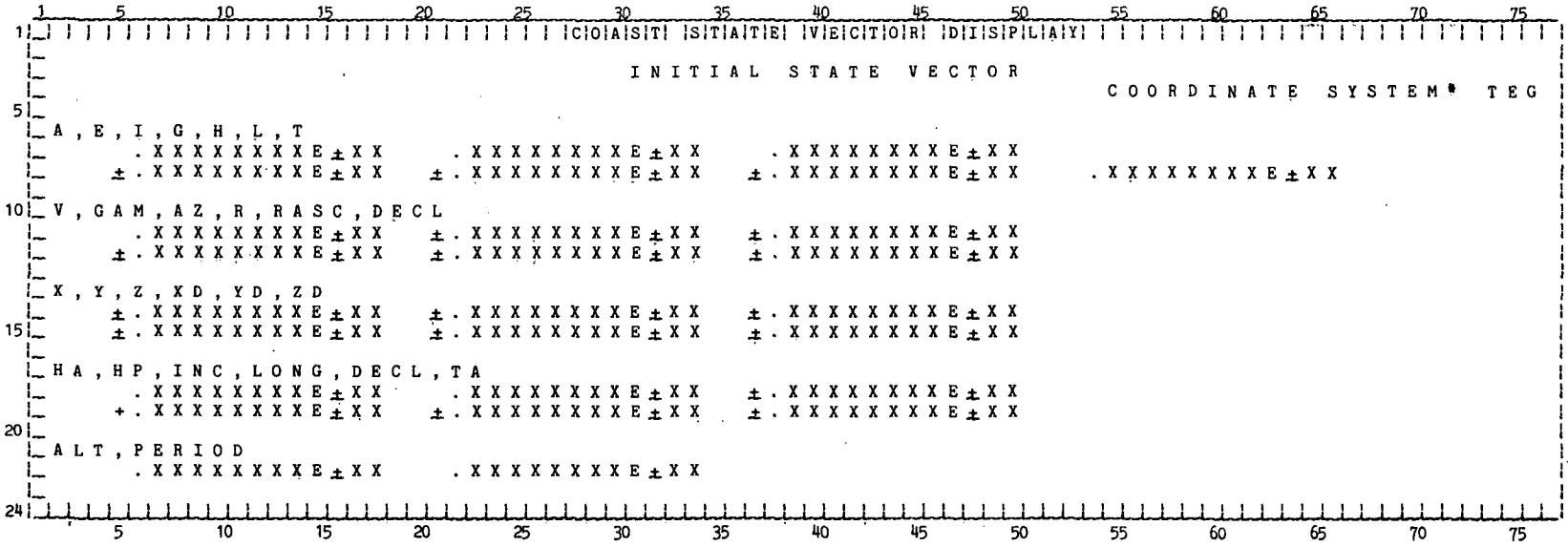


TABLE 4-III.- Concluded

PROCESSOR COAST

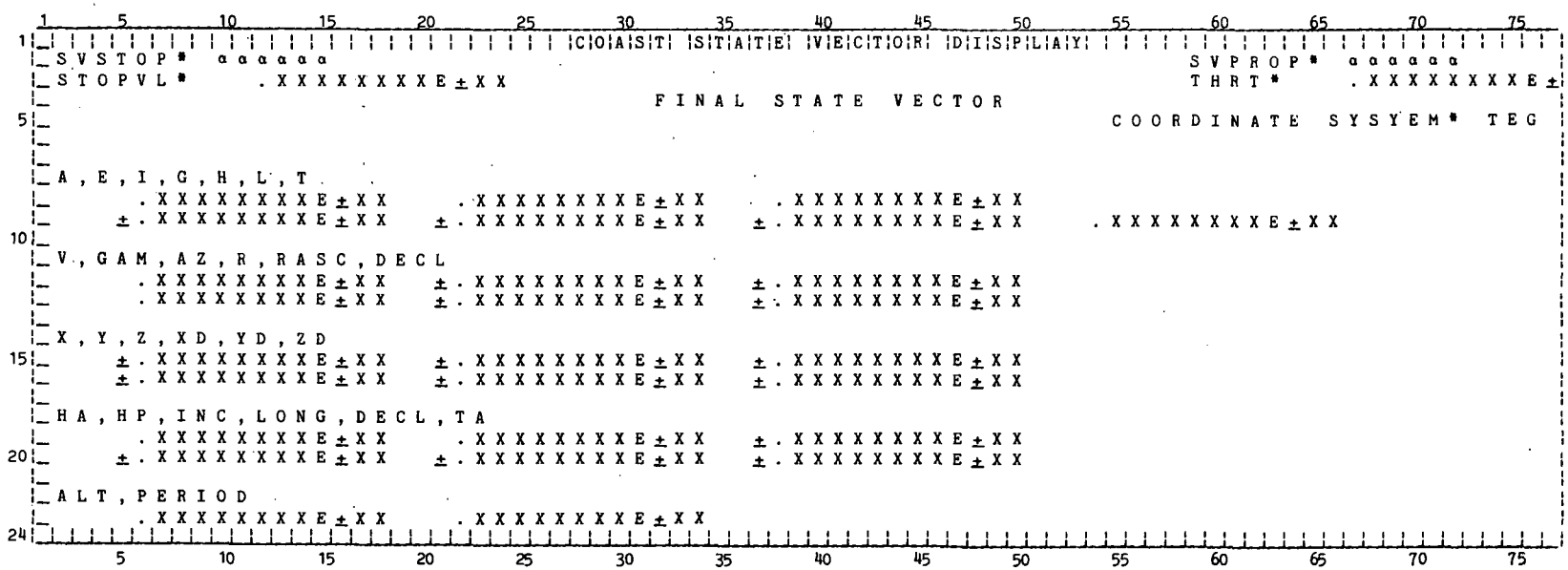


TABLE 4-IV.- DISPLAY PARAMETER DEFINITION TABLE

PROCESSOR COAST

COAST STATE VECTOR DISPLAY	
Display parameter label	Parameter definition
A	Semimajor axis
E	Eccentricity
I	Inclination of orbit plane
G	Argument of perigee
H	Right ascension of orbit plane ascending node
L	Mean anomaly
T	Greenwich mean time of vector
V	Inertial velocity magnitude
GAM	Geocentric flight path angle
AZ	Geocentric flight azimuth
R	Radius magnitude
RASC	Richt ascension
DECL	Declination
X	Cartesian position components, TEG: in X direction
Y	Cartesian position components, TEG: in Y direction
Z	Cartesian position components, TEG: in Z direction
XD	Cartesian velocity components, TEG: in X direction
YD	Cartesian velocity components, TEG: in Y direction
ZD	Cartesian velocity components, TEG: in Z direction
HA	Height of apogee above reference radius
HP	Height of perigee above reference radius
INC	Inclination of orbit plane
LONG	Geographic longitude
DECL	Declination
TA	True anomaly
ALT	Height of vehicle above reference radius
PERIOD	Inertial period of orbit, minutes
SVSTOP	User input stop condition code
STOPVL	User input stop condition value
SVPROP	User input propagation code
THRT	User input-threshold time, seconds

TABLE 4-V.- EXAMPLE OF THE COAST STATE VECTOR DISPLAY

```
                    COAST STATE VECTOR DISPLAY

                         INITIAL STATE VECTOR
                                                    COORDINATE SYSTEM* TEG

A,E,I,G,H,L,T                                          AEIMA
     .22117232E+08    .16253954E-02    .89999985E+02
    -.12999744E+03    .15492438E+03    .19191464E+03    .56816000E+05

V,GAM,AZ,R,RASC,DECL                                   POLRDC
     .25187832E+05   -.19196253E-01    .00000000E+00
     .22154208E+08    .15492438E+03    .61878777E+02

X,Y,Z,XD,YD,ZD                                         CARXYZ
    -.94571840E+07    .44251550E+07    .19537368E+08
     .20124344E+05   -.94164766E+04    .11864549E+05

HA,HP,INC,LONG,DECL,TA                                 HAPLDC
     .12248080E+07    .12156120E+07    .89999985E+02
    -.82457016E+02    .61878777E+02    .19187622E+03

ALT,PERIOD
     .12266680E+07    .91984589E+02

                    COAST STATE VECTOR DISPLAY
SVSTOP* LAT                                            SVPROP* AEG
STOPVL*    .90000000E+02                               THRT*    .00000000E+00
                         FINAL STATE VECTOR
                                                    COORDINATE SYSTEM* TEG

A,E,I,G,H,L,T                                          AEIMA
     .22103192E+08    .21277443E-02    .89999893E+02
    -.91094877E+02    .15492438E+03    .18101920E+03    .57247703E+05

V,GAM,AZ,R,RASC,DECL                                   POLRDC
     .25182305E+05   -.21639327E-02    .11836301E+03
     .22150212E+08   -.86712570E+02    .89999878E+02

X,Y,Z,XD,YD,ZD                                         CARXYZ
     .25180464E+01   -.43838371E+02    .22150212E+08
     .22808855E+05   -.10672605E+05   -.97479451E+00

HA,HP,INC,LONG,DECL,TA                                 HAPLDC
     .12249280E+07    .12159840E+07    .89999893E+02
     .34102249E+02    .89999878E+02    .18101492E+03

ALT,PERIOD
     .12244720E+07    .91985062E+02
$:%
```

TABLE VI.- PROCESSOR MESSAGE TABLE

PROCESSOR COAST

MSG no.	Message ID block	Message text block and explanation
1	*COAST**CINP*	INPUT VECTOR NOT TEG/CARTESIAN Meaning: Incorrect reference axis and/or element set Severity: Processor will terminate and control will be returned to the FDS Executive Action required by user: Transform vector to TEG/Cartesian and resume execution
2	*COAST**CINP*	THE FOLLOWING CODE IS INVALID INPUT Meaning: An invalid code has been entered for SVSTOP, SVPROP, or PRINT Severity: Processor will terminate and control will be returned to the FDS Executive. Action required by user: Use the Interface Table Editor to enter valid code and resume execution.
3	***	VEHICLE REENTERED AT T = ± .XXXXXXXXE±XX Meaning: The orbital period is less than the period of a 40-n. mi. circle. This message is generated by AEG. Severity: Drag is shut off. Processor continues normal execution. Action required by user: None, if reentry is expected (as in a deorbit maneuver). Otherwise, the user should check his data for possible errors.

TABLE 4-VII.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR COAST

Processor name	Processor abstract prompt (maximum 256 characters)
COAST	The COAST processor propagates a given state vector to a specified stop condition and generates the state vector at the terminal condition. Propagation may be forward or backward in time.

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
GLOCON	Global constants array. Master data base element. (Refer to FDS-1, SDD, vol. VI, rev. 1, sec. 1.2.2.3 for description.) Normally defaulted to !!GLCN.
SESCON	Session constants array. (Refer to FDS-1, SDD, vol. VI, rev. 1, sec. 1.2.2.1 for description.) Normally defaulted to !SESCN.
PROCON	Processor constants array. See COAST interface table data array definitions for description.
INVEC	Standard position/velocity input state vector. May be the name of a data element containing a position/velocity state vector.
SVSTOP	Selects state vector stop condition. Valid codes are "DLT", "APS", "TIM", "ALT", "LAT", "LON", "RAD", "APO", "PER", "OPT", "ARG". See COAST interface table definitions for explanation of codes.
STOPVL	Value of state vector stop condition. See COAST interface table definitions for explanation.
THRT	Time increment, maneuver GET, or threshold GET.
SVPROP	State vector propagation selection code. Valid codes are "CON" (for conic (two body) propagation) and "AEG" (for analytical ephemeris generator).
PRINT	Display print code. Valid codes are "YE" or "NO"
PVTAB	Position velocity phase table to contain the initial state vector and the final state vector.

5.0 PROCESSOR ROUTINES

5.1 ROUTINE NAME - MAIN PROGRAM COAST

5.1.1 Purpose

COAST is the main program for the processor COAST. Its purpose is to act as a driver for the search time routine (SIRCH).

5.1.2 Functional Description

COAST is divided into five functional segments. They are (1) a call to the system routine RMPAR, (2) a call to the input segment (CINP), which handles all processor input, (3) a call to the search time routine (SIRCH), which searches for a state vector time based on some specified desired condition and generates the final state vector, (4) a call to the output segment (COUTP), which generates all processor displays and output, and (5) an exit processor segment.

5.1.3 Assumptions and Limitations

None.

5.1.4 Method

None.

5.1.5 Routine Input/Output Variables

The COAST routine input/output variables are presented in table 5.1-I.

5.1.6 Functional Logic Flow

A functional logic flow is not provided. A detailed flow for COAST is presented in figure 5.1-1.

5.1.7 Diagnostics and Debug

The third element of the interface table parameter PROCON is a debug print flag. It is used by the subroutine SIRCH to print out debug information. It is also used by the input segment (CINP) for debug print.

5.1.8 Special Comments

The following subroutines are utilized, and are documented under the indicated sections of the FDS-1 system design document:

XPXIT - Volume VII
SIRCH - Volume III

5.1.9 References

None.

TABLE 5.1-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine COAST

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IPRAM	--	Intg	I/O	--	SP	--	Five parameters retrieved from RTE system

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory
SP = RTE system parameter

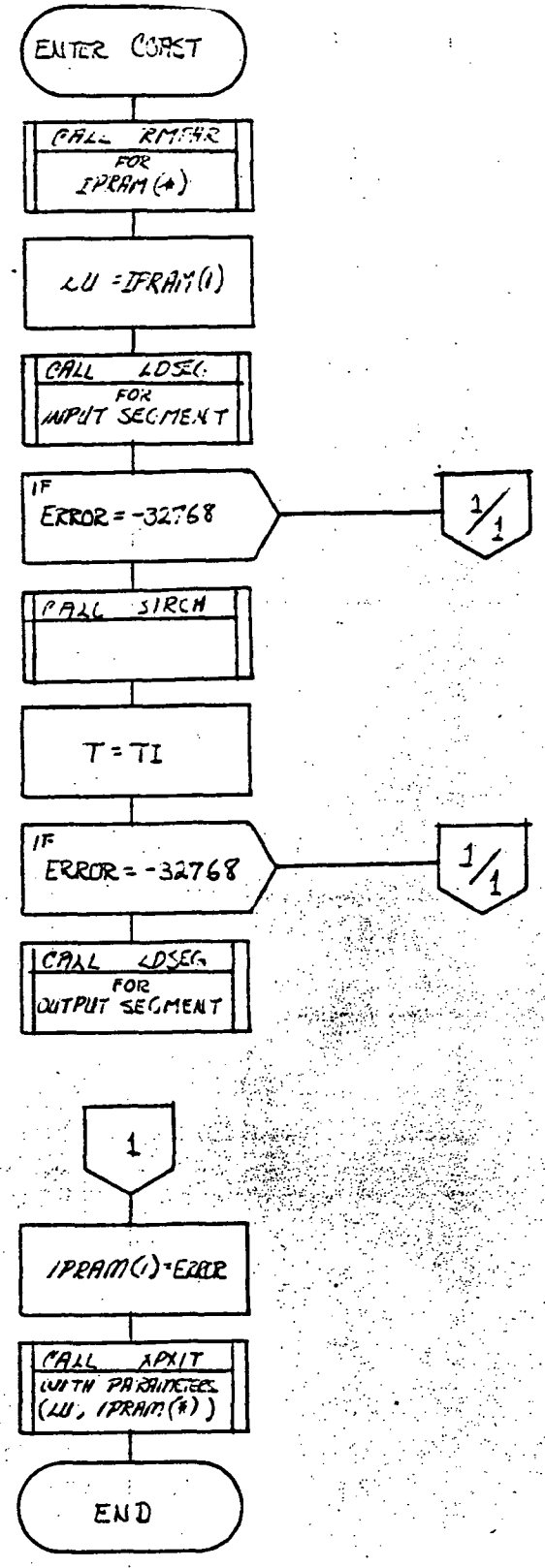


Figure 5.1-1.- Detailed flow for COAST routine.

5.2 ROUTINE NAME - CINP

5.2.1 Purpose

CINP is the input routine for the processor COAST. Its purpose is to provide all input processing for COAST by retrieving interface table data and initializing of constants and variables.

5.2.2 Functional Description

The routine CINP (1) calls a system utility routine to retrieve interface table data, (2) converts variables and arrays to internal units, (3) calls a routine to decode input codes, and (4) stores data in a common block.

5.2.3 Assumptions and Limitations

None.

5.2.4 Method

None.

5.2.5 Routine Input/Output Variables

The CINP routine input/output variables are presented in table 5.2-I. The error message generated by CINP is described in the COAST processor message table (msg. no. 1).

5.2.6 Functional Logic Flow

The functional logic flow for CINP is presented in figure 5.2-1. A detailed flow is presented in figure 5.2-2.

5.2.7 Diagnostics and Debug

If the third element of the interface table array PROCON is set to one, then CINP will print out the initial position, velocity, time, and weight of the vehicle. It also prints out the search option maneuver location parameter and threshold time. A format of the debug print is given in table 5.2-II, and an example display is given in table 5.2-III.

5.2.8 Special Comments

CINP utilizes the following subroutines, which are documented under the indicated sections of the FDS-1 system design document.

XPGET - Volume VII
XRMOV - Volume VII
SSVUC - Volume VII

5.2.9 References

None.

TABLE 5.2-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine CINP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IPRAM		Intg	I		C		Five parameters retrieved from the RTE system.
GLOCON		Real	I		IT	GLOCON	Global constants array.
SESCON		Intg	I		IT	SESCON	Session constants array.
PROCON		Real	I		IT	PROCON	Processor constants array.
BVEC		Real	I		IT	INVEC	Base vector.
SVSTOP		6CH	I		IT	SVSTOP	State vector stop condition code.
STOPVL		Real	I		IT	STOPVL	Value of state vector stop condition.
THRT		Real	I		IT	THRT	Threshold time to begin search for SVSTOP.
PRINT		6CH	I		IT	PRINT	Display print code.
SVPROP		6CH	I		IT	SVPROP	State vector propagator selection flag.
CONST		Real	O		C		COAST constants array.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.2-I.- Concluded

Routine CINP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
CINPUT		Real	O		C		Input parameters array.
RP	--	Real	O	--	C		Vehicle radius vector.
VP	—	Real	O	—	C		Vehicle velocity vector.
T	--	Real	O	--	C		State vector time.
BVEC	—	Real	O	—	C		Base vector in external units.
CVEC	--	Real	O	--	C		Base vector in internal units.
INTBUF	—	Intg	O	—	C		Interface table header.

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.2-II.- DEBUG PRINT DISPLAY FORMAT

PROCESSOR CINP

```
INITIAL CONDITIONS:  X, Y, Z, XD, YD, ZD, T, WEIGHT
   ±.XXXXXXXXE±XX   ±.XXXXXXXXE±XX   ±.XXXXXXXXE±XX
   ±.XXXXXXXXE±XX   ±.XXXXXXXXE±XX   ±.XXXXXXXXE±XX
   ±.XXXXXXXXE±XX   ±.XXXXXXXXE±XX

SEARCH OPTION= XX

MANEUVER LOCATION PARAMETER= XXXX.XXXX

THRESHOLD TIME= ±XXXXXXXX.X
```

TABLE 5.2-III.- EXAMPLE OF THE CINP DEBUG PRINT DISPLAY

```
INITIAL CONDITIONS: X,Y,Z                    EIGHT
   .16723492E+08    .41064                  968430E+08
  -.89373027E+04    .24174262E+05  -.38703101E+04
   .67925125E+05    .29280000E+06

SEARCH OPTION =  2

MANEUVER LOCATION PARAMETER =      .0000

THRESHOLD TIME =      70027.1
```

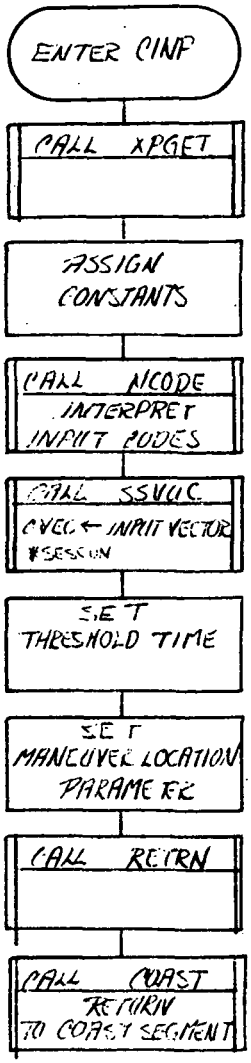
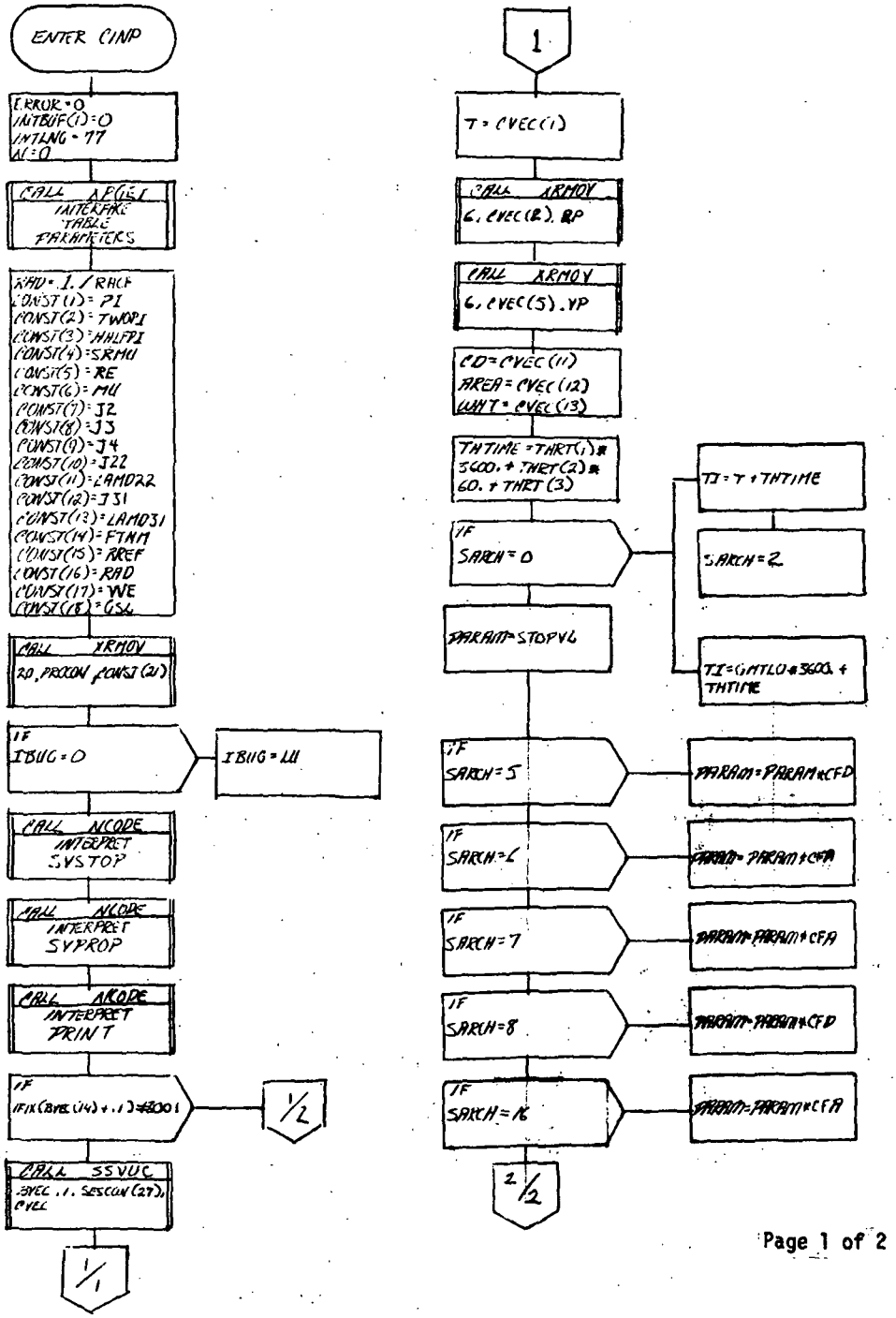


Figure 5.2-1.- CINP functional logic flow.



Page 1 of 2

Figure 5.2-2.- Detailed flow for CINP routine.

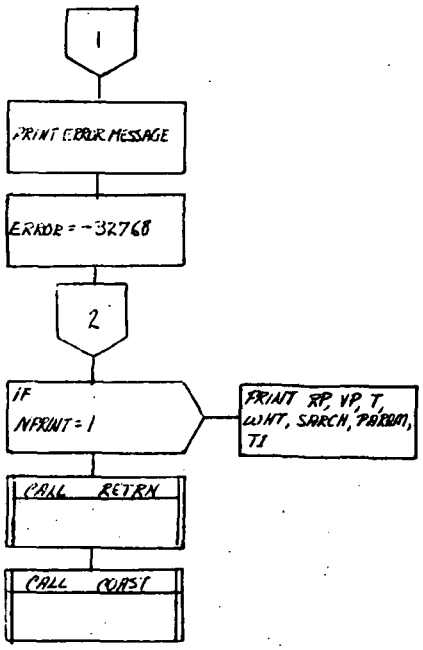


Figure 5.2-2.- Concluded.

5.3 ROUTINE NAME - COUTP

5.3.1 Purpose

The output segment (COUTP) calls a subroutine to generate the COAST state vector display. COUTP then outputs the initial and final state vectors to a phase table.

5.3.2 Functional Description

If the user chooses the COAST state vector display option, COUTP will call a subroutine to generate the processor display. COUTP builds position/velocity phase table vectors from the input and updated state vectors, then calls a system utility routine to output these two vectors to a phase table.

5.3.3 Assumptions and Limitations

None.

5.3.4 Method

None.

5.3.5 Routine Input/Output Variables

The COUTP routine input/output variables are presented in table 5.3-I.

5.3.6 Functional Logic Flow

A functional logic flow is not provided. A detailed flow for COUTP is presented in figure 5.3-1.

5.3.7 Diagnostics and Debug

None.

5.3.8 Special Comments

COUTP utilizes the following subroutines, which are documented under the indicated sections of the FDS-1 system design document.

SSVUC - Volume VII
XPPUT - Volume VII
XRMOV - Volume VII

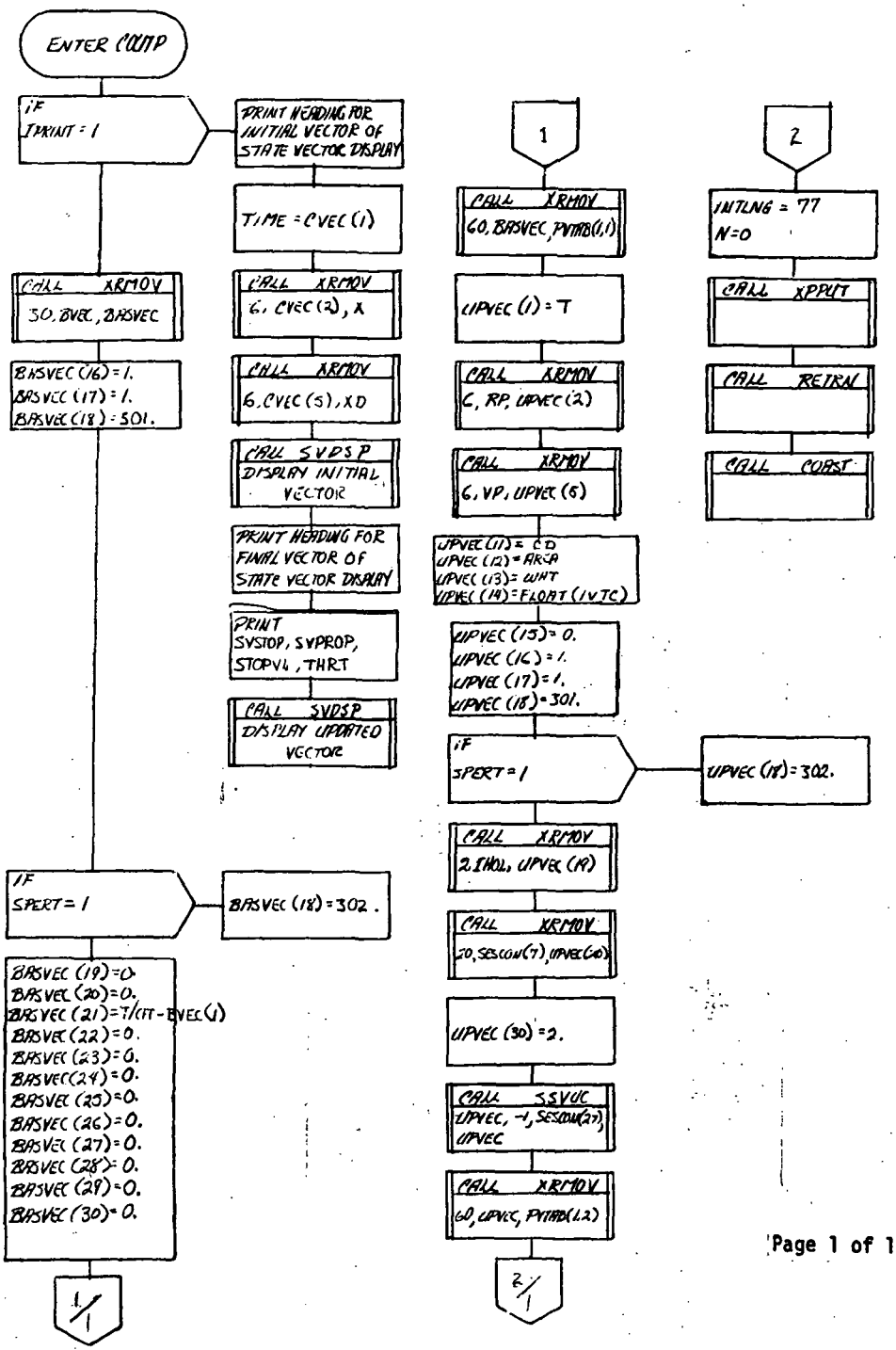
5.3.9 References

None.

TABLE 5.3-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine COUTP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
RP		Real	I		C		Current radius vector.
VP		Real	I		C		Current velocity vector.
T		Real	I		C		State vector time.
SESCON		Real	I		C		Session constants array.
BVEC		Real	I		C		Base vector in external units.
CVEC		Real	I		C		Base vector in internal units.
PVTAB		Real	O		IT		Position velocity phase table.
SVSTOP		6CH	O		T		State vector stop condition code.
SVPROP		6CH	O		T		State vector propagator code.
STOPVL		Real	O		T		State vector stop value.
THTIME		Real	O		T		Threshold time in seconds.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory



Page 1 of 1

Figure 5.3-1.- Detailed flow for COUTP routine.

5.4 ROUTINE NAME - NCODE

5.4.1 Purpose

NCODE interprets the character string codes for SVSTOP, SVPROP, and PRINT.

5.4.2 Functional Description

NCODE receives, through its subroutine calling arguments, the user input character string code for SVSTOP, SVPROP, or PRINT. Using an integer routine flag (also a subroutine calling argument), which determines the input code to be interpreted, control is transferred to the proper logic. At this time NCODE interprets the character string code and assigns an integer code (identifying the character code) to an output argument.

5.4.3 Assumptions and Limitations

None.

5.4.4 Method

None.

5.4.5 Routine Input/Output Variables

The NCODE routine input/output variables are presented in table 5.4-I. The error message generated is described in the COAST processor message table (msg. no. 2).

5.4.6 Functional Logic Flow

A functional logic flow is not provided. A detailed flow for NCODE is provided in figure 5.4-1.

5.4.7 Diagnostics and Debug

None.

5.4.8 Special Comments

NCODE utilizes the following subroutine, which is documented under the indicated section of the FDS-1 system design document.

XRCPR - Volume VII

5.4.9 References

None.

TABLE 5.4-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine NCODE

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IBUG		Intg	I		A		Output unit on which to write all print.
ROUTE		Intg	I		A		Routing flag for code interpretation.
ID		6CH	I		A		Array containing character code.
NUMBER		Intg	O		A		Numerical value identifying the input character code.
ERROR		Intg	O		A		Error return code.

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory

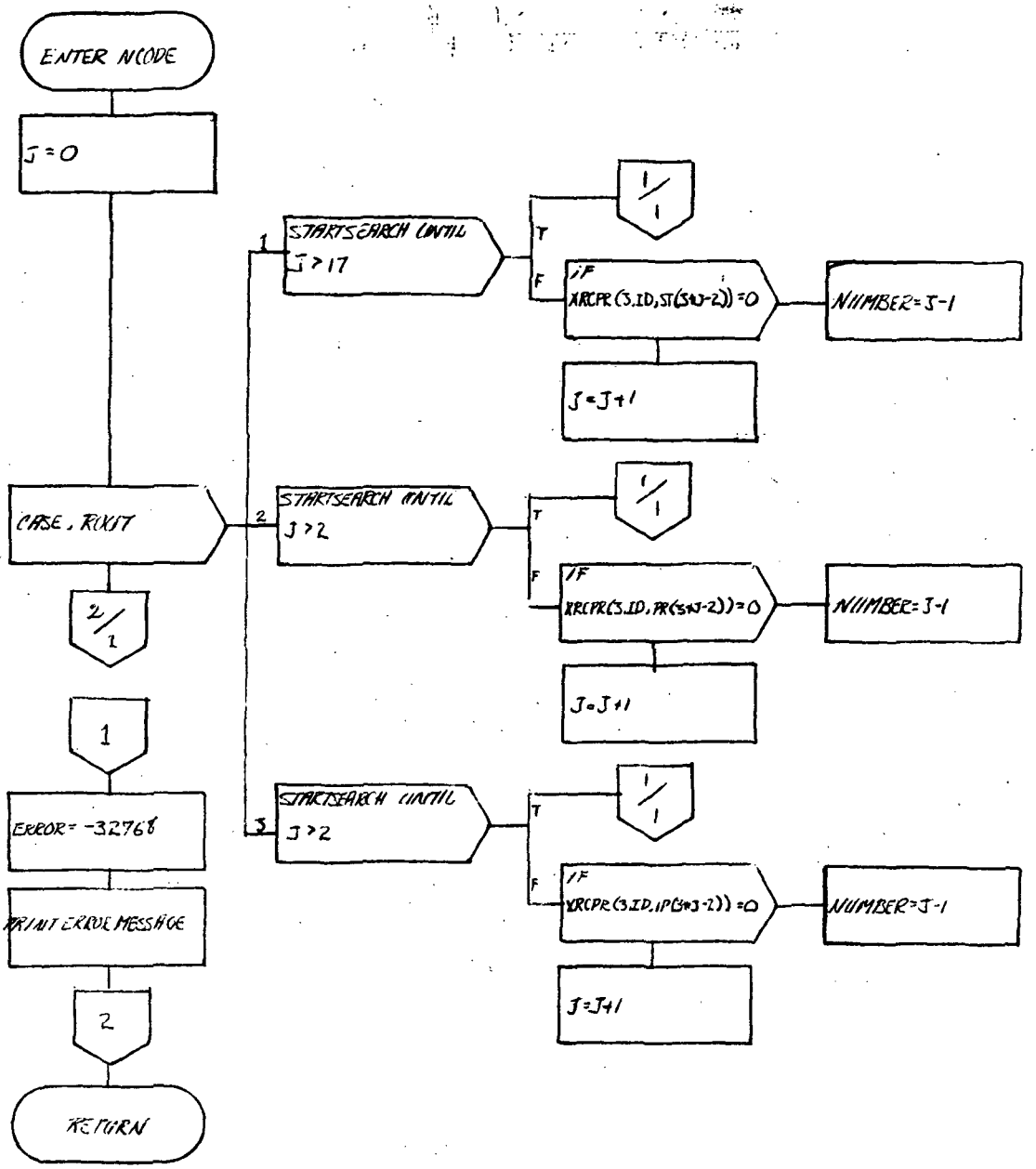


Figure 5.4-1.- Detailed flow for NCODE routine.

5.5 ROUTINE NAME - SVDSP

5.5.1 Purpose

The subroutine SVDSP generates a display of a vehicle state vector.

5.5.2 Functional Description

SVDSP converts a state vector to external units and displays it on the user's terminal. A format of the display is given in the COAST processor description, processor display format.

5.5.3 Assumptions and Limitations

None.

5.5.4 Method

None.

5.5.5 Routine Input/Output Variables

The SVDSP routine input/output variables are presented in table 5.5-I.

5.5.6 Functional Logic Flow

A functional flow is not provided. A detailed flow for SVDSP is presented in figure 5.5-1.

5.5.7 Diagnostics and Debug

None.

5.5.8 Special Comments

SVDSP utilizes the following subroutines, which are documented under the indicated sections of the FDS-1 system design document.

APIE - GPMP processor, vol. III
RVTIM - Undocumented
XYZ2E - Undocumented

5.5.9 References

None.

TABLE 5.5-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine SVDSP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
SESCON		Real	I		C		Session constants array.
IPRAM		Intg	I		C		Array that contains terminal LU.
CONST		Real	I		C		COAST constants array.
X		Real	I		A		Position vector.
XD		Real	I		A		Velocity vector.
T		Real	I		A		Time of state vector.
CD		Real	I		A		Coefficient of drag.
AREA		Real	I		A		Reference area.
WHT		Real	I		A		Weight.
SPERT		6CH	I		A		Vector propagation code.
PER		Real	O		T	PERIOD	Period of orbit in minutes.
ALT		Real	O		T	ALT	Height of vehicle above reference radius.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.5-I.- Continued

Routine SVDSP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
E		Real	O		T	E	Eccentricity
X7		Real	O		T	 T TA	Used to output two different display parameters. Greewich mean time of vector True anomaly
X2		Real	O		T	 G Y	Used to output two different display parameters. Argument of pergiee Cartesian position component in Y direction.
X1		Real	O		T	 A V X	The following variables are used to display three different display parameters each. Semimajor axis Inertial velocity magnitude Cartesian position component in X direction, TEG
X3		Real	O		T	 I P Z	 Inclination of orbit plane Flight azimuth, geocentric Cartesian position component in Z direction, TEG
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.5-I.- Concluded

Routine SVDSP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
X4		Real	O		T	G R XD	Argument of perigee Radius magnitude Cartesian velocity component in X direction, TEG
X5		Real	O		T	H LO YD	Right ascension of orbit plane ascending node Geographic longitude Cartesian velocity component in Y direction, TEG
X6		Real	O		T	L LA ZD	Mean anomaly Geocentric latitude Cartesian velocity component in Z direction, TEG
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

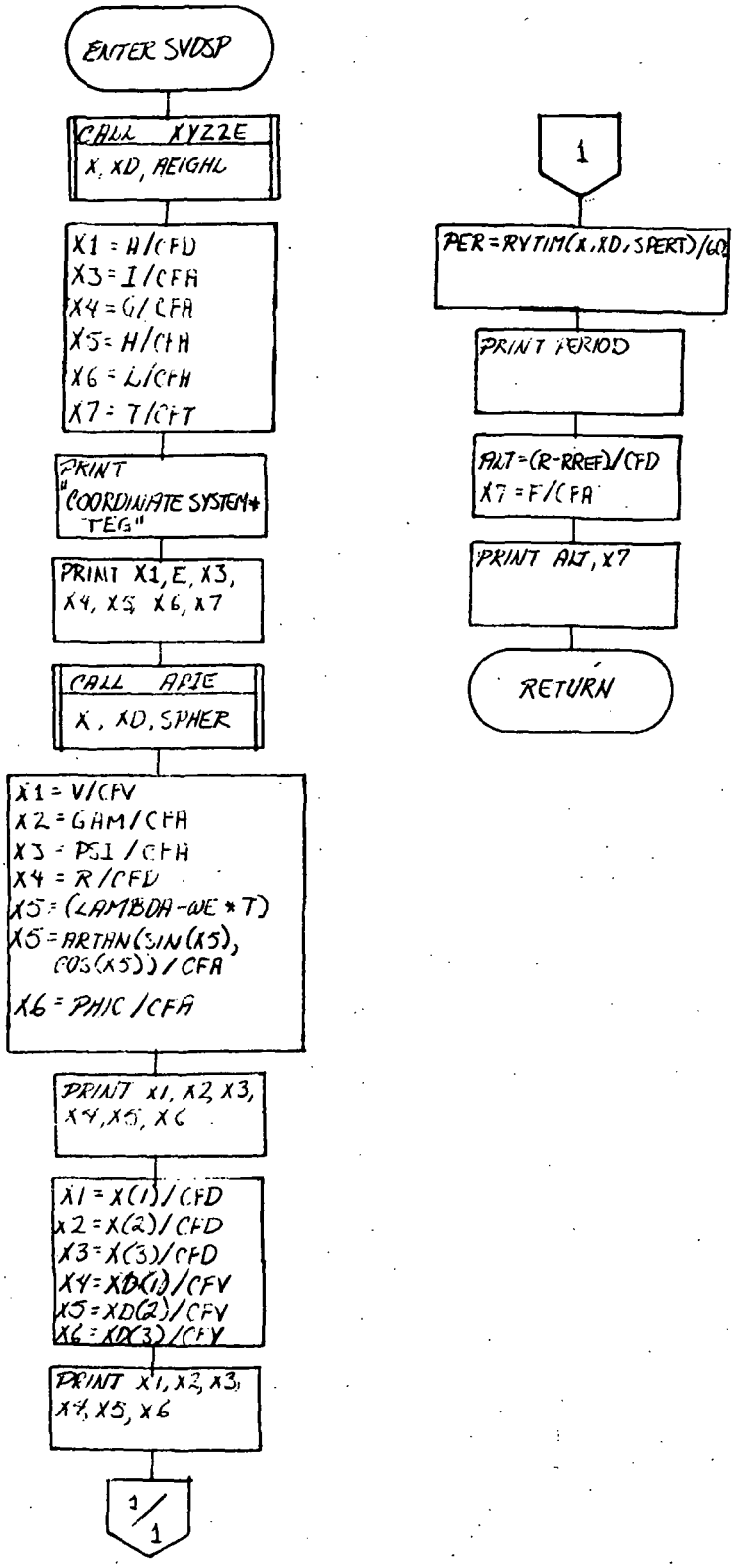


Figure 5.5-1.- Detailed flow for SVDSP routine.

DATA BOX DISPLAY PROCESSOR (DBDSP)

1.0 PURPOSE

The purpose of the data box display processor is to provide a flexible display medium for values of any parameter or pair of parameters generated in the scan mode (see scan processor). This processor aids the user in evaluating the effects that user-defined constraints have on certain computations. More than one processor may be executed in each cycle of the scan, but only one summary table per cycle is stored into the data box by the ENDSC processor.

2.0 FUNCTIONAL DESCRIPTION

Input data for DBDSP is specified or contained in its interface table. The user can specify the following when initiating the data box display:

a.	Data box file name	(req'd.)
b.	List of display variable names	(req'd.)
c.	List of display variable scale factors (powers of ten)	(optional)
d.	List of desired constraint variable names	(optional)
e.	List of desired constraint variable relationships	
f.	List of desired constraint variable values	

The input data box file is read by DBDSP to obtain the required parameter values, parameter names, unit mnemonics, and parameter descriptions. The parameters are stored in an internal array(s). The dimensions of the data box file are (11, 11, and 32) where the first two values represent the maximum number of summary tables (121) that can be generated by a parametric scan (one SCAN/ENDSC pair), and the last size is the maximum number of values allowed including the error indicator in one summary table. Effectively, the maximum number of scan values allowed is 31.

At display time, by user option, either one or two dependent variables (i.e., items in a summary table) may be displayed. A maximum of 16 pages may be displayed giving a maximum of 32 dependent variables displayed in one execution of DBDSP. To allow the maximum number (11) of dependent variables to be displayed on one line, each variable is limited to a numerical value of four digits plus sign.

If, after scaling, the resulting array entry exceeds the four-digit capacity, the value of 9999 with the proper sign will be displayed.

The constraint relationship will be determined at each solution point in the display matrix.

For any violation at a solution point, the value will be suppressed, and an alphabetic will be displayed in its place. The letters A through H will be displayed for their respective constraints that are violated. When at least one constraint has been violated, blanks will be inserted into the array for those constraints that were not violated.

In any case on each display page, the specific constraints, if any, that have been violated on that page will be indicated at the top of the page. If the user inputs an invalid constraint mnemonic or relationship, a message is printed prior to the first page displayed. The invalid constraint will not be used by DBDSP.

The first parameter of each summary table is an error flag that is set by the individual functional processor(s). A nonzero real value indicates that the scan point is invalid and is not to be displayed to the user. The character string $$$$ is displayed at each point where the error flag is set. This display option is only active when the user has applied constraint inputs. If no constraint is specified, each array entry is displayed as found in the data box and the error flag is ignored. Proper selection of a constraint can allow the user to recognize the error condition and not effectively constrain the data box output.

3.0 ASSUMPTIONS AND LIMITATIONS

Listed below are a set of the assumptions and limitations associated with the data box display processor (DBDSP).

a. The first value in each summary table will be an error flag (= 0 indicates no error, ≠ 0 indicates error).

b. Summary tables used to build data boxes via SCAN/ENDSC will be limited to a maximum of 32 values including the error flag.

c. The maximum display size for the dependent variables will be an 11 x 11 matrix. This results from specifying five scan points on each side of the centroid point for both X and Y coordinates.

d. There is a maximum of two dependent (summary) variables per display page.

e. There is a maximum of 16 unique display pages generated per (DBDSP) execution.

f. There is a maximum of a signed four digits per scaled dependent value displayed. For constraints up to eight constraint characters displayed instead of upper/lower dependent values.

g. A maximum of five lines (Y coordinate) can be displayed on the HP2645 display at one time.

h. Users can specify that the display be output to the printer or other LU.

4.0 PROCESSOR INPUT/OUTPUT

a. Processor interface table - The processor interface table for DBDSP is defined in table 4-I and contains the following:

(1) Data box to be displayed

(2) Definitions of 1 to 16 display pages

(3) Definitions of zero to 8 constraints to be applied to the displayed values

The display page definition arguments (PAGE1, PAGE2, PAGE3,..... PAGE15, PAGE16) are input as symbolic strings. Symbolic strings are always input between single quotes.

Each of these arguments is input in one of two forms:

\PAGE1=:'VAR1,n'Δ

\PAGE1=:'VAR1,n,VAR2,m'Δ

The first form specifies that the display page will contain only one dependent variable, VAR1, and that its displayed values will be scaled by 10^n where n is an integer value. The second form specifies that the display page will contain two dependent variables, VAR1 and VAR2, that VAR1 will be scaled by 10^n, and that VAR2 will be scaled by 10^m, where n and m are integer values.

The following gives some examples of values and how they would be displayed for certain scale factors

Value	Scale factor (10^n)	Displayed value
1.23456E5	10^2	1234
1.23456E5	10^3	123
-1.23456	10^{-3}	-1234
1.00E9	10^9	1
1.00E9	10^6	1000

The constraint definition arguments (CONSTA, CONSTB, CONSTC,...CONSTH) are input as symbolic strings and have the form:

\CONSTA=:'VAR1 relation value'Δ

where "relation" is one of

<,>,=,>=,<=,=>, and = <

and "value" is a real number. For example, to constrain the displayed values such that VAR1 is always greater than or equal to 100., then specify:

\CONSTA=:'VAR1>= 100.'Δ

The last letter in the name of each constraint definition argument (A,B,...H) is the letter displayed when that constraint is violated. A "null" symbolic string indicates no constraint relation defined:

\CONSTA=:' 'Δ

b. Interface table data array definitions - The interface table data arrays are provided in table 4-II.

c. Interface table data file definitions - The interface table data file definitions are provided in table 4-III.

d. Processor solicited (prompted) inputs - The processer solicited (prompted) inputs are provided in table 4-IV.

e. Processor displays and display parameter definition tables - The processor display format is shown in table 4-V and a definition of the display variables is provided in table 4-VI.

f. Processor message table - The format and content of the processor message table are provided in table 4-VII.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-VIII.

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR DBDSP

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Value stored in default interface table	Definition
PROCON	AWA	Free	I	3	3		Processor constants (see interface table data array definitions table).
DATBOX	Disk	Free	I				Data box generated by SCAN/ENDSC utility processors.
PAGE1	AWA	Symb	I	16			Definition of display PAGE1; two forms of the symbolic string are allowed: 'NAME1,n' and 'NAME1,n,NAME2,m' NAME1 and NAME2 are the dependent variables to be displayed on this page; the integer values n and m are the exponents of 10, which give the scaling factors for NAME1 and NAME2, respectively. These values must be -99 to 99. An empty symbolic string ('Ƀ') indicates that no display will be defined for this page.
PAGE2	AWA	Symb	I	16		'Ƀ'	Definition of display PAGE2; see PAGE1.
PAGE3 : PAGE16	AWA : AWA	Symb : Symb	I : I	16 : 16		'Ƀ' : 'Ƀ'	Definition of display PAGE3; see PAGE1. : Definition of display PAGE16.

NOTES	CLASS	TYPE			USE
	AWA Disk	Free Intg Real Dubl	2CH 6CH 18CH 36CH	72CH Mix Symb	I = Input O = Output I/O = Input/Output

TABLE 4-I.- Continued

PROCESSOR DBDSP

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Value stored in default interface table	Definition
CONSTA	AWA	Symb	I	9			Constraint violation to be tagged with an "A" in positions where it occurs on all pages of the display. This symbolic string has the form 'NAME relation value' where, NAME is a dependent variable from the summary table and relation is one of <,>,=,<=,>=,=>, or =< and value is a real (floating point) value. A blank ('Ø') symbolic string indicates no constraint checking for this tab ("A").
CONSTB	AWA	Symb	I	9		'Ø'	Constraint violation to be tagged with a "B" (see CONSTA).

NOTES	CLASS	TYPE			USE
		Free	2CH	72CH	I = Input
	AWA	Intg	6CH	Mix	O = Output
	Disk	Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-I.- Concluded

PROCESSOR DBDSP

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Value stored in default interface table	Definition
CONSTC	AWA	Symb	I	9		'%'	Constraint violation to be tagged with a "C".
:	:	:	:	:		:	:
CONSTH	AWA	Symb	I	9		'%'	Constraint violation to be tagged with an "H".

NOTES	CLASS	TYPE			USE
		Free	2CH	72CH	I = Input
	AWA	Intg	6CH	Mix	O = Output
	Disk	Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-II.- INTERFACE TABLE DATA ARRAY DEFINITIONS

PROCESSOR DBDSP

Array name	Index location	Default value	Definition
PROCON	1	0	LU (logical unit) to which the display will be output; value of zero will output the display to the FDS user's terminal; user may also override this be selecting an LU directly when selecting the next display page (see solicited inputs).
	2	0	Indicator specifying user selection of display pages = 0 User to be prompted before each display page is generated (see solicited inputs). ≠ 0 No prompts for solicited inputs issued; all pages with definitions are displayed to indicated LU (see PROCON(1) above).
	3	-20	LU or cartridge number on which the DRDE (DATBOX) is located; this value is used in file manager calls. > 0 Cartridge number < 0 Logical unit number

TABLE 4-III.- INTERFACE TABLE DATA FILE DEFINITIONS

PROCESSOR DBDSP

DRDE DATA FILE DATBOX

(Type 2; record length - 64 words)

Record number	Integer word allocations	Content and definition
1	(1)	Name of FDS processor creating the file
	(4)	Interface table variable name for this file
	(7)	Name of FDS processor updating file (3 ASK11 words of blanks)
	(10)	Interface table variable name for this update (3 ASK11 words of blanks)
2	(1)	Number of entries in SUMTAB (integer 1-32)
	(2)	X scan variable (6 characters)
	(5)	X first subscript (integer or zero if none)
	(6)	X second subscript (integer or zero if none)
	(7)	X units (6 characters)
	(10)	X centroid (real)
	(12)	X increment (real)
	(14)	X number of steps (integer 1-5)
	(15)	Y scan variable (6 chars) or blanks if none
	(18)	Y first subscript (integer or zero if none)
	(19)	Y second subscript (integer or zero if none)
	(20)	Y units (6 characters)
	(23)	Y centroid (real)
	(25)	Y increment (real)
	(27)	Y number of steps (integer 1-5)
	28 - 64	Unused
3	1 - 64	First 64 words of SUMTAB variable names
4	1 - 32	Last 32 words of SUMTAB variable names
	33 - 64	First 32 words of SUMTAB variable units
5	1 - 64	Last 64 words of SUMTAB variable units
6	1 - 64	Values for scan variable(s) X_1 or X_1Y_1

TABLE 4-III.- Concluded

PROCESSOR DBDSP

DRDE DATA FILE DATBOX

(Type 2; record length - 64 words)

Record number	Integer word allocations	Content and definition
7 . .	1 - 64	Values for scan variable(s) X_2 or X_2Y_1
N + 5*	1 - 64	Values for scan variable(s) X_N or X_NY_1
N + 6*	1 - 64	(If there is no Y scan variable, record N + 5 is the last record, else) Values for scan variables X_1Y_2
. . .		
N+M+5*	1 - 64	Values for scan variables X_NY_M
		*Note N = Number of values of X M = Number of values of Y

TABLE 4-IV.- PROCESSOR SOLICITED (PROMPTED) INPUTS

PROCESSOR DBDSP

Prompt	Meaning	Valid responses
PAGE NO. [,ROW,NO.ROWS,LU]:	Specify desired page to be displayed, and optionally, the beginning row number, number of rows, and device LU for the display	One, two, three, or four values separated by commas may be input. The first value (1 to 16) specifies the page number of the next display. The second value is the row number of the data at which the display should begin. Since the HP2645 terminal will only display up to five rows of a display, the user may wish to selectively determine which five rows. The third value is the number of rows to be displayed. The rows are numbered 1 to 11 from the bottom up. Inputting a zero for either the second or third value will cause all rows to be displayed. The fourth value is the device logical unit to which the display will be transmitted. If input as zero or omitted, the LU specified by the PROCON argument in the interface table is used.

TABLE 4-V.- PROCESSOR DISPLAY TABLE

PROCESSOR DBDSP

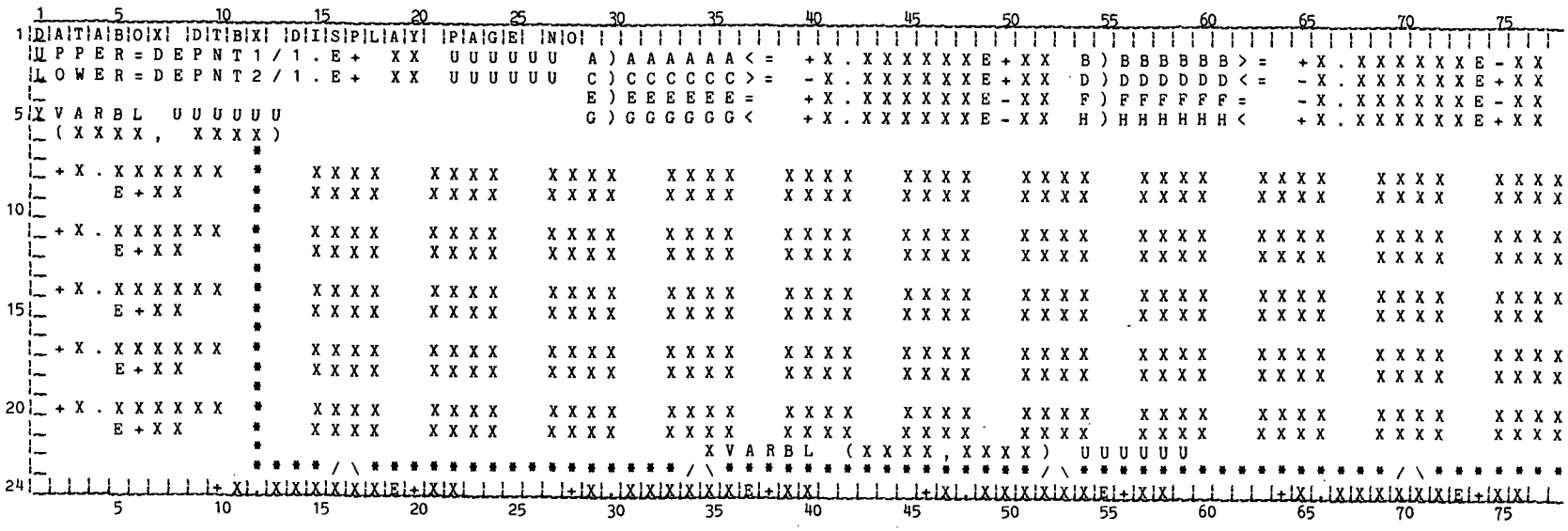


TABLE 4-VI.- DISPLAY PARAMETER DEFINITION TABLE

PROCESSOR DBDSP

DBDSP PROCESSOR DISPLAY	
Display parameter label	Parameter definition
DTBX	Name of Data Box (DRDE) being displayed
NO	Page number being displayed
AAAAAA to HHHHHH	Dependent (summary table) variable used in the particular constraint check
DEPNT1	Dependent (summary table) variable being displayed in the upper position
DEPNT2	Dependent (summary table) variable being displayed in the lower position
1.E+XX	Scale factor; values for YYYYYY and ZZZZZZ have been scaled by their corresponding scale factor (10^{XX})
UUUUUU	Units for the adjacent variable
+X.XXXXXXE+XX	Real number
YVARBL	Y-scale (vertical) independent variable
XVARBL	X-scale (horizontal) independent variable
NNN,MMM	Subscripts (if any) of this independent value
XXXX	Value grid for the two dependent variables (YYYYYY and ZZZZZZ). These two values are digitized and scaled. If any of the defined constraints has been violated at this point, the value is suppressed and the indicator(s) (A - H) for those constraints is displayed.
XXXX	

TABLE 4-VII.- PROCESSOR MESSAGE TABLE

PROCESSOR DBDSP

MSG no.	Message ID block	Message text block and explanation
1	*DBDSP*	"DATA BOX - NOT AVAILABLE." Meaning: The DRDE name provided as the input Data Box is not the name of the DRDE currently logged in the AWA. Severity: High; processor execution abnormally terminated. Action required by user: Modify interface table to specify the correct Data Box or execute a SCAN/ENDSC sequence to generate this Data Box.
2	*DBDSP*	"ERROR IN OPENING DATA BOX - ERROR CODE _____." Meaning: The DRDE file specified as the input Data Box could not be correctly opened via a call to the RTE/FMP routine OPEN. The return code from OPEN is contained in the message text. Severity: Very high; processor execution abnormally terminated. Action required by user: Notify FDS system maintenance personnel.
3	*DBDSP*	"ERROR IN READING DATA BOX - ERROR CODE _____." Meaning: The DRDE file specified as the input Data Box could not be correctly read via a call to the RTE/FMP routine READF. The return code from READF is contained in the message text. Severity: Very high; processor execution abnormally terminated. Action required by user: Notify FDS system maintenance personnel.
4	*DBDSP*	"INVALID PAGE NO. INPUT. MUST BE 1 TO 16." Meaning: The response to the user solicited prompt must specify a display page 1 to 16. Severity: Low; user prompted again and processor execution continues normally. Action required by user: Reinput, with the first field (page number) being the desired display page (as defined in the interface table) 1 to 16.

TABLE 4-VII.- Continued

PROCESSOR DBDSP

MSG no.	Message ID block	Message text block and explanation
5	*DBDSP*	"INVALID VALUE(S) INPUT; ROW NO. + NO. ROWS EXCEEDS AVAILABLE ROWS." Meaning: The values input in response to the user solicited prompt have resulted in an invalid request. The beginning row number (second value input) plus the number of rows to display (third value input) must not be greater than the number of rows in the Data Box (determined by SCAN execution that generated the Data Box). Severity: Low; user is prompted again and processor execution continues normally. Action required by user: Reinput with meaningful values for beginning row number and number of rows. Remember that the entire response must be reinput (i.e., display page number, the first value input, must be reentered).
6	*DBDSP*	"_____IS AN INVALID NAME FOR A CONSTRAINT VARIABLE." Meaning: The name indicated was specified in the interface table as one of the 8 constraint variables but is not one of the summary table variables in this Data Box. Severity: Moderate; the processor execution continues, but the particular constraint is ignored. At the first opportunity, the user may elect to exit DBDSP in order to correct the interface table. Action required by user: Modify the interface table for DBDSP such that all constraint relations utilize only the variables contained in the particular summary table used to build this Data Box.
7	*DBDSP*	"_____IS AN INVALID DEPENDENT VARIABLE NAME." Meaning: The name indicated was specified in the interface table as one of the two dependent variables for the selected page, but is not one of the summary table variables in this Data Box. Severity: Moderate; the processor execution continues, but this display page is ignored. At the first opportunity, the user may elect to exit DBDSP in order to correct the interface table. Action required by user: Modify the interface table for DBDSP such that all display page definitions specify dependent variables that are contained in the summary table used to build this Data Box.

TABLE 4-VII.- Continued

PROCESSOR DBDSP

MSG no.	Message ID block	Message text block and explanation
8	*DBDSP*	"INVALID SYNTAX FOR PAGEXX, IGNORED." Meaning: The syntax for one of the 16 display definitions in the interface table is invalid. There are only two valid forms of a display definition \PAGEXX=:'VAR1,n' and \PAGEXX=:'VAR1,n,VAR2,m' The scale factor n and m must be -99 to 99. Severity: Moderate; the processor execution continues, but this display page is ignored. At the first opportunity, the user may elect to exit DBDSP in order to correct the interface table. Action required by user: Modify the interface table such that the display definition is valid and/or select a different page for display.
9	*DBDSP*	"INVALID SYNTAX FOR CONSTx, IGNORED." Meaning: The syntax for one of the eight constraint violation relationships specified in the interface table is invalid. There is only one valid form of a constraint relationship definition. \CONSTX=:'VAR relation value' where "relation" is one of =, >, <, >=, <=, =>, or =< and "value" is a real number. Severity: Moderate; the processor execution continues, but this constraint violation relationship is ignored. Action required by user: Modify the interface table for DBDSP such that all constraint violation relationships have valid syntax structures.
10	*DBDSP*	"NO DISPLAY PAGES DEFINED." Meaning: The interface table does not contain any valid display page definitions. Severity: Low, processor execution terminates normally. Action required by user: Modify the interface table so that valid display definitions are specified.

TABLE 4-VII.- Concluded

PROCESSOR DBDSP

MSG no.	Message ID block	Message text block and explanation
11	*DBDSP*	"SYNTAX ERROR, PLEASE CHECK AND INPUT AGAIN." Meaning: The form and/or content of the user response did not comply with the syntax for the prompt. The form and content should be: page number[,starting row][,number of rows][,logical unit] where, page number is an integer value 1 to 16 starting row is an integer value 1 to 11 number of rows is an integer value 1 to 11 logical unit is an integer value >0 Severity: Low; user is prompted again and processor execution continues normally. Action required by user: Reinput with data of the proper form.

TABLE 4-VIII.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR DBDSP

Processor name	Processor abstract prompt (maximum 256 characters)
DBDSP	Data box display processor used to build and output 1 to 16 digital displays. Each display will present the scaled values for one or two summary table values over the range determined from the data box. The data box is generated by the SCAN/ENDSC processors.
Parameter keyword name	**Parameter definition prompt (maximum 256 characters)**
PROCON	Processor constants (1) Display LU definition (0 = user's terminal) (2) Solicited inputs flag (0 = user prompted) (3) LU, or cartridge of DATBOX
DATBOX	Data box to be displayed. A data box is a collection of summary tables and is built by SCAN/ENDSC processors.
PAGE 1	Definition of display page 1. Symbolic string giving one or two dependent variables (summary table entries) and their scale factors. 'ALPHA,2,BETA,1' or 'ALPHA,2'
.	.
.	.
.	.
PAGE 16	Definition of display page 16. . . .
CONSTA	Definition of violation constraint A. Symbolic string providing the dependent variable (summary table entry), relational operator, and value to define the constraint. 'ALPHA < = 10.' or 'BETA > 100.'
.	.
.	.
.	.
CONSTH	Definition of violation constraint H. . . .

5.0 PROCESSOR ROUTINES

5.1 ROUTINE NAME - MAIN PROGRAM DBDSP

5.1.1 Purpose

The data box display routine (DBDSP) serves as the main program of the DBDSP processor.

5.1.2 Functional Description

The DBDSP routine calls XZDIN (performs processing of interface table inputs), XZDP1 (reads the data box and supplies constraints), and XZDP2 (outputs the display(s)). In addition, XZDP1 calls XZDMK (applies constraints) and XZDP2 calls XZDOT (outputs a display).

5.1.3 Assumptions and Limitations

None.

5.1.4 Method

None.

5.1.5 Routine Input/Output Variables

None.

5.1.6 Functional Logic Flow

A functional logic flow is not provided. The functional level PDL for DBDSP is presented in figure 5.1-1.

5.1.7 Diagnostics and Debug

None.

5.1.8 Special Comments

None.

```
1 BEGIN DBDSP
2    INITIATE CALL TO XZDIN TO READ INTERFACE TABLE
2    INITIATE CALL TO XZDP1 TO READ DATA FILE AND VERIFY NAME LISTS
2    INITIATE CALL TO XZDP2 TO PROMPT USER AND OUTPUT DATA BOX DISPLAY
1 EXIT
1 END DBDSP
```

Figure 5.1-1.- DBDSP functional level PDL.

5.2 ROUTINE NAME - XZDIN

5.2.1 Purpose

XZDIN retrieves the various interface table inputs, checks their completeness and syntax, and builds arrays in common for use by XZDP1 and XZDP2.

5.2.2 Functional Description

Support subroutine XPGET is called to retrieve values for each of the 26 input arguments. The name of the data box file is constructed and placed into common (DATBOX). The symbolic strings for each of the 16 pages are processed. For each symbolic string that conforms to syntax requirements, the appropriate entries in the dependent variable display table (NDVARL) and display variable scale table (NDVRUL) are made.

The symbolic strings for each of the eight constraints are then processed. For each symbolic string that conforms to syntax requirements, the appropriate entries in the constraint variable table (NCVARL), constraint relation table (NCRELL), and constraint value table (CVALUE) are made.

The PROCON values are also placed into common (LUDSP, SELECT, and CARTRG) and control is returned to DBDSP.

5.2.3 Assumptions and Limitations

None.

5.2.4 Method

None.

5.2.5 Routine Input/Output Variables

All input and output to XZDIN is via common and the appropriate common variables are mentioned in section 5.2.2.

5.2.6 Functional Logic Flow

A functional logic flow is not provided. The functional level PDL for XZDIN is presented in figure 5.2-1.

5.2.7 Diagnostics and Debug

For invalid syntax of page and constraint, definitions messages 6, 7, 8, 9, and 10 may be output. These messages are discussed in table 4-VII.

When DBDSP is scheduled for FDS with bit 4 of the flags parameter set, debug interaction at the user's terminal will occur.

5.2.8 Special Comments

None.

5.2.9 References

None.

```
1 BEGIN XZDIN
2    CALL XPGET TO RETRIEVE PROCON, DATA BOX NAME, 16 DISPLAY DEFINITIONS
2    AND 8 CONSTRAINT RELATIONS
2    USING PROCON, DETERMINE DISPLAY LU
2    SET DISPLAY SELECTION MODE FLAG BASED ON PROCON(2)
2    SET DATA BOX'S CARTRIDGE BASED ON PROCON(3)
2    CLEAR DISPLAY VARIABLE LIST TO ZEROS
2    DO FOR EACH OF 16 DISPLAY DEFINITIONS
3       SET TOKEN POINTER TO 1ST TOKEN
3       DO FOR EACH OF 2 VARIABLES, AND
           WHILE END-OF-STRING NOT REACHED
3       EXIT TO :ERR8: IF TOKEN IS NOT A NAME
4          PUT NAME INTO APPROPRIATE ENTRY OF VARIABLE NAME LIST
4          INCREMENT TOKEN POINTER
3       EXIT TO :ERR8: IF TOKEN IS NOT A COMMA
4          INCREMENT TOKEN POINTER
3       EXIT TO :ERR8: IF TOKEN IS NOT AN INTEGER
4          PUT VALUE INTO APPROPRIATE ENTRY OF VARIABLE SCALE FACTOR LIST
4          INCREMENT TOKEN POINTER
4          IF TOKEN IS COMMA, THEN
5             INCREMENT TO NEXT TOKEN
4          ENDIF
3       ENDDO
2    EXIT TO :ERR8: IF TOKEN IS NOT END-OF-STRING

3       :ERR8: CALL XZMSG - 'INVALID DISPLAY DEFINITION FOR PAGE XX'
3       CALL XZLSS TO PRINT SYMBOLIC STRING
2    ENDDO
1 EXIT TO :ERR10: IF NO VALID DIPLAY PAGES DEFINED
2    CLEAR CONSTAINT DEFINITION LISTS TO ZEROS
2    SET NO. OF CONSTRAINTS TO 0
2    DO FOR EACH OF 8 CONSTRAINT RELATIONS, AND
3       SET TOKEN POINTER TO 1ST TOKEN
3       IF TOKEN IS NOT END-OF-STRING, THEN
3       EXIT TO :ERR9: IF TOKEN IS NOT NAME
4          PUT NAME INTO APPROPRIATE CONTRAINT VARIABLE NAME LIST ENTRY
4          INCREMENT TO NEXT TOKEN
3       EXIT TO :ERR9: IF NOT ONE OF >, <, =, >=, <=, =>, =<
4          SET APPROPRIATE CONTRAINT VARIABLE RELATION LIST ENTRY
4          INCREMENT TO NEXT TOKEN
3       EXIT TO :ERR9: IF NOT A REAL VALUE
4          SET APPROPRIATE CONSTRAINT VARIABLE VALUE LIST ENTRY
4          INCREMENT TO NEXT TOKEN
3       EXIT TO :ERR9: IF NOT END-OF-STRING
4          INCREMENT NO. OF CONSTRAINTS BY 1
3       ENDIF

3       :ERR9: CALL XZMSG - 'INVALID CONSTRAINT DEFINITION X, IGNORED'
3       CLEAR THIS CONSTRAINT VARIABLE NAME LIST ENTRY
3       CALL XZLSS TO PRINT SYMBOLIC STRING
2    ENDDO
1 EXIT XZDIN

2    :ERR10: CALL XZSMG - 'NO DISPLAY PAGES DEFINED'
2    CALL XPXIT TO TERMINATE DBDSP PROCESSOR ABNORMALLY
1 END XZDIN
```

Figure 5.2-1.- XZDIN functional level PDL.

5.3 ROUTINE NAME - XZDP1

5.3.1 Purpose

XZDP1 reads the data box.

5.3.2 Functional Description

XZDP1 reads into common arrays the header information contained in the data box (table 4-III). This includes the names, units, and subscripts of the two independent variables, as well as the names and units for each of the dependent variables (summary table entries).

Next, XZDP1 reads into common (ATABLE) all values for the dependent variables. These values are grouped within ATABLE in ascending order of generation according to the dependent variable; i.e., all values for the first (error indicator) dependent variable are grouped at the beginning of ATABLE followed by the values for the second dependent variable, etc.

XZDP1 then calls XZDMK to apply any defined constraints.

5.3.3 Assumptions and Limitations

None.

5.3.4 Method

None.

5.3.5 Routine Input/Output Variables

THE XZDP1 input/output variables are presented in table 5.3-I.

5.3.6 Functional Logic Flow

A functional logic flow is not presented. The functional level PDL for XZDP1 is presented in figure 5.3-1.

5.3.7 Diagnostics and Debug

For any input/output errors encountered while accessing the data box, messages 1, 2, and 3 may be output. These messages are defined in table 4-VII.

When DBDSP is scheduled by FDS with bit 4 of the flags parameter set, debug interaction at the user's terminal will occur.

5.3.8 Special Comments

None.

5.3.9 References

None.

TABLE 5.3-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine XZDP1

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DATBOX	--	--	I	--	--	--	Data box file name.
XSCNNM YSCNNM	--	--	O	--	--	--	Independent variables names.
XUNITS YUNITS	--	--	O	--	--	--	Independent variables units.
IXSCN1 IXSCN2 IYSCN1 IYSCN2	--	--	O	--	--	--	Independent variables subscripts.
ATABLE	--	--	O	--	C	--	Dependent variables values.
ZTABLE	--	--	O	--	--	--	Names and units of dependent variables.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

```
1 BEGIN XZDP1
2    OPEN DATA FILE
2    READ IN SECOND RECORD CONTAINING SCAN DATA BOX SETUP
2    ERREXIT IF ERROR FLAG SET FROM READ
2    COMPUTE NO. OF X COORDINATES FOR DISPLAY - NX
2    COMPUTE NO. OF Y COORDINATES FOR DISPLAY - NY
2    SET THE NO. OF DEPENDANT VARIABLES SCANNED - NZ
2    COMPUTE THE OUTPUT ARRAY COORDINATE VALUES FOR X AND Y
2    READ IN THIRD,FOURTH, AND FIFTH RECORDS AND ESTABLISH NAME LIST AND UNITS LIST
2    FOR DEPENDANT VARIABLES GENERATED BY SCAN OPERATION
2    DETERMINE IF CONSTRAINT VARIABLES SPECIFIED ARE LISTED AS DEPENDANT
2    VARIABLES GENERATED IN SCAN OPERATION
2    ERREXIT IF NAME IS NOT ON DEPENDANT ON LIST
2    DO FOR EACH Y COORDINATE VALUE ( J= 1,NX )
3       DO FOR EACH X COORDINATE VALUE ( I= 1,NY)
4          READ A SUMMARY TABLE RECORD INTO STORAGE
4          ERREXIT IF ERROR FLAG IS SET
4          DO FOR EACH DEPENDANT VARIABLE ( K= 1,NZ)
5             COMPUTE PROPER ADDRESS IN THREE DIMENSIONAL ARRAY
5             STORE EACH VALUE IN PROPER POSITION IN ARRAY (I,J,K)
4          ENDDO
3       ENDDO
2    ENDDO
2    DO FOR ALL CONSTRAINTS ( I = 1,10)
3       IF CONSTRAINT VARIABLE IS ON DEPENDANT NAME LIST, THEN
4          SET THAT ENTRY TO POINT TO NAME IN DEPENDANT FILE LIST
3       ELSE
4          CLEAR INVALID NAME FOR THAT POSITION IN TABLE
4          REDUCE COUNT OF CONSTRAINTS BY ONE  ( NC )
4          PRINT MESSAGE WARNING OF INVALID CONSTRAINT WHICH WILL BE IGNORED
3       ENDIF
2    ENDDO
2    CALL XZDMK TO CREATE CONSTRAINT VIOLATION MASKS
1 END XZDP1
```

Figure 5.3-1.- XZDP1 functional level PDL.

5.4 ROUTINE NAME - XZDMK

5.4.1 Purpose

XZDMK applies any defined constraints to the data contained in ATABLE.

5.4.2 Functional Description

Three arrays in common (MSKERR, MASK1, and MASK2) are built by XZDMK. Each entry in each of these arrays corresponds to a value position (upper/lower data point) on the display grid. MSKERR is an array of flags set to indicate that this value position is to be printed from MASK1 and MASK2 (instead of from the value in ATABLE). MASK1 and MASK2 are built to contain the ASCII characters (four each) to be printed in the upper and lower portions of each value position, respectively.

XZDMK first scans the error indicators and sets MASK1 and MASK2 to "$$$$" at value positions where the error indicator is nonzero.

Using the three arrays in common, which define the constraints (NCVARL, NCRELL, and CVALUE), XZDMK next performs the constraint checks on all value positions in ATABLE. When a constraint is determined as having been violated, the appropriate ASCII letter (A-H) is placed into MASK1 or MASK2, and the corresponding entry of MSKERR is zeroed to indicate a violation at this value position.

5.4.3 Assumptions and Limitations

None.

5.4.4 Method

None.

5.4.5 Routine Input/Output Variables

All input/output variables to XZDMK are via common, and the appropriate common variables are defined in section 5.4.2.

5.4.6 Functional Logic Flow

A functional logic flow is not provided. The functional level PDL for XZDMK is presented in figure 5.4-1.

5.4.7 Diagnostics and Debug

None.

5.4.8 Special Comments

None.

5.4.9 References

None.

```
1 BEGIN XZDMK
2    CLEAR CONSTRAINT VARIABLE VIOLATION TABLE
3    DO FOR REQUESTED ROWS (J=NY,1,-1)
3       DO FOR ALL COLUMNS ( I=1,11 )
4          SET MASK TO ERROR INDICATOR (N=1,NC)
4          DO FOR ALL CONSTRAINT VARIABLES
5             IF CONSYRAINT IS SET,THEN
6                CASE CONSTRAINT VARIABLE REL(:LT:,:LE:,:EQ:,:GE:,:GT:)

7                   :LT:
7                   IF DEPENDANT VALUE PASSES CONSTRAINT TEST (N), THEN
8                      BLANK OUT POSITION IN CONSTRAINT MASK
7                   ELSE
8                      SET INDICATOR IN CONSTRAIN VARIABLE VIOLATION TABLE
8                      SET ERROR INDICATION IN MASK FOR THIS ENTRY
8                      SET UP SYMBOL TO BE ADDED TO MASK BASED ON N
7                   ENDIF

7                   :LE:
7                   IF DEPENDANT VALUE PASSES CONSTRAINT TEST (N), THEN
8                      BLANK OUT POSITION IN CONSTRAINT MASK
7                   ELSE
8                      SET INDICATOR IN CONSTRAINT VAR. VIOLATION MASK
8                      SET ERROR INDICATION IN MASK FOR THIS ENTRY
8                      SET UP SYMBOL TO BE ADDED TO MASK BASED ON N
7                   ENDIF

7                   :EQ:
7                   IF DEPENDANT VALUE PASSES CONSTRAINT TEST (N), THEN
8                      BLANK OUT POSITION IN CONSTRAINT MASK
7                   ELSE
8                      SET INDICATOR IN CONSTRAINT VAR. VIOLATION MASK
8                      SET ERROR INDICATION IN MASK FOR THIS ENTRY
8                      SET UP SYMBOL TO BE ADDED TO MASK BASED ON N
7                   ENDIF

7                   :GE:
7                   IF DEPENDANT VALUE PASSES CONSTRAINT TEST (N), THEN
8                      BLANK OUT POSITION IN CONSTRAINT MASK
7                   ELSE
8                      SET INDICATOR IN CONSTRAINT VAR. VIOLATION MASK
8                      SET ERROR INDICATION IN MASK FOR THIS ENTRY
8                      SET UP SYMBOL TO BE ADDED TO MASK BASED ON N
7                   ENDIF

7                   :GT:
7                   IF DEPENDANT VALUE PASSES CONSTRAINT TEST (N), THEN
8                      BLANK OUT POSITION IN CONSTRAINT MASK
7                   ELSE
8                      SET INDICATOR IN CONSTRAINT VAR. VIOLATION MASK
8                      SET ERROR INDICATION IN MASK FOR THIS ENTRY
8                      SET UP SYMBOL TO BE ADDED TO MASK BASED ON N
7                   ENDIF
6                ENDCASE
5             ELSE
6                BLANK THIS POSITION IN MASK
5             ENDIF
5             COMPUTE K BASED ON N SO THAT CHARACTER IN PROPER POSITION OF 6 WD SET
5             LOGICAL "OR" MASK SYMBOL INTO MASK FOR THIS ENTRY
4          ENDDO   FOR  N
4          IF NO CONSTRAINT VIOLATION FLAG SET AND # CONSTR < 10, THEN
5             INSERT BLANKS IN MASK BETWEEN NC AND 10
4          ENDIF
3       ENDDO FOR  I
2    ENDDO FOR  J
1 END XZDMK
```

Figure 5.4-1.- XZDMK functional level PDL.

5.5 ROUTINE NAME - XZDP2

5.5.1 Purpose

XZDP2 prompts the user and calls XZDOT to output a display.

5.5.2 Functional Description

If the SELECT flag in common indicates that the user is not to be prompted, then for each page defined for display (NDVARL), indices defining the upper and lower variables (K1 and K2, respectively), as well as the page number (NPAGE), are put into common and XZDOT is called to output the display.

Otherwise, the user is prompted to provide the page number to be displayed (table 4-IV). The user may, optionally, also input the starting row number (NSTROW), number of rows (NROWS), and display logical unit (LUDSP). These quantities are also placed in common for use by XZDOT. Following return from XZDOT, the user is prompted again. When a "%" is input, control is returned to DBDSP.

5.5.3 Assumptions and Limitations

None.

5.5.4 Method

None.

5.5.5 Routine Input/Output Variables

All input and output to XZDP2 is via common, and the appropriate common variables are mentioned in section 5.5.2.

5.5.6 Functional Logic Flow

A functional logic flow is not provided. The functional level PDL for XZDP2 is presented in figure 5.5-1.

5.5.7 Diagnostics and Debug

When the user inputs an undefined page, invalid starting/number of rows combination, or illegal syntax in response to the prompt, an error message will be output. Messages 5, 11, and 12 are defined in table 4-VII.

5.5.8 Special Comments

None.

5.5.9 References

None.

```
1 BEGIN XZDP2
2    IF PROMPT OPTION CHOSEN, THEN
3       DO UNTIL USER INPUTS VALID PAGE NUMBER OR % SIGN
4          PROMPT AND CALL XPRDM TO OBTAIN USER RESPONSE
3       EXIT IF % SIGN ENTERED
4          IF PAGE NUMBER IS > 16 OR < 0, THEN
5             PRINT INVALID PAGE NUMBER
4          ELSE
5             DO FOR NUMBER OF DISPLAY DEPENDANT VARIABLES
6                STARTSEARCH UNTIL DISPLAY DEP. VAR. ARE CHECKED AGAINST MSTR NAME LIST
6                EXITIF DISPLAY DEPENDANT VARIABLE FOUND TO BE VALID
6                ENDLOOP
7                   SET INDICATOR(J)=-1
7                   PRINT MESSAGE NAME NOT FOUND
6                ENDSEARCH
6                SET INDICATOR(J) TO MASTER LIST INDEX VALUE
5             ENDDO
5             IF NEITHER DEPENDANT VARIABLE IS FOUND, THEN
6                PRINT ERROR MESSAGE
5             ELSE
6                INITIALIZE PARAMETERS FOR THIS DISPLAY PAGE
6                IF THERE IS NO FIRST VARIABLE, THEN
7                   PROMOTE SECOND VARIABLE TO FIRST ,CLEAR SECOND VARIABLE
7                   CALL XZDOT TO OUTPUT THE DISPLAY PAGE
6                ENDIF
5             ENDIF
4          ENDIF
3       ENDDO
2    ELSE
3       DO UNTIL ALL PAGES HAVE BEEN OUTPUT
4          PREPARE PAGE FOR OUTPUT TO LU
4          CALL XZDOT TO OUTPUT THE DISPLAY PAGE
3       ENDDO
2    ENDIF
2    RETURN
1 END XZDP2
```

Figure 5.5-1.- XZDP2 functional level PDL.

5.6 ROUTINE NAME - XZDOT

5.6.1 Purpose

XZDOT outputs one display to the designated logical unit.

5.6.2 Functional Description

The heading information containing the data box name, page number, dependent (upper and lower) variable names, units, and scale factors, as well as any constraints violated, are built into print lines and output to the selected LU (LUDSP).

Dependent on the starting row (NSTROW) and number of rows (NOROW), loop indices are computed for generating the proper rows to be output. For each row output, two print lines are built. The first contain the Y-axis value and a value or constraint mask for each of the upper value positions. The second print line contains a value or constraint mask for each of the lower value positions. These two print lines are output within the loop.

Finally, print lines defining the X-axis variable and values are built and output.

5.6.3 Assumptions and Limitations

None.

5.6.4 Method

None.

5.6.5 Routine Input/Output Variables

The XZDOT input/output variables are presented in table 5.6-I.

5.6.7 Diagnostics and Debug

When DBDSP is scheduled by FDS with bit 4 of the flags parameter set, debug interaction at the user's terminal will occur.

5.6.8 Special Comments

None.

5.6.9 References

None.

TABLE 5.6-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine XZDOT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
K1			I				Index to upper dependent variable.
K2			I				Index to lower dependent variable.
ZTABLE			I				Array of names and units for all dependent variables.
ATABLE			I				Array of dependent variables' values.
NOVRUL			I				Array of scale factors for each page.
NPAGE			I				Page number.
DTBX			I				Data box name.
NCVARL			I				Constraint variable names.
NCRELL			I				Constraint relations.
CVALUE			I				Constraint values.
XSCNNM YSCNNM			I				Independent variables names.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.6-I.- Concluded

Routine XZDOT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
XUNITS YUNITS			I				Independent variables units.
IXSCN1 IXSCN2 IYSCN1 IYSCN2			I				Independent variables subscripts.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

```
1 BEGIN XZDOT
2    CONSTRUCT AND DISPLAY PAGE HEADER INFORMATION
2    BLANK DISPLAY SPACE
2    DO FOR NY
3       OUTPUT BLANK LINE
3       CONVERT AND STORE Y-AXIS LABEL
3       DO TWICE FOR K1 AND K2
4          DO FOR ALL NX
5             IF ERROR FLAG NOT SET ,THEN
6                SCALE AND CONVERT VALUE
5             ELSE
6                PERFORM CONSTRAINT MASKING
5             ENDIF
5             OUTPUT LINE
5             BLANK Y-AXIS LABEL
4          ENDDO
3       ENDDO
2    ENDDO
2    CONSTRUCT X - AXISAND DISPLAY TRAILER INFORMATION
1 END XZDOT
```

Figure 5.6-1.- XZDOT functional level PDL.

DATA BOX VARIABLE EXTRACTOR PROCESSOR (DBEXT)

1.0 PURPOSE

The purpose of the data box variable extractor (DBEXT) is to provide the FDS user a means by which either an independent or dependent variable value can be collected from a specified data box. The data box must have been generated by SCAN/ENDSC or DBINT processors, at which time the user-specified range(s) for the independent variable(s) were recorded in the data box.

2.0 FUNCTIONAL DESCRIPTION

The input data for DBEXT is specified in its interface table. When initiating the DBEXT, the FDS user can specify the following:

a. Data box file name

b. Name of the independent or dependent variable to be extracted

c. Indicator designating the type of variable being extracted (independent or dependent)

d. Name and value of the independent variable to be held constant if a dependent variable is being extracted and there are two independent variables in the data box

The input parameters are retrieved, then stored internally for later use. The input data is used by DBEXT to obtain the required parameter names, parameter value, and variable indicator. The maximum size for the data box file to be searched by DBEXT is 121 (11 x 11) summary vectors.

If an independent variable has been specified for extraction, the range of incremental values (1 to 11) for that independent variable are collected for output.

If a dependent variable has been specified and this data box contains only one independent variable, the values (1 to 11) for this dependent variable are extracted from the summary vectors in the data box for output.

If a dependent variable has been specified and this data box contains two independent variables, the user must indicate which of the two independent variables should be fixed (and at what value) in order that the values (1 to 11) for the dependent variable over the range of the other independent variable may be extracted from the data box.

The user-specified constant value (if any) for an independent variable must correspond to one of the incremented (1 to 11) occurrences of that independent variable.

Since there is a limit of only one independent or dependent variable to be extracted, the maximum size of the extracted variable array is 11 elements. The array generated by DBEXT is placed into the FDS user's AWA and is available for subsequent use.

3.0 ASSUMPTIONS AND LIMITATIONS

Listed below are the assumptions and limitations associated with the DBEXT processor.

a. DBEXT can be utilized to extract variables from a data box generated by either the SCAN/ENDSC or DBINT processors.

b. DBEXT will extract only one independent or one dependent variable per execution of the processor.

c. The maximum size of the input data box from which the specified variable is to be extracted is 121 summary vectors and each summary vector is limited to 32 values.

d. The name of the variable to be extracted must be one of the independent or dependent variables contained in the specified data box.

4.0 PROCESSOR INPUT/OUTPUT

a. Processor interface table - The processor interface table for DBEXT is defined in table 4-I and contains:

   (1) Processor constants array

   (2) Data box to be searched

   (3) Variable to be extracted

   (4) Indicator for type variable to be extracted

   (5) Variable to be held constant for the search (not used if the user specified an independent variable). Must be set to "DUMMY" if not used.

   (6) Value for variable to be held constant (not used if the type indicator specifies an independent variable).

   (7) Output data array of 1 to 11 real values.

b. Interface table data array definitions - The contents and size of the output data array depend on the particular data box and variables specified as input.

c. Interface table data file definitions - The interface table data file definitions are provided in table 4-II.

d. Processor solicited (prompted) inputs - None.

e. Processor displays and display parameter definition table - None.

f. Processor message table - The format and content of the processor message table is provided in table 4-III.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-IV.

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR DBEXT

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Value stored in default interface table	Definition
PROCON	AWA	Intg	I	1	1	20	Cartridge number for data box file
DATBOX	Disk	Free	I	—	—	--	Data box generated by SCAN/ENDSC or DBINT utility processors
EXTVAR	AWA	6CH	I	3	1	--	Name of variable to be extracted: "NAMEXX"
TYPE	AWA	2CH	I	1	1	--	Indicator of type variable to be extracted: "IN" = Independent "DE" = Dependent
CONVAR	AWA	6CH	I	3	1		Independent variable to be held constant if a dependent variable is being extracted and there are two independent variables in the data box
CONST	AWA	Real	I	2	1	--	Constant value for the independent variable (CONVAR).

NOTES

CLASS: AWA, Disk

TYPE: Free, Intg, Real, Dubl, 2CH, 6CH, 18CH, 36CH, 72CH, Mix, Symb

USE: I = Input; O = Output; I/O = Input/Output

TABLE 4-I.- Concluded

PROCESSOR DBEXT

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Value stored in default interface table	Definition
OUTDAT	AWA	Real	O	2-22	1-11	--	Array of 1 to 11 real values of output data

NOTES	CLASS	TYPE			USE
		Free	2CH	72CH	I = Input
	AWA	Intg	6CH	Mix	O = Output
	Disk	Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-II.- INTERFACE TABLE DATA FILE DEFINITIONS

PROCESSOR DBEXT

DRDE DATA FILE DATBOX

(Type 2; Record Length - 64 words)

Record number	Integer word allocations	Content and definition
1	64	(1) Name of FDS processor creating the file (4) Interface table variable name for this file (7) Name of FDS processor updating file (three ASCII words or blanks) (10) Interface table variable name for this update (three ASCII words or blanks)
2	64	(1) Number of entries in sumtab (Integer 1-32) (2) X-scan variable (6 chars) (5) X-first subscript (Integer, or null if none) (6) X-second subscript (Integer, or null if none) (7) X units (6 characters) (10) X centroid (real) (12) X increment (real) (14) X number of steps (Integer 0-5) (15) Y-scan variable (6 chars, or null if none) (18) Y-first subscript (Integer, or null if none) (19) Y-second subscript (Integer, or null if none) (20) Y units (6 characters) (23) Y centroid (real) (25) Y increment (real) (27) Y number of steps (Integer 0-5)
3	64	First 64 words of sumtab variable name
4	64	Last 32 words of sumtab variable names First 32 words of sumtab variable units
5	64	Last 64 words of sumtab variable units
6	64	Values for scan variable(s) X_1 or X_1Y_1

TABLE 4-II.- Concluded

PROCESSOR DBEXT

DRDE DATA FILE DATBOX

Record number	Integer word allocations	Content and definition
7 . . . n+5*	64 64	Values for scan variable(s) X_2 or X_2Y_1 Values for scan variable(s) X_n or X_nY_1
n+6* . . .	64	(If there is no Y-scan variable, record n+5 is the last record, else) Values for scan variables X_1 Y_2
n+m+5*	64	Values for scan variables X_nY_m *Note: n = number of values of X m = number of values of Y

TABLE 4-III.- PROCESSOR MESSAGE TABLE

PROCESSOR DBEXT

MSG no.	Message ID block	Message text block and explanation
1	*DBEXT*	"DATA BOX COULD NOT BE FOUND." Meaning: The FDS user specified the name for a data box that could not be found. Severity: Execution of DBEXT processor terminated. Action required by user: Execute DBEXT with the correct data box name.
2	*DBEXT*	"INDEPENDENT VARIABLE NAME NOT IN THE SPECIFIED DATA BOX." Meaning: The FDS user specified an independent variable not found in the specified data box. Severity: Execution of DBEXT processor terminated. Action required by user: Execute DBEXT with the correct independent variable name.
3	*DBEXT*	"DEPENDENT VARIABLE NAME NOT IN THE SPECIFIED DATA BOX." Meaning: The FDS user specified a dependent variable not found in the specified data box. Severity: Execution of DBEXT processor terminated. Action required by user: Execute DBEXT with the correct dependent variable name.
4	*DBEXT*	"INDICATOR FOR THE TYPE OF VARIABLE BEING EXTRACTED IS INVALID." Meaning: The FDS user specified an invalid response for the TYPE parameter. Valid type parameters are either "IN" for an independent variable to be extracted, or "DE" for a dependent variable to be extracted. Severity: Execution of DBEXT processor terminated. Action required by user: Correct TYPE parameter and reexecute.
5	*DBEXT*	"VALUE FOR CONST IS OUTSIDE THE RANGE OF THE SPECIFIED DATA BOX." Meaning: The value specified for CONST is outside the range of the data box. Severity: Execution of DBEXT processor terminated. Action required by user: Correct the value for CONST and reexecute the DBEXT processor.

TABLE 4-III.- Concluded

PROCESSOR DBEXT

MSG no.	Message ID block	Message text block and explanation
6	*DBEXT*	"ERROR IN OPENING DATA BOX--ERROR CODE_____." Meaning: The DRDE file specified as the input could not be successfully opened. The return code from OPEN is contained in the message text. Severity: Execution of DBEXT processor terminated. Action required by user: Notify FDS system maintenance personnel.
7	*DBEXT*	"ERROR IN READING DATA BOX--ERROR CODE_____." Meaning: The DRDE file specified as the input data box could not be successfully read. The return code is contained in the message text. Severity: Execution of DBEXT processor terminated. Action required by user: Notify FDS system maintenance personnel.
8	*DBEXT*	"ONLY ONE INDEPENDENT VARIABLE IN THE DATA BOX." Meaning: The FDS user did not set interface table parameter "CONVAR" to "DUMMY", and there is only one independent variable in the data box. Severity: Execution of DBEXT terminated. Action required by user: If data box contains only one independent variable, set interface table parameter "CONVAR" to "DUMMY".
9	*DBEXT*	"THERE ARE TWO INDEPENDENT VARIABLES IN THE DATA BOX." Meaning: The FDS user set interface table parameter "CONVAR" to "DUMMY", and there were two independent variables in the data box. Severity: Execution of DBEXT terminated. Action required by user: Set "CONVAR" to the name of the independent variable to be held constant.
10	*DBEXT*	"DBEXT PROCESSOR ABNORMALLY TERMINATED." Meaning: The DBEXT processor encountered an error from which it could not recover. Severity: Execution of DBEXT terminated. Action required by user: Determine problem; correct and execute DBEXT again.

TABLE 4-IV.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR DBEXT

Processor name	Processor abstract prompt (maximum 256 characters)
DBEXT	DBEXT processor extracts either independent or dependent variable values from a specified box. The user-specified range(s) for the independent variable(s) were recorded in the data box as it was generated.
Parameter keyword name	**Parameter definition prompt (maximum 256 characters)**
PROCON	PROCON is the cartridge number for the data box file (one integer value).
DATBOX	DATBOX is the DRDE; contents are five header records and one summary table for each iteration of SCAN or DBINT.
EXTVAR	EXTVAR is the variable to be extracted from the data box (six characters).
TYPE	TYPE is the indicator of the type variable to be extracted from the data box. Valid inputs are "IN" for an independent variable or "DE" for a dependent variable (two characters).
CONVAR	CONVAR is the name of the independent variable to be held constant if a dependent variable is being extracted and there are two independent variables in the data box (six characters).
CONST	CONST is the constant value for the independent variable to be held constant if a dependent variable is being extracted and there are two independent variables in the data box (one real value).
OUTDAT	OUTDAT is the array of 1 to 11 real values of output data extracted from the data box.

5.0 PROCESSOR ROUTINES

The only available routine documentation is contained on the comment cards in the software listing, with the exception of the PDL provided in figure 5-1.

```
CD*****************************************************
CDØ
CDØ   DBEXT - DATA BOX VARIABLE EXTRACTOR PROCESSOR
CDØ
CDØ         - SCHEDULED BY FDS
CDØ
CD*****************************************************
CD1
CD1   DBEXT PROVIDES THE FDS USER A MEANS BY WHICH EITHER AN
CD1   INDEPENDENT OR DEPENDENT VARIABLE CAN BE COLLECTED FROM A
CD1   SPECIFIELD DATA BOX.
CD1
CD*****************************************************
CD2
CD2   INPUTS TO DBEXT FROM THE INTERFACE TABLE
CD2
CD2     PROCON - CARTRIDGE # FOR DATA BOX
CD2     DATBOX - DATA BOX FILE NAME
CD2     EXTVAR - VARIABLE TO BE EXTRACTED
CD2     TYPE   - INDICATOR OF THE TYPE VARIABLE TO BE EXTRACTED
CD2     CONVAR - INDEPENDENT VARIABLE TO BE HELD CONSTANT IF A
CD2              DEPENDENT VARIABLE IS BEING EXTRACTED AND THERE ARE
CD2              TWO INDEPENDENT VARIABLES IN THE DATA BOX
CD2     CONST  - CONSTANT VALUE FOR THE INDEPENDENT VARIABLE
CD2
CD2
CD2   INPUTS TO DBEXT FROM THE DRDE FILE
CD2
CD2      RECORD 1
CD2        (1)  - NAME OF THE FDS PROCESSOR CREATING THE FILE
CD2        (4)  - INTERFACE TABLE VARIABLE NAME FOR THIS FILE
CD2        (7)  - NAME OF THE FDS PROCESSOR UPDATING THE FILE
CD2                  (3 ASCII WORDS OR BLANKS)
CD2        (1Ø) - INTERFACE TABLE VARIABLE NAME FOR THIS UPDATE
CD2                  (3 ASCII WORDS OR BLANKS)
CD2
CD2
CD2      RECORD 2
CD2        (1)  - NUMBER OF ENTRIES IN THE SUMMARY TABLE
CD2        (2)  - X SCAN VARIABLE (6 CHARACTERS)
CD2        (5)  - X FIRST SUBSCRIPT (INTG OR ZERO IF NONE)
CD2        (6)  - X SECOND SUBSCRIPT (INTG OR ZERO IF NONE)
CD2        (7)  - X UNITS (6 CHARACTERS)
CD2        (1Ø) - X CENTROID (REAL)
CD2        (12) - X INCREMENT (REAL)
CD2        (14) - X NUMBER OF STEPS (INTEGER Ø-5)
CD2        (14) - Y SCAN VARIABLE (6 CHARACTERS)
CD2        (18) - Y FIRST SUBSCRIPT (INTG OR ZERO IF NONE)
CD2        (19) - Y SECOND SUBSCRIPT (INTG OR ZERO IF NONE)
CD2        (2Ø) - Y UNITS (6 CHARACTERS)
CD2        (23) - Y CENTROID (REAL)
CD2        (25) - Y INCREMENT (REAL)
CD2        (27) - Y NUMBER OF STEPS (INTEGER Ø-5)
CD2
CD2     SUMMARY TABLE RECORDS
CD2       - EACH SUMMARY TABLE CONTAINS VALUES FOR EACH DEPENDENT
CD2         VARIABLE SCANNED (UP TO 32 VALUES INCLUDING THE ERROR
CD2         FLAG WHICH IS THE FIRST VALUE IN THE SUMMARY TABLE)
CD2
CD2     DEPENDENT VARIABLE NAME AND UNITS ARE IN RECORDS 3,4, AND 5
CD2     OF THE DRDE FILE
CD2
CD*****************************************************
```

Figure 5-1.- DBEXT functional level PDL.

```
CD3
CD3      OUTPUT FROM DBEXT
CD3
CD3        - ARRAY OF 1 TO 11 REAL VALUES OF OUTPUT DATA
CD3
CD*****************************************************
CD4
CD4      DEFINE VARIABLES
CD4
CD4    CNSTVP - POINTER TO INDEPENDENT VARIABLE TO BE HELD CONSTANT
CD4    CONVAR - INDEPENDENT VARIABLE TO BE HELD CONSTANT IF A
CD4             DEPENDENT VARIABLE IS BEING EXTRACTED AND THERE ARE
CD4             TWO INDEPENDENT VARIABLES IN THE DATA BOX
CD4    CONST  - VALUE FOR THE INDEPENDENT VARIABLE TO BE HELD
CD4             CONSTANT
CD4    DATBOX - DATA BOX NAME
CD4    DATBUF - BUFFER FOR SUMMARY TABLE DESCRIPTORS
CD4    DE     - CONSTANT TO COMPARE AGAINST TYPE
CD4    ENDARY - END POINT FOR VALARY
CD4    EXTVAR - VARIABLE TO BE EXTRACTED
CD4    IN     - CONSTANT TO COMPARE AGAINST TYPE
CD4    INTBUF - INTERFACE TABLE HEADER BUFFER
CD4    LU     - LOGICAL UNIT NUMBER
CD4    MINCR  - INCREMENT MINUS INCREMENT/4
CD4    MRBUFF - MANAGER REQUEST BUFFER
CD4    NUMENT - NUMBER OF ENTRIES IN THE SUMMARY TABLE
CD4    NXSTEP - NUMBER OF STEPS ON EITHER SIDE OF X CENTROID
CD4    NYSTEP - NUMBER OF STEPS ON EITHER SIDE OF Y CENTROID
CD4    PINCR  - INCREMENT PLUS INCREMENT/4
CD4    POINT  - POINTER TO THE LOCATION OF THE DEPENDENT VARIABLE TO
CD4             EXTRACTED
CD4    PROCON - CARTRIDGE # FOR THE DATA BOX
CD4    TYPE   - INDICATOR OF THE TYPE VARIABLE TO BE HELD CONSTANT
CD4    VALARY - X AND Y COMPUTED VALUES ARRAY EQUIVALENCED TO XVV
CD4             AND YVV
CD4    VALPT  - POINTER TO THE VALUE TO HOLD CONSTANT
CD4    VAP    - VALUE ARRAY POINTER
CD4    XCENT  - X CENTROID
CD4    XINCR  - X INCREMENT
CD4    XSCANV - X SCAN VARIABLE NAME
CD4    XSTEPS - TOTAL NUMBER OF STEPS FOR X VARIABLE
CD4    XVV    - COMPUTED VALUES FOR X VARIABLE
CD4    YCENT  - Y CENTROID
CD4    YINCR  - Y INCREMENT
CD4    YSCANV - Y SCAN VARIABLE NAME
CD4    YSTEPS - TOTAL NUMBER OF STEPS FOR Y VARIABLE
CD4    YVV    - COMPUTED VALUES FOR Y VARIABLE
CD4
CD*****************************************************
CD*****************************************************
CD5
CD5      EXTERNAL REFERENCES
CD5
CD*****************************************************
CD*****************************************************
CD5
CD5      FDS SUBROUTINES USED:
CD5
CD5    XPGET                  XPPUT
CD5    XPXIT                  XRCPR
CD5    XREXT                  XRI6
CD5    XUDBG
CD5
CD5      RTE SUBROUTINES USED:
```

Figure 5-1.- Continued.

```
BEGIN DBEXT
   PERFORM INITIALIZATION
   IF "TYPE" MATCHES "DE"
   THEN  [DEPENDENT VARIABLE PROCESSING                                    ]
      COMPUTE VALUES FOR INDEPENDENT VARIABLES
      READ THIRD RECORD CONTAINING DEPENDENT VARIABLE NAMES
      ERROREXIT (ERR3) IF READ ERROR FLAG SET
      SEARCH [FIND DEPENDENT VARIABLE TO BE EXTRACTED                      ]
         DO
            EXITIF DEPENDENT VARIABLE TO BE EXTRACTED FOUND
            EXITTHEN [DEPENDENT VARIABLE FOUND                             ]
               SET LOCATION OF DEPENDENT VARIABLE FOUND IN SUMMARY TABLE
            EXITENDIF
         UNTIL ALL DEPENDENT VARIABLE NAMES ARE CHECKED
         ENDDO
         ERROREXIT (ERR7) IF DEPENDENT VARIABLE NAME NOT IN DATA BOX
      ENDSEARCH
      IF "CONVAR" DOES NOT MATCH "DUMMY"
      THEN
         ERROREXIT (ERR8) IF ONLY ONE INDEPENDENT VARIABLE NAME IN DATA BOX
         SEARCH [EACH OF TWO INDEPENDENT VARIABLES                         ]
            FOR ONE OF TWO INDEPENDENT VARIABLES
            DO
               EXITIF "CONVAR" MATCHES INDEPENDENT VARIABLE NAME
               EXITTHEN [CORRECT INDEPENDENT VARIABLE NAME FOR CONSTANT VALUE FOUND    ]
                  SEARCH [FIND "CONST" IN COMPUTED VALUES FOR INDEPENDENT VARIABLE      ]
                     DO
                        FOR ALL INDEPENDENT VARIABLE VALUES
                        EXITIF "CONST" .EQ. A COMPUTED VALUE OF THE PROPER INDEPENDENT VARIABL
                        EXITTHEN [A MATCH BETWEEN "CONST" AND A COMPUTED VALUE IS FOUND          ]
                           DO
                              READ NEXT SUMMARY TABLE RECORD
                              ERROREXIT (ERR3) IF READ ERROR FLAG SET
                              EXTRACT DEPENDENT VARIABLE VALUE BASED ON LOCATION
                              STORE VALUE IN "OUTDAT"
                           UNTIL OTHER INDEPENDENT VARIABLE IS INCREMENTED OVER ITS RANGE
                           ENDDO
                           CALL XPPUT TO MOVE "OUTDAT" TO AWA
                        EXITENDIF
                     ENDDO
                     ERROREXIT (ERR6) IF "CONST" .NE. A COMPUTED VALUE
                  ENDSEARCH
               EXITENDIF
            ENDDO
            ERROREXIT (ERR4) IF INDEPENDENT VARIABLE NAME DOES NOT MATCH DATA BOX
         ENDSEARCH
      ELSE
         ERROREXIT (ERR9) IF THERE ARE TWO INDEPENDENT VARIABLE NAMES IN THE DA
         DO
            READ NEXT SUMMARY TABLE RECORD
            ERROREXIT (ERR3) IF READ ERROR FLAG SET
            EXTRACT DEPENDENT VARIABLE VALUE BASED ON LOCATION
            STORE VALUE IN "OUTDAT"
         UNTIL INDEPENDENT VARIABLE IS INCREMENTED OVER ITS RANGE
         ENDDO
```

Figure 5-1.- Continued.

```
         CALL XPPUT TO MOVE "OUTDAT" TO AWA
      ENDIF
   ELSE  [INDEPENDENT VARIABLE PROCESSING                                    ]
      IF "TYPE" MATCHES "IN"
      THEN
         SEARCH [ONE OR TWO INDEPENDENT VARIABLES                            ]
            DO
               EXITIF "EXTVAR" MATCHES INDEPENDENT VARIABLE NAME
               EXITTHEN [CORRECT INDEPENDENT VARIABLE FOUND                  ]
                  GET PROPER CENTROID AND NUMBER OF STEPS
                  COMPUTE VALUES FOR THE INDEPENDENT VARIABLE TO BE EXTRACTED
                  STORE VALUES IN PROPER LOCATION IN "OUTDAT"
                  CALL XPPUT TO MOVE "OUTDAT" TO USER AWA
               EXITENDIF
            UNTIL BOTH INDEPENDENT VARIABLES ARE CHECKED
            ENDDO
            ERROREXIT (ERR4) IF INDEPENDENT VARIABLE NOT FOUND
         ENDSEARCH
      ELSE
         ERROREXIT (ERR5) IF "TYPE" DOES NOT MATCH "IN" OR "DE"
      ENDIF
   ENDIF
EXIT DBEXT
*************************************************************************

   *************************************************************************
   BEGINSEGMENT INITIALIZATION
      CALL XPGET TO RETRIEVE PROCON, DATA BOX NAME, NAME AND TYPE OF
      VARIABLE TO BE EXTRACTED, VARIABLE TO BE HELD CONSTANT,
      AND CONSTANT VALUE
      SET DATA BOX CARTRIDGE NUMBER BASED ON PROCON
      OPEN DRDE FILE
      ERROREXIT (ERR2) IF DATA BOX COULD NOT BE FOUND
      ERROREXIT (ERR1) IF OPEN ERROR FLAG SET
      READ SECOND RECORD CONTAINING DATA BOX SETUP
      ERROREXIT (ERR3) IF READ ERROR FLAG SET
   ENDSEGMENT INITIALIZATION
   *************************************************************************

   *************************************************************************
   BEGINERRORSEGMENT
      CASE
         TYPE OF ERROR
         PART(ERR1)
         CALL EXEC-'ERROR IN OPENING DATA BOX--ERROR CODE X'
         PART(ERR2)
         CALL EXEC-'NAME SPECIFIED NOT A DATA BOX'
         PART(ERR3)
         CALL EXEC-'ERROR IN READING DATA BOX--ERROR CODE X'
         PART(ERR4)
         CALL EXEC-'INDEPENDENT VARIABLE NAME NOT IN THE SPECIFIED DATA BOX'
         PART(ERR5)
         CALL EXEC-'INDICATOR FOR THE TYPE OF VARIABLE BEING EXTRACTED IS INVAL
         PART(ERR6)
         CALL EXEC-'VALUE FOR CONST IS OUTSIDE THE RANGE OF THE SPECIFIED DATA
         PART(ERR7)
         CALL EXEC-'DEPENDENT VARIABLE NAME NOT IN THE SPECIFIED DATA BOX'
         PART(ERR8)
         CALL EXEC-'ONLY ONE INDEPENDENT VARIABLE IN THE DATA BOX'
         PART(ERR9)
         CALL EXEC-'THERE ARE TWO INDEPENDENT VARIABLES IN THE DATA BOX'
      ENDCASE
      CALL EXEC-'DBEXT PROCESSOR ABNORMALLY TERMINATED'
      CALL XPXIT TO TERMINATE DBEXT ABNORMALLY

      ENDERRORSEGMENT
      *************************************************************************

   END DBEXT
```

Figure 5-1.- Concluded

DATA BOX INTERPOLATOR PROCESSOR (DBINT)

1.0 PURPOSE

The purpose of the data box interpolator (DBINT) is to provide the FDS user with the capability to perform a linear interpolation of a dependent variable over a specified range for the independent variables, X and Y. This linear interpolation will generate a new data box.

2.0 FUNCTIONAL DESCRIPTION

The input data for DBINT is specified in its interface table. When initializing the data box interpolator the FDS user will specify the following:

a. Input data box name

b. Centroid value of the X-independent variable in the new data box

c. Centroid value of the Y-independent variable in the new data box

d. Increment value of the X-independent variable in the new data box

e. Increment value of the Y-independent variable in the new data box

f. Number of steps in the X direction (maximum of five steps) in the new data box

g. Number of steps in the Y direction (maximum of five steps) in the new data box

Based on the new range(s) specified for the independent variable(s), dependent variable(s) values are used with a linear or bilinear interpolation technique to generate a new output data box. The range(s) of this new data box correspond to those specified by the FDS user (see above).

The maximum dimensions of the data box file are (11, 11, 32) where the first two values represent the maximum number of summary tables (121) that can be generated by one SCAN/ENDSC pair or by a previous execution of DBINT. The last value is the maximum number of values allowed, including the error indicator, in one summary table.

The first parameter of each summary table is an error flag which is set by the individual functional processor(s). A nonzero real value indicates that the point is invalid and should not be used to generate a new box. If, during the generation of a summary vector (dependent variables) by DBINT, an invalid point is encountered in the input data box, this summary vector will be marked invalid in the output data box. An error message will be transmitted for only the first such invalid point encountered.

3.0 ASSUMPTIONS AND LIMITATIONS

Listed below are a set of the assumptions and limitations associated with the DBINT processor.

a. The first value in each summary table will be an error flag (= 0 indicates no error, ≠ indicates error).

b. Summary tables used to build data boxes will be limited to a maximum of 32 values, including the error flag.

c. The number of steps (parameter keywords XSTEPS and YSTEPS) can only have an integer value of 0-5 inclusive.

d. The range(s) of the independent variable(s) specified for the new data box may not go beyond the range(s) of the independent variable(s) defined in the input data box.

e. DBINT can be utilized for the linear interpolation of values from a data box generated by either SCAN/ENDSC or a previous DBINT execution.

4.0 PROCESSOR INPUT/OUTPUT

a. Processor interface table - The processor interface table for DBINT is defined in table 4-I and contains:

   (1) Processor constants

   (2) Input data box file name

   (3) Centroid value of X-independent variable (XCENTR) for the new data box

   (4) Centroid value of Y-independent variable (YCENTR) for the new data box

   (5) Increment value of X-independent variable (XINCR) for the new data box

   (6) Increment value of Y-independent variable (YINCR) for the new data box

   (7) Number of steps on each side of XCENTR (XSTEPS) for the new data box

   (8) Number of steps on each side of YCENTR (YSTEPS) for the new data box

   (9) Output data box name

b. Interface table data array definitions - None.

c. Interface table data file definitions - The format and definitions of the output data file are provided in table 4-II.

d. Processor solicited (prompted) inputs - None.

e. Processor displays and display parameter definition table - None.

f. Processor message table - The DBINT processor error messages are provided in table 4-III.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-IV.

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR DBINT

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Value stored in default interface table	Definition
PROCON	AWA	Intg	I	1	1	20	Cartridge number for data box
INDTBX	Disk	Free	I	VAR	—		Input data box
XCENTR	AWA	Real	I	2	1	--	X centroid
XINCR	AWA	Real	I	2	1	--	X increment
XSTEPS	AWA	Intg	I	1	1	--	Number of steps for X
YCENTR	AWA	Real	I	2	1	--	Y centroid
YINCR	AWA	Real	I	2	1	--	Y increment
YSTEPS	AWA	Intg	I	1	1	--	Number of steps for Y
OTDTBX	Disk	Free	O	VAR	--	--	Output data box

NOTES

CLASS: AWA, Disk

TYPE: Free 2CH 72CH; Intg 6CH Mix; Real 18CH Symb; Dubl 36CH

USE: I = Input; O = Output; I/O = Input/Output

TABLE 4-II.- INTERFACE TABLE DATA FILE DEFINITIONS

PROCESSOR DBINT

DRDE DATA FILE INDTBX and OTDTBX

(Type 2, Record Length - 64 words)

Record number	Integer word allocations	Content and definition
1	64	(1) Name of FDS processor creating the file (4) Interface table variable name for this file (7) Name of FDS processor updating file (3 ASCII words or blanks) (10) Interface table variable name for this update (3 ASCII words or blanks)
2	64	(1) Number of entries in sumtab (integer 1-32) (2) X-scan variable (6 characters or NULL if none) (5) X-first subscript (integer or NULL if none) (6) X-second subscript (integer or NULL if none) (7) X-units (6 characters) (10) X-centroid (real) (12) X-increment (real) (14) X-number of steps (integer 0-5) (15) Y-scan variable (6 characters or NULL if none) (18) Y-first subscript (integer or NULL if none) (19) Y-second subscript (integer or NULL if none) (20) Y-units (6 characters) (23) Y-centroid (real) (25) Y-increment (real) (27) Y-number of steps (integer 0-5)
3	64	First 64 words of sumtab variable name
4	64	Last 32 words of sumtab variable names First 32 words of sumtab variable units
5	64	Last 64 words of sumtab variable units
6	64	Values for scan variable(s) X_1 or X_1Y_1

TABLE 4-II.- Concluded

PROCESSOR DBINT

DRDE DATA FILE INDTBX and OTDTBX

(Type 2, Record Length = 64 words)

Record number	Integer word allocations	Content and definition
7 . . . n+5	64	Values for scan variable(s) X_2 or X_2Y_1 Values for scan variable(s) X_n or X_nY_1
n+6* . . . n+m+5	64	(If there is no Y-scan variable, record n+5 is the last record, else) values for scan variables X_1 Y_2 Values for scan variables X_nY_m
		*Note: n = Number of values of X m = Number of values of Y

TABLE 4-III.- PROCESSOR MESSAGE TABLE

PROCESSOR DBINT

MSG no.	Message ID block	Message text block and explanation
1	*DBINT*	"DATA BOX COULD NOT BE FOUND." Meaning: The FDS user specified the name for a data box that could not be found on the DRDE file. Severity: Execution of DBINT processor is terminated. Action required by user: Execute DBINT with the correct data box name.
2	*DBINT*	"ERROR IN OPENING OLD DATA BOX--ERROR CODE_____." Meaning: The DRDE file specified as the input could not be successfully opened. The return code from OPEN is contained in the message text. Severity: Execution of DBINT processor is terminated. Action required by user: Notify the FDS system maintenance personnel.
3	*DBINT*	"NUMBER OF STEPS MUST BE BETWEEN 0 AND 5, INCLUSIVE." Meaning: Either the value for XSTEPS and/or YSTEPS is <0 or >5. Severity: Execution of DBINT processor is terminated. Action required by user: Set the value for XSTEPS and/or YSTEPS to a valid value and reexecute DBINT.
4	*DBINT*	"ERROR IN READING DATA BOX--ERROR CODE_____." Meaning: The DRDE file specified as the input data box could not be successfully read. The return code is contained in the message text. Severity: Execution of DBINT processor is terminated. Action required by user: Notify the FDS system maintenance personnnel.
5	*DBINT*	"INPUT DATA BOX CONTAINS AN INVALID SUMMARY VECTOR." Meaning: During the generation of a summary vector, an invalid point was encountered in the input data box. Severity: Normal execution of DBINT continues. Action required by user: None. The summary vector is marked invalid in the output data box. Only one such message per execution of DBINT is generated.

TABLE 4-III.- Concluded

PROCESSOR DBINT

MSG no.	Message ID block	Message text block and explanation
6	*DBINT*	"ERROR IN WRITING DATA BOX--ERROR CODE_____." Meaning: The DRDE file specified as the output could not be successfully written. The return code is contained in the message text. Severity: Execution of DBINT processor is terminated. Action required by user: Notify the FDS system maintenance personnel.
7	*DBINT*	"ERROR IN PURGE OF NEW DATA BOX--ERROR CODE_____." Meaning: Error returned from purge (other than FMGR ERROR-006) during the purge of the new data box. Severity: Execution of DBINT processor is terminated. Action required by user: Notify the FDS system maintenance personnel.
8	*DBINT*	"ERROR IN CREATING NEW DATA BOX--ERROR CODE_____." Meaning: The DRDE file specified as the output could not be successfully created. Severity: Execution of DBINT processor is terminated. Action required by user: Notify the FDS system maintenance personnel.
9	*DBINT*	"ERROR IN OPENING NEW DATA BOX--ERROR CODE_____." Meaning: The DRDE file specified as the output could not be successfully opened. The return code from OPEN is contained in the message text. Severity: Execution of DBINT processor is terminated. Action required by user: Notify the FDS system maintenance personnel.
10	*DBINT*	"RANGE OF NEW DATA BOX OUTSIDE OLD RANGE." Meaning: The range specified for the new data box goes beyond the boundaries of the old data box. Severity: Execution of DBINT processor is terminated. Action required by user: Determine the correct range and reexecute the processor.
11	*DBINT*	"ERROR WHILE CLOSING NEW DATA BOX." Meaning: The DRDE file created as the new data box could not be sucessfully closed. Severity: Execution of DBINT is terminated. Action required by user: Notify the FDS system maintenance personnel.

TABLE 4-IV.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR DBINT

Processor name	Processor abstract prompt (maximum 256 characters)
DBINT	The data box interpolator provides the FDS user with the capability to perform a linear interpolation of a dependent variable over a specified range for the indenpedent variables X and Y.
Parameter keyword name	**Parameter definition prompt (maximum 256 characters)**
PROCON	PROCON is the cartridge number for the data box file (one integer value).
INDTBX	INDTBX is the DRDE file; contents are five header records and one summary table for each iteration of SCAN or DBINT
XCENTR	XCENTR is the X-centroid value (one real value).
YCENTR	YCENTR is the Y-centroid value (one real value).
XINCR	XINCR is the X increment used to obtain the next value of X to be used in the interpolation.
YINCR	YINCR is the Y increment used to obtain the next value of Y to be used in the interpolation.
XSTEPS	XSTEPS is the number of steps on either side of the centroid to be executed for X.
YSTEPS	YSTEPS is the number of steps on either side of the centroid to be executed for Y.
OTDTBX	OTDTBX is the output data box. Contents are five header records and one summary table for each iteration of DBINT.

5.0 PROCESSOR ROUTINES

The only available routine documentation is contained on the comment cards in the software listing, with the exception of the PDL provided in figure 5-1.

```
CD*****************************************************
CD0
CD0    DBINT - DATA BOX INTERPOLATOR PROCESSOR
CD0
CD0          - SCHEDULED BY FDS
CD0
CD*****************************************************
CD1
CD1    DBINT PROVIDES THE FDS USER WITH THE CAPABILITY TO PERFORM
CD1    A LINEAR INTERPOLATION OF A DEPENDENT VARIABLE VALUE
CD1    OVER A SPECIFIED RANGE FOR THE INDEPENDENT VARIABLES,
CD1    X AND Y.  THIS LINEAR INTERPOLATION WILL GENERATE A
CD1    NEW DATA BOX.
CD1
CD*****************************************************
CD2
CD2    INPUTS TO DBINT FROM THE INTERFACE TABLE
CD2
CD2      PROCON    - CARTRIDGE # FOR DATA BOX FILE
CD2      INDTBX    - INPUT DATA BOX NAME
CD2      XCENTR    - X CENTROID
CD2      XINCR     - X INCREMENT
CD2      XSTEPS    - # OF STEPS FOR X
CD2      YCENTR    - Y CENTROID
CD2      YINCR     - Y INCR
CD2      YSTEPS    - # OF STEPS FOR Y
CD2      OTDTBX    - OUTPUT DATA BOX NAME
CD2
CD2
CD2    INPUTS TO DBINT FROM THE DRDE FILE
CD2
CD2       RECORD 1
CD2         (1)  - NAME OF THE FDS PROCESSOR CREATING THE FILE
CD2         (4)  - INTERFACE TABLE VARIABLE NAME FOR THIS FILE
CD2         (7)  - NAME OF THE FDS PROCESSOR UPDATING THE FILE
CD2                   (3 ASCII WORDS OR BLANKS)
CD2         (10) - INTERFACE TABLE VARIABLE NAME FOR THIS UPDATE
CD2                   (3 ASCII WORDS OR BLANKS)
CD2
CD2
CD2       RECORD 2
CD2         (1)  - NUMBER OF ENTRIES IN THE SUMMARY TABLE
CD2         (2)  - X SCAN VARIABLE (6 CHARACTERS)
CD2         (5)  - X FIRST SUBSCRIPT (INTG OR ZERO IF NONE)
CD2         (6)  - X SECOND SUBSCRIPT (INTG OR ZERO IF NONE)
CD2         (7)  - X UNITS (6 CHARACTERS)
CD2         (10) - X CENTROID (REAL)
CD2         (12) - X INCREMENT (REAL)
CD2         (14) - X NUMBER OF STEPS (INTEGER 0-5)
CD2         (14) - Y SCAN VARIABLE (6 CHARACTERS)
CD2         (18) - Y FIRST SUBSCRIPT (INTG OR ZERO IF NONE)
CD2         (19) - Y SECOND SUBSCRIPT (INTG OR ZERO IF NONE)
CD2         (20) - Y UNITS (6 CHARACTERS)
CD2         (23) - Y CENTROID (REAL)
CD2         (25) - Y INCREMENT (REAL)
CD2         (27) - Y NUMBER OF STEPS (INTEGER 0-5)
CD2
CD2       SUMMARY TABLE RECORDS
CD2         - EACH SUMMARY TABLE CONTAINS VALUES FOR EACH DEPENDENT
CD2           VARIABLE SCANNED (UP TO 32 VALUES INCLUDING THE ERROR
CD2           FLAG WHICH IS THE FIRST VALUE IN THE SUMMARY TABLE)
CD2
CD2       DEPENDENT VARIABLE NAME AND UNITS ARE IN RECORDS 3,4, AND 5
```

Figure 5-1.- DBINT functional level PDL.

```
CD2      OF THE DRDE FILE
CD2
CD*************************************************************
CD3
CD3      OUTPUT FROM DBINT
CD3
CD3        - OUTPUT WILL BE A DATA BOX OF THE TYPE GENERATED BY
CD3          SCAN/ENDSC.
CD3
CD*************************************************************
CD4
CD4      ROUTINE VARIABLES
CD4
CD4      BUF-OUTPUT BUFFER
CD4      ELOC1--LOCATION OF SUMMARY VECTOR ERROR FLAG FOR POINT 1
CD4      ELOC2--LOCATION OF SUMMARY VECTOR ERROR FLAG FOR POINT 2
CD4      ELOC3--LOCATION OF SUMMARY VECTOR ERROR FLAG FOR POINT 3
CD4      ELOC4--LOCATION OF SUMMARY VECTOR ERROR FLAG FOR POINT 4
CD4      ERROR--ERROR FLAG FOR NEW DATA BOX
CD4      IDCB1--DCB FOR OLD DATA BOX
CD4      IDCB2--DCB FOR NEW DATA BOX
CD4      INDTBX--INPUT DATA BOX NAME
CD4      INTPT1--INTERPOLATED POINT 1
CD4      INTPT2--INTERPOLATED POINT 2
CD4      INPNTX--FLAG FOR MAX NUMBER OF X VALUES
CD4      INPNTY--FLAG FOR MAX NUMBER OF Y VALUES
CD4      MASTER--ARRAY WHICH CONTAINS OLD DATA BOX
CD4      NNXS--NUMBER OF NEW X STEPS
CD4      NNYS--NUMBER OF NEW Y STEPS
CD4      NOXS--NUMBER OF OLD X STEPS
CD4      NOYS--NUMBER OF OLD Y STEPS
CD4      NUMENT--NUMBER OF ENTRIES IN TABLE
CD4      NXCORD--NEW X VALUES
CD4      NYCORD--NEW Y VALUES
CD4      OTDTBX--NEW DATA BOX NAME
CD4      OXCENT OLD X CENTROID
CD4      OXCORD--OLD X VALUES
CD4      OXINCR--OLD X INCREMENT
CD4      OXSTP--NUMBER OF OLD X STEPS
CD4      OYCENT--OLD Y CENTROID
CD4      OYCORD--OLD Y VALUES
CD4      OYINCR--OLD Y INCREMENT
CD4      OYSTP--NUMBER OF OLD Y STEPS
CD4      PNT1-POINTER TO VALUE 1
CD4      PNT2--POINTER TO VALUE 2
CD4      PNT3--POINTER TO VALUE 3
CD4      PNT4--POINTER TO VALUE 4
CD4      PNTRX--X COUNTER
CD4      PNTRY--Y COUNTER
CD4      SLOPX--SLOPE FOR X
CD4      SLOPY--SLOPE FOR Y
CD4      XCENTR--NEW X CENTROID
CD4      XINCR--NEW X INCREMENT
CD4      XSTEPS--NUMBER OF NEW X STEPS
CD4      YCENTR--NEW Y CENTROID
CD4      YINCR--NEW Y INCREMENT
CD4      YSTEPS--NUMBER OF NEW Y STEPS
CD4
CD*************************************************************
CD5
CD5      EXTERNAL REFERENCES
CD5
CD*************************************************************
CD5      FDS SUBROUTINES USED:
```

Figure 5-1.- Continued.

```
BEGIN DBINT
   PERFORM INITIALIZATION
   UPDATE FIRST RECORD WITH DBINT PROCESSOR DATA
   FOR I = 2, 5
   DO
      READ RECORD # I
      ERROREXIT (ERR4) ERROR IN READING OLD DATA BOX
      CASE
         RECORD # I
         PART(2)
         COMPUTE VALUES FOR X & YS FOR OLD DATA BOX
         COMPUTE VALUES FOR X & YS FOR NEW DATA BOX
         ERROREXIT (ERR10)  IF NEW DATA BOX RANGE OUTSIDE OLD DATA BOX RANGE
         UPDATE SECOND RECORD WITH APPROPRIATE DATA ("XCENTR", "YCENTR")
         ("XINCR", "YINCR", "XSTEPS", "YSTEPS")
      ELSE
         KEEP RECORD AS IS
      ENDCASE
      CALL XPATR TO GET NAME FOR OUTPUT DATA BOX
      CALL CREATE TO CREATE NEW DATA BOX FILE
      ERROREXIT (ERR8) ERROR OCCURRED DURING THE CREATE FOR NEW DATA BOX
      WRITE RECORD # 1-5 TO THE NEW DATA BOX
      ERROREXIT (ERR6) ERROR IN WRITING NEW DATA BOX
   ENDDO
   FOR NUMBER OF SUMMARY ENTRIES IN OLD DATA BOX
   DO
      READ RECORD FROM DATA BOX AND SAVE FOR LATER USE
      ERROREXIT (ERR4) IF A READ ERROR OCCURRED
   ENDDO
   IF OLD Y VALUES EXIST
   THEN
      FOR ALL NEW Y VALUES
      DO
         WHILE OLD Y VALUE < NEW Y VALUE
         DO
            INCREMENT TO NEXT OLD Y VALUE
         ENDDO
         IF OLD Y VALUE = NEW Y VALUE
         THEN
            SET EQUAL FLAG AND SAVE POINTERS
            SET SLOPE FOR Y = Ø.Ø
         ELSE
            SAVE POINTERS WHICH BOUND NEW Y VALUE
```

Figure 5-1.- Continued.

```
            COMPUTE SLOPE FOR Y
         ENDIF
      ENDDO
      IF OLD X VALUES EXIST
      THEN
         FOR ALL NEW X VALUES
         DO
            WHILE OLD X VALUE < NEW X VALUE
            DO
               INCREMENT TO NEXT OLD X VALUE
            ENDDO
            IF OLD X VALUE = NEW X VALUE
            THEN
               SET EQUAL FLAG AND SAVE POINTERS
               SET SLOPE FOR X = Ø.Ø
            ELSE
               SAVE POINTERS WHICH BOUND NEW X VALUES
               COMPUTE SLOPE FOR X
            ENDIF
         ENDDO
         IF NEW X & Y EXIST
         THEN
            COMPUTE LOCATIONS OF ERROR SUMMARY FOR 4 POINTS (1, 2, 3, 4)
            PERFORM SUMMARYERROR
         ELSE
            IF ONLY X EXIST
            THEN
               COMPUTE LOCATIONS FOR ERROR SUMMARY FOR 2 X POINTS
               PERFORM SUMMARYERROR
            ELSE
               COMPUTE LOCATIONS FOR ERROR SUMMARY FOR 2 Y POINTS
               PERFORM SUMMARYERROR
            ENDIF
         ENDIF
         IF NO SUMMARY ERROR
         THEN
            FOR I=2, NUMBER OF ENTRIES IN SUMMARY TABLE
            DO
               IF X VALUES EXIST
               THEN
                  COMPUTE LOCATION OF DATA TO BE USED FOR THE INTERPOLATION
                  PERFORM INTERPOLATE  (FOR INTERPOINT1 FROM OLD 1 & 2)
               ELSE
                  INTERPOINT1 SET TO X1
               ENDIF
               IF X & Y VALUES EXIST
               THEN
                  COMPUTE LOCATION OF DATA TO BE USED FOR THE INTERPOLATION
                  PERFORM INTERPOLATE  (FOR INTERPOINT2 FROM 3 & 4)
               ELSE
                  INTERPOINT2 SET TO INTERPOINT1
               ENDIF
               IF Y VALUES EXIST
               THEN
                  PERFORM INTERPOLATE  (FOR INTERPOINTX FROM INTERPOINT1 & 2)
               ELSE
                  INTERPOINTX SET TO INTERPOINT1
               ENDIF
               WRITE ONE SUMMARY VECTOR  (1 RECORD OF 64 WORDS)
               ERROREXIT (ERR6) IF WRITE ERROR OCCURRED
            ENDDO
         ENDIF
      ENDDO
   ENDDO
```

Figure 5-1.- Continued.

```
   SET RETURN PARAMETERS FOR NORMAL EXIT
   CALL CLOSE FOR OLD DATA BOX
   CALL PURGE FOR NEW DATA BOX NAME
   CALL RENAME  TO RENAME TEMP DATA BOX
   CALL XPPUT FOR NEW DATA BOX
   CALL XPXIT FOR NORMAL RETURN
EXIT DBINT
*****************************************************************

*************************************************
   ********************************************************************
   BEGINSEGMENT INITIALIZATION
      CALL RMPAR TO RECEIVE LU AND DEBUG FLAGS
      CALL XPGET TO RETRIEVE INTERFACE TABLE DATA
      OPEN DRDE FILE FOR OLD DATA BOX
      ERROREXIT (ERR1) IF DATA BOX COULD NOT BE FOUND
      ERROREXIT (ERR2) IF DATA BOX COULD NOT BE OPENED
      ERROREXIT (ERR3) NUMBER OF "XSTEPS" OR "YSTEPS" <Ø OR >5
   ENDSEGMENT INITIALIZATION
   ********************************************************************

   *************************************************
   ********************************************************************
   BEGINSEGMENT INTERPOLATE
       [INTERPOLATE NEW VALUES FROM OLD VALUES                          ]
      COMPUTE SLOPE FOR THIS ITERATION
      INTERPOLATE TO THE NEW VALUE
   ENDSEGMENT INTERPOLATE
   ********************************************************************

   *************************************************
   ********************************************************************
   BEGINSEGMENT SUMMARYERROR
      IF SUMMARY VECTOR ERROR FLAG SET IN OLD DATA BOX
      THEN
         SET NEW DATA BOX SUMMARY VECTOR ERROR FLAG
         IF FIRST SUMMARY VECTOR ERROR
         THEN
            CALL EXEC ' WARNING   WARNING   WARNING'
            CALL EXEC 'DATA BOX CREATED WITH AN INVALID SUMMARY VECTOR'
         ENDIF
      ENDIF
   ENDSEGMENT SUMMARYERROR
   ********************************************************************

   *************************************************
   ********************************************************************
   BEGINERRORSEGMENT
      CASE
         TYPE OF ERROR
         PART(ERR1)
         CALL EXEC -'DATA BOX COULD NOT BE FOUND'
         PART(ERR2)
         CALL EXEC -'ERROR IN OPENING OLD DATA BOX---ERROR CODE X'
         PART(ERR3)
         CALL EXEC -'NUMBEROF STEPS MUST BE BETWEEN Ø & 5, INCLUSIVE'
         PART(ERR4)
         CALL EXEC -'ERROR IN READING DATA BOX---ERROR CODE X'
         PART(ERR5)
         CALL EXEC -'INPUT DATA BOX CONTAINS AN INVALID SUMMARY VECTOR'
         PART(ERR6)
         CALL EXEC -'ERROR IN WRITING DATA BOX---ERROR CODE X'
         PART(ERR7)
         CALL EXEC -'ERROR IN THE PURGE OF THE NEW DATA BOX---ERROR CODE X'
```

Figure 5-1.- Continued.

```
        PART(ERR8)
        CALL EXEC -'ERROR IN CREATING NEW DATA BOX---ERROR CODE X'
        PART(ERR9)
        CALL EXEC -'ERROR IN OPENING NEW DATA BOX---ERROR CODE X'
        PART(ERR10)
        CALL EXEC ='RANGE OF NEW DATA BOX OUTSIDE OLD RANGE'
      ENDCASE
      CALL CLOSE TO CLOSE OLD DATA BOX
      CALL CLOSE TO CLOSE NEW DATA BOX
      CALL PURGE (DELETE NEW DATA BOX IF THERE WAS AN ERROR                    ]
      SET RETURN PARAMETERS FOR ABNORMAL EXIT
      CALL XPXIT TO TERMINATE DBINT ABNORMALLY
   ENDERRORSEGMENT
   ************************************************************************

   ******************************************************
END DBINT
```

Figure 5-1.- Concluded.

DATA ELEMENT DEFINITION (DEFIN)

1.0 PURPOSE

The DEFIN utility processor provides the FDS user with the capability to define, allocate storage for, and initialize one or more data elements in the active work area (AWA).

2.0 FUNCTIONAL DESCRIPTION

The DEFIN processor interprets the input data supplied via a symbolic string entry for the interface table parameter keyword DEFINE. It isolates the information that defines each data element (name, optional dimensions, and type) and ensures the correct syntax. (See table 4-I for detailed explanation of correct syntax.)

If the syntax is valid, DEFIN deletes any existing data element of the same name, allocates a new data element and initializes it to zeros (blanks for character data). If the syntax is found to be in error, the data element name and the error are displayed and processing continues with the next data element. If there is insufficient space in the AWA for a data element, the user is notified, execution of the sequence table is terminated, and control is returned to the FDS executive.

3.0 ASSUMPTIONS AND LIMITATIONS

Listed below are the assumptions and limitations associated with the DEFIN processor.

a. The DEFIN processor can only allocate memory resident (AWA) data elements of types free, integer, real, double, and character strings.

b. Where symbolic strings are used as input data, only limited checking can be done by the interface table editor to ensure that the data and punctuation are correct. Therefore, it is possible that a syntax error will not be discovered until processor execution time. An error thus found will result in a message being issued, execution of the sequence table terminated and control returned to the executive. Data correction is through the interface table editor.

c. The allowable length of the symbolic string is determined by an internal communication buffer of 247 words. The buffer contains token codes and data in internal representations. For DEFIN, the guaranteed minimum number of data element definitions is 13; the maximum is a function of the number of definitions using optional fields and may be up to 24 definitions.

d. If a specific data element already exists in the AWA, it will be deleted and a new data element allocated with the specified dimensions and type.

4.0 PROCESSOR INPUT/OUTPUT

The inputs required and outputs for the DEFIN processor are listed as follows:

a. Processor interface table - The interface table defined in table 4-I contains the symbolic string DEFINE. The format of this string is:

'data element name [(idim [,jdim])]/type/[,...]'

where:

(1) Data element name is an alphanumeric name of up to six characters for the data element to be defined.

(2) [(idim [,jdim])] is an optional dimension in logical entries for an array being defined. If entered, idim and jdim must be positive integers greater than zero; default values are (1,1).

(3) /type/ is a required type field.

Valid types are:

F - free or typeless

I - integer

R - single precision real

D - double precision real

Cn - character string of n characters, where n = 2, 6, 18, 36, or 72

b. Interface table data array definitions - None.

c. Interface table data file definitions - None.

d. Processor solicited (prompted) inputs - None.

e. Processor displays and display parameter definition table - None.

f. Processor messages - The DEFIN processor error messages are provided in table 4-II.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-III.

h. Memory resident (AWA) data elements - DEFIN outputs the specified data elements to the AWA and initializes them to zeros (or blanks for character strings).

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR DEFIN

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
DEFINE	AWA	SS	I	VAR	247		Symbolic string of data element definitions of the format: 'data element name [(idim[,jdim])], /type/[,...] where, data element name is an alphanumeric name of up to 6 characters for the data element to be defined. [(idim[,jdim])] is an optional dimension for an array being defined in logical entities. If entered, IDIM and JDIM must be positive integers greater than zero; the default values are (1,1) /type/ is a required type field. Valid types are: F - free I - integer R - single precision real D - double precision real Cn - character string of N characters, where N = 2, 6, 18, 36, or 72

NOTES	CLASS	TYPE			USE
	Symb	Free	2CH	72CH	I = Input
	AWA	Intg	6CH	Mix	O = Output
	Disk	Real	18CH		I/O = Input/Output
		Dubl	36CH		

TABLE 4-II.- PROCESSOR MESSAGE TABLE

PROCESSOR DEFIN

MSG no.	Message ID block	Message text block and explanation
1	*DF01*	"DATA ELEMENT CCCCCC SIZE EXCEEDS 1200 WORDS." Meaning: Data element size (IDIM * JDIM * size of type) exceeds maximum of 1200 words. Severity: Minor; processing of this element is terminated, but continues to next element (if any). Action required by user: Adjust dimensions of data element to be within the limit using the internal table editor.
2	*DF02*	"SERIOUS SYNTAX ERROR FOUND IN THE # NN ELEMENT - SCAN TERMINATED." Meaning: There is an unrecognizable data element name in DEFINE symbolic string at element # NN. Severity; Fatal, DEFINE is aborted; no elements are allocated. Action required by user: Use interface table editor to enter only alphanumeric names of up to 6 characters as a data element name.
3	*DF03*	"WARNING ... DATA ELEMENT CCCCCC NOT TERMINATED BY A /." Meaning: There was no end of type field slash for data element CCCCCC. Severity: Warning message only. Action required by user: None required.
4	*DF04*	"DATA ELEMENT CCCCCC HAS INVALID IDIM." Meaning: IDIM field is not an integer value > 0. Severity: Minor; processing of this element is terminated but continues to next element (if any). Action required by user: Use interface table editor to enter only valid IDIM fields.
5	*DF05*	"DATA ELEMENT CCCCCC HAS INVALID JDIM." Meaning: JDIM field is not an integer value > 0. Severity: Minor; processing of this element is terminated but continues to next element (if any). Action required by user: Use interface table editor to enter only valid JDIM fields.
6	*DF06*	"DATA ELEMENT CCCCCC DIMENSIONS ARE NOT DELIMITED BY PARENTHESIS." Meaning: There is an unrecognizable field where the dimension(s) should be. Severity: Minor; processing of this element is terminated but continues to next element (if any). Action required by user: Ensure the correct format of each data element, with parenthesis enclosing dimensions.

TABLE 4-II.- Concluded

PROCESSOR DEFIN

MSG no.	Message ID block	Message text block and explanation
7	#DF07#	"DATA ELEMENT CCCCCC HAS A MISSING OR INVALID TYPE FIELD." Meaning: Either the type field was not included or the type specified was not F, I, B, D, or Cn. Severity: Minor, processing of this element is terminated but continues to the next element (if any). Action required by user: Use interface table editor to give data element CCCCCC a valid type field.

TABLE 4-III.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR DEFIN

Processor name	Processor abstract prompt (maximum 256 characters)
DEFIN	Utility processor for defining AWA data elements. The size and type of the data elements to be allocated may be specified.

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
DEFINE	Symbolic string specifying the name and attributes of data elements to be allocated. The list is of the form 'DATA ELEMENT NAME [(I-DIMENSION[,J-DIMENSION])] /TYPE/, ...' where type is I, R, D, C2, C6, C18, C36, C72 or F.

5.0 PROCESSOR ROUTINES

The only available routine documentation is contained on the comment cards in the software listing. Additional material may be found in JSC IN 77-FM-18, volume IV, revision 2 dated April 1978, and in JSC IN 77-FM-18, volume IX (to be published).

SEQUENCE ITERATION PROCESSORS (DO/ENDDO)

1.0 PURPOSE

The DO/ENDDO utility processors, used as a pair of sequence table entries, provide the FDS user with the capability to perform conditional looping within a sequence table.

2.0 FUNCTIONAL DESCRIPTION

The processors DO/ENDDO create and control a loop over a series of processors as defined by a set of sequence table entries. The loop begins with the sequence entry immediately subsequent to the DO and continues to the sequence entry of the ENDDO.

DO interface table parameter keywords provide the identification of the loop conditions. This information includes the type of loop (UNTIL or WHILE) and two single precision real data element components to be compared for a specified relationship. Control information is saved in a manager data array named &DOSTK while the loop is active.

Depending on the loop type, WHILE/UNTIL, the control conditions are evaluated at the beginning/end of each loop to determine if termination conditions have been satisfied. This information is retrieved from the current &DOSTK entry such that loops may be executed in a nested fashion; i.e., loops may appear within loops.

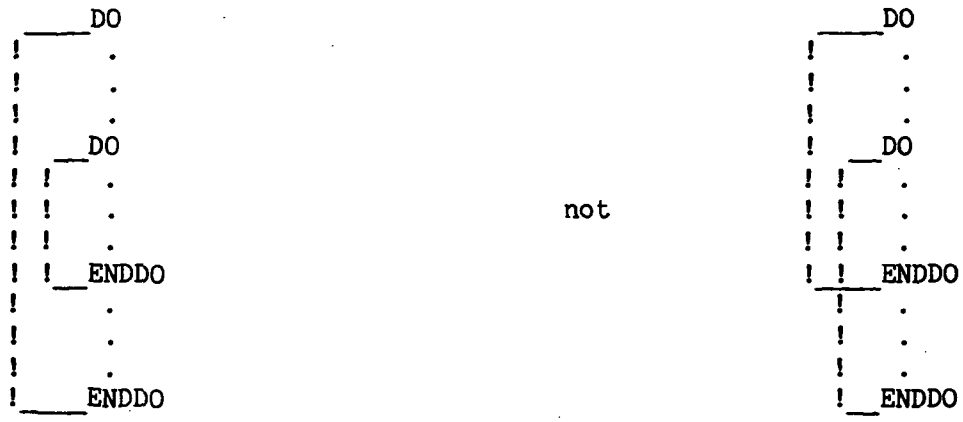
3.0 ASSUMPTIONS AND LIMITATIONS

Listed below are the assumptions and limitations associated with the DO/ENDDO processors.

a. DO and ENDDO must be used as a pair to ensure correct results for a loop execution. Use of DO with no ENDDO will result in an error message and execution termination for the WHILE option and a single execution through the remainder of the sequence table for the UNTIL option. Use of only ENDDO results in an error message and execution termination.

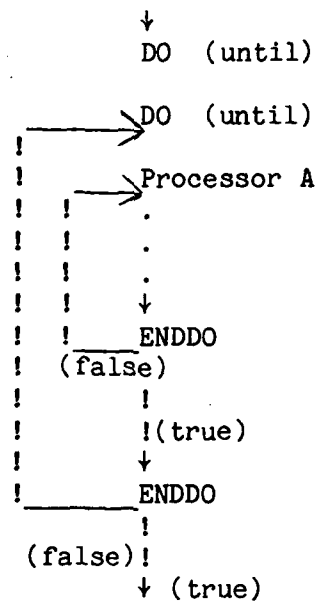
b. There is no interface table for ENDDO. DO builds &DOSTK control block from which ENDDO receives all inputs.

c. Loops are executed in a nested fashion; i.e., the first ENDDO encountered will be matched with the last previous DO encountered. For example:



The maximum number of nested DOs is 4.

Note: (1) When a DO is followed by another DO in a predefined sequence table, the first DO is executed with the second DO as the top of the loop. The second DO is executed with the subsequent processor as the top of the inner loop. In this manner, the execution of the two loops is nested. For example,



(2) A few special considerations are required when executing DO in the semiautomatic mode. These conditions do not apply to an automatic execution.

When any processor except DO is overridden by the user with a DO, the top of the loop will be the original sequence entry; i.e., the DO will be an insert prior to the original entry (see (a) as follows). However, if DO is overridden with a DO, the override will be executed with the subsequent entry, not the original DO, as the top of the loop (see (b) as follows). In this case, if the user concurs with

the reprompt for execution of the original DO, two loops will be formed with the same table entry as the top of the loop; therefore, if a duplicate ENDDO is also inserted, the net effect is that the two loops will not be nested but will execute sequentially through the same range (see (c) as follows).

```
$ 100 = processor A:Δ                    100 = processor A

$ 200 = processor B:DOΔ                  DO(temporary insert)

$ = DO:Δ                    results in   200 = processor B
                            the sequence!
$ 200 = processor B:Δ                    !  300 = processor C
                                         !
$ 300 = processor C:Δ                    !_ 700 = ENDDO
      .
      .
      .
$ 700 = ENDDO:Δ
```

(a) Inserting a DO

```
$ 100 = processor A:Δ                    100 = processor A

$ 200 = DO, IT1:DO,IT2Δ                  200 = DO,IT2(override)

$ = DO,IT2:&Δ               results in   300 = processor C
                            the sequence!
$ 200 = DO,IT1:Δ                         !      .
                                         !      .
$ 300 = processor C                      !      .
                                         !
      .                                  !_ 700 = ENDDO
      .
      .
$ 700 = ENDDO:Δ
```

(b) Overriding a DO

```
$ 100 = processor A:Δ                    100 = processor A

$ 200 = DO,IT1:DO,IT2Δ                   200 = DO,IT1 (original DO)

$ = DO,IT2:Δ                             300 = processor C
                                         !
$ 200 = DO,IT1:Δ                         !      .
                            results in   !      .
$ 300 = processor C:Δ       the sequence!      .
      .                                  !
      .                                  !_ 700 = ENDDO (original
      .                                                  ENDDO)
                                           200 = DO,IT2 (insert DO)
```

```
$ 700 = ENDDO;ENDDOΔ

$ = ENDDO:Δ

$ 700 = ENDDO:Δ

$ 800 = processor D:Δ
```

```
   ┌→300 = processor C
   !     .
   !     .
   !     .
   !
   !__700 = ENDDO (insert ENDDO)

     800 = processor D
```

(c) Inserting a loop for sequential execution

(3) The operands used to control an UNTIL type loop need not be defined in the AWA until just prior to execution of the ENDDO processor. Other input parameters must exist when the DO is first executed and will not be retrieved once inside the loop; i.e., changes to DOTYPE and RELATN, made after loop execution begins, have no effect on loop control.

(4) Nesting errors involving DO, SCAN and IF are not explicitly caught, but may be detected due to unmatched DOs, ENDDOs, IFs, or ELSEs or by exceeding the limits on the number of SCAN or DO loops activated. For example:

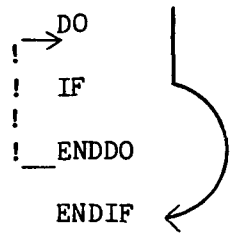
```
  ┌→DO
  !
  ! ┌→SCAN
  !!
  !!__ENDDO
   !
   !__ENDSC
```

will terminate for exceeding the scan nesting limit of four or by executing the ENDDO after the loop has been satisfactorily completed.

```
    IF
  ┌→DO
  !
  !  ELSE
  !
  !__ENDDO

    ENDIF
```

will terminate for never finding the matching ENDDO or for executing the ENDDO with no matching DO.



If the IF is never false, the loop and other processors will execute normally. However, if the IF condition is false, the matching ENDDO will not be found and termination will produce a diagnostic message.

(5) Execution of DO or ENDDO in the MANUAL mode is not meaningful and results in a diagnostic message and no execution.

(6) Comparison of the two real data element components using the equality (=) or inequality (#) relationships may not always be meaningful. Equality of two real values requires that they be identical in all 32 bits. This is seldom the case if either operand results from any form of arithmetical computation.

4.0 PROCESSOR INPUT/OUTPUT

Listed below are the inputs required and the outputs for the DO/ENDDO processors.

a. Processor interface table - The interface table for DO is defined in table 4-I and contains:

- Type of loop control
- Two single precision real values to be compared
- A relationship to be applied to the two values

b. Interface table data array definitions - None.

c. Interface table data file definitions - None.

d. Processor solicited (prompted) inputs - None.

e. Processor displays and display parameter definition table - None.

f. Processor message table - The DO/ENDDO error messages are provided in table 4-II.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-III.

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR DO/ENDDO

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
DOTYPE	AWA	6CH	I	3	1	--	Type of loop to be executed; i.e., "UNTIL" or "WHILE"
OPRND1	AWA	Real	I	2	1	--	First quantity to be compared for loop control
RELATN	AWA	2CH	I	1	1	--	Relational symbol for control condition from the set "#" not equal "<" less than "<=" less than or equal "=<" less than or equal "=" equal ">=" greater than or equal "=>" greater than or equal ">" greater than
OPRND2	AWA	Real	I	2	1	--	Second quantity to be compared for loop control

NOTES

CLASS
AWA
Disk

TYPE
Free 2CH 72CH
Intg 6CH Mix
Real 18CH Symb
Dubl 36CH

USE
I = Input
O = Output
I/O = Input/Output

TABLE 4-II.- PROCESSOR MESSAGE TABLE

PROCESSOR DO/ENDDO

MSG no.	Message ID block	Message text block and explanation
1	*DO01*	"MAXIMUM NUMBER OF NESTED LOOPS IS 4." Meaning: The sequence table contains more than four concurrently executing loops. Severity: Fatal; sequence execution is aborted. Action required by user: Use the sequence table editor to restructure the logic to use fewer nested loops.
2	*DO02*	"DO CANNOT BE EXECUTED IN THE MANUAL MODE." Meaning: Since it is only meaningful to loop on a sequence of processors, DO may only be used in SEMIAUTOMATIC or AUTOMATIC modes of execution. Severity: Fatal; processor execution is aborted. Action required by user: Use the sequence table editor to build the desired processor sequence and execute the sequence.
3	*DO03*	"ENDDO CANNOT BE EXECUTED WITHOUT A MATCHING DO." Meaning: Execution of ENDDO was attempted without previously executing a corresponding DO. Severity: Fatal; sequence execution is aborted. Action required by user: Examine the sequence table for logic errors such as overlapping SCAN and DO ranges, DO executions that were skipped due to IF conditions or omissions of the DO processor.
4	*DO04*	"DO CANNOT BE EXECUTED WITHOUT A MATCHING ENDDO." Meaning: Execution of DO was attempted without subsequent appearance of a matching ENDDO in the sequence table. This condition is detected for WHILE options of the DO. Severity: Fatal; sequence execution is aborted. Action required by user: Use the sequence table editor to correct the sequence and supply the missing ENDDO.
5	*DO05*	"INVALID RELATION ... USED IN CONDITION FOR DO." Meaning: The value of the parameter RELATN was not one from the supported set (see table 4-I). Severity: Fatal; sequence execution is aborted. Action required by user: Correct the value of RELATN to be one of the supported conditions.

TABLE 4-II.- Concluded

PROCESSOR DO/ENDDO

MSG no.	Message ID block	Message text block and explanation
6	*DO06*	"NO AWA SPACE TO STORE THIS DO'S CONTROL TABLE." Meaning: There is not enough space in the AWA for &DOSTK. Severity: Fatal; sequence execution is aborted. Action required by user: Clear enough room in the AWA for 27 words for each nested DO.
7	*DO07*	"INVALID DOTYPE ... USED FOR DO. MUST BE UNTIL OR WHILE." Meaning: The value of the parameter DOTYPE was not "UNTIL" or "WHILE". Severity: Fatal; sequence execution is aborted. Action required by user: Correct the value of DOTYPE.

TABLE 4-III.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR DO

Processor name	Processor abstract prompt (maximum 256 characters)
DO	DO, when paired with ENDDO, defines a series of sequence table entries to be repeatedly executed. The loop may execute either while a condition is TRUE or until a condition becomes TRUE.

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
DOTYPE	A character string indicating whether the loop is to be executed "UNTIL" a condition is met or "WHILE" a condition is TRUE.
OPRND1	A single real value to be compared under the specified relation to OPRND2 for loop control.
RELATN	A character string specifying the type of relationship, which, if TRUE, will meet the loop control condition. Valid relations are #,<,<=,=<,=,>=,=>, and >.
OPRND2	A single real value to be compared under the specified relation to OPRND1 for loop control.

TABLE 4-III.- Concluded

PROCESSOR ENDDO

Processor name	Processor abstract prompt (maximum 256 characters)
ENDDO	The loop end designator for the DO processor. All sequence table entries between the corresponding DO and ENDDO will be cyclically executed as defined by the DO conditions.

Parameter keyword name	Parameter definition prompt (maximum 256 characters)

5.0 PROCESSOR ROUTINES

The only available routine documentation is contained on the comment cards in the software listing.

Additional material may be found in JSC IN 77-FM-18, volume IV, revision 2 dated April 1978.

DOCUMENT PROCESSOR (DOC)

All documentation for the document processor (DOC) is presented in JSC IN 77-FM-18, volume V, revision 1 dated June 1979.

DEORBIT TARGET MODULE PROCESSOR (DTM)

1.0 PURPOSE

The DTM processor computes a deorbit maneuver that provides entry conditions to ensure a compatible velocity, flightpath angle, and range at entry interface to satisfy temperature constraints during entry.

2.0 FUNCTIONAL DESCRIPTION

The DTM computes the guidance targets for the deorbit maneuver that will provide the best entry conditions within the input constraints. The basis of the DTM targeting is to ensure the attainment of a correlated velocity, flightpath angle, and range target at EI. This is done by propagating the state to an estimated Tig (actual Tig in the Tig-fixed mode) with a precision predictor, generating an analytic PEG-4 maneuver solution to first-guess targets that output burnout and EI conditions, and then calling the entry target generator (ETG) target line to update the EI targets. PEG-4 performs guidance computations necessary to ensure achievement of the desired horizontal and vertical velocity components. It determines the velocity to be gained, time to go until thrust termination, and the resultant thrust direction vector. The ETG target line is given an EI velocity magnitude output from PEG-4 and returns the desired flightpath angle and range. The velocity and flightpath angle are converted to a new velocity-flightpath angle line constraint for PEG-4 and the range target is converted to a new PEG-4 position target by appropriately adjusting Tig or the Tig-to-EI transfer angle θ_{EI}. The above logic is then recycled with the new targets.

The targeting options available in the DTM are listed in the matrix below. In each case, any of the lower level options are available for any of the next higher level options, except as noted. All references here and later to Tig-fixed and Tig-free refer to the deorbit maneuver Tig.

OPTIONS

First level Tig mode	Second level propellant usage	Third level thrust system selection
Tig-free	Propellant wasting	Solution ΔV biased to a minimum value compatible with the designated prime and backup thrusters
Tig-fixed	Inplane	Solution for any designated thruster when prime and backup thrusters are the same

The first option to be set is whether Tig is free or fixed. If Tig is free, a solution will be found within one revolution beginning at the input threshold time.

Next, the propellant usage option must be designated. Fuel wasting is considered to be nominal solution. This option generates an out-of-plane deorbit maneuver so that a specified amount of OMS propellant will be expended to attain an acceptable center of gravity at EI. In the Tig-free mode, this option computes a solution so that the out-of-plane thrust angle is the same for the primary and backup propulsion systems.

The inplane solution option computes a delta-V solution to a nominal EI target that lies in the Tig trajectory plane. In the Tig-free mode, this option computes a solution so that the amount of OMS propellant expended is the same for both the backup and primary systems; thus, ensuring adequate propellant in case of a primary system failure.

Finally, for each propellant usage mode, a thruster designation must be made. A biased solution may be found for any allowed combination of primary and backup thrusters or a solution may be found for any single designated thruster system. The currently defined primary/backup thruster options are:

2 OMS

1 OMS

RCS

After a solution has been found, offset targets are computed that will account for the effects of higher order gravity during the coast from engine cutoff to EI. These offset targets will ensure that the required velocity, flightpath angle, and range set defined by the ETG will be met at EI.

3.0 ASSUMPTIONS AND LIMITATIONS

a. Tig time must be after the input vector time and is always input in hours, minutes, seconds GET.

b. Tthrsh must be after the input vector time and within one revolution of the desired Tig. It is always input in hours, minutes, seconds GET.

c. Tffmin should be less than 5400 seconds.

d. Rngip should be from 0.0 to 25 000. miles.

e. The prime and backup system delta-V versus Tig lines must cross for a fuel wasting solution to be found using equal yaw angles and equal delta-V. Thus the RCS is not a valid backup for the Shuttle in the fuel wasting mode. However, the RCS may be used for primary system or in-plane solutions.

4.0 PROCESSOR INPUT/OUTPUT

a. Processor interface table - The definition of the DTM interface table parameters is provided in table 4-I. All default value entries are in the FDS-1 internal units set.

b. Interface table data array definitions - The definition of the data arrays named within the interface table is provided in table 4-II. The DTMCON array units are in the internal standard set.

c. Interface table data file definition - None.

d. Processor solicited (prompted) inputs - None.

e. Processor displays and display parameter definition table - The DTM display format is shown in table 4-III and the definition of all display variables in table 4-IV. The display lower portion is printed only when the full print option flag IPOUT is set equal to "FLAGS" in the interface table. The lower portion consists of option flags useful only occasionally to the user.

f. Processor message table - The DTM processor messages are shown in table 4-V for error conditions resulting in processor termination and for nonterminal conditions resulting in warning messages.

g. Interface table extended prompts - The processor extended prompts for each interface table parameter keyword are provided in table 4-VI.

TABLE 4-I.- PROCESSOR INTERFACE TABLE

PROCESSOR DTM

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
ITIGFR	AWA	6CH	I	3			Deorbit ignition mode "FIXED" or "FREE"
TIG	AWA	Real	I	6	3		Deorbit ignition time in hours, minutes, seconds for ITIGFR = "FIXED"
TTHRSH	AWA	Real	I	6	3		Deorbit threshold time in hours, minutes, seconds for ITIGFR = "FREE"
IFUEL	AWA	2CH	I	1			Propellant use mode "FW" = Fuel wasting "IP" = Inplane
WCGOMS	AWA	Real	I	2			Weight of OMS to be burned
INGPR	AWA	6CH	I	3			Primary engine configuration "1OMS" - One OMS thrusting "2OMS" - Two OMS thrusting "RCS" - RCS thrusting
INGBU	AWA	6CH	I	3			Backup engine configuration same definitions as INGPR above
C1IP	AWA	Real	I	2			Ordinate intercept on the $V_{vertical}$ - $V_{horizontal}$ target line

NOTES	CLASS	TYPE		USE
		Free	2CH 72CH	I = Input
	AWA	Intg	6CH Mix	O = Output
	Disk	Real	18CH Symb	I/O = Input/Output
		Dubl	36CH	

TABLE 4-I.- Continued

PROCESSOR DTM

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
C2IP	AWA	Real	I	2			Slope of the V_v - V_h target line
TFFMIN	AWA	Real	I	2			Minimum allowed free fall time
RNGIP	AWA	Real	I	2			Range target from EI to landing site
KDGUID	AWA	6CH	I	3			PEG guidance option "PEG4" = V_v - V_h "PEG7" = External V
DTMVEC	AWA	Real	I	30	15		State vector
INTEGF	AWA	6CH	I	3		AEG	Propagation flag - "AEG"
IPOUT	AWA	6CH	I	3		FINAL	Print options "FINAL" = Final print only "FLAGS" = Final print with flag settings "DEBUG" = Interim printing with flags
IOUNIT	AWA	Intg	I	1		0	Output unit for print not sent to user terminal
TLATD	AWA	Real	I	2		.6091555	Landing site geodetic latitude, rad

NOTES

CLASS: AWA, Disk

TYPE: Free 2CH 72CH; Intg 6CH Mix; Real 18CH Symb; Dubl 36CH

USE: I = Input; O = Output; I/O = Input/Output

TABLE 4-I.- Continued

PROCESSOR DTM

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
TLONG	AWA	Real	I	2		-2.056745	Landing site geodetic longitude, rad
TALTD	AWA	Real	I	2		2220.66	Landing site geodetic altitude, ft
RA2	AWA	Real	I	2		3.317173	Landing site azimuth, rad
FOMS	AWA	Real	I	2		6000.	OMS one engine thrust, lb
FRCS	AWA	Real	I	2		3614.74	RCS four engine thrust, lb
XMDOMS	AWA	Real	I	2		19.157088	OMS one engine flow rate, lb/sec
XMDRCS	AWA	Real	I	2		13.623	RCS four engine flow rate, lb/sec
SESCON	AWA	Free	I	90	90	!SESCN	Session constants array
DTMCON	AWA	Real	I	82	41		Processor constants array
FVECON	AWA	Real	I	30	15		Entry target line array for FVE
GLOCON	AWA	Free	I	180	180	!!GLCN	Global constants array
SUMTAB	AWA	Free	O	328	8,41		Processor summary table

NOTES	CLASS	TYPE			USE
		Free	2CH	72CH	I = Input
	AWA	Intg	6CH	Mix	O = Output
	Disk	Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-I.- Concluded

PROCESSOR DTM

Parameter keyword name	Class	Type	Use	Size	Array dimension (I,J)	Values stored in default interface table	Definition
PVTABP	AWA	Real	O	240	30,4		Primary solution phase table
PVTABB	AWA	Real	O	240	30,4		Backup solution phase table

NOTES	CLASS	TYPE			USE
		Free	2CH	72CH	I = Input
	AWA	Intg	6CH	Mix	O = Output
	Disk	Real	18CH	Symb	I/O = Input/Output
		Dubl	36CH		

TABLE 4-II.- INTERFACE TABLE DATA ARRAY DEFINITIONS

PROCESSOR DTM

Array name	Index location	Default value	Definition
GLOCON	(1) . . . (180)	!!GLCN	Global constants array defined in vol. I, table 7.2-III
SESCON	(1) . . . (90)	!SESCN	Session constants array defined in vol. I, table 7.2-II
DTMVEC	(1) . . . (15)		Input state vector - contents defined in vol. I, figure 7.3-2.
FVECON	(1) . . . (15)		Velocity, flightpath angle, and range target line used by the FVE processor
	(1)	25600.	VEL(1)
	(2)	25700.	VEL(2)
	(3)	25727.87	VEL(3) — Velocity, fps
	(4)	25800.	VEL(4)
	(5)	25900.	VEL(5)
	(6)	-.01782556	FPA(1)
	(7)	-.02034792	FPA(2)
	(8)	-.02096274	FPA(3) — Flightpath angle, rad
	(9)	-.02253098	FPA(4)
	(10)	-.02447867	FPA(5)
	(11)	24926650.	RNGG(1)
	(12)	24847060.	RNGG(2)
	(13)	24864070.	RNGG(3) — Range, ft
	(14)	24873190.	RNGG(4)
	(15)	24984800.	RNGG(5)

TABLE 4-II.- Continued

PROCESSOR DTM

Array name	Index location	Default value	Definition
DTMCON	(1)	31993.	DBARR - Distance from runway to TAEM alinement circle
	(2)	20000.	RTURNN - Alinement circle radius
	(3)	10.	DC10 - Initial C1 increment
	(4)	50.	DC1MAX - Maximum allowed C1 increment
	(5)	.00029147	RNGTOL - Range convergence tolerance
	(6)	10.	TFFTOL - Tolerance used for increasing free fall time
	(7)	100.	DTIGO - Initial value of TIG adjustment
	(8)	200.	DTGMAX - Maximum value of TIG adjustment
	(9)	.34906	DS4S5 - GOLPAR routine constant
	(10)	25.	DWTOL - Weight tolerance on equal OMS check
	(11)	.01745329	DSITOL - Tolerance on equal yaw angle check
	(12)	.0001745329	GAMTOL - Tolerance on gamma convergence
	(13)	.01745329	SSS(1)
	(14)	.01745329	SSS(2)
	(15)	5000.	SSS(3) GOLPAR constants array
	(16)	0.	SSS(4)
	(17)	6.283185	SSS(5)
	(18)	10.	NMAX1 - Maximum iterations during initialization
	(19)	20.	NSEG - Number of steps used in gravity prediction
	(20)	.75	RHOI - Factor for computing the damping factor applied to V_{miss} for correcting V_{go}.
	(21)	.001	SMISSI - PEG convergence factor
	(22)	6.	TGOMIN - Guidance delta time before thrust termination
	(23)	0.	XLD - Circular orbit rate multiplier for PEG
	(24)	25.	DTMAXX - Maximum time step for gravity prediction
	(25)	2.	DTMINN - Minimum time step for gravity prediction
	(26)	.381966	GGOL - GOLPAR constant
	(27)	192.38	WBIAS - OMS propellant tolerance
	(28)	1.27409	THEIIP - Initial theta EI for TIG fixed mode
	(29)	19.25	CDENMN - Minimum denominator in C1 incrementation
	(30)	.0001	CRTOL - Crossrange computation tolerance
	(31)	.0001	TDENMN - Minimum denominator in crossrange logic
	(32)	10.0	DTCRO - Initial delta time in crossrange logic
	(33)	1.0	TGDOMS - Minimum denominator in equal OMS logic
	(34)	.001745329	TGDSI - Minimum denominator in equal out-of-plane angle logic
	(35)	60.0	TTHRB - Bias time for minimum delta-V TIG free solution
	(36)	10.0	TGSLPV - Initial slope for equal OMS prediction
	(37)	572.96	TGSLPY - Initial slope for equal yaw angle logic
	(38)	20.0	NCMAX - DTM iteration limit
	(39)	0.	POLE(1)
	(40)	0.	POLE(2) Unit vector in direction of Earth axis of rotation
	(41)	1.	POLE(3)

TABLE 4-II.- Continued

PROCESSOR DTM

Array name	Index location	Default value	Definition
SUMTAB	(1,1)	0	"ERROR0" 000000 Error condition flag for summary table
	(1,2)		"TIG" SESCON(17) Ignition time
	(1,3)		"TBOPR" SESCON(17) Burnout time - primary system
	(1,4)		"TBOBU" SESCON(17) Burnout time - backup system
	(1,5)		"TEIPR" SESCON(17) Time at entry interface - primary
	(1,6)		"TEIBU" SESCON(17) Time at entry interface - backup
	(1,7)		"DVPR" SESCON(19) Delta-V - primary
	(1,8)		"DVBU" SESCON(19) Delta-V - backup
	(1,9)		"DVOBPR" SESCON(19) Delta-V available - primary
	(1,10)		"DVOBBU" SESCON(19) Delta-V available - backup
	(1,11)		"DWPR" SESCON(21) Propellant weight expended - primary
	(1,12)		"DWBU" SESCON(21) Propellant weight expended - backup
	(1,13)		"PSIPR" SESCON(13) Out-of-plane yaw angle - primary
	(1,14)		"PSIBU" SESCON(13) Out-of-plane yaw angle - backup
	(1,15)		"XRNGPR" SESCON(15) Crossrange - primary
	(1,16)		"XRNGBU" SESCON(15) Crossrange - backup
	(1,17)		"CA1" SESCON(19) C1 target
	(1,18)		"CA2" 000000 C2 target
	(1,19)		"WCG" SESCON(21) OMS propellant expended target
	(1,20)		"THETEI" SESCON(13) TIG to EI transfer angle target
	(1,21)		"TPCHPR" SESCON(13) Thrust pitch angle - primary
	(1,22)		"TPCHBU" SESCON(13) Thrust pitch angle - backup
	(1,23)		"TYAWPR" SESCON(13) Thrust yaw angle - primary
	(1,24)		"TYAWBU" SESCON(13) Thrust yaw angle - backup
	(1,25)		"RANGPR" SESCON(13) Range from EI to landing site - primary
	(1,26)		"RANGBU" SESCON(13) Range from EI to landing site - backup
	(1,27)		"THELSD" SESCON(13) Range from EI for ETG
	(1,28)		"ETGRNG" SESCON(15) Range from EI computed by ETG
	(1,29)		"DTCSTP" SESCON(17) Time from burnout to EI - primary
	(1,30)		"DTCSTB" SESCON(17) Time from burnout to EI - backup
	(1,31)		"WBOPR" SESCON(21) Weight at burnout - primary
	(1,32)		"WBOBU" SESCON(21) Weight at burnout - backup
	(1,33)		"VGOXPR" SESCON(19)
	(1,34)		"VGOYPR" SESCON(19) Primary thrust system LVLH delta-V components
	(1,35)		"VGOZPR" SESCON(19)
	(1,36)		"VGOXBU" SESCON(19)
	(1,37)		"VGOYBU" SESCON(19) Backup thrust system LVLH delta-V components
	(1,38)		"VGOZBU" SESCON(19)
	(1,39)		"BTPR" SESCON(17) Burn time - primary
	(1,40)		"BTBU" SESCON(17) Burn time - backup
	(1,41)		"DVIMP" SESCON(19) Impulsive delta-V
			"VGXBU" SESCON(19)
			"VGYBU" SESCON(19) Backup thrust system inertial delta-V components
			"VGZBU" SESCON(19)

TABLE 4-II.- Concluded

PROCESSOR DTM

Array name	Index location	Default value	Definition
SUMTAB			"RENKX" SESCON(15) "RENKY" SESCON(15) } Radial components of the integrated entry interface vector "RENKZ" SESCON(15) "VENKX" SESCON(19) "VENKY" SESCON(19) } Velocity components of the integrated entry interface vector "VENKZ" SESCON(19)
PVTABP			Position/velocity phase table for the primary thrust system solution. Four states will be saved.
	(1,1)		(1-30,1) - Initial state - (XYZI,XYZIO)
	(1,2)		(1-30,2) - Ignition state - (RAAA,VA)
	(1,3)		(1-30,3) - Burnout state - (ROA,VOA)
	(1,4)		(1-30,4) - Entry interface state - (RENK,VENK)
PVTABB			Position/velocity phase table for the backup thrust system solution. The same four states defined for PVTABP will be saved.

TABLE 4-III.- PROCESSOR DISPLAY FORMAT

PROCESSOR DTM

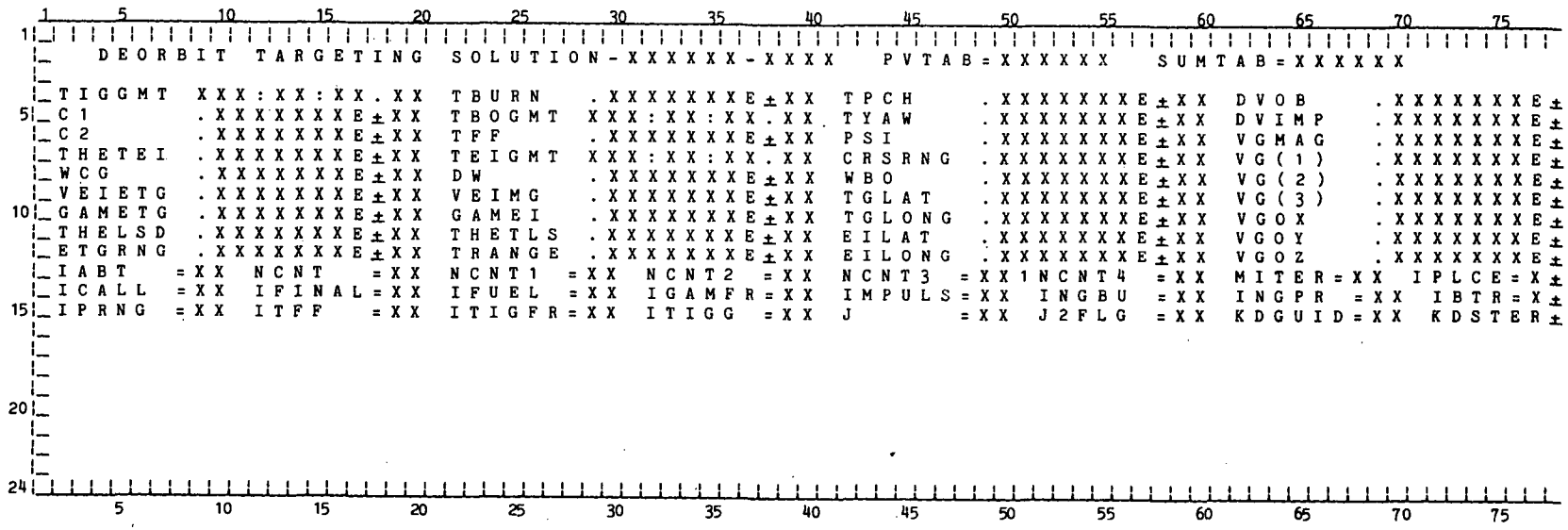


TABLE 4-IV.- DISPLAY PARAMETER DEFINITION TABLE

PROCESSOR DTM

DEORBIT TARGETING SOLUTION	
Display parameter label	Parameter definition
C1	Ordinate intercept on the V_v - V_h target line
C2	Slope of the V_v - V_h target line
CRSRNG	Entry crossrange, n. mi.
DVIMP	Impulsive delta-V
DVOB	Available delta-V
DW	Propellant expended
EILAT	Entry interface latitude
EILONG	Entry interface longitude
ETGRNG	Target range from EI to the landing site, n. mi.
GAMETG	Target EI flightpath angle
PSI	Out-of-plane yaw angle
PVTAB	Position/velocity phase table name
SUMTAB	Summary table name
TBOGET	Time of deorbit maneuver burnout - GET, hr., min., sec
TBURN	Deorbit burn time
TEIGET	Time of entry interface - GET, hrs., min., sec
TFF	Time from burnout to entry interface
TGLAT	Tig latitude
TGLONG	Tig longitude
THELSD	Transfer angle from EI to landing site
THETEI	Transfer angle from TIG to EI
THETLS	EGRT entry interface range angle
TIGGET	Ignition time - GET, hrs., min., sec
TPCH	Thrust pitch angle
TYAW	Thrust yaw angle
TRANGE	Computed range from EI to landing site, n. mi.
GAMEI	Computed EI flightpath angle
VEIETG	Target entry interface velocity
VEIMG	Computed entry interface velocity
VG(1), VG(2), VG(3)	Inertial components of the velocity to be gained vector at ignition time
VGMAG	Inertial delta-V magnitude
VGOX, VGOY, VGOZ	LVLH components of the velocity to be gained vector at ignition time
WBO	Weight at burnout
WCG	OMS propellant to be expended

TABLE 4-IV.- Concluded

PROCESSOR DTM

DEORBIT TARGETING SOLUTION

Display parameter label	Parameter definition
	The following parameters appear in the "FLAGS" output display and their definitions are found in the section 5.0, vol. III description of the DTMOT routine

IABT	NCNT	NCNT1	NCNT2	NCNT3	NCNT4	MITER	IPLACE
ICALL	IFINAL	IFUEL	IGAMFR	IMPULS	INGBU	INGPR	IBTR
IPRNG	ITFF	ITIGFR	ITIGG	J	J2FLG	KDGUID	KDSTER

TABLE 4-V.- PROCESSOR MESSAGE TABLE

PROCESSOR DTM

MSG no.	Message ID block	Message text block and explanation
1	*DTM*	"MAX ITERATIONS EXCEEDED - MITER = N" Meaning: In one of the DTM iteration loops the maximum iterations were used. The value of MITER printed out identifies the logic area in which the maximum was exceeded. Severity: Processor terminated Action required by user: Examine interface table inputs for gross errors and contact DTM expert for analysis.
2	*DTT1*	"STATE VECTOR IS NOT IN THE T.E.G. COORDINATE SYSTEM" Meaning: The input state vector is in a coordinate system other than the true Equator and Greenwich meridian system required for input to the processor. Severity: Processor terminated Action required by user: Use the vector conversion utility or input a correct vector through the interface table and rerun the processor.
3	*DTT1*	"ERRONEOUS HOLLERITH INPUT - ITIGFR, IFUEL, INGPR, INGBU OR KDGUID INCORRECT" Meaning: User has input an incorrect Hollerith option for one of the above variables in the interface table. Severity: Processor terminated Action required by user: Rerun the processor and use the interface table extended prompts if required to ensure the correct Hollerith inputs.
4	*DTM9*	"FREE FALL TIME FORCED TO TFFMIN CONSTRAINT" Meaning: An acceptable solution was found but the freefall time minimum was encountered. Thus, equal yaw angles were not achieved. Severity: Normal termination Action required by user: If the equal yaw angle constraint is required, rerun with a relaxed TFFMIN constraint.
5	*DTM9*	"FREE FALL TIME CONSTRAINT VIOLATED" Meaning: A solution was found but a freefall time less than the input minimum was required. Severity: Normal termination Action required by user: Use a TIG fixed solution to raise the freefall time if the result was unacceptable.

TABLE 4-V.- Concluded

PROCESSOR DTM

MSG no.	Message ID block	Message text block and explanation
6	*DTM9*	"FREE FALL TIME FORCED TO TFFMIN CONSTRAINT - SOLUTION UNACCEPTABLE IN DELTA-V" Meaning: An unacceptable delta-V resulted even after reaching the minimum freefall time. Severity: Processor terminated Action required by user: Freefall time (TFFMIN) or OMS propellant available (WCGOMS) must be increased to obtain an acceptable solution.
7	*PEG*	"NO PHYSICAL SOLUTION FOUND" Meaning: No solution was possible from the LTVCON process. Severity: Processor terminated Action required by user: Check for gross input errors or contact a PEG expert for analysis.
8	*PEG*	"VGO EQUAL ZERO" Meaning: A zero delta-V solution found Severity: Processor terminated Action required by user: Check for gross input errors including the input vector or contact a PEG expert for analysis.
9	*PEG*	"NO CONVERGENCE - MAX ITERATIONS EXCEEDED" Meaning: No PEG solution was found within the maximum allowed PEG iterations. Severity: Processor terminated. Action required by user: Check for gross input errors including the NMAX and NSEG iteration limits in DTMCON or contact a PEG expert for analysis.
10	*PEG*	"INPUT THRUSTER UNACCEPTABLE FOR PEG GUIDANCE" Meaning: The thruster selection for INGPR or INGBU was not valid. Severity: Processor terminated. Action required by user: Rerun with correct thruster selection for INGPR and INGBU.

TABLE 4-VI.- INTERFACE TABLE EXTENDED PROMPTS

PROCESSOR DTM

Processor name	Processor abstract prompt (maximum 256 characters)
DTM	The DTM computes a deorbit maneuver and associated targets to ensure a compatible velocity, flightpath angle, and range at entry interface.

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
ITIGFR	Deorbit ignition mode: "FIXED" = Tig fixed, "FREE" = Tig free
TIG	Ignition time in GET hrs., min., sec. - input when ITIGFR = "FREE"
TTHRSH	Threshold time in GET hrs., min., sec. - input when ITIGFR = "FREE"
IFUEL	Propellant use mode: "FW" = fuel wasting, "IP" = inplane
WCGOMS	Weight of OMS propellant to be burned during a fuel wasting deorbit
INGPR	Primary engine configuration: "1OMS" = one OMS thrusting "2OMS" = two OMS thrusting "RCS" = RCS thrusting
INGBU	Backup engine configuration: "1OMS" = one OMS thrusting "2OMS" = two OMS thrusting "RCS" = RCS thrusting
C1IP	Ordinate intercept on the $V_{vertical}$-$V_{horizontal}$ target line
C2IP	Slope of the $V_{vertical}$-$V_{horizontal}$ target line

TABLE 4-VI.- Continued

PROCESSOR DTM

Processor name	Processor abstract prompt (maximum 256 characters)
Parameter keyword name	**Parameter definition prompt (maximum 256 characters)**
TFFMIN	Minimum allowed free-fall time from burnout to entry interface
RNGIP	Target range from entry interface to landing site
KDGUID	Guidance option: "PEG4" = target to V_v-V_h line "PEG7" = external delta-V targets
DTMVEC	Position and velocity state vector in the TEG coordinate system and at a time prior to the input TIG or threshold time
INTEGF	Integration option flag: "AEG" = analytic ephemeris generator
IPOUT	Print options: "FINAL" = final print display only "FLAGS" = final print with flag settings "DEBUG" = all terminal displays and off-line debug
IOUNIT	Output unit for off-line debug print
TLATD	Landing site geodetic latitude, rad
TLONG	Landing site geodetic longitude, rad
TALTD	Landing site geodetic altitude, ft

TABLE 4-VI.- Continued

PROCESSOR DTM

Processor name	Processor abstract prompt (maximum 256 characters)
Parameter keyword name	**Parameter definition prompt (maximum 256 characters)**
RAZ	Landing site azimuth, rad
FOMS	OMS one-engine thrust, lb
FRCS	RCS four-engine thrust, lb
XMDOMS	OMS one-engine flow rate, lb/sec
XMDRCS	RCS four-engine flow rate, lb/sec
SESCON	Sesson constants array
DTMCON	DTM internal constants array
FVECON	Entry target line consisting of five velocities, five gammas, and five ranges. The array entries are in ft/sec, radians, and feet, respectively.
GLOCON	FDS1 global constants array
SUMTAB	DTM summary table output array
PVTABP	Position/velocity phase table for primary thruster solution

TABLE 4-VI.- Concluded

PROCESSOR DTM

Processor name	Processor abstract prompt (maximum 256 characters)

Parameter keyword name	Parameter definition prompt (maximum 256 characters)
PVTABB	Position/velocity phase table for backup thruster solution

5.0 PROCESSOR ROUTINES

5.1 ROUTINE NAME - MAIN PROGRAM DTM

The DTM routine contains the top level calls to the input and output routines and calls to the other executive routines responsible for computational control of the processor.

5.1.1 Purpose

The DTM routine is the top level of a multilevel executive which directs the deorbit targeting processor through its processing functions.

5.1.2 Functional Description

The DTM issues calls for input and initialization of data via the interface table. In addition it issues calls to other executive routines as a function of the user specified options to be processed. Finally, it issues calls to the output routines to build displays, summary tables, and phase tables.

5.1.3 Assumptions and Limitations

None.

5.1.4 Method

DTM is the top level of a two level executive. It along with all the second level executive routines are core resident for the duration of processing. DTM calls the executives below it by subroutine "call" statements. It calls the overlayed computational segments by calling the routine "LDSEG" with the segment name as an argument.

5.1.5 Routine Input/Output Variables

Table 5.1-I contains the definitions of the input/output variables stored in common. Table 5.1-II contains the definitions of all the input/output variables for the DTM main executive routine.

5.1.6 Functional Logic Flow

Figure 5.1-1 contains the functional logic flow of the DTM executive routine.

5.1.7 Diagnostics and Debug

Debug capability is provided within the computational routines and will be discussed in those sections. Diagnostics come from the DTM main executive in the form of the error messages contained in table 4-V.

5.1.8 Special Comments

None.

5.1.9 References

Ives, D. G.; Garland, B. J. (JSC); Drapela, L.; and Bedford, J. (MDTSCO): OFT MCC Level C Requirements for the Deorbit Planning Processor, Revision 1. JSC IN 76-FM-93 dated February 14, 1977.

TABLE 5.1-I.- DEFINITIONS OF THE INPUT/OUTPUT VARIABLES STORED IN COMMON

PROGRAM VARIABLES
COMMON VARIABLES - COM(606),ICOM(1212)

NAME	LOCATION	DEFINITION
AAA(10)	COM(1)	- GOLPAR HISTORICAL DATA ARRAY
UNUSED(5)	COM(11)	- UNUSED 5 REAL WORDS
ATP(3)	COM(16)	- UNIT VECTOR IN THRUST DIRECTION COMPUTED IN PEG
ATPBU(3)	COM(19)	- UNIT VECTOR IN THRUST DIRECTION FOR BACKUP SYSTEM
ATPPR(3)	COM(22)	- UNIT VECTOR IN THRUST DIRECTION FOR PRIME SYSTEM
CA(2)	COM(25)	- INTERCEPT AND SLOPE OF THE V(V)-V(H) TARGET LINE
UNUSED	COM(27)	- UNUSED 1 REAL WORD

TABLE 5.1-I.- Continued

ZEROT	COM(28)	- GMT OF INITIAL INPUT VECTOR
TIG1	COM(29)	- ADJUSTED TIG FROM DTT5(BTR)
DY1	COM(30)	- TIG ADJUST PARAMETER FOR DTT5
REC(3,3)	COM(31)	- EARTH FIXED TO TOPODETIC MATRIX
CRRNBU	COM(40)	- CROSSRANGE FOR BACKUP SYSTEM
CRRNPR	COM(41)	- CROSSRANGE FOR PRIMARY SYSTEM
CRSRNG	COM(42)	- CROSSRANGE AT THE LANDING SITE
C1IP	COM(43)	- INPUT INTERCEPT OF THE V(V)-V(H) TARGET LINE
C2IP	COM(44)	- INPUT SLOPE OF THE V(V)-V(H) TARGET LINE
DC1	COM(45)	- RANGE ADJUST PARAMATER = 0 IN FDS1
DC11	COM(46)	- C1 OFFSET VALUE
DELAZ	COM(47)	- VARIABLE TO COMPUTE SIGN OF THETLS
DGAM	COM(48)	- FLIGHT PATH ANGLE ERROR USED IN OFFSET TARGETING
DNRNG	COM(49)	- DOWNRANGE FROM EI TO LANDING SITE
UNUSED(2)	COM(50)	- UNUSED 2 REAL WORDS
DTCOST	COM(52)	- COAST TIME FROM BURNOUT TO EI
DTHELS	COM(53)	- ANGLE FROM EI TO LANDING SITE
DTMCON(41)	COM(54)	- DTM PROCESSOR CONSTANTS ARRAY DEFINED IN 77-FM-18 VOL III
DTMVEC(15)	COM(95)	- INPUT VECTOR FOR DTM PROCESSING
UNUSED	COM(110)	- UNUSED 1 REAL WORD
DVBU	COM(111)	- BACKUP SYSTEM DEORBIT DELTA-V
DVPR	COM(112)	- PRIME SYSTEM DEORBIT DELTA-V
DVIMP	COM(113)	- IMPULSIVE DELTA-V
DVOBBU	COM(114)	- BACKUP SYSTEM DELTA-V AVAILABLE
DVOBPR	COM(115)	- PRIMARY SYSTEM DELTA-V AVAILABLE
DWBU	COM(116)	- BACKUP SYSTEM WEIGHT OF PROPELLANT EXPENDED
DWPR	COM(117)	- PRIMARY SYSTEM WEIGHT OF PROPELLANT EXPENDED
DY2	COM(118)	- TIG ADJUST PARAMETER = DWBU-DWPR
EARPOL(3)	COM(119)	- UNIT VECTOR IN THE DIRECTION OF EARTH ROTATION AXIS
TIGHMS(3)	COM(122)	- TIG G.E.T. IN HRS/MIN/SEC
UNUSED(2)	COM(125)	- UNUSED 2 REAL WORDS
ERAEI	COM(127)	- EARTH ROTATION ANGLE AT TIME OF EI
ERAI	COM(128)	- EARTH ROTATION ANGLE AT INITIAL INPUT VECTOR TIME
ETGRNG	COM(129)	- REQUIRED ENTRY RANGE FROM DTT14
FBU	COM(130)	- BACKUP SYSTEM THRUST
FFF	COM(131)	- THRUST OF SELECTED ACTIVE SYSTEM
FOMS	COM(132)	- THRUST OF ONE OMS ENGINE
FPR	COM(133)	- PRIMARY SYSTEM THRUST
FRCS	COM(134)	- THRUST OF FOUR RCS ENGINES
FT	COM(135)	- THRUST OF SELECTED SYSTEM IN PEG
FVECON(15)	COM(136)	- INPUT ENTRY TARGET LINE
FWYAW	COM(151)	- OUT OF PLANE FUEL WASTING ANGLE
GAMETG	COM(152)	- REQUIRED ENTRY FLIGHT PATH ANGLE
UNUSED(2)	COM(153)	- UNUSED 2 REAL WORDS
GLOCON(180)	COM(309)	- FDS1 GLOBAL CONSTANTS ARRAY DEFINED IN 77-FM-18
UNUSED	COM(245)	- UNUSED 1 REAL WORD
HEINK	COM(246)	- ALTITUDE AT ESTIMATED EI TIME FOR OFFSET TARGETING

TABLE 5.1-I.- Continued

HEIS	COM(247)	- DESIRED ALTITUDE OF EI
TTRHMS	COM(248)	- TTHRSH G.E.T. IN HRS/MIN/SEC
UNUSED	COM(251)	- UNUSED 1 REAL WORDS
HTGT	COM(252)	- PEG ENTRY INTERFACE ALTITUDE
UNUSED	ICOM(505)	- UNUSED 1 INTEGER WORD
IAME	ICOM(506)	- ABORT SELECTION FLAG = 0 IN FDS1
UNUSED	ICOM(507)	- UNUSED 1 INTEGER WORD
IAOA	ICOM(508)	- DTM ABORT FLAG = 0 IN FDS1
IBTR	ICOM(509)	- RANGE ADJUSTMENT PASS FLAG 0 = NOT FIRST PASS 1 = FIRST PASS
ICALLL	ICOM(510)	- ETG CALL SELECT FLAG 0 = NO ETG CALL 1 = CALL ETG
UNUSED	ICOM(511)	- UNUSED 1 INTEGER WORD
IDTM	ICOM(512)	- DTM MAIN ROUTINE PASS FLAG 1 = FIRST PASS >1 = TIG < TTHRSH PASS
IERR	ICOM(513)	- DTM ERROR DETECTION FLAG 0 = NO ERRORS DETECTED 1 = ERROR DETECTED - NO SOLUTION
IFFPTM	ICOM(514)	- FREE FLIGHT PREDICTOR ROUTE FLAG 0 = INTEGRATE TO EI 1 = INTEGRATE TO TIG
IFINAL	ICOM(515)	- ETG FINAL CALL FLAG 0 = CALL ETG IN APPROXIMATE MODE 1 = CALL ETG IN PRECISE MODE
IFUEL	ICOM(516)	- FUEL USAGE MODE FLAG 1 = FUEL WASTING 2 = INPLANE
IGAMFR	ICOM(517)	- ETG MODE FLAG 0 = FIXED MODE 1 = FREE MODE
ITCST	ICOM(518)	- CLOSEST APPROACH INTEGRATION FLAG SET = 0 IN FDS1
IMPULS	ICOM(519)	- IMPULSIVE SOLUTION FLAG 0 = FINITE BURN SOLUTION 1 = IMPULSIVE BURN SOLUTION
IMSG	ICOM(520)	- DTM ERROR MESSAGE FLAG. SEE 77-FM-18, VOL. III, SECTION 5.2.11 FOR VALUE DEFINITIONS
INCR	ICOM(521)	- SHALLOW TARGET LINE FLAG = 0 IN FDS1
INGBU	ICOM(522)	- BACKUP SYSTEM SELECTION FLAG 1 = RCS THRUST SYSTEM 14 = ONE OMS THRUST SYSTEM 16 = TWO OMS THRUST SYSTEM
INGPR	ICOM(523)	- PRIMARY SYSTEM SELECTION FLAG 1 = RCS THRUST SYSTEM 14 = ONE OMS THRUST SYSTEM 16 = TWO OMS THRUST SYSTEM
INTEGF(3)	ICOM(524)	- INPUT HOLLERITH FOR INTEGRATOR SELECTION
UNUSED	ICOM(527)	- UNUSED 1 INTEGER WORD
INTBUF(224)	ICOM(528)	- FDS1 INTERFACE TABLE I-O BUFFER
IOUNIT	ICOM(752)	- LOGICAL UNIT FOR OFFLINE OUTPUT
UNUSED	ICOM(753)	- UNUSED 1 INTEGER WORD
IPLACE	ICOM(754)	- DTM PROCESSING LOCATION FLAG. REFER TO 76-FM-50, TABLE 4.4.3-II,

TABLE 5.1-I.- Continued

		PAGE 66 FOR DEFINITIONS
IPOUT(3)	ICOM(755)	- INPUT HOLLERITH FOR OUTPUT DESIRED "FINAL" = FINAL DISPLAY ONLY "FLAGS" = FINAL DISPLAY + FLAG SETTINGS "DEBUG" = FINAL DISPLAY,FLAG SETTINGS, AND OFFLINE DEBUG PRINT
IPRETG	ICOM(758)	- ETG PRECISION MODE FLAG 0 = APPROXIMATE RANGE SOLUTION 1 = PRECISE RANGE SOLUTION
IPRNG	ICOM(759)	- ETG RANGE SELECTION FLAG 0 = USE ETG COMPUTED RANGE 1 = USE INPUT RANGE,C1,AND C2
UNUSED	ICOM(760)	- UNUSED 1 INTEGER WORD
ISTOP	ICOM(761)	- ETG PROCESSOR ABORT FLAG = 0 IN FDS1
ITIGFR	ICOM(762)	- TIG MODE SELECTED FLAG 0 = TIG FIXED MODE 1 = TIG FREE MODE 2 = BTR INTERNAL TIG FIXED
ITERM	ICOM(763)	- PROCESSING TERMINATION FLAG 0 = CONTINUE PROCESSING 1 = TERMINATE PROCESSING
ITFF	ICOM(764)	- FREE FALL TIME CONSTRAINT FLAG 1 = NORMAL SOLUTION 2 = TFF FORCED TO CONSTRAINT 3 = DELTA-V EXCEEDS DELTA-V AVAILABLE 4 = TFF CONSTRAINT EXCEEDED
ITIGG	ICOM(765)	- DTTS(BTR) PASS FLAG 0 = FIRST PASS SET ITIGFR = 2 1 = FIRST PASS COMPUTATIONS >1 = SECOND OR GREATER PASS
JDGUID(3)	ICOM(766)	- INPUT GUIDANCE MODE HOLLERITH
JFUEL(1)	ICOM(769)	- INPUT FUEL USE MODE HOLLERITH
JJ	ICOM(770)	- GOLPAR STATUS FLAG
JNGPR(3)	ICOM(771)	- INPUT PRIMARY SYSTEM SELECTION HOLLERITH
JNGBU(3)	ICOM(774)	- INPUT BACKUP SYSTEM SELECTION HOLLERITH
JSYS(3)	ICOM(777)	- THRUST SYSTEM HOLLERITH OUTPUT
JTIGFR(3)	ICOM(780)	- INPUT TIG MODE HOLLERITH
J2FLG	ICOM(783)	- J2 TERM SELECTION FLAG 0 = NO J2 TERM 1 = USE J2 TERM
LU	ICOM(784)	- LOGICAL UNIT OF USERS TERMINAL
KACS(1)	ICOM(785)	- NUMBER OF 4 THRUSTER RCS ACTIVE
KDGUID	ICOM(786)	- PEG GUIDANCE MODE FLAG 3 = PEG4 SOLUTION 7 = PEG7 SOLUTION
UNUSED(2)	ICOM(787)	- UNUSED 2 INTEGER WORDS
KDSTER	ICOM(789)	- STEERING MODE FLAG IN PEG 2 = NOMINAL PEG4 TURNING RATE 1 = ZERO TURNING RATE FOR INERTIAL HOLD
KDTHR	ICOM(790)	- ENGINE CONFIGURATION FLAG IN PEG 1 = RCS THRUST MODE 14 = ONE OMS THRUST MODE 16 = TWO OMS THRUST MODE
UNUSED(2)	ICOM(791)	- UNUSED 2 INTEGER WORDS
KINIT	ICOM(793)	- PEG INITIAL PASS FLAG

TABLE 5.1-I.- Continued

		0 = NO INITIALIZATION 1 = FIRST PASS INITIALIZATION
KITER	ICOM(794)	- PEG TIG STATE INITIALIZATION FLAG = 1 IN FDS
KMODE	ICOM(795)	- GUIDANCE MODE FLAG IN DTM - SAME AS KDGUID
KOMS(1)	ICOM(796)	- NUMBER OF OMS THRUSTING
UNUSED(2)	ICOM(797)	- UNUSED 2 INTEGER WORDS
KSTEER	ICOM(799)	- STEERING MODE FLAG IN DTM - SAME AS KDSTER
NCNT	ICOM(800)	- DTM ITTERATION COUNTER
NCNT1	ICOM(801)	- DTT21 ITTERATION COUNTER
NCNT2	ICOM(802)	- DTT5 ITTERATION COUNTER
NCNT3	ICOM(803)	- DTM8 ITTERATION COUNTER
NCNT4	ICOM(804)	- DTM4 ITTERATION COUNTER
NSYS	ICOM(805)	- SYSTEM SELECTION FLAG 1 = PRIMARY SYSTEM 2 = BACKUP SYSTEM
MITER	ICOM(806)	- MAX ITTERATION DIAGNOSTICS FLAG. REFER TO 76-FM-93,TABLE 4.4.3-I, PAGE 64 FOR DEFINITIONS
UNUSED	COM(404)	- UNUSED 1 REAL WORD
PSIBU	COM(405)	- BACKUP SYSTEM OUT OF PLANE YAW ANGLE
PSIPR	COM(406)	- PRIMARY SYSTEM OUT OF PLANE YAW ANGLE
RAAA(3)	COM(407)	- POSITION VECTOR AT TIG.
RGRAV(3)	COM(410)	- GRAVITY EFFECT ON POSITION VECTOR OVER THE BURN ARC
RAZ	COM(413)	- INPUT RUNWAY AZIMUTH ANGLE
RLS(3)	COM(414)	- EARTH FIXED POSITION VECTOR
RCHMAG	COM(417)	- EARTH RADIUS AT THE LANDING SITE
RC1(3)	COM(418)	- INITIAL POSITION VECTOR USED IN PRECISE PREDICT
RDA(3)	COM(421)	- BURNOUT POSITION VECTOR
REIS(3)	COM(424)	- EI POSITION VECTOR (PRE OFFSET TGT)
REISM	COM(427)	- MAGNITUDE OF REIS VECTOR
RENK(3)	COM(428)	- EI POSITION VECTOR (POST OFFSET TGT)
RNGIP	COM(431)	- INPUT RANGE
RTA(3)	COM(432)	- EI POSITION VECTOR
UNUSED	COM(435)	- UNUSED 1 REAL WORD
RTE	COM(436)	- GEOCENTRIC RADIUS OF LANDING SITE
SBD	COM(437)	- SIGN OF OUT OF PLANE BURN DIRECTION
SMISS	COM(438)	- CONVERGANCE CRITERIA FOR VMISS
SESCON(90)	ICOM(877)	- FDS1 SESSION CONSTANTS ARRAY DEFINED IN 77-FM-18
TALTD(1)	COM(484)	- INPUT GEODETIC LANDING SITE ALTITUDE
UNUSED	COM(485)	- UNUSED 1 REAL WORD
TC	COM(486)	- TIME OF CLOSEST APPROACH
TEINK	COM(487)	- TIME AT EI (POST OFFSET TARGETING)
TIG	COM(488)	- IGNITION TIME
TEIFVE	COM(489)	- FVE TIME TO EI = 1800. IN FDS1
UNUSED	COM(490)	- UNUSED 1 REAL WORD
TEIS	COM(491)	- TIME AT EI (PRE OFFSET TARGETING)
TFFBU	COM(492)	- BACKUP SYSTEM FREE FALL TIME
TFFMIN	COM(493)	- INPUT MINIMUM ALLOWED FREE FALL TIME
TFFPR	COM(494)	- PRIMARY SYSTEM FREE FALL TIME
TGO	COM(495)	- BURN DELTA-T
UNUSED	COM(496)	- UNUSED 1 REAL WORD
THEPEI	COM(497)	- PAST VALUE OF THETEI IN MINIMUM DELTA-V SEARCH
THETAT	COM(498)	- DESIRED TRANSFER ANGLE TIG TO EI

TABLE 5.1-I.- Continued

THETD	COM(499)	- ORBITAL RATE
THETDR	COM(500)	- RELATIVE CLOSING RATE BETWEEN THE VEHICLE AND LANDING SITE
THETEI	COM(501)	- TIG TO EI TRANSFER ANGLE
THETLS	COM(502)	- ACTUAL RANGE ANGLE EI TO LANDING SITE
THELSD	COM(503)	- DESIRED RANGE ANGLE EI TO LANDING SITE
TGDMN	COM(504)	- MIMIMUM VALUE OF DENOMINATOR IN EQUAL OMS LOGIC
TGSLPP	COM(505)	- INITIAL SLOPE FOR EQUAL OMS YAW LOGIC
UNUSED	COM(506)	- UNUSED 1 REAL WORD
TLATC(1)	COM(507)	- GEOCENTRIC LATITUDE OF LANDING SITE
TLATD(1)	COM(508)	- GEODETIC LATITUDE OF LANDING SITE
TLONG(1)	COM(509)	- GEODETIC LONGITUDE OF LANDING SITE
TF	COM(510)	- TIME OF BURNOUT
TPCHB	COM(511)	- BACKUP SYSTEM LVLH PITCH ANGLE
TPCHP	COM(512)	- PRIMARY SYSTEM LVLH PITCH ANGLE
TRANG	COM(513)	- EGRT COMPUTED RANGE
UNUSED	COM(514)	- UNUSED 1 REAL WORD
TT	COM(515)	- TIME OF ENTRY INTERFACE
TTHRSH	COM(516)	- INPUT TIG THRESHOLD TIME
TYAWB	COM(517)	- BACKUP SYSTEM LVLH YAW ANGLE
TYAWP	COM(518)	- PRIMARY SYSTEM LVLH YAW ANGLE
UFIMP(3)	COM(519)	- UNIT VECTOR IN IMPULSIVE THRUST DIRECTION
UXDTM(3)	COM(522)	- LVLH DOWNRANGE UNIT VECTOR
UYDTM(3)	COM(525)	- LVLH OUT OF PLANE UNIT VECTOR
UZDTM(3)	COM(528)	- LVLH RADIAL UNIT VECTOR
VA(3)	COM(531)	- VELOCITY VECTOR AT TIG
XYZEN(3)	COM(534)	- MEAN OF 50 POSITION VECTOR
VC1(3)	COM(537)	- INITIAL VELOCITY VECTOR USED IN PRECISE PREDICT
VDA(3)	COM(540)	- VELOCITY VECTOR AT BURNOUT
VEIS(3)	COM(543)	- VELOCITY VECTOR AT EI (PRE OFFSET TGT)
VEISM	COM(546)	- MAGNITUDE OF VEIS VECTOR
VENK(3)	COM(547)	- VELOCITY VECTOR AT EI(POST OFFSET TGT)
VEXX	COM(550)	- TOTAL EFFECTIVE EXAUST VELOCITY
UNUSED(2)	COM(551)	- UNUSED 2 REAL WORDS
VG(3)	COM(553)	- INERTIAL DELTA-V COMPONENTS
VGIMP	COM(556)	- IMPULSIVE DELTA-V MAGNITUDE
VGMAG	COM(557)	- MAGNITUDE OF VG FROM PEG
VGRAV(3)	COM(558)	- GRAVITY EFFECT ON VELOCITY VECTOR OVER THE BURN ARC
VGOX	COM(561)	- LVLH DELTA-V(X) COMPONENT
VGOY	COM(562)	- LVLH DELTA-V(Y) COMPONENT
VGOZ	COM(563)	- LVLH DELTA-V(Z) COMPONENT
VTAA(3)	COM(564)	- VELOCITY VECTOR AT EI
DTAVG	COM(567)	- STEP SIZE FOR GRAVITY PREDICTION
VEIM	COM(568)	- DESIRED ENTRY INTERFACE VELOCITY
VRHOTD(3)	COM(569)	- V-RHO TOPODETIC
WBO	COM(572)	- BURNOUT WEIGHT
WCG	COM(573)	- WEIGHT OF FUEL TO BE BURNED X SBD
WD	COM(574)	- FUEL FLOW RATE OF SELECTED SYSTEM
WDBU	COM(575)	- BACKUP SYSTEM FUEL FLOW RATE
WDPR	COM(576)	- PRIMARY SYSTEM FUEL FLOW RATE
UNUSED(2)	COM(577)	- UNUSED 2 REAL WORDS
XMD(1)	COM(579)	- FUEL FLOW RATE OF SELECTED SYSTEM
UNUSED(2)	COM(580)	- UNUSED 2 REAL WORDS

TABLE 5.1-I.- Continued

XMDOMS	COM(582)	- ONE OMS FUEL FLOW RATE
XMDRCS	COM(583)	- FOUR RCS FUEL FLOW RATE
XOFCN	COM(584)	- ITTERATIONS REQUIRED IN DTT7 TO CONVERGE ON TEI FOR OFFSET TARGETING
XYZED(3)	COM(585)	- MEAN OF 50 VELOCITY VECTOR
PSUM1	COM(588)	- **********
PSUM2	COM(589)	- *
PSUM3	COM(590)	- *
PSUM4	COM(591)	- * SUMMARY TABLE OUTPUT QUANTITY
PSUM5	COM(592)	- * WORKSPACE ARRAY
PSUM6	COM(593)	- *
PSUM7	COM(594)	- *
PSUM8	COM(595)	- *
PSUM9	COM(596)	- *
PSUM10	COM(597)	- **********
DTTHRS	COM(598)	- DELTA-T BETWEEN TIG AND TTHRSH
VHS	COM(599)	- CONIC HORIZONTAL VELOCITY COMPONENT AT ENTRY INTERFACE
VVS	COM(600)	- CONIC VERTICAL VELOCITY COMPONENT AT ENTRY INTERFACE
WCGOMS	COM(601)	- INPUT WEIGHT OF FUEL AVAILABLE
IPARM(5)	ICOM(1203)	- FDS1 INITIALIZATION PARAMETER ARRAY
IRPARM(5)	ICOM(1208)	- FDS1 RETURN CODE PARAMETER ARRAY

COMMON VARIABLES - PEGCOM(70),IPGCOM(140)

NAME	LOCATION	DEFINITION
UNUSED(3)	PEGCOM(1)	- UNUSED 3 REAL WORDS
ATR	PEGCOM(4)	- ESTIMATED THRUST ACCELERATION
CLMDXZ	PEGCOM(5)	- FRACTIONAL MULTIPLIER OF ORBIT RATE
UNUSED	PEGCOM(6)	- UNUSED 1 REAL WORD
DTPEG	PEGCOM(7)	- DELTA-T TO UPDATE STATE VECTOR FROM TIGT TO TIGA
DVSS(3)	PEGCOM(8)	- CHANGE IN ACCUMULATED SENSED VELOCITY
FTS	PEGCOM(11)	- SAVED THRUST VALUE FOR SELECTED SYSTEM
UNUSED	PEGCOM(12)	- UNUSED 1 REAL WORD
IPS	IPGCOM(25)	- FLAG FOR UPDATING THRUST AND FLOW RATE AT GUIDANCE PHASING
IV	IPGCOM(26)	- ACTIVE VEHICLE = 1 IN FDS1
KABORT	IPGCOM(27)	- ABORT GUIDANCE FLAG = 0 IN FDS1
KCUTOF	IPGCOM(28)	- ACTIVE GUIDANCE TERMINATION FLAG
KFLAG	IPGCOM(29)	- PEG ERROR INDICATOR FLAG
KFUELD	IPGCOM(30)	- OUT OF PLANE FUEL DEPLETION FLAG
KSTOP	IPGCOM(31)	- PEG SOLUTION CONVERGANCE FLAG
NCYC	IPGCOM(32)	- NUMBER OF PEG CALLS PER GUIDANCE CYCLE
NMAX	IPGCOM(33)	- MAX GUIDANCE ITTERATIONS ALLOWED
NMAX2	IPGCOM(34)	- MAX PEG ITTERATIONS ALLOWED
NOMS	IPGCOM(35)	- NUMBER OF OMS USED IN TWO PHASE MANEUVER
NPHASE	IPGCOM(36)	- NUMBER OF ACTIVE GUIDANCE THRUST PHASES USED
RHOO	PEGCOM(19)	- SCALAR DAMPING FACTOR APPLIED TO VMISS TO CORRECT VGO
TBRCS	PEGCOM(20)	- DELTA-T OF RCS DURING PARALLEL BURN

TABLE 5.1-I.- Concluded

TGDP	PEGCOM(21)	- PREVIOUS VALUE OF GUIDANCE TIME
TIGA	PEGCOM(22)	- ACTUAL TIG
TLAM	PEGCOM(23)	- REFERENCE TIME OF XLAM AND XLDA
TLAMC	PEGCOM(24)	- REFERENCE TIME OF XLAMC AND XLAMDC
UNUSED	PEGCOM(25)	- UNUSED 1 REAL WORD
TRCSOF	PEGCOM(26)	- RCS CUTOFF TIME DURING AOA
TTT	PEGCOM(27)	- CURRENT GUIDANCE TIME
UNUSED	PEGCOM(28)	- UNUSED 1 REAL WORD
UYY(3)	PEGCOM(29)	- UNIT VECTOR NORMAL TO DESIRED TRAJECTORY PLANE
UNUSED	PEGCOM(32)	- UNUSED 1 REAL WORD
VEXS	PEGCOM(33)	- SAVED VALUE OF TOTAL EFFECTIVE EXAUST VELOCITY
VGOXLV	PEGCOM(34)	- LVLH X COMPONENT OF DELTA-V
VGOYLV	PEGCOM(35)	- LVLH Y COMPONENT OF DELTA-V
VGOZLV	PEGCOM(36)	- LVLH Z COMPONENT OF DELTA-V
VGOP(3)	PEGCOM(37)	- INERTIAL VELOCITY TO BE GAINED
VP(3)	PEGCOM(40)	- PREDICTED THRUST CUTOFF VELOCITY
VSP(3)	PEGCOM(43)	- PREVIOUS PASS SENSED VELOCITY
XLAM(3)	PEGCOM(46)	- UNIT POSITION VECTOR IN VGO DIRECTION
XLAMC(3)	PEGCOM(49)	- UNIT POSITION VECTOR IN VGO DIRECTION
XLAMDC(3)	PEGCOM(52)	- UNIT VELOCITY VECTOR IN VGO DIRECTION
XLDA(3)	PEGCOM(55)	- UNIT VELOCITY VECTOR IN VGO DIRECTION
XLDXZ	PEGCOM(58)	- FRACTIONAL MULTIPLIER OF ORBIT RATE
XMASSE	PEGCOM(59)	- CURRENT MASS OF VEHICLE
XMBO	PEGCOM(60)	- DESIRED MASS OF VEHICLE AT THRUST CUTOFF
XMDOT	PEGCOM(61)	- CURRENT MASS FLOW RATE
UNUSED(6)	PEGCOM(62)	- UNUSED 6 REAL WORDS
UYD(3)	PEGCOM(68)	- UNIT VECTOR NORMAL TO TRAJECTORY PLANE

COMMON VARIABLES - P03COM(24),IP03CM(48)

NAME	LOCATION	DEFINITION
IYAW	IP03CM(1)	- FLAG FOR PWYAW COMPUTATION
UNUSED	IP03CM(2)	- UNUSED 1 INTEGER WORD
QPRIME	P03COM(2)	- INTERMEDIATE THRUST INTEGRAL VARIABLE
RC2(3)	P03COM(3)	- FINAL POSITION VECTOR COMPUTED IN PRECISE PREDICT
RGO(3)	P03COM(6)	- POSITION TO GO VECTOR INCLUDING RANGE BIAS
RP(3)	P03COM(9)	- PREDICTED THRUST CUTOFF POSITION
TIJOL	P03COM(12)	- BURN CENTROID DELTA-T
TIS	P03COM(13)	- TWICE THE THRUST INTEGRAL FROM 0 TO T PLUS 0 TO TGO
TAU	P03COM(14)	- RATIO OF MASS TO MASS FLOW RATE
TPREV	P03COM(15)	- PREVIOUS PREDICTED BURNOUT TIME VALUE
VGOYS	P03COM(16)	- SQUARE OF VGO OUT OF PLANE COMPONENT
VGOD	P03COM(17)	- DESIRED MAGNITUDE OF VGO FOR FUEL DEPLETION
VMISS(3)	P03COM(18)	- DIFFERENCE BETWEEN THE PREDICTED AND DESIRED CUTOFF VELOCITY
VC2(3)	P03COM(21)	- PREDICTED CUTOFF VELOCITY FROM PRECISE PREDICT

TABLE 5.1-II.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTM

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
	No math symbols are used in the flow chart logic. Refer to the code symbols alone.			All units are the FDS1 standard internal units of ft, ft/sec, sec, lbf, lbm, and radians unless otherwise noted in these tables.		The name of a variable as it is found in the interface table (on an output display or in common) will be found here.	See table 5.1-I for definition of all variables passed through common. The definitions for variables used in calling arguments and other variables not found in common will be found in this table.

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.1-II.- Continued

Routine DTM

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
C1IP		Real	I/O	ft/sec	IT	C1IP	Refer to table 5.1-I for all code symbol definitions.
C2IP		Real	I/O	--	IT	C2IP	
DTCOST		Real	O	sec	C		
DTMCON		Real	I	--	IT	DTMCON	
FOMS		Real	I/O	lbf	IT	FOMS	
FRCS		Real	I/O	lbf	IT	FRCS	
IAME		Intg	O	--	C		
IDTM		Intg	O	--	C		
IFFPTM		Intg	O	--	C		
IMSG		Intg	I/O	--	C		
INTEGF		6CH	I/O	-	IT	INTEGF	

NOTES:

All variables input via the interface table are output to common.

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.1-II.- Continued

Routine DTM

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IOUNIT		Intg	I/O	—	IT	IOUNIT	
IPLACE		Intg	O	--	C		
IPOUT		6CH	I/O	—	IT	IPOUT	
ITIGFR		Intg	O	--	C		
JDGUID		6CH	I/O	—	IT	KDGUID	
JFUEL		2CH	I/O	--	IT	IFUEL	
JNGPR		6CH	I/O	—	IT	INGPR	
JNGBU		6CH	I/O	—	IT	INGBU	
JTIGFR		6CH	I/O	—	IT	ITIGFR	
KDGUID		Intg	I/O	--	C		
KDSTER		Intg	I	—	C		
KMODE		Intg	O	--	C		

NOTES:

All variables input via the interface table are output to common.

TYPE

Free	Dubl	18CH	Mix
Intg	2CH	36CH	Char
Real	6CH	72CH	Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.1-II.- Continued

Routine DTM

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
NSYS		Intg	O	--	C		
IPARM		Intg	O	--	C		
IRPARM		Intg	O	--	C		
RGRAV		Real	I	ft	C		
RAZ		Real	I/O	rad	IT	RAZ	
RC1		Real	I	ft	C		
RC2		Real	O	ft	C		
RDA		Real	I/O	ft	C		
RNGIP		Real	I/O	n. mi.	IT	RNGIP	
SESCON		Free	I/O	--	IT	SESCON	
TALTD		Real	I/O	ft	IT	TALTD	
TIMEC		Real	I/O	sec	IT	DTMVEC(1)	

NOTES:

All variables input via the interface table are output to common.

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.1-II.- Continued

Routine DTM

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TIG		Real	O	sec	C		
TFFMIN		Real	I/O	sec	IT	TFFMIN	
TGO		Real	I	sec	C		
TIGHMS		Real	I/O	hr,min, sec	IT	TIG	
TLATD		Real	I/O	rad	IT	TLATD	
TLONG		Real	I/O	rad	IT	TLONG	
TP		Real	I	sec	C		
TT		Real	I	sec	C		
TTRHMS		Real	I/O	hr,min, sec	IT	TTHRSH	
TTHRSH		Real	I	sec	C		
UXDTM		Real	I	—	C		
UYDTM		Real	I	--	C		

NOTES:

All variables input via the interface table are output to common.

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.1-II.- Continued

Routine DTM

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
U2DTM		Real	I	--	C		
VC1		Real	I	ft/sec	C		
VC2		Real	O	ft/sec	C		
VDA		Real	I/O	ft/sec	C		
VG		Real	I	ft/sec	C		
VGRAV		Real	I	ft/sec	C		
VGOX		Real	O	ft/sec	C		
VGOY		Real	O	ft/sec	C		
VGOZ		Real	O	ft/sec	C		
XMDOMS		Real	I/O	slugs/sec	IT	XMDOMS	
XMDRCS		Real	I/O	slugs/sec.	IT	XMDRCS	
NOTES: All variables input via the interface table are output to common.		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.1-II.- Concluded

Routine DTM

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
WE		Real	I/O	ft	IT	GLOCON(13)	
WCGOMS		Real	I/O	lbm	IT	WCGOMS	

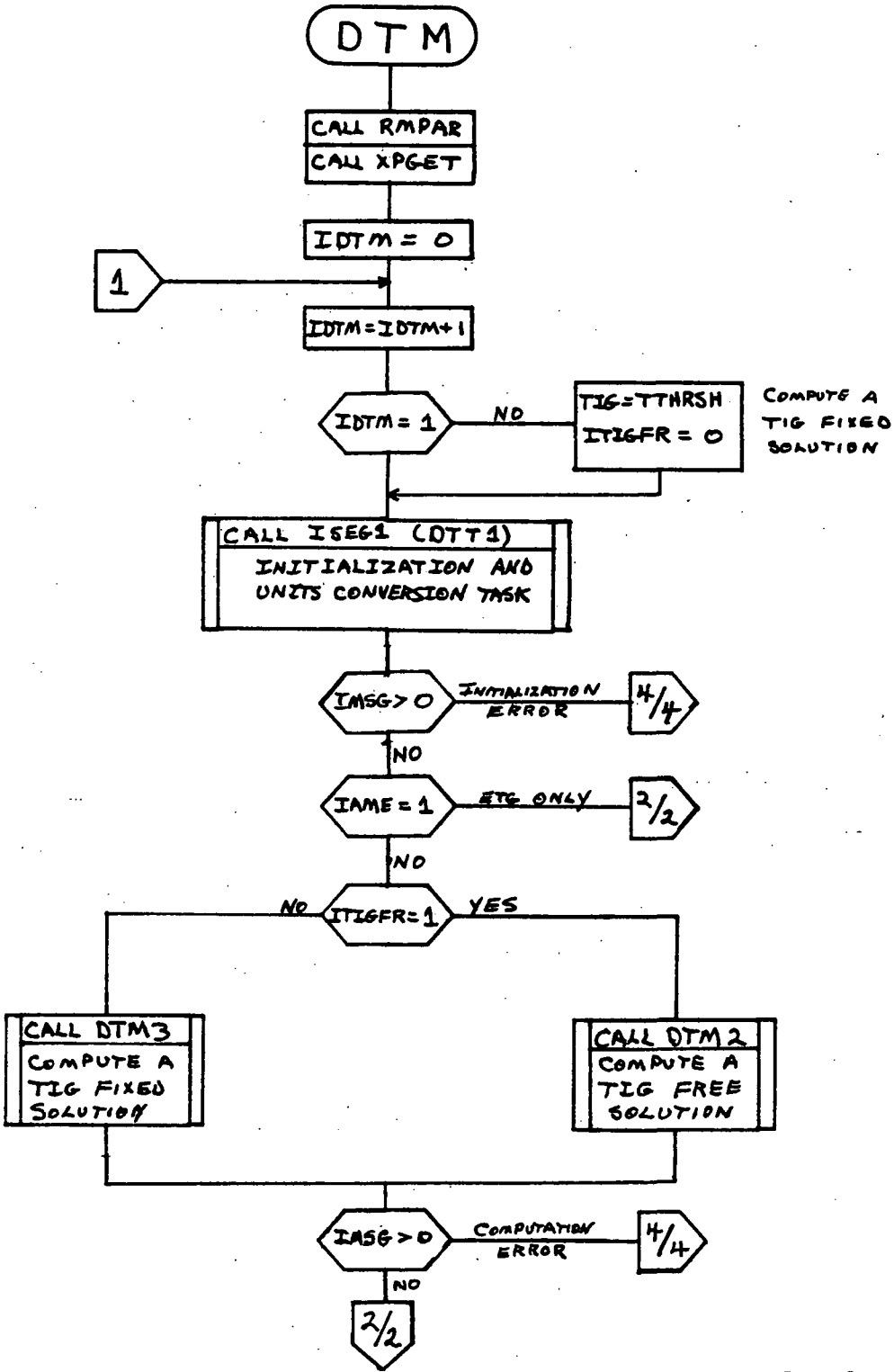
NOTES:

All variables input via the interface table are output to common.

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory



Page 1 of 4

Figure 5.1-1.- Functional flow for the DTM executive routine.

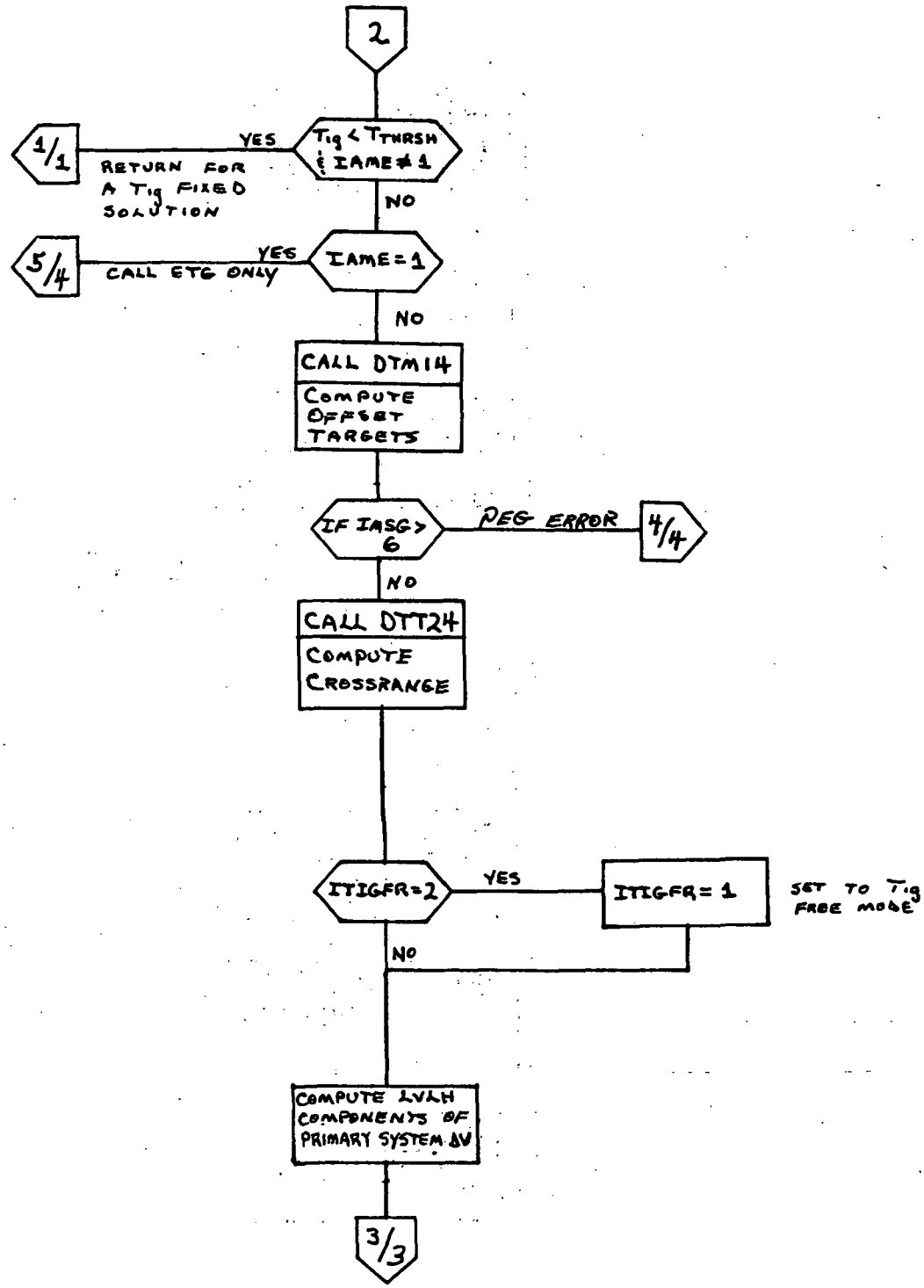


Figure 5.1-1.- Continued.

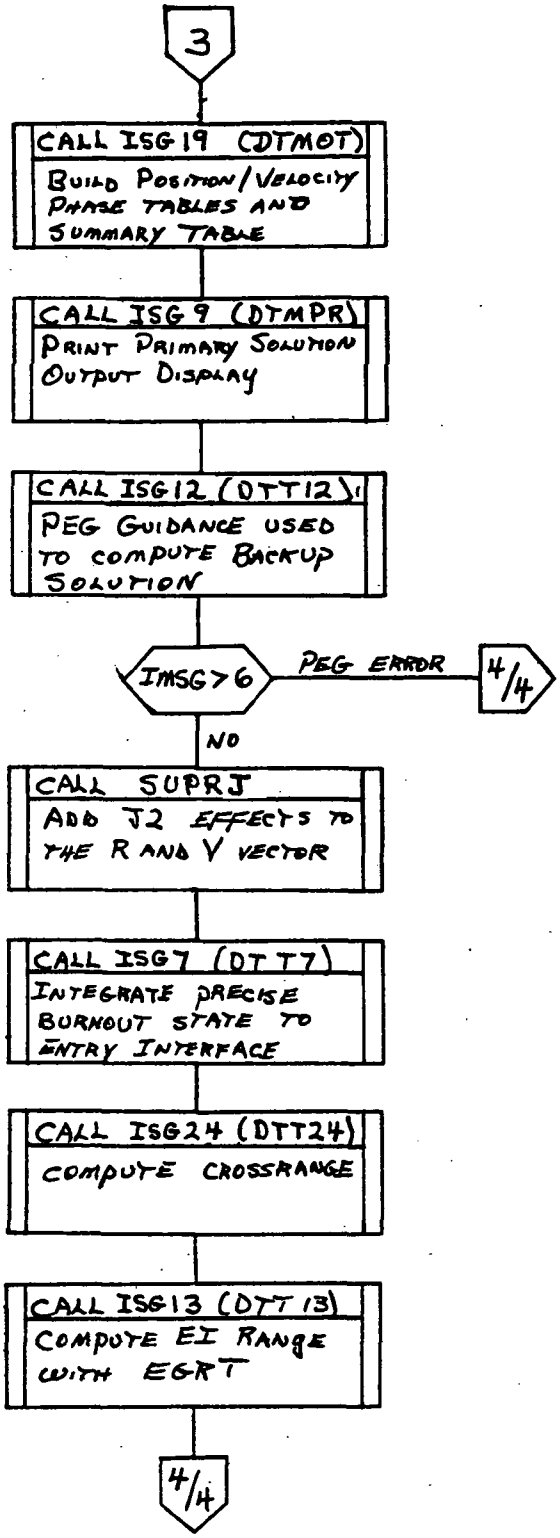


Figure 5.1-1.- Continued.

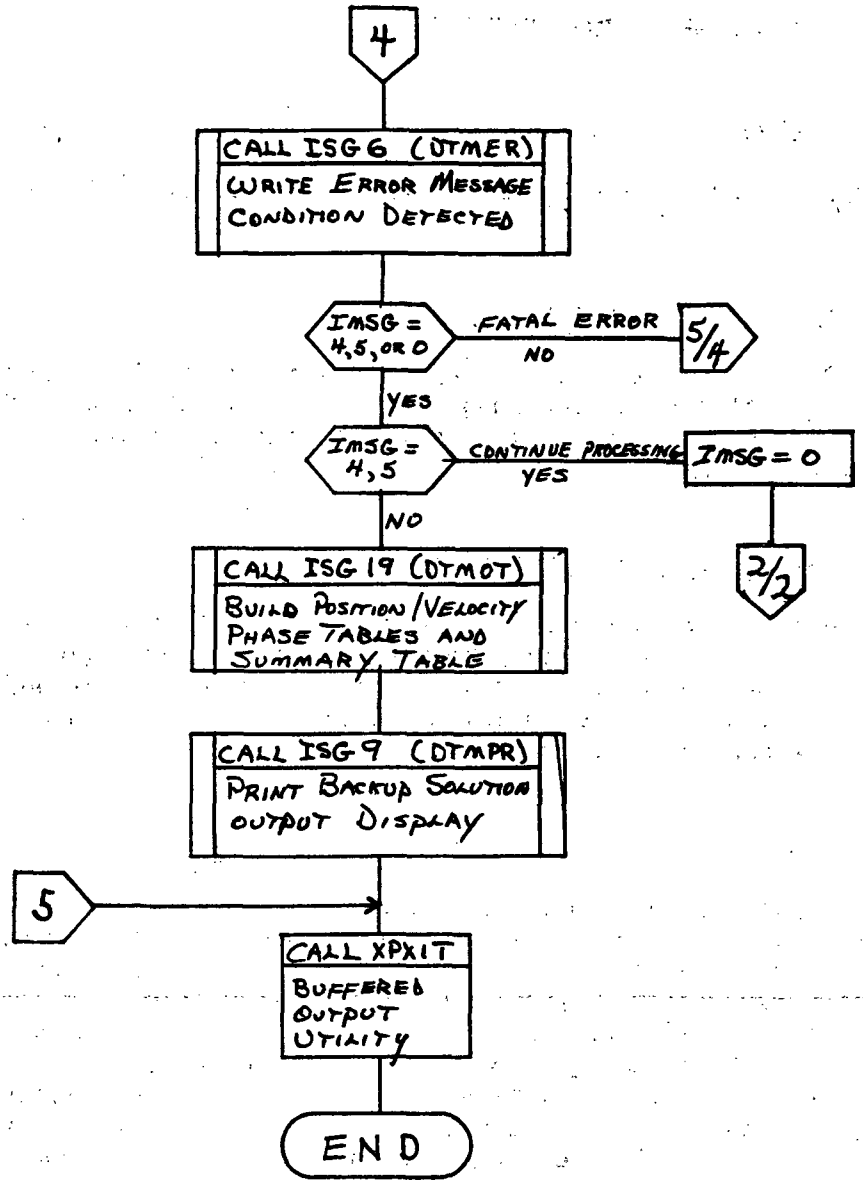


Figure 5.1-1.- Concluded.

5.2 ROUTINE NAME - DTM2 EXECUTIVE ROUTINE

5.2.1 Purpose

The DTM2 routine is the executor for computation of Tig free solutions within the DTM.

5.2.2 Functional Description

The DTM2 routine calls other executive and computational routines to determine an ignition time after the input threshold time at which a fuel wasting or inplane deorbit maneuver will result in entry conditions compatible with the input target line.

5.2.3 Assumptions and Limitations

The freefall time constraint (Tffmin) is only applied for fuel wasting solutions. Also, if fuel wasting is used and equal yaw angles cannot be achieved for the prime and backup systems within the input OMS propellant weight (WCGOMS) then an inplane equal OMS solution is found. In this case, the weight of OMS used may exceed the WCGOMS amount.

5.2.4 Method

Initially an inplane impulsive minimum delta-V solution is found. Then a finite burn solution is found for the prime and backup systems. When an inplane solution has been requested the result is an equal OMS propellant usage by the prime and backup systems. If a fuel wasting solution has been requested then a solutions resulting in equal out-of-plane yaw angles for the prime and backup systems is computed. If equal yaw angles are not possible then the fuel wasting mode defaults to find an inplane equal OMS solution if possible. Ignition time is adjusted as required to achieve the equal yaw angles or equal OMS usage, and to keep the freefall time above the input minimum. However, for inplane solutions the freefall time constraint is not enforced.

5.2.5 Routine Input/Output Variables

Table 5.2-I contains the definitions of all the input/output variables for the DTM2 routine.

5.2.6 Functional Logic Flow

Figure 5.2-1 contains the functional logic flow for the DTM2 executive routine.

5.2.7 Diagnostics and Debug

The diagnostic messages in table 4-V are invoked from DTM2 when the appropriate error flag has been set in a computational routine.

5.2.8 Special Comments

None.

5.2.9 References

None.

TABLE 5.2-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTM2

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
C1IP		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
DTCOST		Real	I		C		
DWTOL		Real	I		C	DTMCON(10)	
DWBU		Real	I		C		
DWPR		Real	I		C		
IAME		Intg	I		C		
IADA		Intg	I		C		
IERR		Intg	I		C		
IFUEL		Intg	I		C		
TIG		Real	I		C		
TFFBU		Real	I		C		
TFFMIN		Real	I		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.2-I.- Continued

Routine DTM2

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TTHRSH		Real	I		C		
TTHRB		Real	I		C	DTMCON(35)	
CA		Real	O		C		
DTTHRS		Real	O		C		
ICALLL		Intg	O		C		
IFINAL		Intg	O		C		
IMSG		Intg	O		C		
IPLACE		Intg	O		C		
IPRNG		Intg	O		C		
ITFF		Intg	O		C		
NCNT		Intg	O		C		
MITER		Intg	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

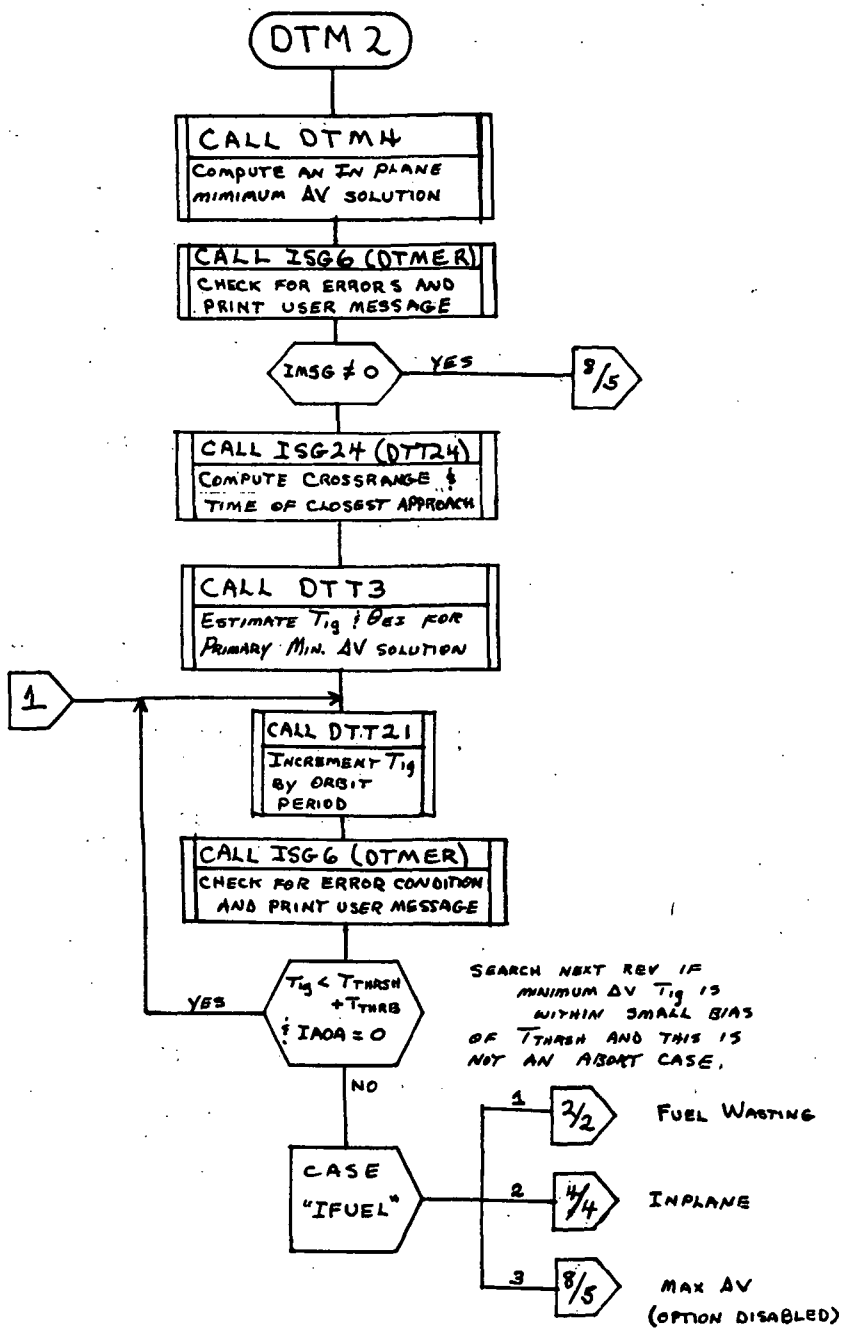
SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.2-I.- Concluded

Routine DTM2

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
SBD		Real	O		C		
WCGOMS		Real	I		C		

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory



Page 1 of 5

Figure 5.2-1.- Functional logic flow for the DTM2 executive routine.

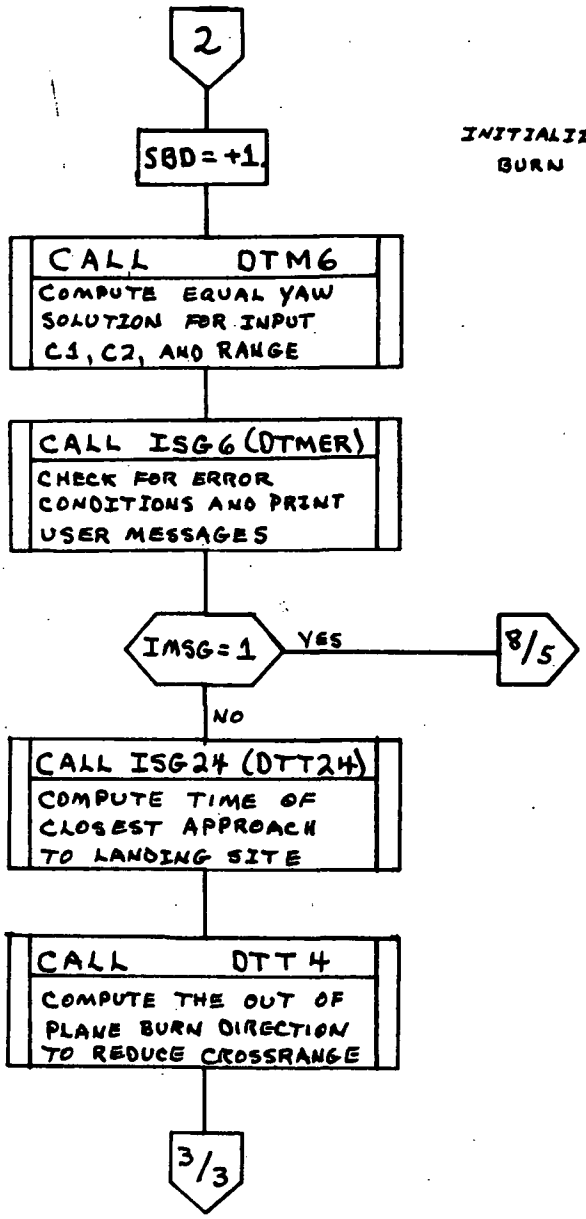


Figure 5.2-1.- Continued.

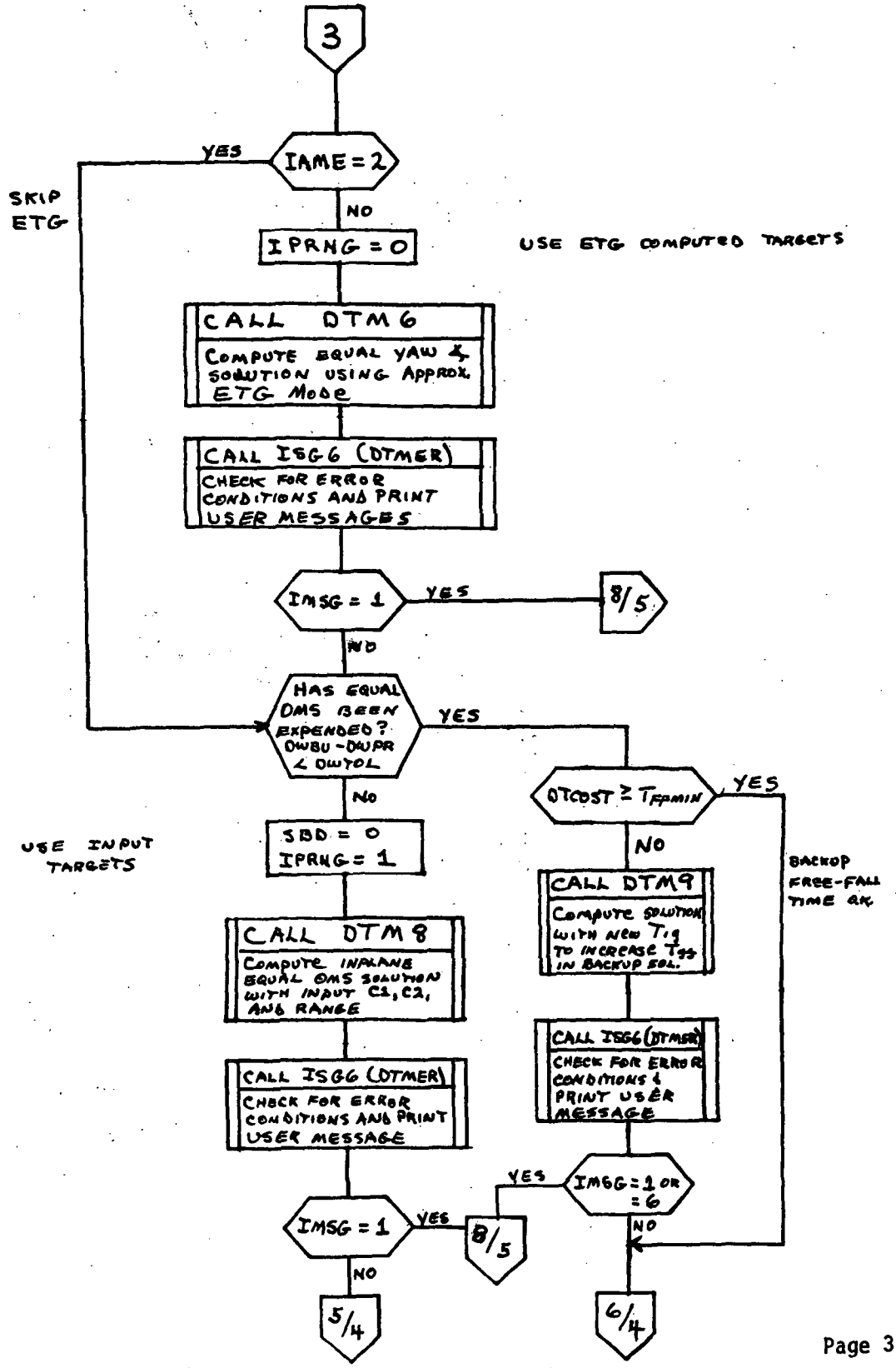


Figure 5.2-1.- Continued.

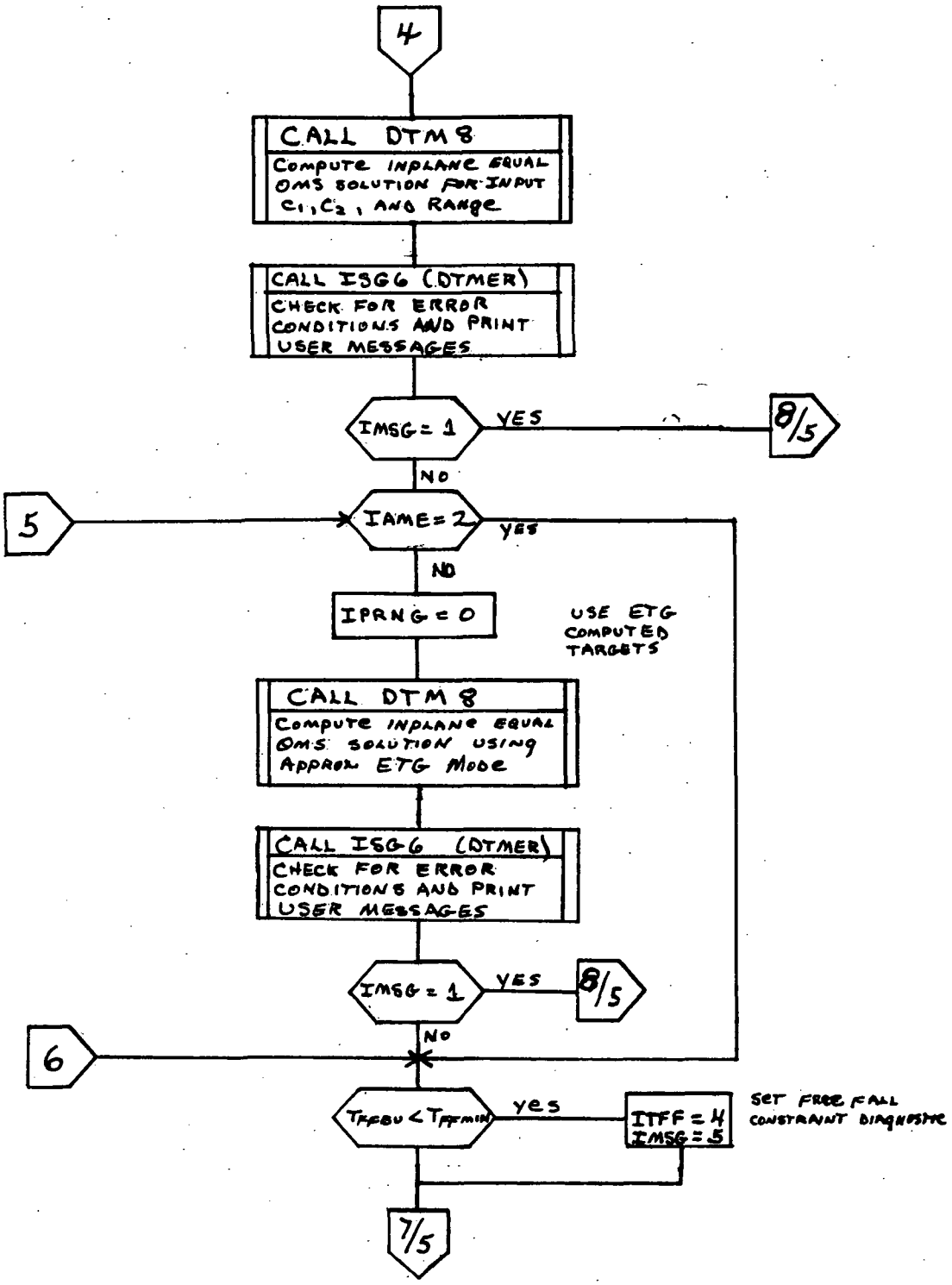


Figure 5.2-1.- Continued.

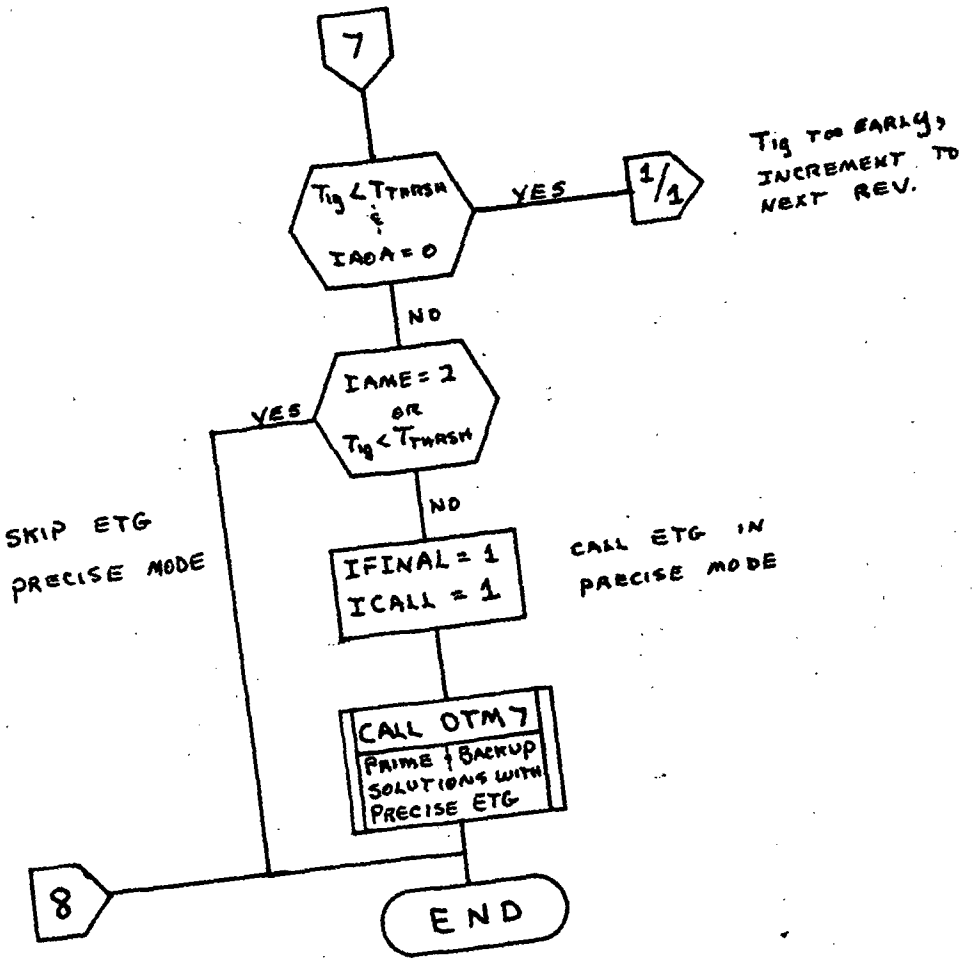


Figure 5.2-1.- Concluded.

5.3 ROUTINE NAME - DTM3 EXECUTIVE ROUTINE

5.3.1 Purpose

The DTM3 routine is the executor for computation of Tig fixed solutions within the DTM processor.

5.3.2 Functional Description

The DTM3 routine calls other executive and computational routines using the input fixed ignition time to compute a fuel wasting or an inplane deorbit maneuver that will result in entry conditions compatible with the input target line.

5.3.3 Assumptions and Limitations

The freefall time constraint is not applied at all for Tig fixed solutions because the fixed ignition time also fixes a value for freefall time. In the fuel wasting mode the OMS fuel available amount (WCGOMS) is used to constrain the delta-V used for the out-of-plane solutions, however, it does not constrain the OMS usage or delta-V required for an inplane solution.

5.3.4 Method

A finite burn solution is found for the prime and backup thrust systems with ignition time at the input Tig. For an inplane solution the prime and backup solutions will result in compatible entry ranges but will not normally use equal amounts of OMS fuel. Similarly, for a fuel wasting solution equal out-of-plane yaw angles will not normally be found unless the Tig selected happened to be one where equal yaw angles result. The logic simply computes a solution at the input Tig and the OMS usage and yaw angles fall out.

5.3.5 Routine Input/Output Variables

Table 5.3-I contains the definitions of all the input/output variables for the DTM3 executive routine.

5.3.6 Functional Logic Flow

Figure 5.3-1 contains the functional logic flow for the DTM3 executive routine.

5.3.7 Diagnostics and Debug

The diagnostic messages in table 4-V are invoked from DTM3 when the appropriate error flag has been set in a computational routine.

5.3.8 Special Comments

None.

5.3.9 References

None.

TABLE 5.3-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTM3

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DWBU		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
DWPR		Real	I		C		
DWTOL		Real	I		C		
IAME		Intg	I		C		
IERR		Intg	I		C		
IFUEL		Intg	I		C		
TFFBU		Real	I		C		
TFFMIN		Real	I		C		
WCGOMS		Real	I		C		
ICALLL		Intg	O		C		
IFINAL		Intg	O		C		
IMPULS		Intg	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.3-I.- Concluded

Routine DTM3

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IMSG		Intg	O		C		
IPLACE		Intg	O		C		
IPRNG		Intg	O		C		
ITFF		Intg	O		C		
NCNT		Intg	O		C		
MITER		Intg	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

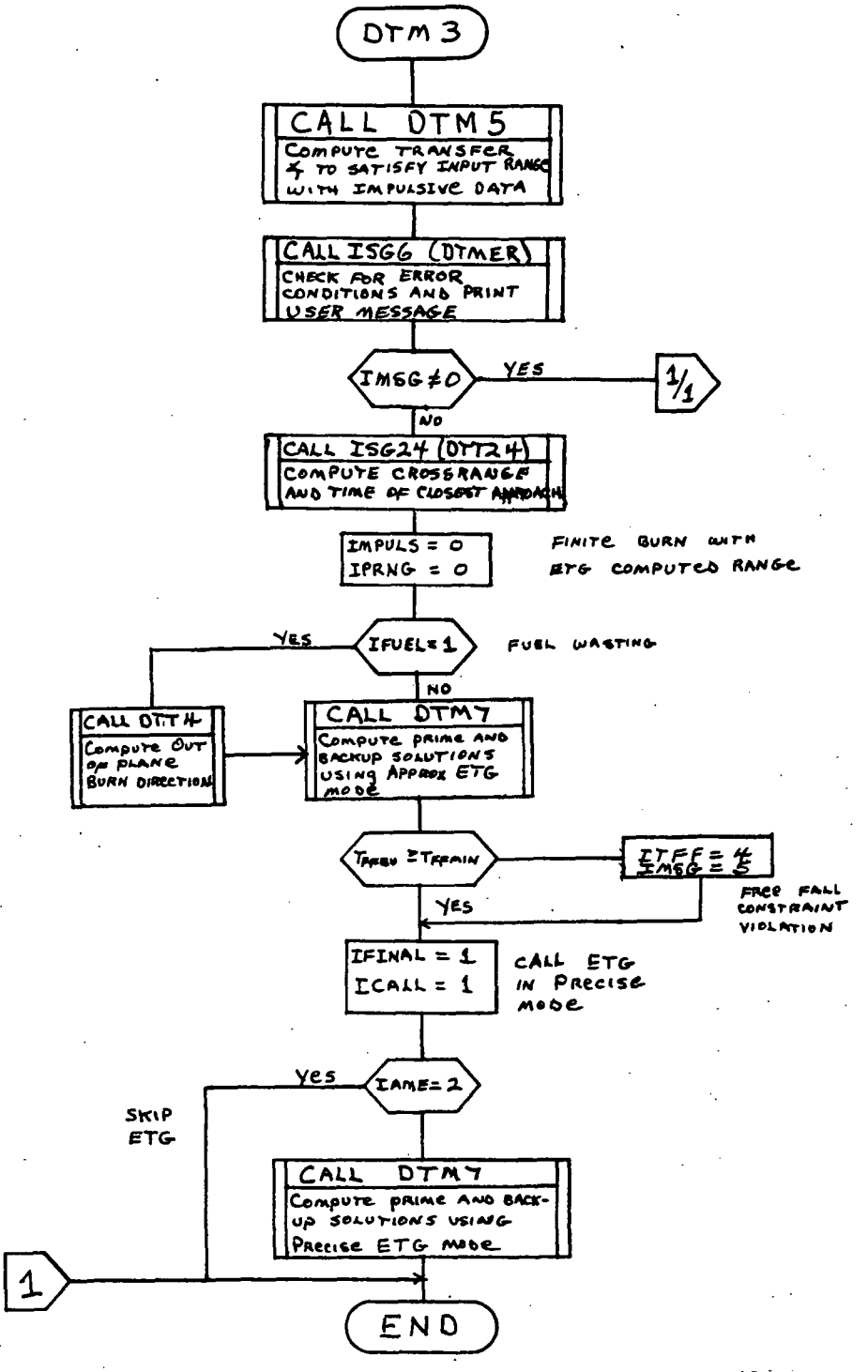


Figure 5.3-1.- Functional logic flow for the DTM3 executive routine.

5.4 ROUTINE NAME - DTM4 EXECUTIVE ROUTINE

5.4.1 Purpose

The DTM4 routine is the executor for the impulsive minimum delta velocity computations within the DTM processor.

5.4.2 Functional Description

The DTM4 routine sets up calls to the optimization routine (DTT2) and the PEG basic targeting routine (DTM5) until a minimum delta-V is found impulsively.

5.4.3 Assumptions and Limitations

None.

5.4.4 Method

An optimization/minimization routine (DTT2) is called to compute a central angle of travel from ignition time to the entry interface point. This angle is then used as a target for the PEG guidance to compute an impulsive maneuver which will result in that central angle of travel. This entire process is then repeated until the delta-V computed is found to be a minimum.

5.4.5 Routine Input/Output Variables

Table 5.4-I contains the definitions of all the input/output variables for the DTM4 executive routine.

5.4.6 Functional Logic Flow

Figure 5.4-1 contains the functional logic flow for the DTM4 executive routine.

5.4.7 Diagnostics and Debug

None.

5.4.8 Special Comments

None.

5.4.9 References

None.

TABLE 5.4-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTM4

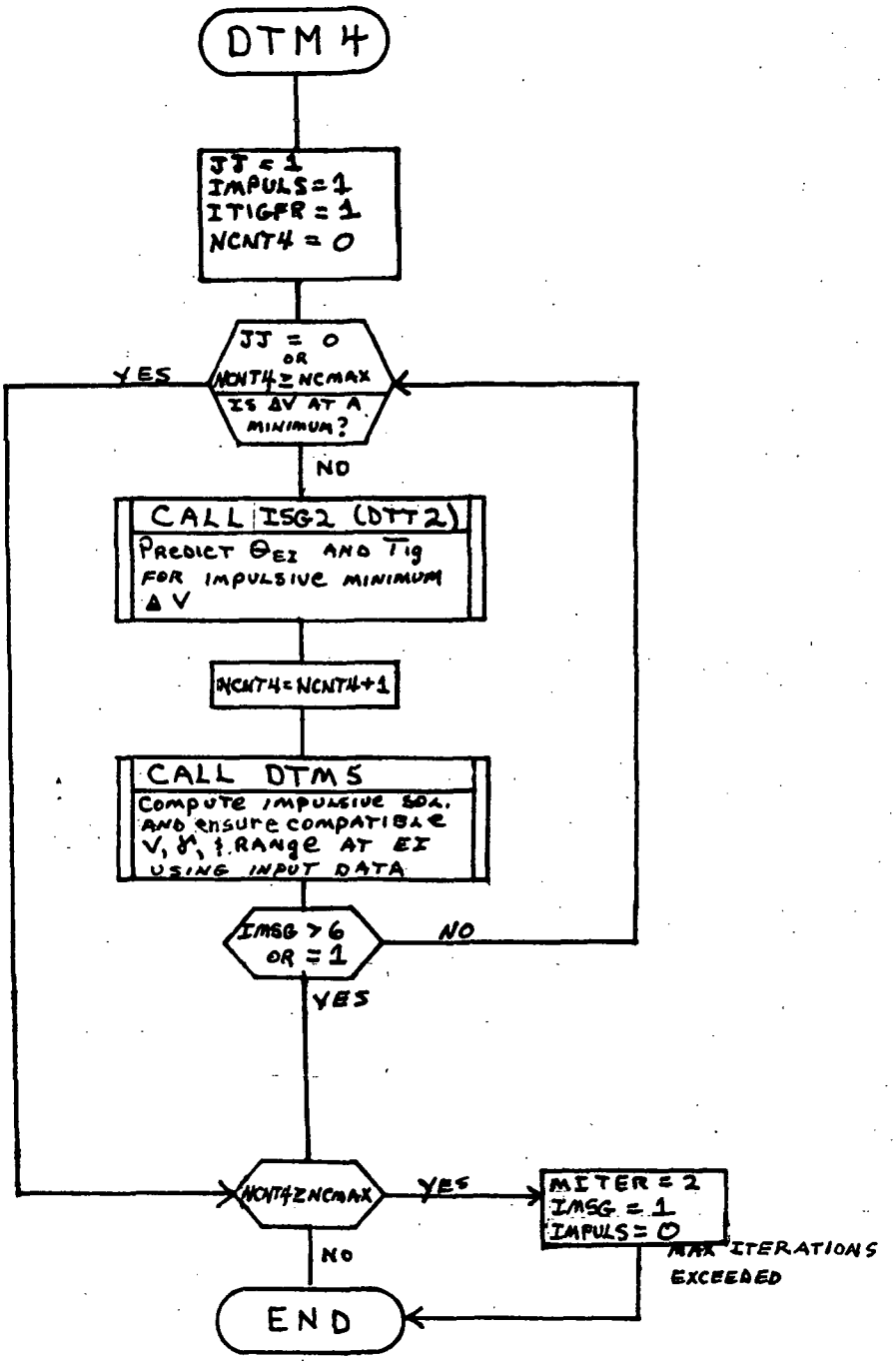
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DTMCON		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
IMPULS		Intg	O		C		
IMSG		Intg	O		C		
IPLACE		Intg	O		C		
ITIGFR		Intg	O		C		
JJ		Intg	O		C		
NCNT4		Intg	O		C		
MITER		Intg	O		C		
NCMAX		Intg	I		C	DTMCON(38)	

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory



Page 1 of 1

Figure 5.4-1.- Functional logic flow for the DTM4 executive routine.

5.5 ROUTINE NAME - DTM5 EXECUTIVE ROUTINE

5.5.1 Purpose

The DTM5 routine is the executor for the basic targeting to ensure a compatible velocity, flightpath angle, and range at entry interface.

5.5.2 Functional Description

The DTM5 routine does all necessary calls and logical routing to the computational routines in order to compute the actual PEG targeted deorbit burn. This routine is the executive for the DTM targeting and as such is the heart of the DTM computational capability. It sets up calls to almost all of the computational segments and routines within DTM including the entry guidance, PEG guidance, and coasting flight propagator.

5.5.3 Assumptions and Limitations

The DTM5 targeting is a cyclic process in which the maximum allowable number of passes is set by an input in the DTMCON array in the interface (NCMAX). A value of from 10 to 20 passes is usually sufficient unless the range tolerance (RNGTOL) is set too small or unless a gross input error has been made in the initial vector. The indication of such errors will be a MITER=3 error message.

5.5.4 Method

The DTM5 routine uses the coasting propagator and the PEG guidance to predict a set of entry interface conditions after a finite deorbit burn. Then the entry guidance range to target routine and the entry target generator (simulated with an input target line) are used to obtain the actual and desired range from entry interface to landing. If the actual and desired range are not within a small tolerance of each other then an adjustment is made to null the range error. For Tig free solutions the ignition time is adjusted and for Tig fixed solutions the target central angle of travel from Tig to entry interface is adjusted. The logic then recycles to obtain new solutions until the range tolerance is met or the maximum allowable cycles is reached.

5.5.5 Routine Input/Output Variables

Table 5.5-I contains the definition of all the input/output variables for the DTM5 executive routine.

5.5.6 Functional Logic Flow

Figure 5.5-1 contains the functional logic flow for the DTM5 executive routine.

5.5.7 Diagnostics and Debug

A printout of the actual and desired ranges and the range tolerance is produced when the user selects debug print.

5.5.8 Special Comments

None.

5.5.9 References

None.

TABLE 5.5-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTM5

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DTMCON		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
IOUNIT		Intg	I		C		
IMPULS		Intg	I		C		
IPOUT		6CH	I		C		
ITIGFR		Intg	I		C		
RNGTOL		Real	I/O		C,T	DTMCON(5)	
THETLS		Real	I/O		C,T		
THELSD		Real	I/O		C,T		
IMSG		Intg	O		C		
IPLACE		Intg	O		C		
MITER		Intg	O		C		
NCNT2		Intg	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.5-I.- Concluded

Routine DTM5

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
NSYS		Intg	O		C		
NCMAX		Intg	I		C	DTMCON(38)	

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory

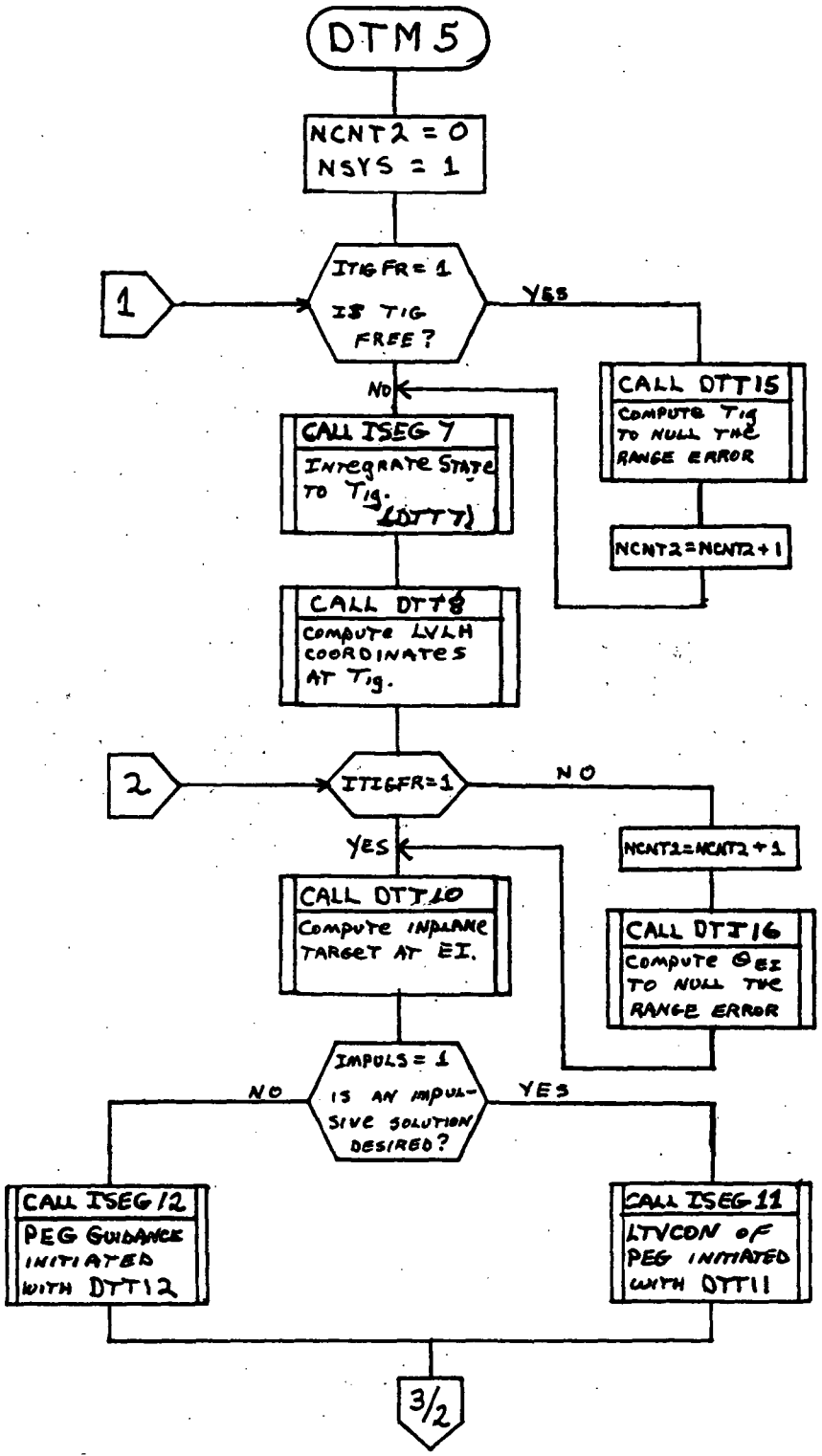


Figure 5.5-1.- Functional logic flow for the DTM5 executive routine.

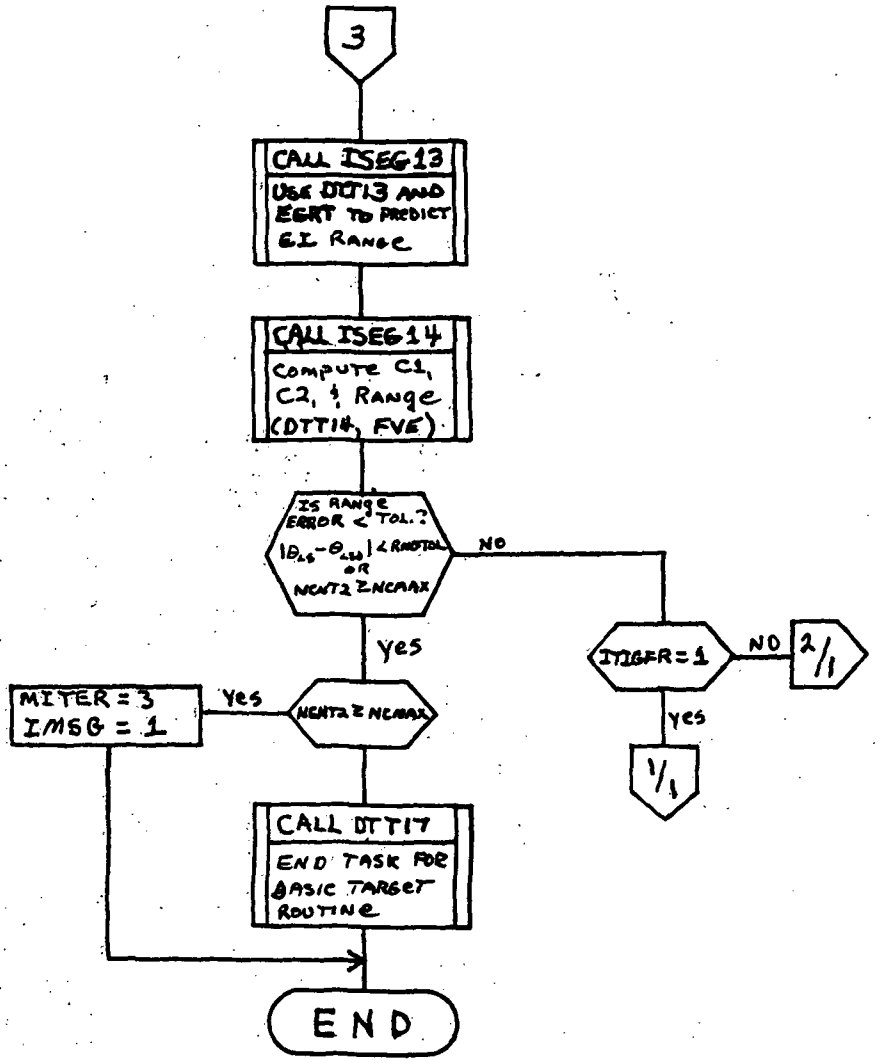


Figure 5.5-1.- Concluded.

5.6 ROUTINE NAME - DTM6 EXECUTIVE ROUTINE

5.6.1 Purpose

The DTM6 routine is the executor for the equal yaw angle computations within the DTM processor.

5.6.2 Functional Description

The DTM6 routine is used for Tig free fuel wasting solutions. It adjusts the ignition time until a Tig is found for which the out-of-plane yaw angles are within a small tolerance of each other.

5.6.3 Assumptions and Limitations

There will be no equal yaw angle or equal OMS solution for the prime and backup systems unless the engine characteristics (thrust and flow rate) are such that the characteristic curves of OMS propellant required versus ignition time have an intersection. For the Shuttle vehicle this means that RCS combined with OMS in any configuration of prime and backup will not produce an equal yaw angle or equal OMS propellant solution.

5.6.4 Method

The DTM6 routine calls the Tig adjustment routine (DTT5) to compute a Tig to produce equal yaw angles. The basic targeting routine (DTM5) is then called to produce a prime and backup system solution at that Tig. The resultant yaw angles are tested and the logic cycles until the yaw angles are within a small tolerance of each other or until the maximum cycle has been reached.

5.6.5 Routine Input/Output Variables

Table 5.6-I contains the definition of all the input/output variables for the DTM6 executive routine.

5.6.6 Functional Logic Flow

Figure 5.6-1 contains the functional logic flow for the DTM6 executive routine.

5.6.7 Diagnostics and Debug

None.

5.6.8 Special Comments

None.

5.6.9 References

None.

TABLE 5.6-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTM6

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DSITOL		Real	I		C	DTMCON(11)	Refer to table 5.1-I for all code symbol definitions.
TGDSI		Real	I		C	DTMCON(34)	
TGSLPV		Real	I		C	DTMCON(37)	
PSIBU		Real	I		C		
PSIPR		Real	I		C		
NCMAX		Intg	I		C	DTMCON(38)	
DV2		Real	O		C		
IMSG		Intg	O		C		
IPLACE		Intg	O		C		
ITIGG		Intg	O		C		
NCNT		Intg	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.6-I.- Concluded

Routine DTM6

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
MITER		Intg	O		C		
TGDMN		Real	O		C		
TGSLPP		Real	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

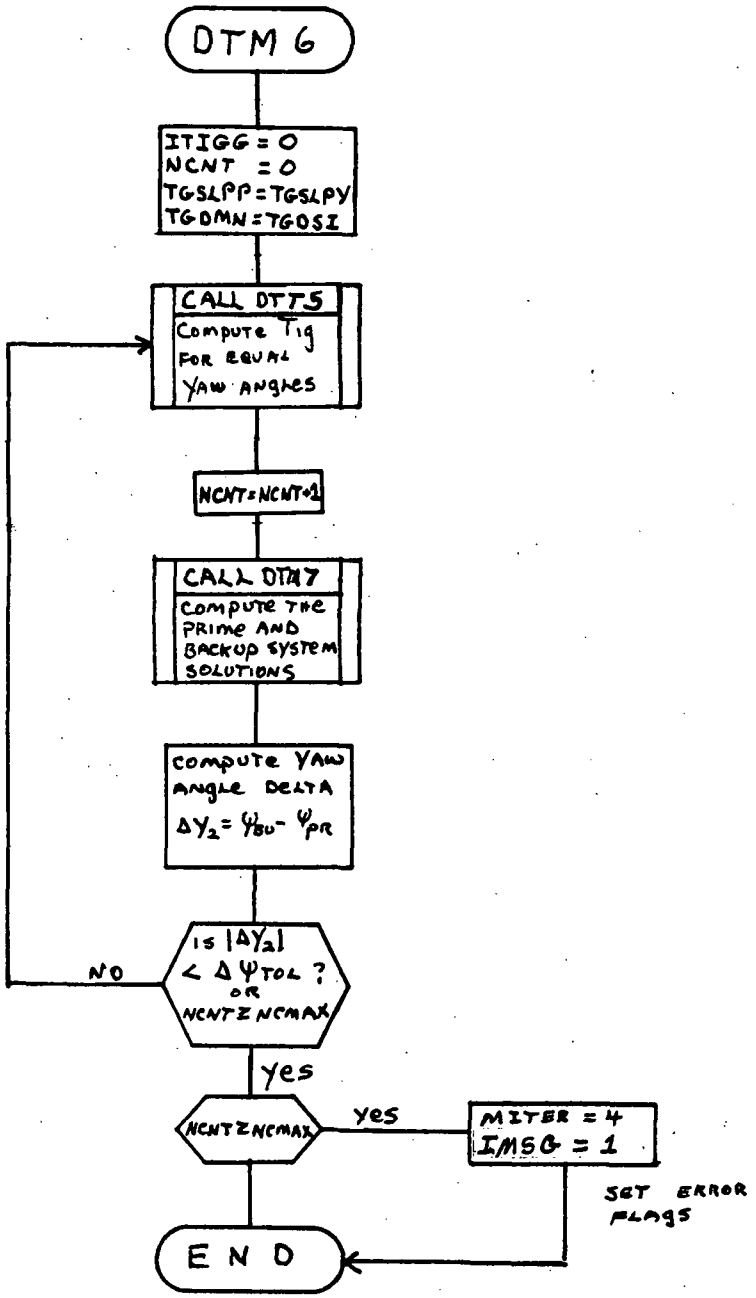


Figure 5.6-1.- Functional logic flow for the DTM6 executive routine.

5.7 ROUTINE NAME - DTM7 EXECUTIVE ROUTINE

5.7.1 Purpose

The DTM7 routine is the executor for the prime and backup thruster system solution computation within the DTM processor.

5.7.2 Functional Description

The DTM7 routine uses the basic targeting routine (DTM5) to compute the primary system solution. The PEG guidance routines (DTT12) are then used directly to calculate the backup system solution. The cross range and entry range for each solution are also computed by calls to DTT24 and DTT13.

5.7.3 Assumptions and Limitations

None.

5.7.4 Method

None.

5.7.5 Routine Input/Output Variables

Table 5.7-I contains the definitions of all the input/output variables for the DTM7 executive routine.

5.7.6 Functional Logic Flow

Figure 5.7-1 contains the functional logic flow for the DTM7 executive routine.

5.7.7 Diagnostics and Debug

None.

5.7.8 Special Comments

None.

5.7.9 References

None.

TABLE 5.7-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTM7

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IPLACE		Intg	O		C		Refer to table 5.1-I for all code symbol definitions.

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory

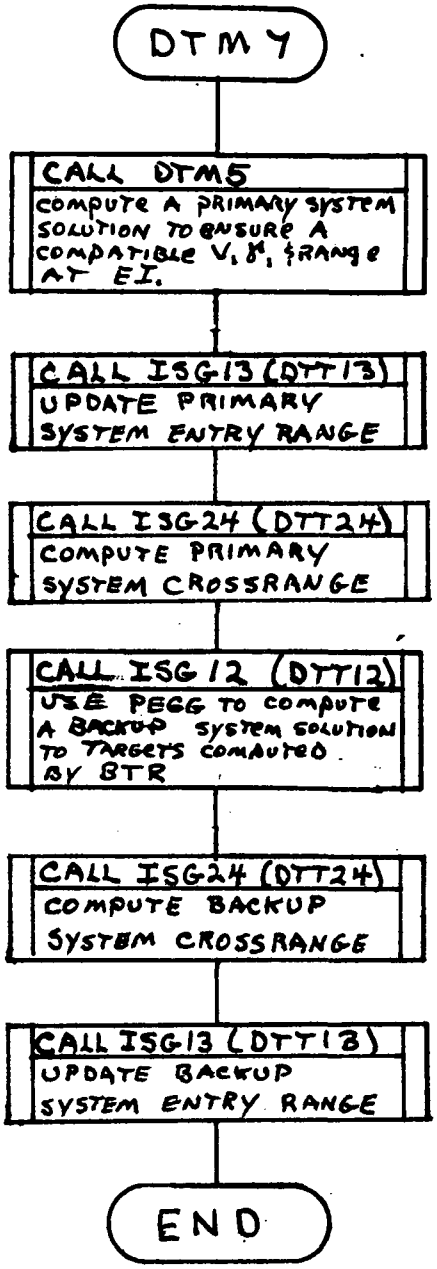


Figure 5.7-1.- Functional logic flow for the DTM7 executive routine.

5.8 ROUTINE NAME - DTM8 EXECUTIVE ROUTINE

5.8.1 Purpose

The DTM8 routine is the executor for the equal OMS computation within the DTM processor.

5.8.2 Functional Description

The DTM8 routine is used for Tig FREE, inplane solutions. It calls the Tig adjustment routine (DTT5) to compute a Tig from which equal OMS will be expended by the prime and backup systems. The PEG targeting is then used to compute maneuvers at that Tig and the actual OMS usage is obtained. The prime and backup system OMS usages are compared and the logic cycles until they are within a small tolerance of each other.

5.8.3 Assumptions and Limitations

Equal OMS usage cannot be achieved unless the characteristic curves of propellant weight used versus ignition time have an intersection. For the Shuttle this means that the RCS and OMS system are not a valid prime/backup system combination.

5.8.4 Method

None.

5.8.5 Routine Input/Output Variables

Table 5.8-I contains the definitions of all the input/output variables for the DTM8 executive routine.

5.8.6 Functional Logic Flow

Figure 5.8-1 contains the functional logic flow for the DTM8 executive routine.

5.8.7 Diagnostics and Debug

None.

5.8.8 Special Comments

None.

5.8.9 References

None.

TABLE 5.8-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTM8

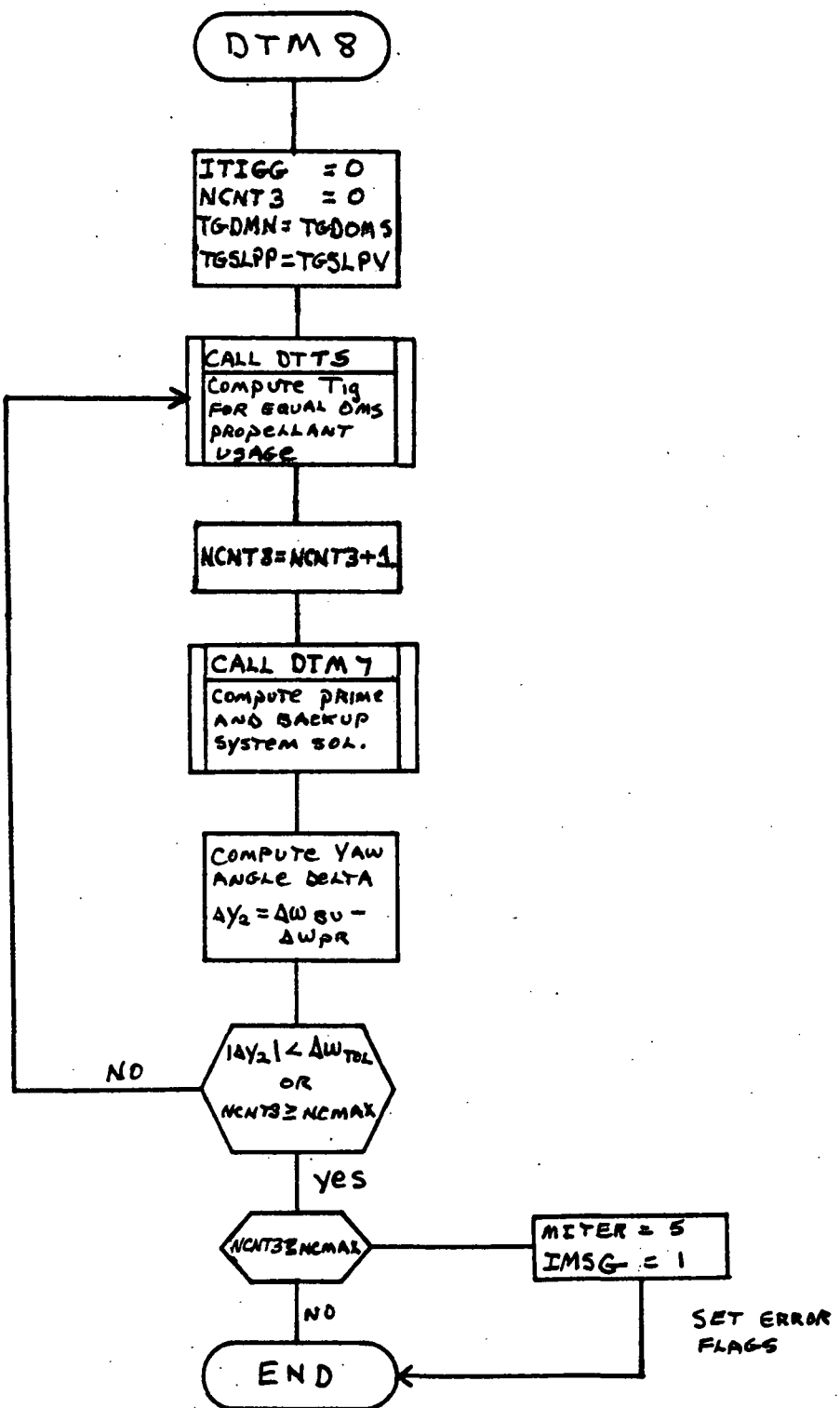
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DWBU		Real	I		C		Refer to table 5.1-I for code symbol definitions.
DWPR		Real	I		C		
DWTOL		Real	I		C	DTMCON(10)	
NCMAX		Intg	I		C	DTMCON(38)	
TGDOMS		Real	I		C	DTMCON(33)	
TGSLPV		Real	I		C	DTMCON(36)	
DV2		Real	O		C		
IMSG		Intg	O		C		
IPLACE		Intg	O		C		
ITIGG		Intg	O		C		
NCNT3		Intg	O		C		
MITER		Intg	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.8-I.- Concluded

Routine DTM8

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TGDMN		Real	O		C		
TGSLPP		Real	O		C		

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory



Page 1 of 1

Figure 5.8-1.- Functional logic flow for the DTM8 executive routine.

5.9 ROUTINE NAME - DTM9 EXECUTIVE ROUTINE

5.9.1 Purpose

The DTM9 routine is the executor for the ignition time computation to increase the backup system freefall time within the DTM processor.

5.9.2 Functional Description

The DTM9 routine calls a Tig adjustment routine (DTT18) to compute a Tig time that will increase the backup system freefall time from burnout to entry interface. Prime and backup solutions are then recomputed by calling DTM7 and the actual freefall times obtained. The logic cycles until the backup system freefall times is greater than the minimum allowed freefall time (Tffmin) or until the maximum number of cycles has been exceeded.

5.9.3 Assumptions and Limitations

If the adjustment of Tig to meet the Tffmin constraint results in OMS usage above the input WCGOMS amount, then no solution is possible and processing will stop.

5.9.4 Method

None.

5.9.5 Routine Input/Output Variables

Table 5.9-I contains the definitions of all the input/output variables for the DTM9 executive routine.

5.9.6 Functional Logic Flow

Figure 5.9-1 contains the functional logic flow for the DTM9 executive routine.

5.9.7 Diagnostics and Debug

None.

5.9.8 Special Comments

None.

5.9.9 References

None.

TABLE 5.9-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTM9

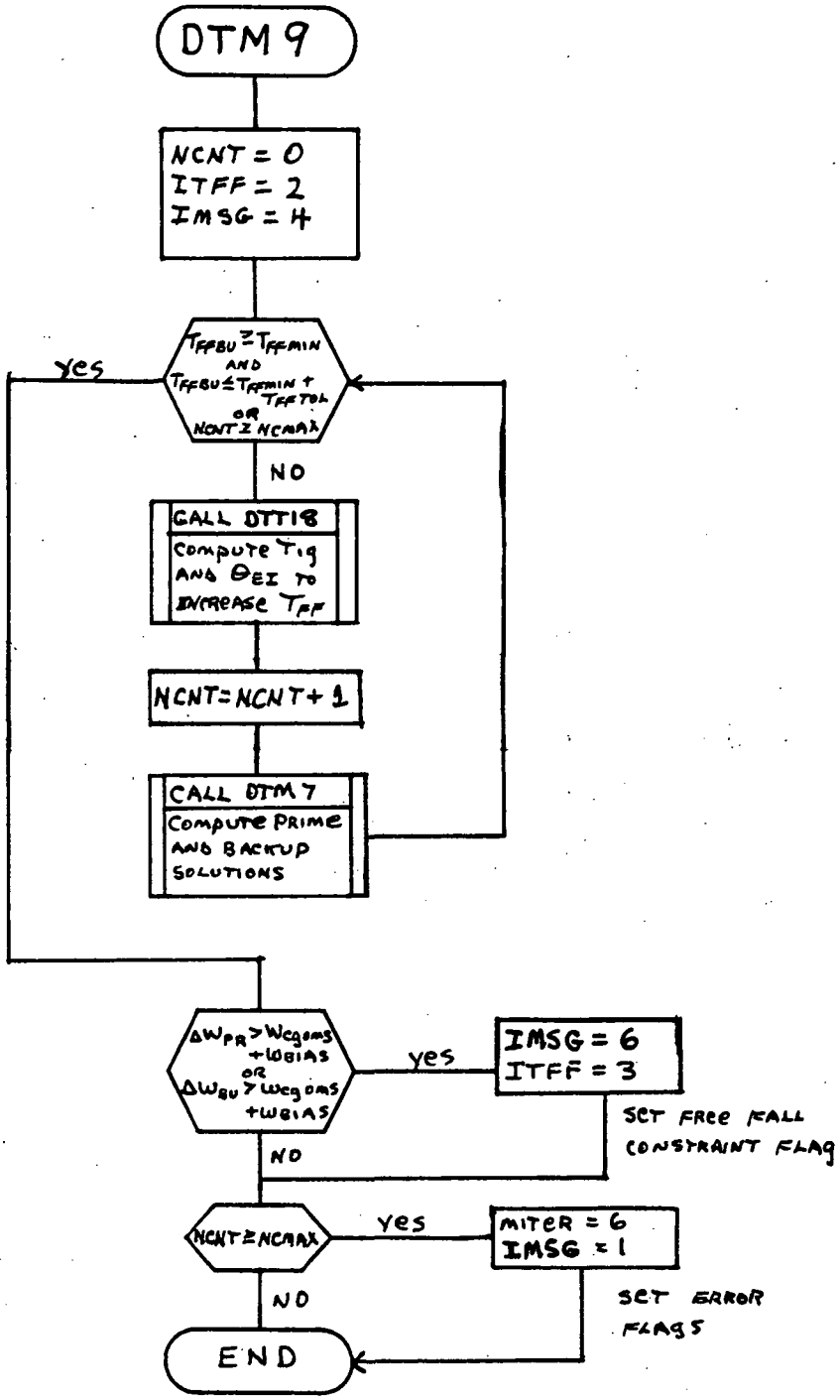
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DWBU		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
DWPR		Real	I		C		
TFFBU		Real	I		C		
TFFMIN		Real	I		C		
TFFTOL		Real	I		C	DTMCON(6)	
WBIAS		Real	I		C	DTMCON(27)	
WCGOMS		Real	I		C		
IMSG		Intg	O		C		
IPLACE		Intg	O		C		
ITFF		Intg	O		C		
NCNT		Intg	O		C		
MITER		Intg	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.9-I.- Concluded

Routine DTM9

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
NCMAX		Intg	I		C	DTMCON(38)	

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory



Page 1 of 1

Figure 5.9-1.- Functional logic flow for the DTM9 executive routine.

5.10 ROUTINE NAME - DTM14 EXECUTIVE ROUTINE

5.10.1 Purpose

The DTM14 routine is the executor for the offset targeting computation within the DTM processor.

5.10.2 Functional Description

DTM14 computes a set of offset targets that include the use of J2 effects during the burn. These targets, when used for an onboard conic solution, will produce a solution that includes the higher order gravity effects.

5.10.3 Assumptions and Limitations

None.

5.10.4 Method

The conic targeted burnout vector from the DTM solution is adjusted to include J2 effects by the SURRJ routine. This vector is propagated to entry interface and the entry conditions compared to the original solution. The entry ranges are compared and the difference is used to adjust the PEG target angle of travel from Tig to entry interface. The entry velocities and flightpath angles are compared and used to adjust the C1 target line constant. The PEG guidance (DTT12) is then used to recompute the burn solution to the new targets and the logic cycles until the computed entry interface conditions match the original DTM solution.

5.10.5 Routine Input/Output Variables

Table 5.10-I contains the definition of all the input/output variables for the DTM14 executive routine.

5.10.6 Functional Logic Flow

Figure 5.10-1 contains the functional logic flow for the DTM14 executive routine.

5.10.7 Diagnostics and Debug

None.

5.10.8 Special Comments

None.

5.10.9 References

None.

TABLE 5.10-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTM14

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
CA		Real	I/O		C		Refer to table 5.1-I for all code symbol definitions.
DC11		Real	I		C		
DGAM		Real	I		C		
DTHELS		Real	I/O		C		
RGRAV		Real	I		C		
RNGTOL		Real	I		C	DTMCON(5)	
GAMTOL		Real	I		C	DTMCON(12)	
NCMAX		Intg	I		C	DTMCON(38)	
RC1		Real	I		C		
RDA		Real	I/O		C		
TGO		Real	I		C		
THETEI		Real	I/O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.10-I.- Continued

Routine DTM14

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
THETLS		Real	I		C		
THELSD		Real	I		C		
VC1		Real	I		C		
VDA		Real	I/O		C		
VGRAV		Real	I		C		
IFFPTM		Intg	O		C		
IMSG		Intg	O		C		
IPLACE		Intg	O		C		
NCNT		Intg	O		C		
NSYS		Intg	O		C		
MITER		Intg	O		C		
RC2		Real	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.10-I.- Concluded

Routine DTM14

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
VC2		Real	O		C		

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory

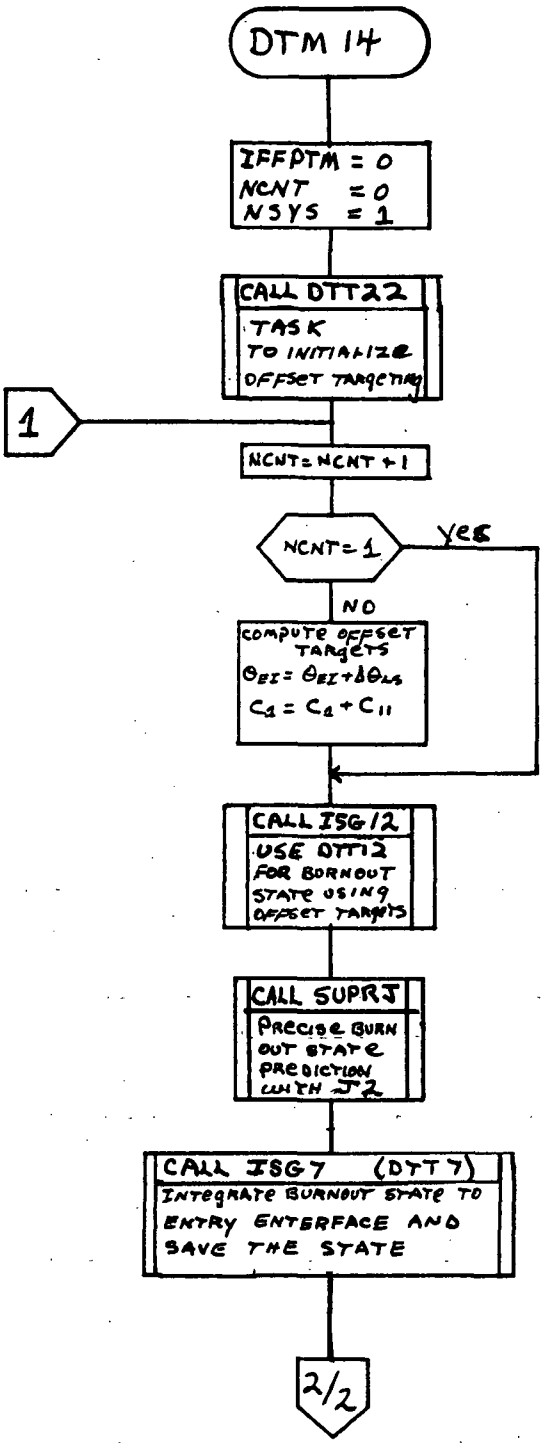


Figure 5.10-1.- Functional logic flow for the DTM14 executive routine.

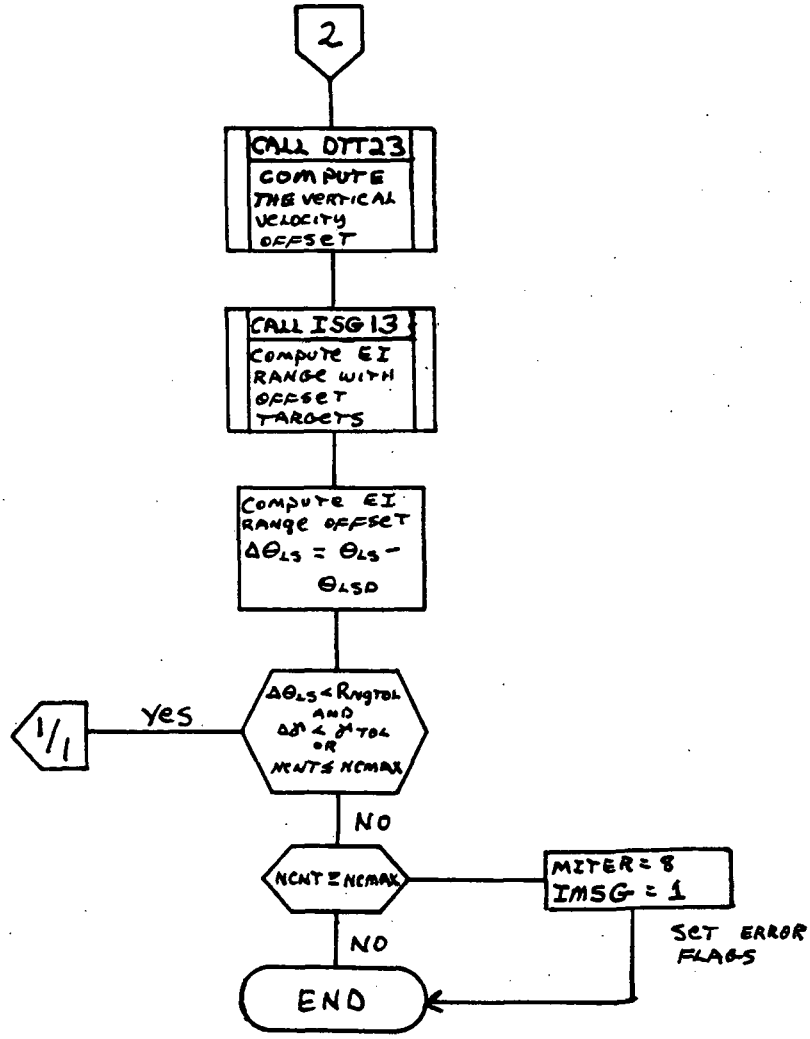


Figure 5.10-1.- Concluded

5.11 ROUTINE NAME - DTT3 COMPUTATIONAL ROUTINE

5.11.1 Purpose

The DTT3 routine adjusts the ignition time and central angle of travel from Tig to entry interface to approximate a finite burn.

5.11.2 Functional Description

DTT3 computes the burn time for a primary system maneuver of the input delta-V magnitude. One half of the burn time is then subtracted from the current Tig time for a new Tig. The product of one half of the burn time and the orbital rate is added to the current central angle of travel from Tig to EI for a new central angle.

5.11.3 Assumptions and Limitations

The finite burn is an approximation with the midpoint of the burn located at the impulsive maneuver time.

5.11.4 Method

None.

5.11.5 Routine Input/Output Variables

Table 5.11-I contains the definition of all the input/output variables for the DTT3 computational routine.

5.11.6 Functional Logic Flow

Figure 5.11-1 contains the functional logic flow for the DTT3 computational routine.

5.11.7 Diagnostics and Debug

None.

5.11.8 Special Comments

None.

5.11.9 References

None.

TABLE 5.11-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT3

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DVIMP		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
FPR		Real	I		C		
STP		Real	I		C	GLOCON(117)	
TIG		Real	I/O		C		
THETD		Real	I		C		
THETEI		Real	I/O		C		
WDPR		Real	I		C		
WT		Real	I		C	DTMVEC(13)	
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

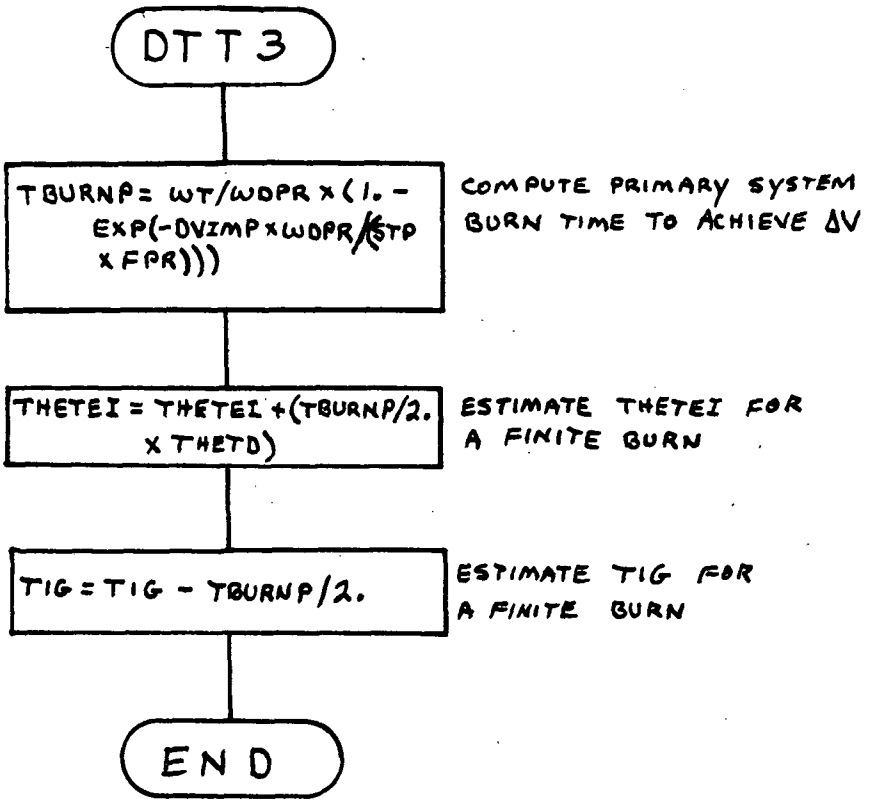


Figure 5.11-1.- Functional logic flow for the DTT3 computational routine.

5.12 ROUTINE NAME - DTT4 COMPUTATIONAL ROUTINE

5.12.1 Purpose

The DTT4 routine determines the direction in which out-of-plane delta-V should be expended in order to reduce the cross range at the landing site.

5.12.2 Functional Description

In the fuel wasting mode the excess fuel available is expended out-of-plane in a direction such that the cross range is minimized. DTT4 determines the direction of the out-of-plane burn (SBD) as being plus when the out-of-plane delta-V should be in the direction of the angular momentum vector, or negative when the out-of-plane delta-V should be applied opposite to the angular momentum vector. The determination of this direction is based upon the landing site location at entry interface relative to the trajectory plane at ignition. The direction is computed such that the trajectory plane will be shifted toward the landing site.

5.12.3 Assumptions and Limitations

None.

5.12.4 Method

None.

5.12.5 Routine Input/Output Variables

Table 5.12-I contains the definitions of all the input/output variables for the DTT4 computational routine.

5.12.6 Functional Logic Flow

Figure 5.12-1 contains the functional logic flow for the DTT4 computational routine.

5.12.7 Diagnostics and Debug

When debug printout is requested by the user the DTT4 routine will output the sign of the burn direction (SBD), the down range unit vector (UX), the landing site unit vector (ULS), the burn time of the maneuver (TBURN), and the value of one half the burn arc (THHAFB).

5.12.8 Special Comments

None.

5.12.9 References

None.

TABLE 5.12-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT4

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DVOBPR		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
ERAI		Real	I		C		
FPR		Real	I		C		
IOUNIT		Intg	I		C		
IPOUT		6CH	I		C		
STP		Real	I		C	GLOCON(117)	
TC		Real	I		C		
TLATC		Real	I		C		
TLONG		Real	I		C		
THETD		Real	I		C		
UXDTM		Real	I/O		C,T		
UYDTM		Real	I		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.12-I.- Concluded

Routine DTT4

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
UZDTM		Real	I		C		
WE		Real	I		C	GLOCON(13)	
WDPR		Real	I		C		
WT		Real	I		C	DTMVEC(13)	
SBD		Real	O		C,T		
ULS		Real	O		T		Landing site unit vector at TC
TBURN		Real	O		T		Time of burn
THHAFB		Real	O		T		Time of half-burn arc
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

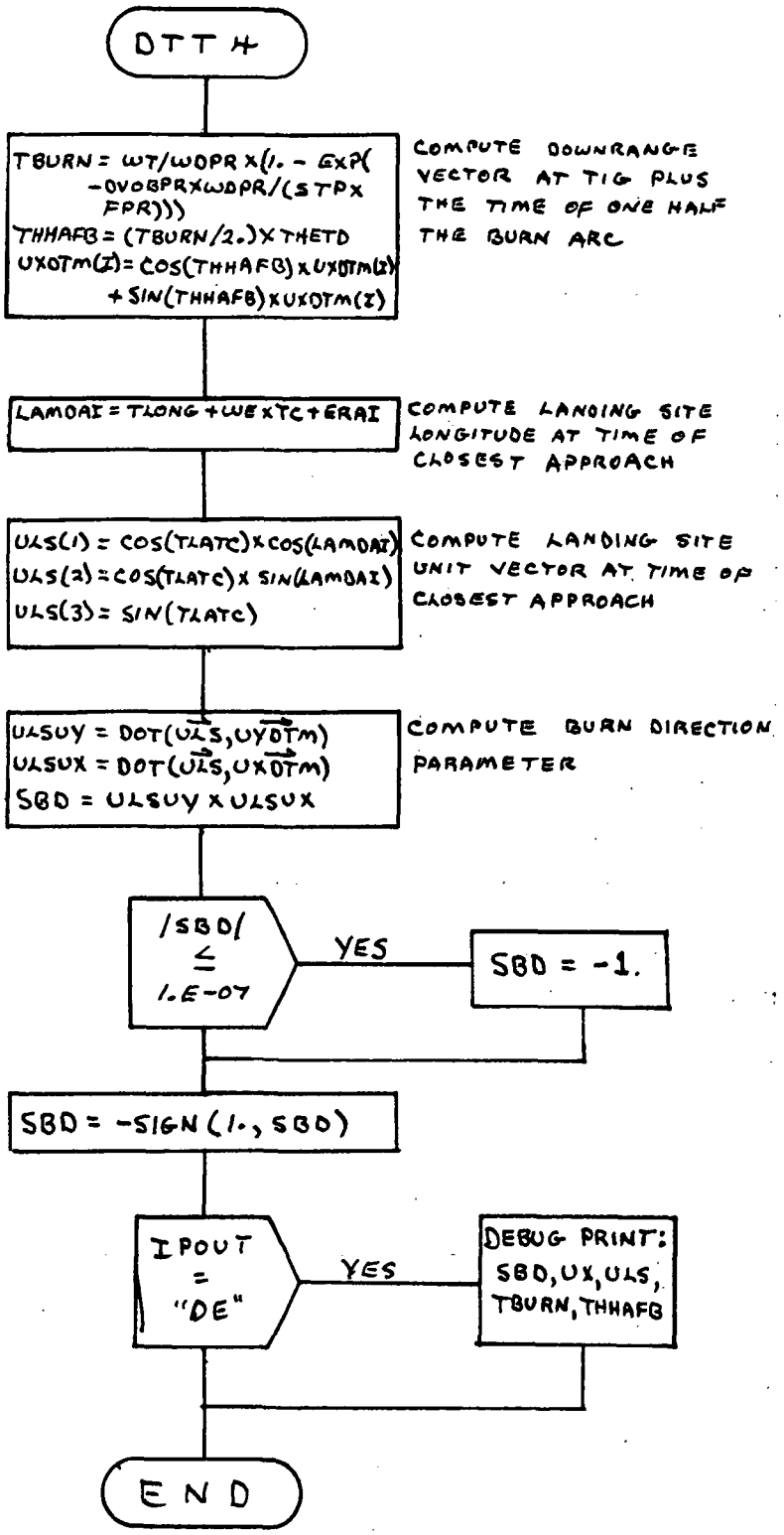


Figure 5.12-1.- Functional logic flow for the DTT4 computational routine.

5.13 ROUTINE NAME - DTT5 COMPUTATIONAL ROUTINE

5.13.1 Purpose

The DTT5 routine is used to compute an ignition time adjustment such that equal yaw angles or equal OMS usage can be achieved between the primary and backup systems in the Tig free mode.

5.13.2 Functional Description

The DTT5 routine computes a Tig adjustment and adjusts the Tig time and the central angle of travel from Tig to entry interface. The Tig adjustment is computed such that the difference between the primary and backup system out-of-plane yaw angles will be driven toward zero in the Tig free fuel wasting mode or the difference between the prime and backup system OMS usage will be driven toward zero in the Tig free inplane mode. The routine is called repeatedly until the yaw angle or OMS differences are within a small input tolerance of each other.

5.13.3 Assumptions and Limitations

None.

5.13.4 Method

The DTT5 uses a linear interpolation method to compute the Tig adjustment needed. The adjustment is based upon the current and one pass previous values of the Tig time and the corresponding effect that the Tig times have on the yaw angles or on the OMS usage.

5.13.5 Routine Input/Output Variables

Table 5.13-I contains the definitions of all the input/output variables for the DTT5 computational routine.

5.13.6 Functional Logic Flow

Figure 5.13-1 contains the functional logic flow for the DTT5 computational routine.

5.13.7 Diagnostics and Debug

None.

5.13.8 Special Comments

None.

5.13.9 References

None.

TABLE 5.13-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT5

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DTGMAX		Real	I		C	DTMCON(8)	Refer to table 5.1-I for all code symbol definitions.
DTIGO		Real	I		C	DTMCON(7)	
DV1		Real	I/O		C		
DV2		Real	I		C		
ITIGG		Intg	I/O		C		
THETD		Real	I		C		
THETEI		Real	I/O		C		
TGDMN		Real	I		C		
TGSLPP		Real	I/O		C		
TIG		Real	I/O		C		
TIG1		Real	I/O		C		
ITIGFR		Intg	0		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

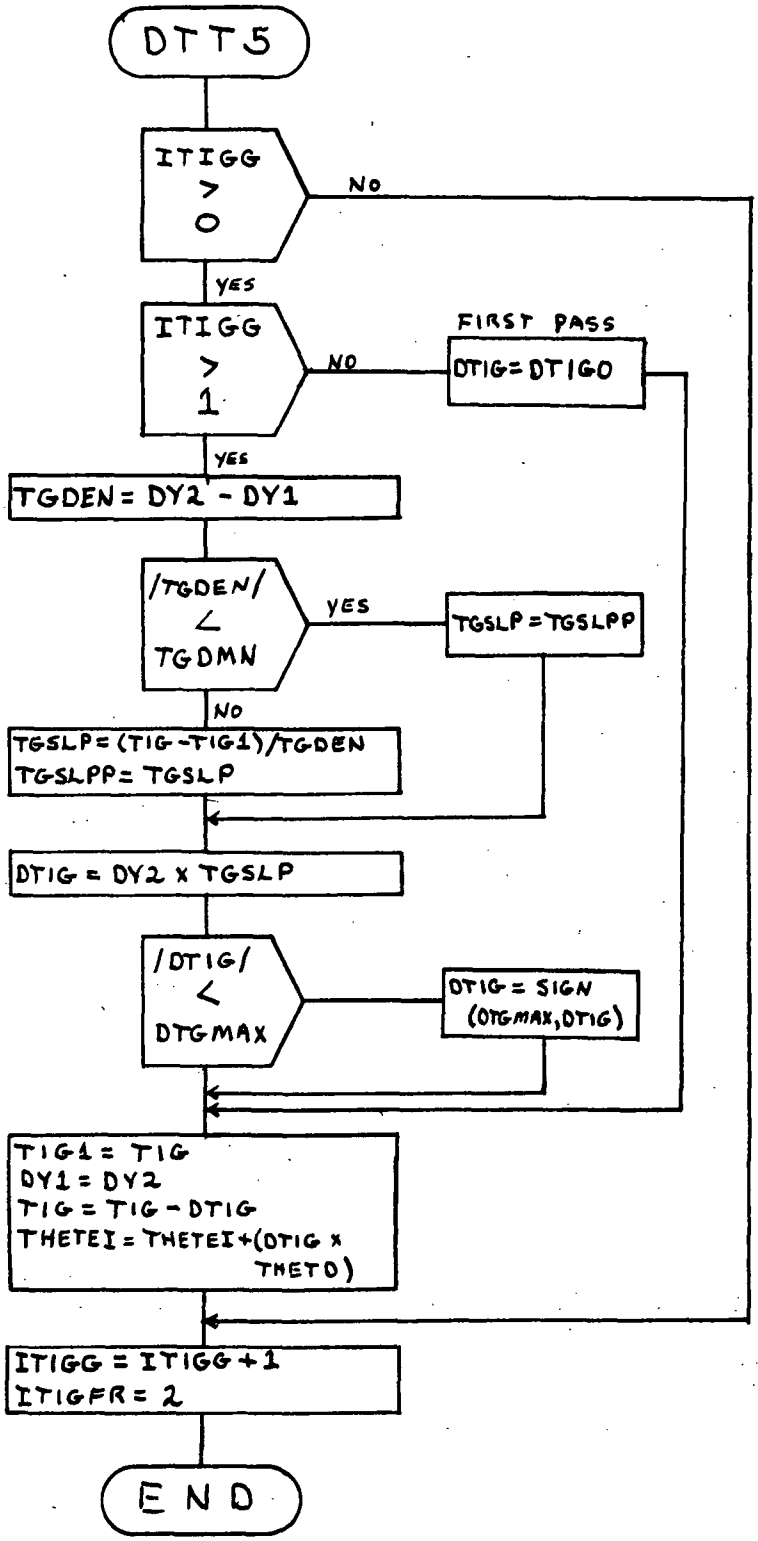


Figure 5.13-1.- Functional logic flow for the DTT5 computational routine.

5.14 ROUTINE NAME - DTT8 COMPUTATIONAL ROUTINE

5.14.1 Purpose

The DTT8 routine computes a local vertical reference frame at the time of deorbit ignition.

5.14.2 Functional Description

A local vertical frame is computed with the Z-axis down along the radius vector at Tig, the Y-axis out-of-plane in the direction of $\dot{\bar{V}}x\dot{\bar{R}}$ and the X-axis pointing down range.

5.14.3 Assumptions and Limitations

None.

5.14.4 Method

None.

5.14.5 Routine Input/Output Variables

Table 5.14-I contains the definitions of all the input/output variables for the DTT8 computational routine.

5.14.6 Functional Logic Flow

Figure 5.14-1 contains the functional logic flow for the DTT8 computational routine.

5.14.7 Diagnostics and Debug

None.

5.14.8 Special Comments

None.

5.14.9 References

None.

TABLE 5.14-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT8

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
RAAA		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
VA		Real	I		C		
UXDTM		Real	O		C		
UYDTM		Real	O		C		
UZDTM		Real	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

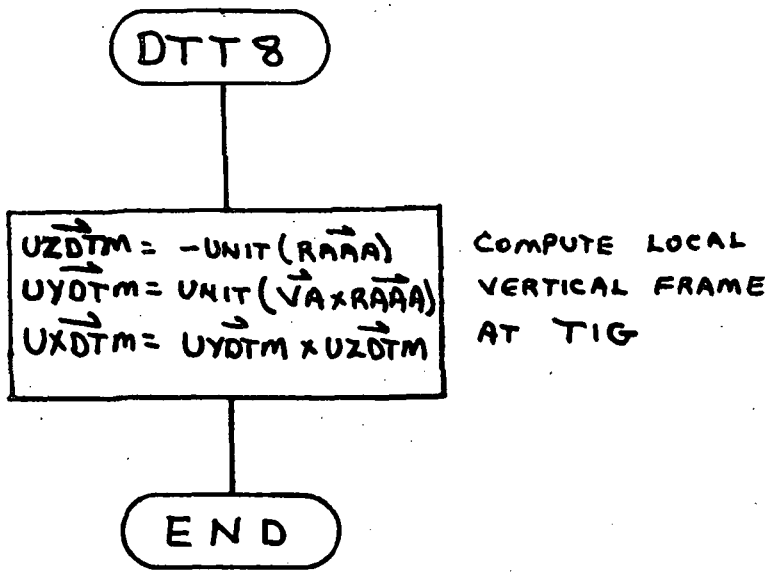


Figure 5.14-1.- Functional logic flow for the DTT8 computational routine.

5.15 ROUTINE NAME - DTT10 COMPUTATIONAL ROUTINE

5.15.1 Purpose

The DTT10 routine computes an inplane inertial position vector at entry interface to be used by DTM as an impulsive solution target.

5.15.2 Functional Description

A unit vector in the direction of entry interface is computed at Tig using the central angle of travel from Tig to EI as the basis. The radius vector at entry interface is then computed from the input entry interface altitude plus the geocentric radius at the entry interface latitude. The geocentric radius is computed from the function HLIP.

5.15.3 Assumptions and Limitations

None.

5.15.4 Method

None.

5.15.5 Routine Input/Output Variables

Table 5.15-I contains the definitions of all the input/output variables for the DTT10 computational routine and for the HLIP function.

5.15.6 Functional Logic Flow

Figure 5.15-1 contains the functional logic flow for the DTT10 computational routine and the HLIP function.

5.15.7 Diagnostics and Debug

When the user selects the debug output option the following variables are printed, UXDTM, UYDTM, THETEI, RTA, UEI, and REIMAG.

5.15.8 Special Comments

None.

5.15.9 References

None.

TABLE 5.15-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT10

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ELLIPT		Real	I		C	GLOCON(27)	Refer to table 5.1-I for all code symbol definitions.
EQRAD		Real	I		C	GLOCON(23)	
EIALT		Real	I		C	GLOCON(57)	
IOUNIT		Intg	I		C		
IPOUT		6CH	I		C		
POLE		Real	I		C	DTMCON(39)	
THETEI		Real	I/O		C,T		
UXDTM		Real	I/O		C,T		
UZDTM		Real	I/O		C,T		
RTA		Real	O		C		
REIMAG		Real	O		T		Radius magnitude at E.I.
UEI		Real	O		G		Unit vector in direction of $\overrightarrow{RTA}$
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.15-I.- Concluded

Routine HLIP

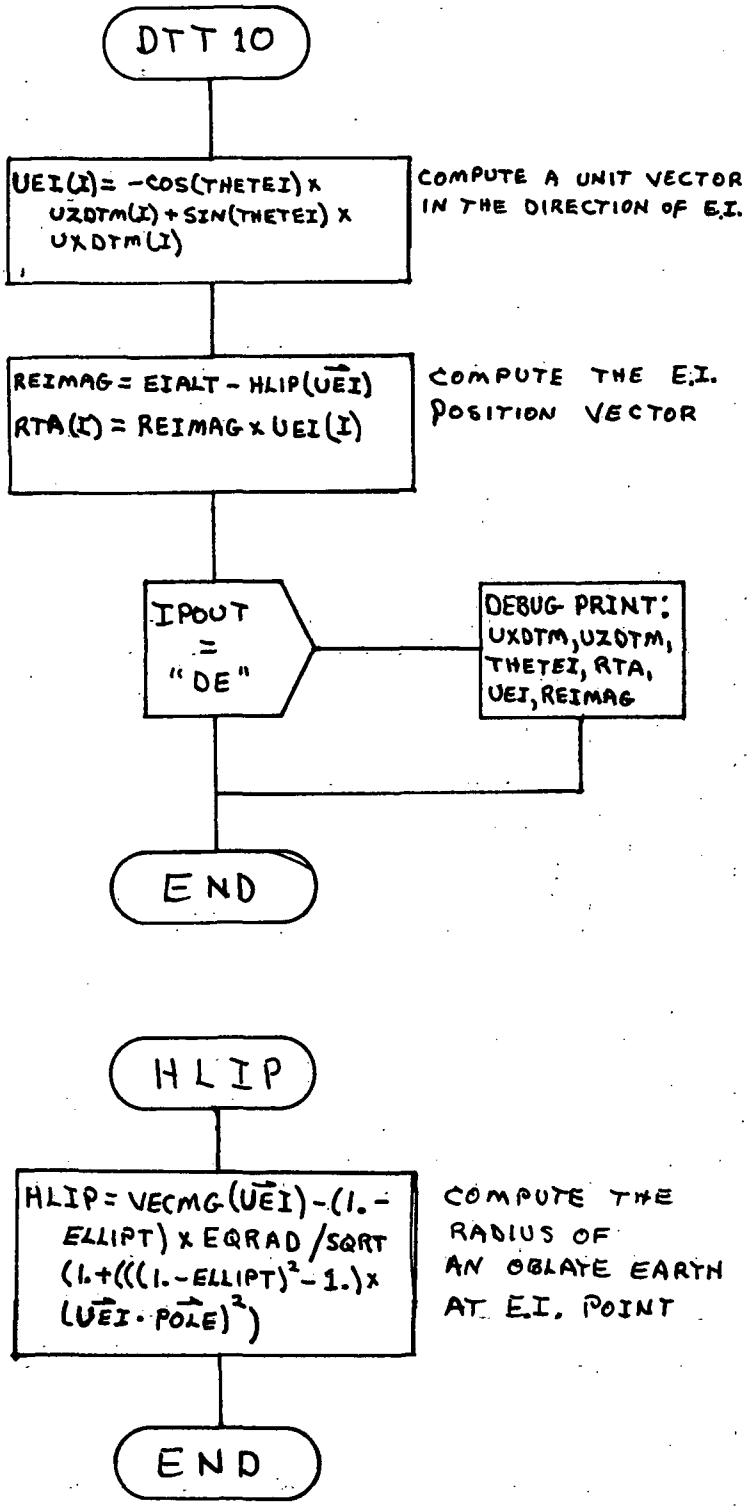
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
R		Real	I		A		ECI position vector
POLE		Real	I		A	DTMCON(39)	
EQRAD		Real	I		A	GLOCON(23)	
ELLIPT		Real	I		A	GLOCON(27)	
HLIP		Real	O		A		Altitude above oblate Earth

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory



Page 1 of 1

Figure 5.15-1.- Functional logic flow for the DTT10 computational routine and the HLIP function.

5.16 ROUTINE NAME - DTT15 COMPUTATIONAL ROUTINE

5.16.1 Purpose

The DTT15 routine adjusts the ignition time to eliminate the difference between the actual and desired range from entry interface to the landing site.

5.16.2 Functional Description

The difference between the actual and the desired range from entry interface to the landing site is divided by the orbital rate and then added to the ignition time. This produces a new Tig time that should null the range differences. This routine is called as required until the range error is within a small input tolerance.

5.16.3 Assumptions and Limitations

None.

5.16.4 Method

None.

5.16.5 Routine Input/Output Variables

Table 5.16-I contains the definitions of all the input/output variables for the DTT15 routine.

5.16.6 Functional Logic Flow

Figure 5.16-1 contains the functional logic flow for the DTT15 computational routine.

5.16.7 Diagnostics and Debug

None.

5.16.8 Special Comments

None.

5.16.9 References

None.

TABLE 5.16-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT15

Code symbol	*Math* symbol	Type	Use	Units	Source	External label	Definition
IBTR		Intg	I/O		C		Refer to table 5.1-I for all code symbol definitions.
THETDR		Real	I		C		
THETLS		Real	I		C		
THELSD		Real	I		C		
TIG		Real	I/O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

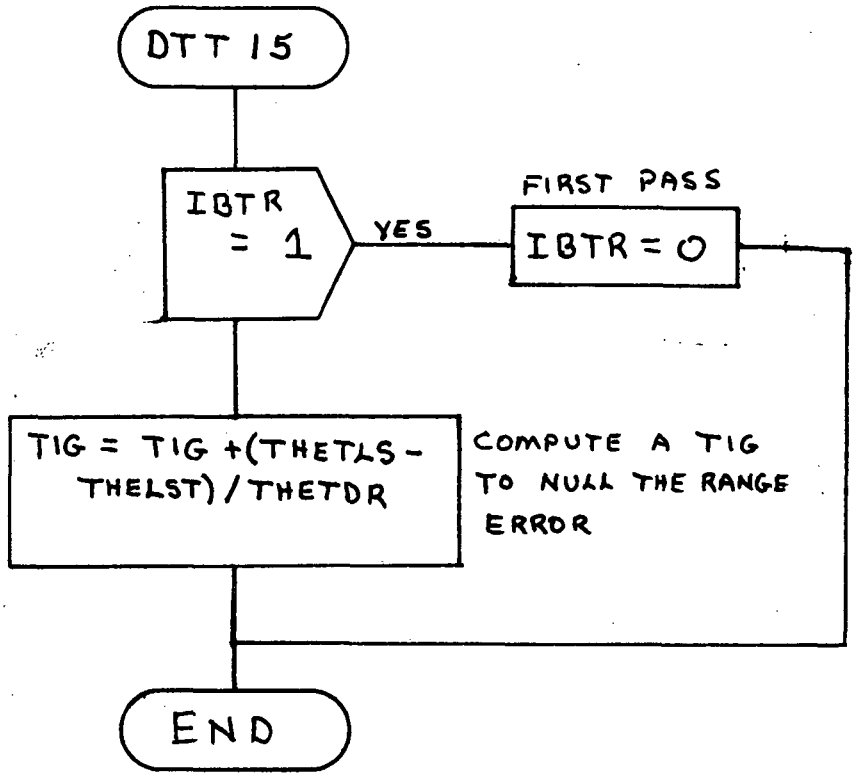


Figure 5.16-1.- Functional logic flow for the DTT15 computational routine.

5.17 ROUTINE NAME - DTT16 COMPUTATIONAL ROUTINE

5.17.1 Purpose

The DTT16 routine adjusts the target central angle of travel from Tig to entry interface to eliminate the difference between the actual and desired range from entry interface to the landing site.

5.17.2 Functional Description

The difference between the actual and the desired range from entry interface to the landing site is added to the target range from Tig to entry interface. This new range is used as a new deorbit target. The routine is called as required until the range error is within a small input tolerance.

5.17.3 Assumptions and Limitations

None.

5.17.4 Method

None.

5.17.5 Routine Input/Output Variables

Table 5.17-I contains the definitions of all the input/output variables for the DTT16 computational routine.

5.17.6 Functional Logic Flow

Figure 5.17-1 contains the functional logic flow for the DTT16 computational routine.

5.17.7 Diagnostics and Debug

None.

5.17.8 Special Comments

None.

5.17.9 References

None.

TABLE 5.17-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT16

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IBTR		Intg	I		C		Refer to table 5.1-I for all code symbol definitions.
THETEI		Real	I/O		C		
THELSD		Real	I		C		
THETLS		Real	I		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

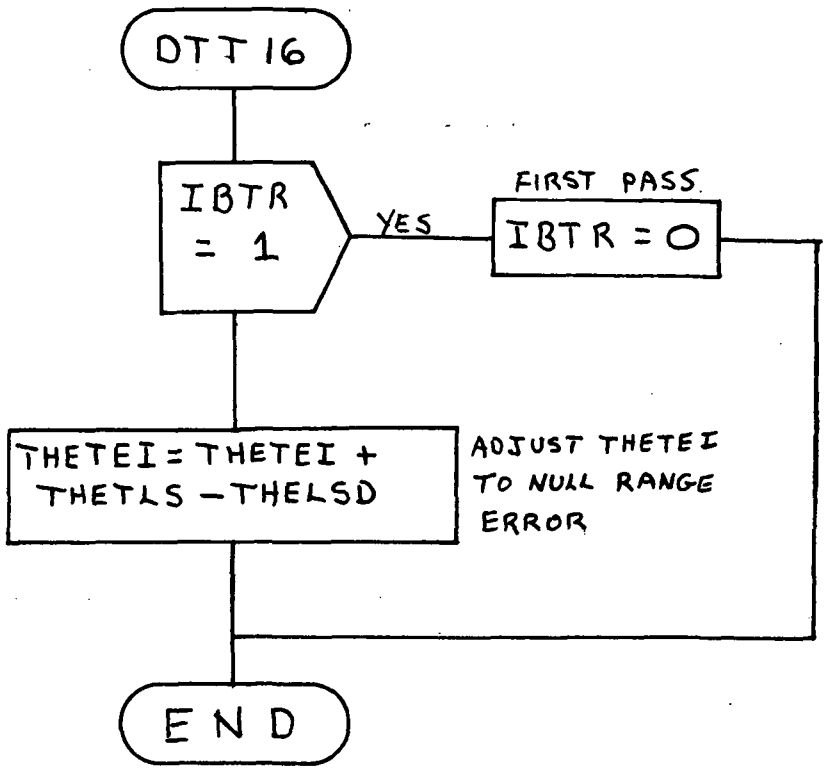


Figure 5.17-1.- Functional logic flow for the DTT16 computational routine.

5.18 ROUTINE NAME - DTT17 COMPUTATIONAL ROUTINE

5.18.1 Purpose

The DTT17 is the end task for the basic targeting routine (DTM5).

5.18.2 Functional Description

DTT17 resets flags and saves the entry interface vector for later use in offset targeting.

5.18.3 Assumptions and Limitations

None.

5.18.4 Method

None.

5.18.5 Routine Input/Output Variables

Table 5.18-I contains the definitions of all the input/output variables for the DTT17 computational routine.

5.18.6 Functional Logic Flow

Figure 5.18-1 contains the functional logic flow for the DTT17 computational routine.

5.18.7 Diagnostics and Debug

None.

5.18.8 Special Comments

None.

5.18.9 References

None.

TABLE 5.18-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT17

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
RTA		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
TT		Real	I		C		
VTAA		Real	I		C		
IBTR		Intg	O		C		
NSYS		Intg	O		C		
REIS		Real	O		C		
TEIS		Real	O		C		
VEIS		Real	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

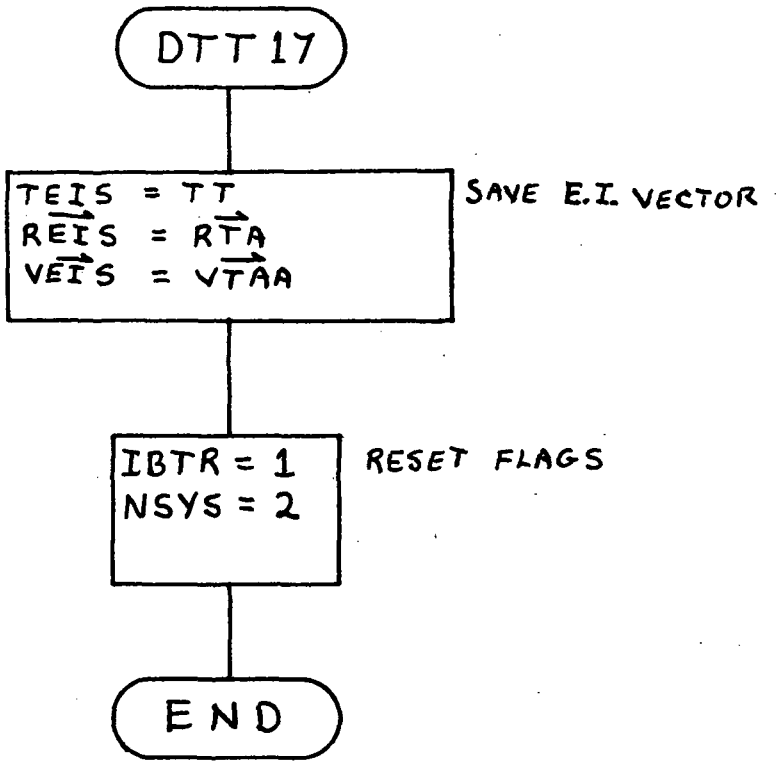


Figure 5.18-1.- Functional logic flow for the DTT17 computational routine.

5.19 ROUTINE NAME - DTT18 COMPUTATIONAL ROUTINE

5.19.1 Purpose

For Tig free fuel wasting cases the DTT18 routine computes a new Tig time and target central angle of travel from Tig to entry interface such that the free-fall time from burnout to entry interface will be more than the input minimum freefall time.

5.19.2 Functional Description

In the Tig free fuel wasting mode the Tig time is backed up such that the backup system freefall time will be greater than the input freefall time minimum by a small input tolerance. A new target central angle of travel from Tig to EI is computed corresponding to the adjusted Tig time.

5.19.3 Assumptions and Limitations

None.

5.19.4 Method

None.

5.19.5 Routine Input/Output Variables

Table 5.19-I contains the definitions of all the input/output variables for the DTT18 computational routine.

5.19.6 Functional Logic Flow

Figure 5.19-1 contains the functional logic flow for the DTT18 computational routine.

5.19.7 Diagnostics and Debug

None.

5.19.8 Special Comments

None.

5.19.9 References

None.

TABLE 5.19-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT18

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DTCOST		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
TFFMIN		Real	I		C		
TFFTOL		Real	I		C	DTMCON(6)	
THETEI		Real	I/O		C		
THETD		Real	I		C		
TIG		Real	I/O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

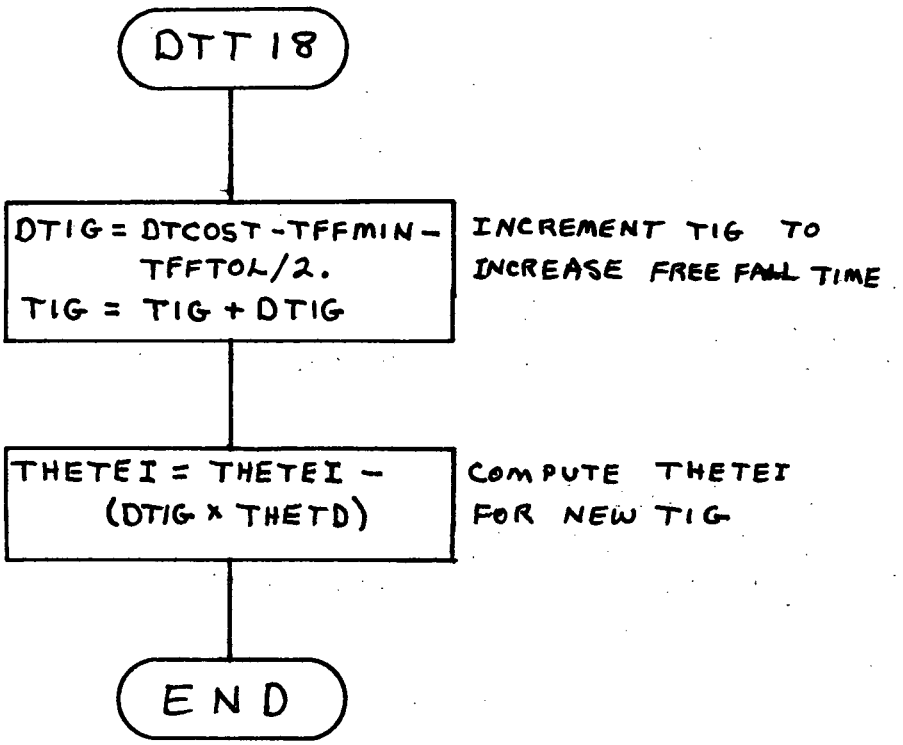


Figure 5.19-1.- Functional logic flow for the DTT18 computational routine.

5.20 ROUTINE NAME - DTT21 COMPUTATIONAL ROUTINE

5.20.1 Purpose

The DTT21 routine advances the ignition time by one orbit period when the Tig computed becomes less than the input threshold time.

5.20.2 Functional Description

In the Tig free mode the computed Tig time may move to a time prior to the input threshold time. If this occurs the DTT21 routine adds one orbit period to the Tig time for a new Tig time.

5.20.3 Assumptions and Limitations

The user input threshold time is used as an absolute minimum time for a solution. If Tig occurs prior to this time, a solution in the next revolution is computed.

5.20.4 Method

None.

5.20.5 Routine Input/Output Variables

Table 5.20-I contains the definitions of all the input/output variables for the DTT21 computational routine.

5.20.6 Functional Logic Flow

Figure 5.20-1 contains the functional logic flow for the DTT21 computational routine.

5.20.7 Diagnostics and Debug

None.

5.20.8 Special Comments

None.

5.20.9 References

None.

TABLE 5.20-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT21

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
C1IP		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
C2IP		Real	I		C		
NCNT1		Intg	I/O		C		
NCMAX		Intg	I		C	DTMCON(38)	
THETD		Real	I		C		
TIG		Real	I/O		C		
TWOPI		Real	I		C	GLOCON(61)	
CA		Real	O		C		
IMSG		Intg	O		C		
IPRNG		Intg	O		C		
INCR		Intg	O		C		
MITER		Intg	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

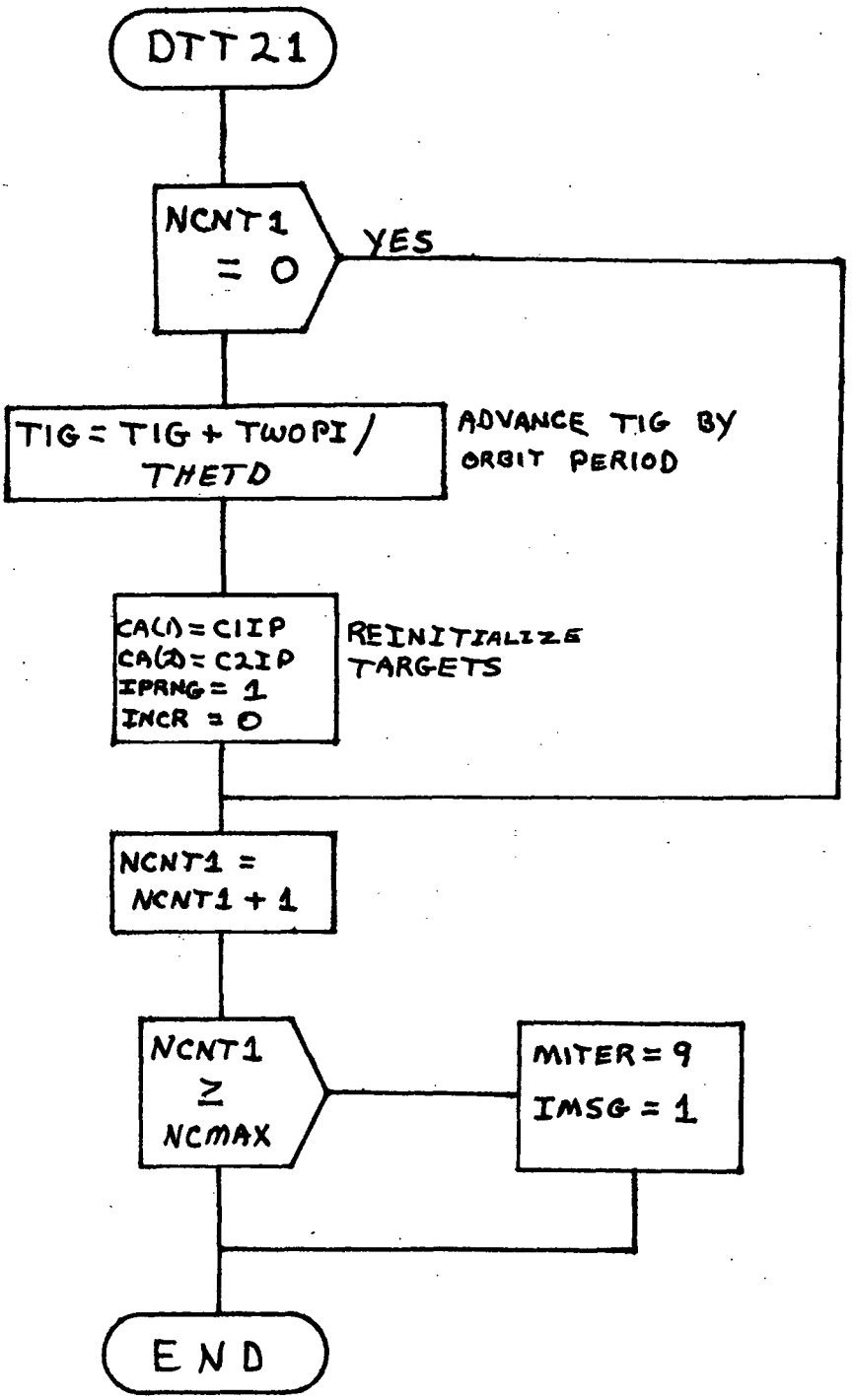


Figure 5.20-1.- Functional logic flow for the DTT21 computational routine.

5.21 ROUTINE NAME - DTT22 COMPUTATIONAL ROUTINE

5.21.1 Purpose

The DTT22 routine is an initialization routine for the offset targeting logic in DTT24.

5.21.2 Functional Description

The DTT22 routine computes a conic flightpath angle at entry interface based upon entry position and velocity vector. It also computes conic vertical and horizontal velocities from the entry velocity vector and flightpath angle.

5.21.3 Assumptions and Limitations

None.

5.21.4 Method

None.

5.21.5 Routine Input/Output Variables

Table 5.21-I contains the definitions of all the input/output variables for the DTT22 computational routine.

5.21.6 Functional Logic Flow

Figure 5.21-I contains the functional logic flow for the DTT22 computational routine.

5.21.7 Diagnostics and Debug

None.

5.21.8 Special Comments

None.

5.21.9 References

None.

TABLE 5.21-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT22

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
REIS		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
VEIS		Real	I		C		
VHS		Real	O		C		
VVS		Real	O		C		

NOTES:

TYPE

Free	Dubl	18CH	Mix
Intg	2CH	36CH	Char
Real	6CH	72CH	Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

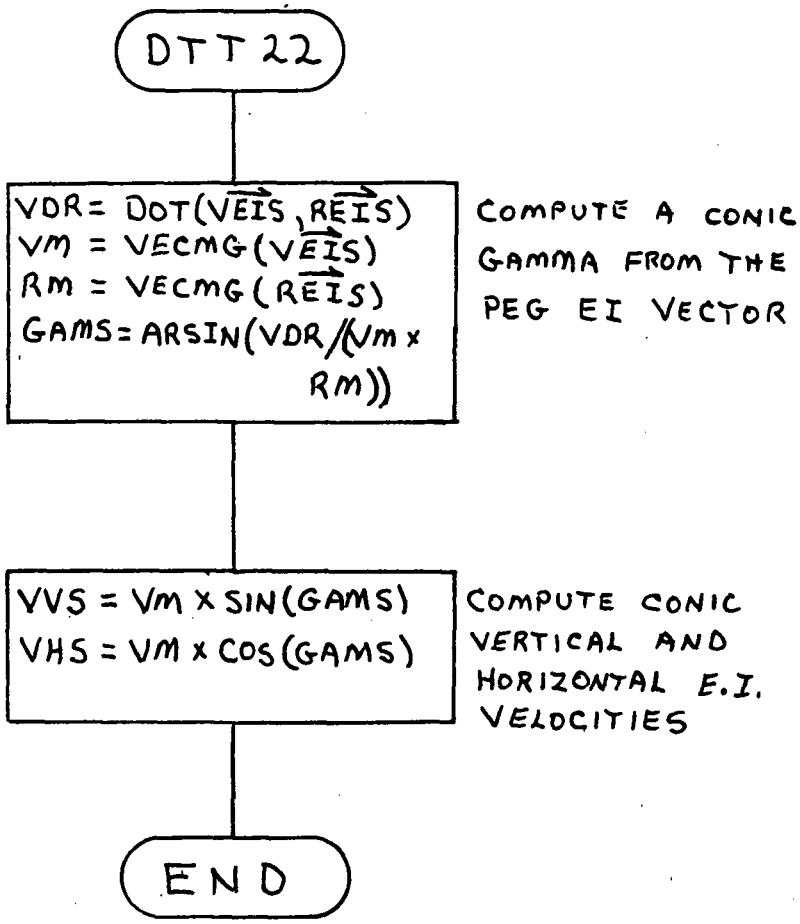


Figure 5.21-1.- Functional logic flow for the DTT22 computational routine.

5.22 ROUTINE NAME - DTT23 COMPUTATIONAL ROUTINE

5.22.1 Purpose

The DTT23 routine computes a correction factor to apply to the vertical velocity-horizontal velocity target line in order to null the error between the entry interface flightpath angles obtained from the conic prediction based upon the original target line and from the integration to entry interface of the J2 effects burnout vector.

5.22.2 Functional Description

The flightpath angle, vertical velocity, and horizontal velocity at entry interface are computed using the vector obtained from integrating the J2 effects burn out vector to entry interface. Also a conic flightpath angle based upon the vertical velocity-horizontal velocity target line is computed. From these a flightpath angle difference and a delta to the target line slope is computed. This routine is called as required until the flightpath angle difference is less than a small input tolerance.

5.22.3 Assumptions and Limitations

None.

5.22.4 Method

None.

5.22.5 Routine Input/Output Variables

Table 5.22-I contains the definitions of all the input/output variables for the DTT23 computational routine.

5.22.6 Functional Logic Flow

Figure 5.22-1 contains the functional logic flow for the DTT23 computational routine.

5.22.7 Diagnostics and Debug

None.

5.22.8 Special Comments

None.

5.22.9 References

None.

TABLE 5.22-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT23

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
CA		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
RENK		Real	I		C		
VENK		Real	I		C		
VHS		Real	I		C		
VVS		Real	I		C		
DC11		Real	O		C		
DGAM		Real	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

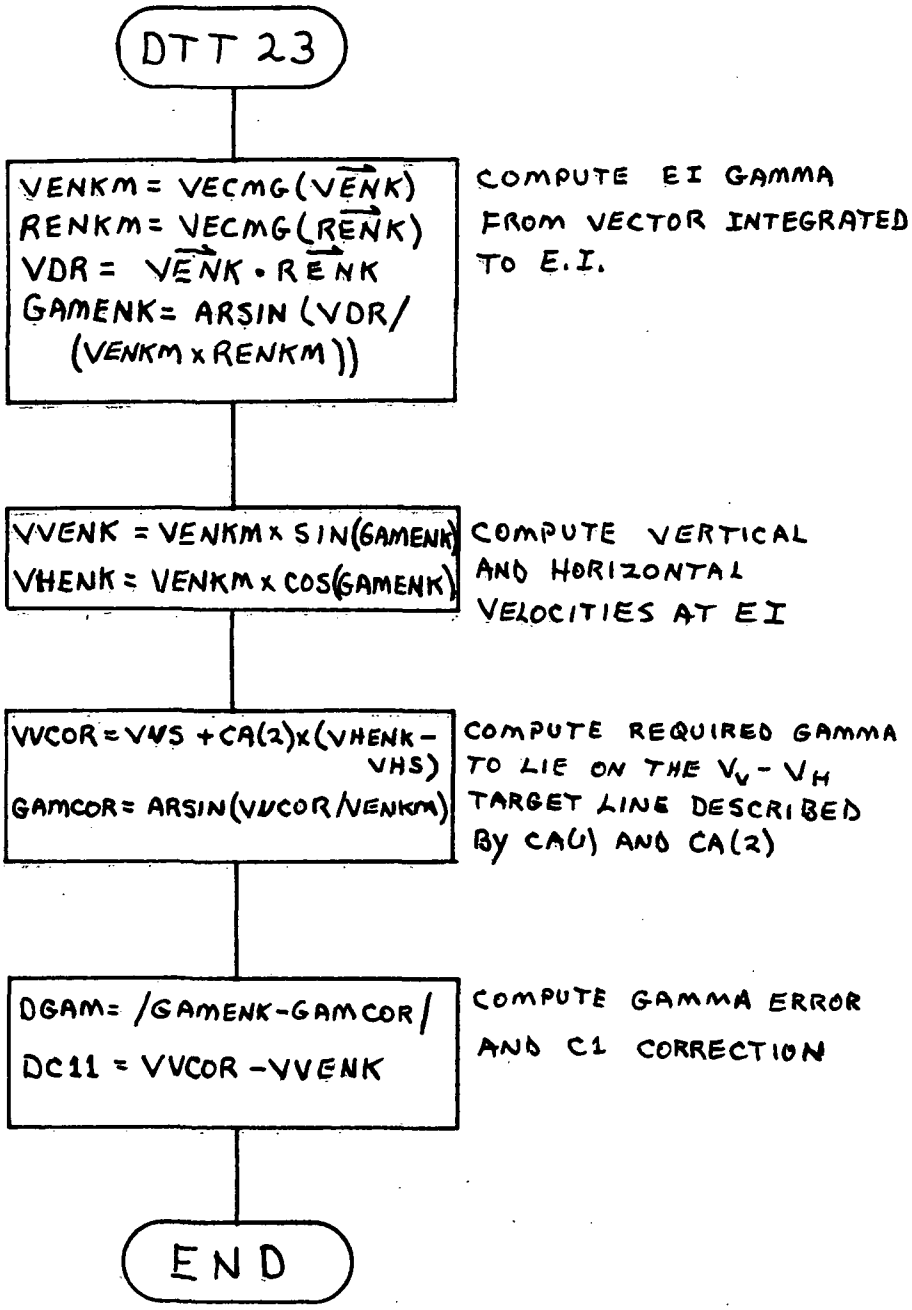


Figure 5.22-1.- Functional logic flow for the DTT23 computational routine.

5.23 ROUTINE NAME - SUPRJ COMPUTATIONAL ROUTINE

5.23.1 Purpose

The SUPRJ routine does a state extrapolation to account for the J2 gravitational effects in the deorbit burnout vector.

5.23.2 Functional Description

The Tig vector is propagated to the burnout time by an iterative series of calls to a GRAVJ function to compute the current gravity acceleration vector at the input vector position with the J2 effects included. The state vector is extrapolated through a computed number of time steps using the gravity acceleration vector that is updated with each step. With each step, the position and velocity vectors are updated as a function of the current velocity vector and the gravitational acceleration vector. The extrapolation is complete when the burnout time is reached.

5.23.3 Assumptions and Limitations

None.

5.23.4 Method

None.

5.23.5 Routine Input/Output Variables

Table 5.23-I contains the definitions of all the input/output variables for the SUPRJ computational routine.

5.23.6 Functional Logic Flow

Figure 5.23-1 contains the functional logic flow for the SUPRJ computational routine.

5.23.7 Diagnostics and Debug

When the user selects the debug print option the following variables are printed, RC2, VC2, TGO, and DTAVG.

5.23.8 Special Comments

None.

5.23.9 References

None.

TABLE 5.23-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine SUPRJ

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DTAVG		Real	I/O		C,T		Refer to table 5.1-I for all code symbol definitions.
IOUNIT		Intg	I		C		
IPOUT		Intg	I		C		
RC2		Real	I/O		C,T		
TGO		Real	I/O		C,T		
VC2		Real	I/O		C,T		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

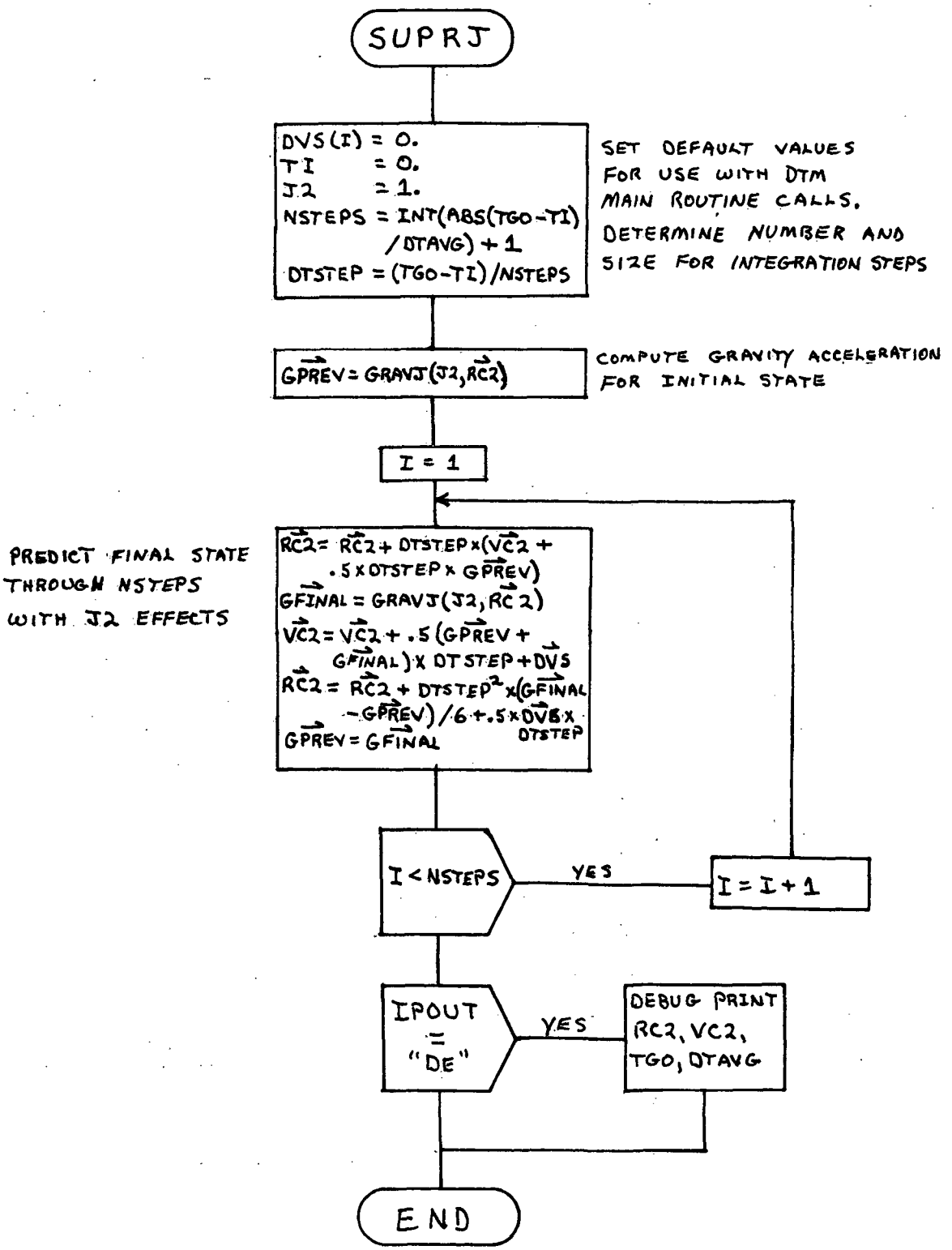


Figure 5.23-1.- Functional logic flow for the SUPRJ computational routine.

5.24 ROUTINE NAME - GRAVJ COMPUTATIONAL ROUTINE

5.24.1 Purpose

The GRAVJ routine computes the gravity acceleration vector for the SUPRJ and SUPRG routines.

5.24.2 Functional Description

The GRAVJ routine computes the current gravity acceleration vector with J2 effects included. Given a current position vector the routine computes the spherical Earth gravity vector at the current position and then adds the J2 effects to the vector if desired.

5.24.3 Assumptions and Limitations

The potential model is a spherical Earth with J2 effects included if desired.

5.24.4 Method

None.

5.24.5 Routine Input/Output Variables

Table 5.24-I contains the definitions of all the input/output variables for the GRAVJ computational routine.

5.24.6 Functional Logic Flow

Figure 5.24-1 contains the functional logic flow for the GRAVJ computational routine.

5.24.7 Diagnostics and Debug

None.

5.24.8 Special Coments

None.

5.24.9 References

None.

TABLE 5.24-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine GRAVJ

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
EQRAD		Real	I		C	GLOCON(23)	Refer to table 5.1-I for all code symbol definitions.
GM		Real	I		C	GLOCON(1)	
J2		Real	I		A		
POLE		Real	I		C	DTMCON(39)	
RF		Real	I		A		
XJ2		Real	I		C	GLOCON(31)	
GRAV		Real	O		A		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

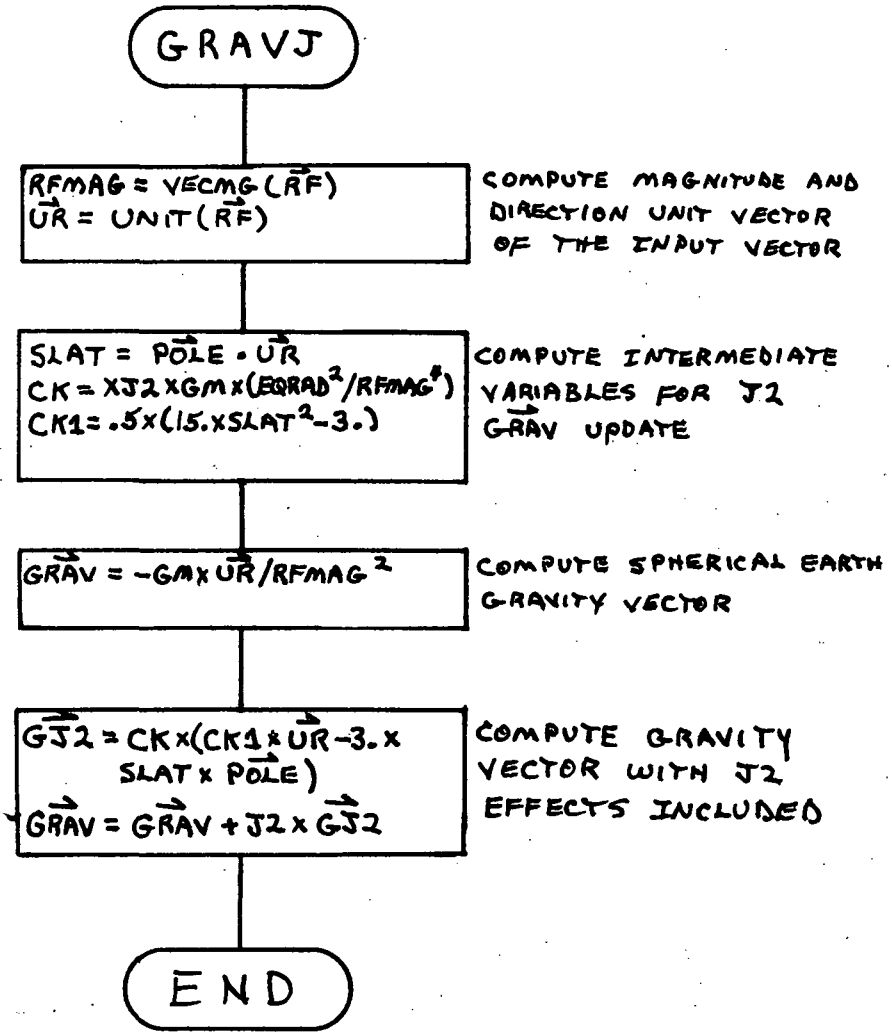


Figure 5.24-1.- Functional logic flow for the GRAVJ computational routine.

5.25 ROUTINE NAME - DTT1 COMPUTATIONAL ROUTINE - SEGMENT 1

5.25.1 Purpose

The DTT1 routine does parameter initialization, units conversion, and engine characteristics computations for the DTM processor. Along with the GEOD routine it is part of the processing segment 1.

5.25.2 Functional Description

This routine is used by DTM once at the beginning of processing. It initializes a large number of flag and switch settings for DTM and verifies that the user Hollerith inputs are acceptable. Following this it converts from the users selected external units set to the internal units required for processing. Finally it computes the thrust, flow rate, and delta-V onboard for the prime and backup engine configurations and it computes landing site position and closing rates.

5.25.3 Assumptions and Limitations

None.

5.25.4 Method

None.

5.25.5 Routine Input/Output Variables

Table 5.25-I contains the definitions of all the input/output variables for the DTT1 computational routine.

5.25.6 Functional Logic Flow

Figure 5.25-1 contains the functional logic flow for the DTT1 computational routine.

5.25.7 Diagnostics and Debug

The values of all variables in common will be dumped at the completion of DTT1 processing when the user selects the debug output option.

5.25.8 Special Comments

None.

5.25.9 References

None.

TABLE 5.25-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT1

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AMILE		Real	I		C	GLOCON(89)	Refer to table 5.1-I for all code symbol definitions.
CA		Real	O		C		
C1IP		Real	I/O		C		
C2IP		Real	I		C		
DTTHRS		Real	I		C		
FPA		Real	I/O	deg	C	FVECON(6)	
GGOL		Real	I		C	DTMCON(26)	
IDTM		Intg	I		C		
INTEGF		Intg	I		C		
IPOUT		6CH	I		C		
ITIGFR		Intg	I/O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.25-I.- Continued

Routine DTT1

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
JDGUID		6CH	I		C		
JFUEL		2CH	I		C		
JNGBU		6CH	I		C		
JNGPR		6CH	I		C		
JTIGFR		6CH	I		C		
RADIAN		Real	I		C	GLOCON(83)	
RNGG		Real	I/O	n. mi.	C	FVECON(11)	
RNGIP		Real	I/O	n. mi.	C		
RTE		Real	I		C		
SESCON		Free	I		C		
SMISSI		Real	I		C	DTMCON(21)	
SSS		Real	I		C	DTMCON(13)	

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory

TABLE 5.25-I.- Continued

Routine DTT1

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
STP		*Real*	*I*		*C*	*GLOCON(117)*	
TIMEC		Real	I/O		C	DTMVEC(1)	
THEIIP		Real	I		C	DTMCON(28)	
THETEI		Real	I/O		C		
THETD		Real	I/O		C		
TFFMIN		Real	I/O		C		
TIG		Real	I/O		C		
TLATD		Real	I		C		
TTHRSH		Real	I/O		C		
WE		Real	I		C	GLOCON(13)	
WCGOMS		Real	I/O		C		
WT		Real	I/O		C	DTMVEC(13)	

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.25-I.- Continued

Routine DTI1

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
XMDOMS		Real	I/O	slugs/sec	C		
XMDRCS		Real	I/O	slugs/sec	C		
XMU		Real	I		C	GLOCON(1)	
XYZI		Real	I/O		C	DTMVEC(2)	
XYZID		Real	I/O		C	DTMVEC(5)	
DVOBBU		Real	O		C		
DVOBPR		Real	O		C		
ERAI		Real	O		C		
FBU		Real	O		C		
FPR		Real	O		C		
IABT		Intg	O		C		
IAME		Intg	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.25-I.- Continued

Routine DTT1

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IBTR		Intg	O		C		
ICALLL		Intg	O		C		
ICAPT		Intg	O		C		
IERR		Intg	O		C		
IFFPTM		Intg	O		C		
IFINAL		Intg	O		C		
IFUEL		Intg	O		C		
IGAMFR		Intg	O		C		
IMPULS		Intg	O		C		
IMSG		Intg	O		C		
INCR		Intg	O		C		
INGBU		Intg	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.25-I.- Continued

Routine DTT1

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
INGPR		Intg	O		C		
IRPARM		Intg	O		C		
IPRNG		Intg	O		C		
ITERM		Intg	O		C		
ITFF		Intg	O		C		
KRCS		Intg	O		C		
KDGUID		Intg	O		C		
KDSTER		Intg	O		C		
KDTHR		Intg	O		C		
KITER		Intg	O		C		
KOMS		Intg	O		C		
KSTEER		Intg	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.25-I.- Continued

Routine DTT1

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
NCNT1		Intg	O		C		
MITER		Intg	O		C		
SBD		Real	O		C		
SMISS		Real	O		C		
THEPEI		Real	O		C		
THETDR		Real	O		C		
TLATC		Real	O		C		
UYDTM		Real	O		C		
WD		Real	O		C		
WDBU		Real	O		C		
WDPR		Real	O		C		
XMD		Real	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.25-I.- Concluded

Routine DTT1

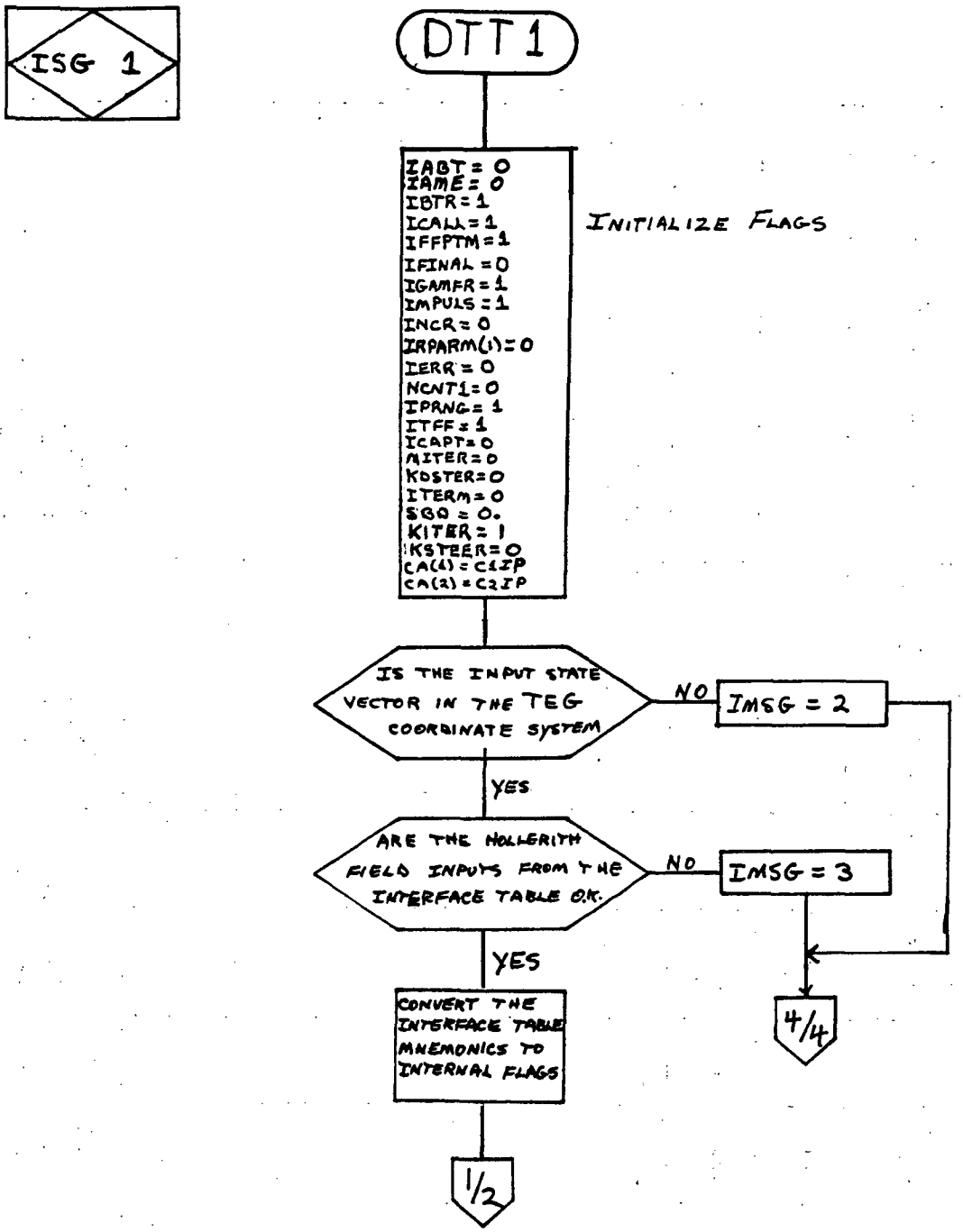
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ZEROT		Real	O		C		
TIGHMS		Real	I	hr,min, sec	C		
TTRHMS		Real	I	hr,min, sec	C		
COM		Real	O		T		Real variable common
ICOM		Intg	O		T		Integer variable common

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory



Page 1 of 4

Figure 5.25-1.- Functional logic flow for the DTT1 computational routine.

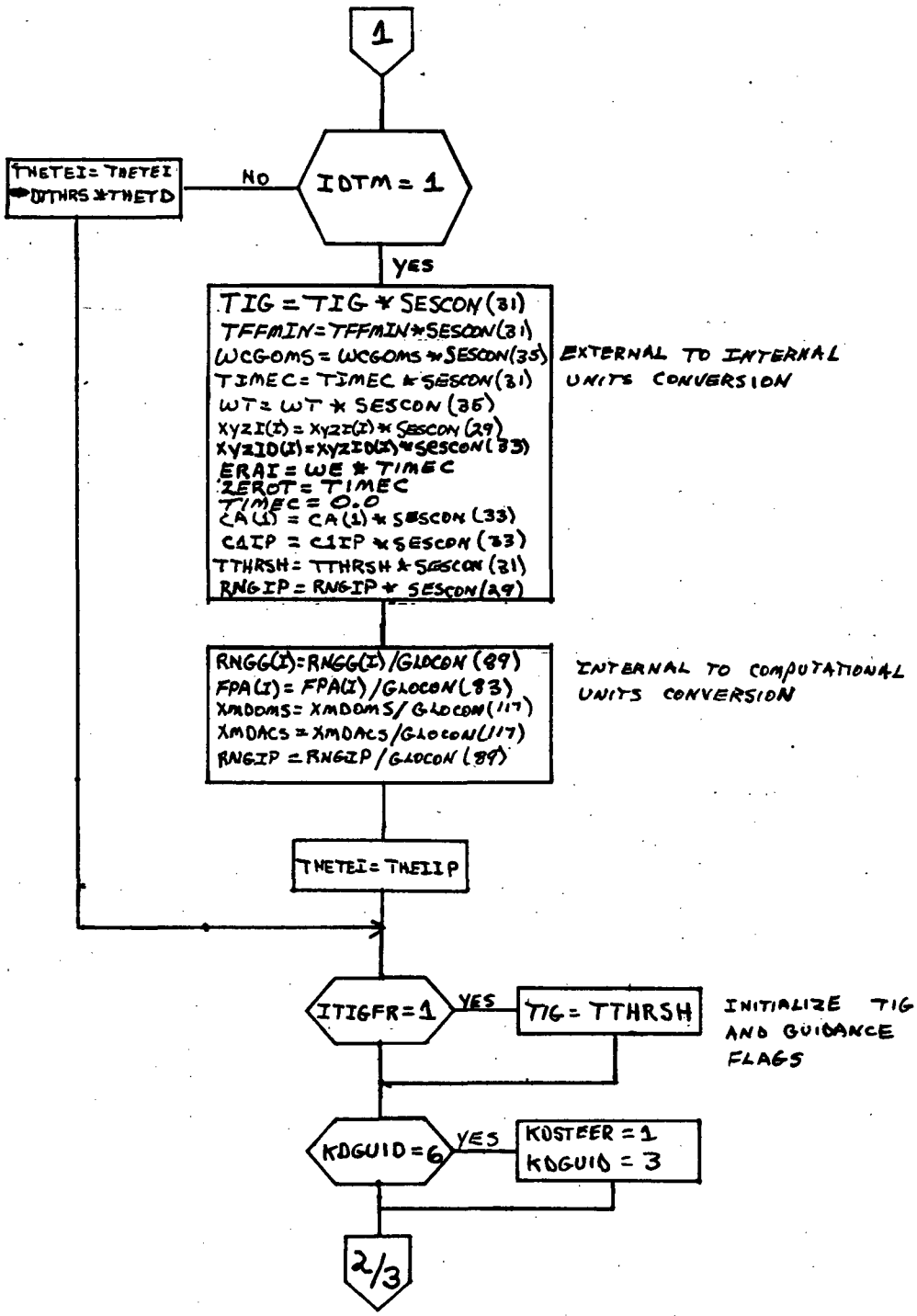


Figure 5.25-1.- Continued.

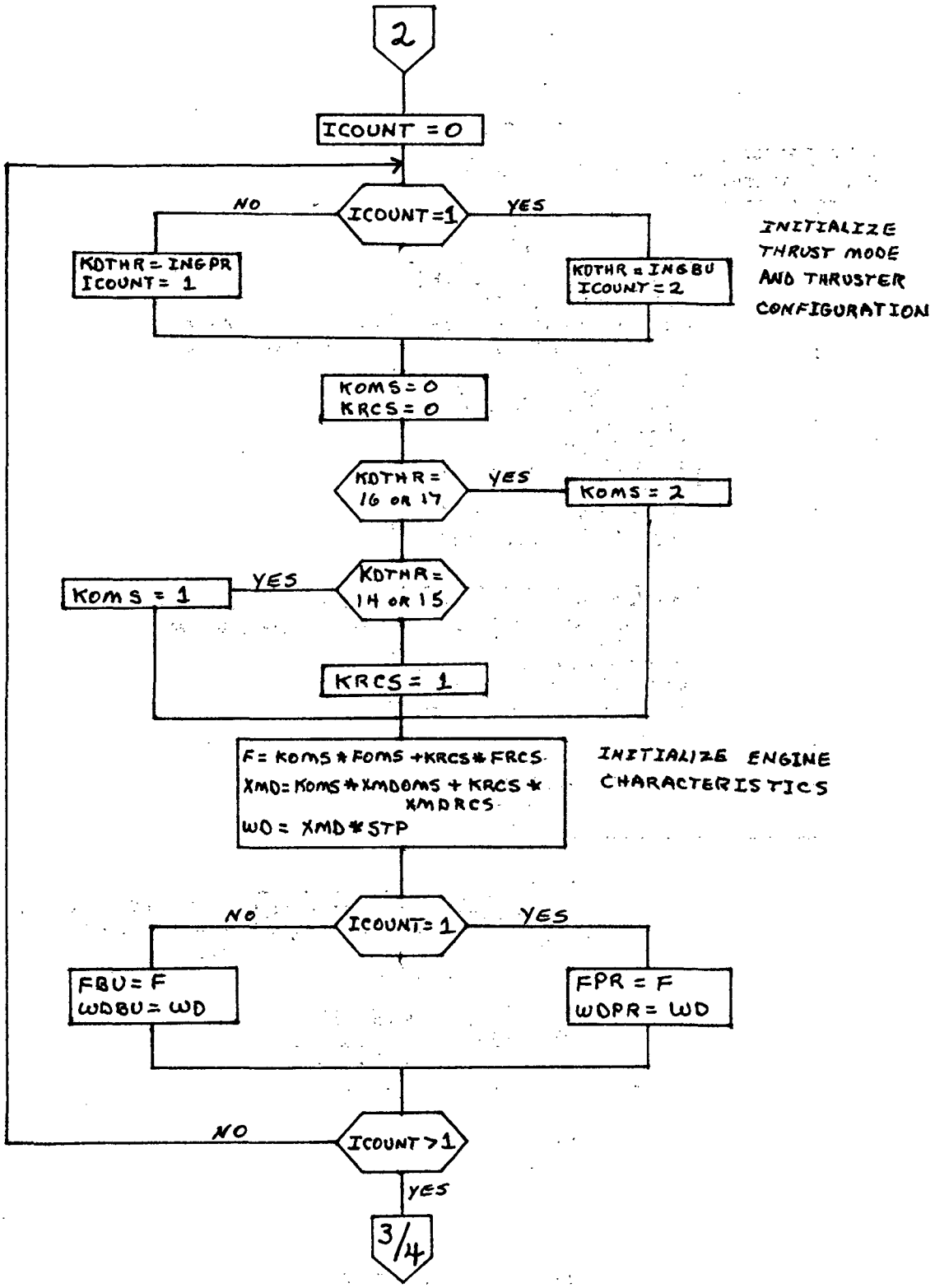


Figure 5.25-1.- Continued.

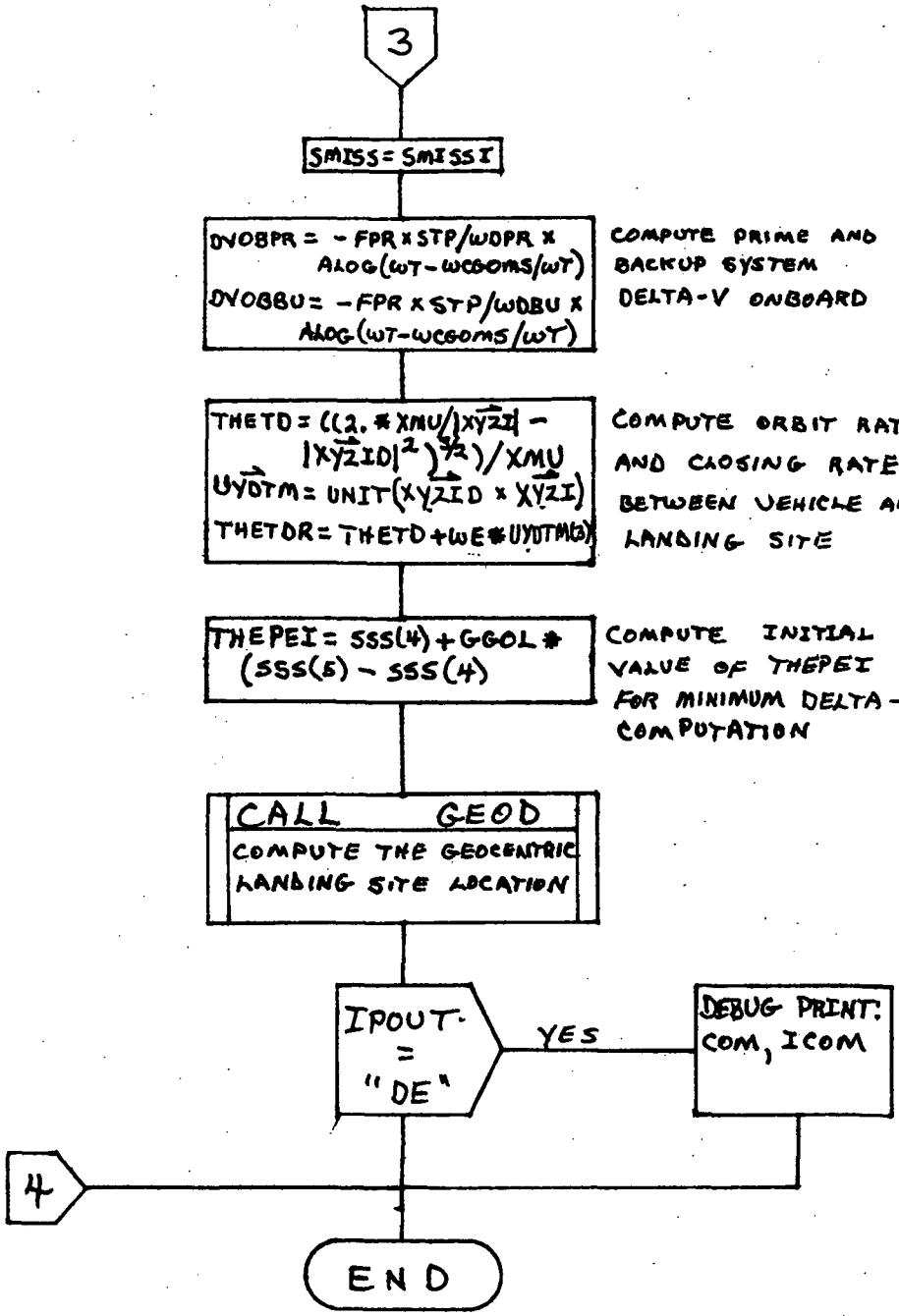


Figure 5.25-1.- Concluded.

5.26 ROUTINE NAME - GEOD COMPUTATIONAL ROUTINE

5.26.1 Purpose

The GEOD routine computes geocentric radius and latitude from geodetic altitude and latitude.

5.26.2 Functional Description

Geodetic altitude and latitude are converted to a geocentric radius and latitude. The Earth ellipticity is established by the ellipsoid equitorial and polar radii contained in the global constants.

5.26.3 Assumptions and Limitations

None.

5.26.4 Method

None.

5.26.5 Routine Input/Output Variables

Table 5.26-I contains the definitions of all the input/output variables for the GEOD computational routine.

5.26.6 Functional Logic Flow

Figure 5.26-1 contains the functional logic flow for the GEOD computational routine.

5.26.7 Diagnostics and Debug

None.

5.26.8 Special Comments

None.

5.26.9 References

None.

TABLE 5.26-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine GEOD

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
EQRAD		Real	I		C	GLOCON(23)	Refer to table 5.1-I for all code symbol definitions.
GHD		Real	I		A		Geodetic altitude
GPHID		Real	I		A		Geodetic latitude
POLRAD		Real	I		C	GLOCON(25)	Refer to table 5.2.11.5-I for all code symbol definitions.
GR		Real	O		A		Geocentric radius
GPHIC		Real	O		A		Geocentric latitude
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

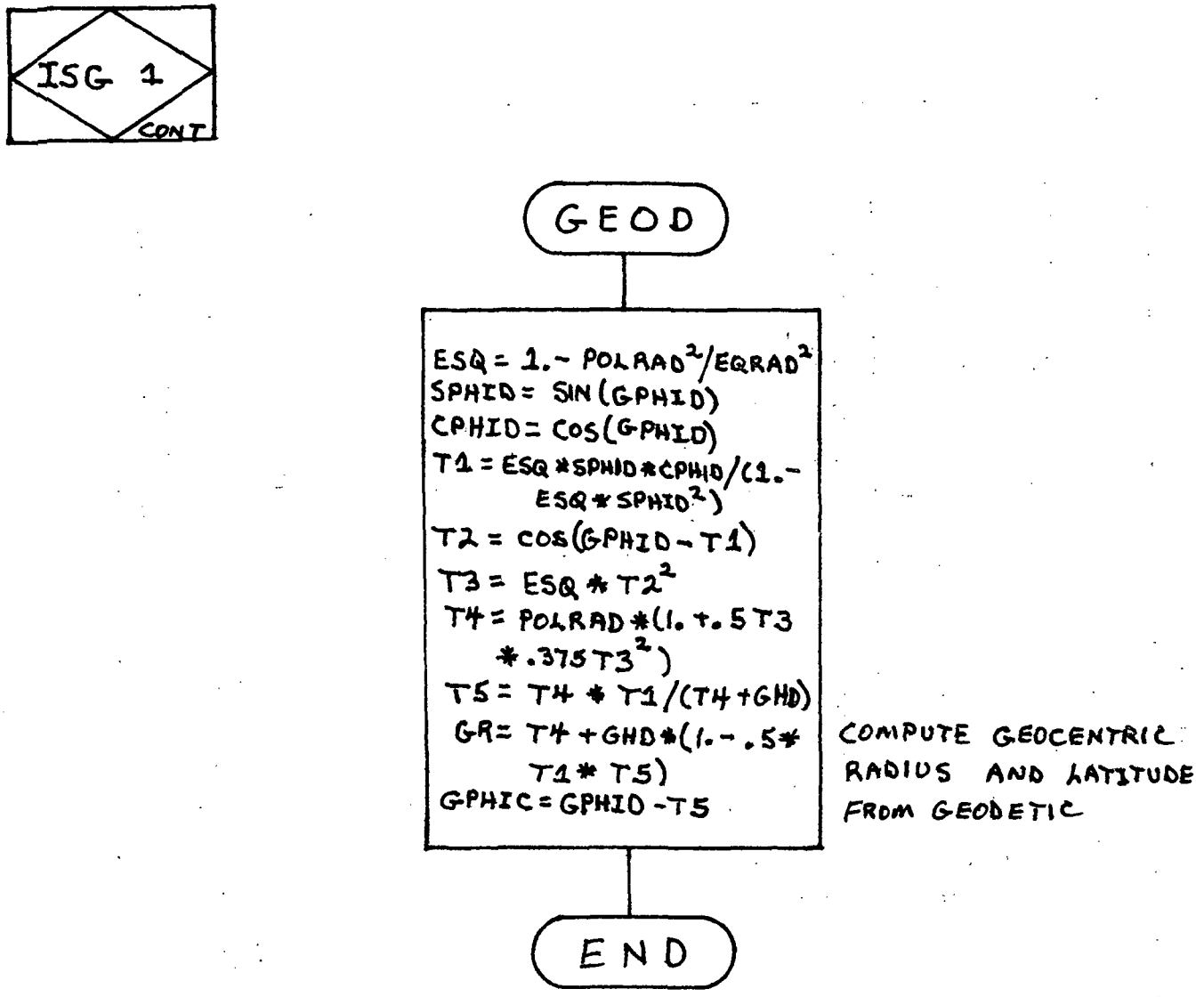


Figure 5.26-1.- Functional logic flow for the GEOD computational routine.

5.27 ROUTINE NAME - DTT2 COMPUTATIONAL ROUTINE - SEGMENT 2

5.27.1 Purpose

The DTT2 routine computes a Tig time and central angle of travel from Tig to entry that provide a minimum delta-V inplane solution. This routine is the first step in Tig free processing to obtain a Tig estimate prior to doing finite burn processing. Along with the GLPRP routine it is part of computational segment 2.

5.27.2 Functional Description

The DTT2 routine calls the minimization routine GLPRP to obtain a transfer angle that provides a minimum delta-V deorbit. Then the Tig time is adjusted to insure that the range from entry interface to landing remains constant.

5.27.3 Assumptions and Limitations

None.

5.27.4 Method

None.

5.27.5 Routine Input/Output Variables

Table 5.27-I contains the definitions of all the input/output variables for the DTT2 computational routine.

5.27.6 Functional Logic Flow

Figure 5.27-1 contains the functional logic flow for the DTT2 computational routine.

5.27.7 Diagnostics and Debug

When the user selects the debug print option the following variables are output - TIG, THETEI, AAA, JJJ. The AAA array is the input constants to the GLPRP routine and the JJJ flag indicates a minima has been reached when it has a value of zero.

5.27.8 Special Comments

None.

5.27.9 References

None.

TABLE 5.27-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT2

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AAA		Real	I/O		C,T		Refer to table 5.1-I for all code symbol definitions.
JJ		Real	I/O		C,T		
IPOUT		Intg	6CH		C		
IOUNIT		Intg	I		C		
THETD		Real	I		C		
THEPEI		Real	I/O		C		
THETEI		Real	I/O		C,T		
TIG		Real	I/O		C,T		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

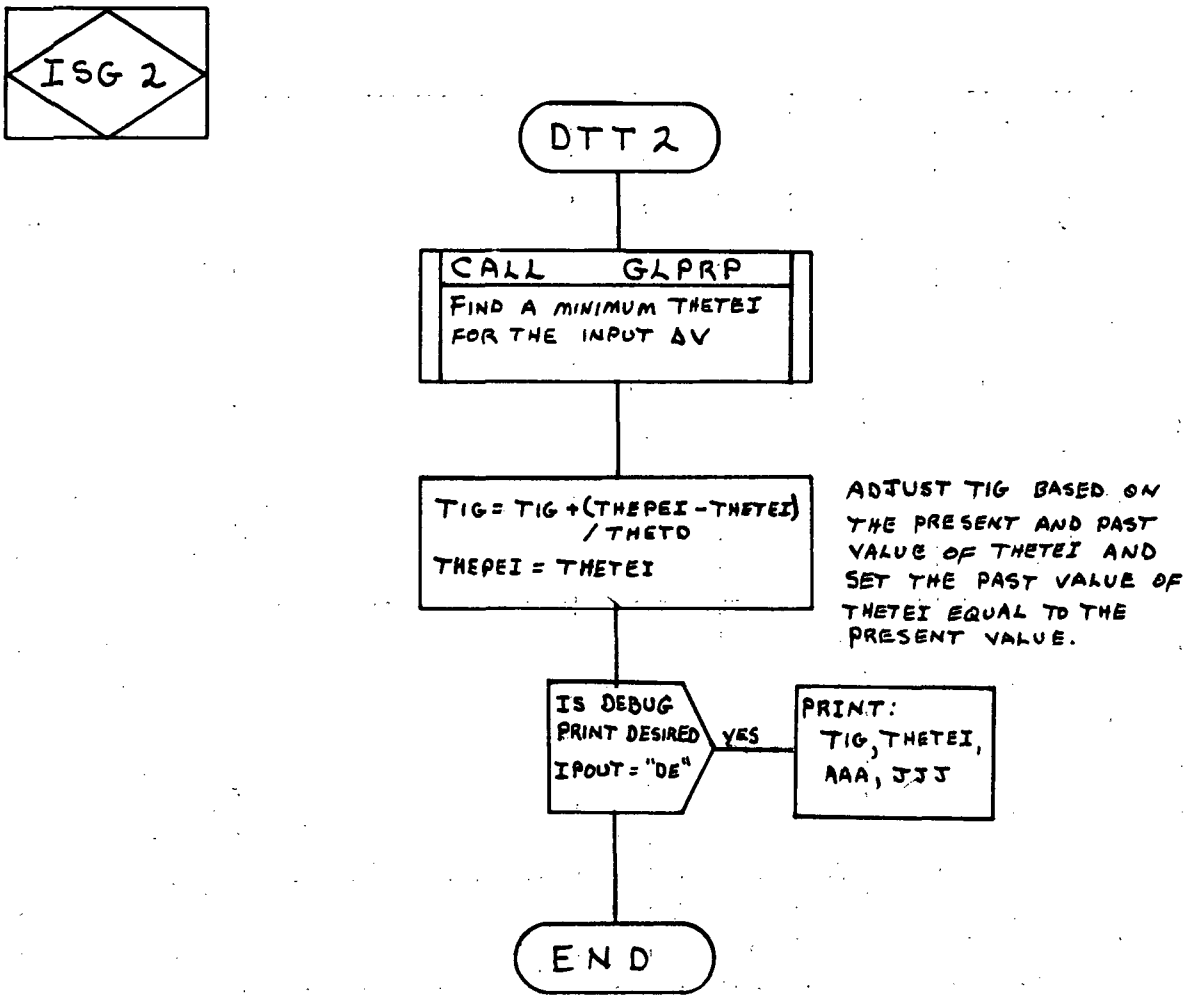


Figure 5.27-1.- Functional logic flow for the DTT2 computational routine.

5.28 ROUTINE NAME - GLPRP COMPUTATIONAL ROUTINE

5.28.1 Purpose

The GLPRP routine computes a value of X which minimizes some Y = f(X). In the case of DTM the X is the central angle of travel from Tig to entry and the Y is the impulsive delta-V required for deorbit.

5.28.2 Functional Description

GLPRP searches for a minimum value of Y = f(X) using both a golden section method and a parabolic fit method. The minimum is obtained by a series of calls to the routine each time reducing the range of values of X which could produce a minimum value of Y. The search is completed when the range of X values remaining is less than an input tolerance.

5.28.3 Assumptions and Limitations

The function being searched must be piecewise continuous as well as its first derivative. In addition the function should have no flat spots or multiple local minima in order to assure a convergence.

5.28.4 Method

The golden section search method makes an initial prediction of X based on a delta-X (the difference between an initial X max and X min). With each pass through the routine, a new X max and X min is calculated thus reducing the value of delta-X and limiting the range of X values from which a minimum Y may be found. This process is continued until the predicted value of X has converged within an input tolerance.

In addition to the golden section prediction, a parabolic estimate of X to minimize Y is computed after two passes through the routine. The parabola is fitted to the current set of three points generated by the golden section logic lying closest to the Y minimum. When two successive parabolic predictions of X differ by less than the input tolerance the search is also terminated. In this case, the last parabolic prediction of X is considered to be the solution.

5.28.5 Routine Input/Output Variables

Table 5.28-I contains the definitions of all the input/output variables for the GLPRP computational routine.

5.28.6 Functional Logic Flow

Figure 5.28-1 contains the functional logic flow for the GLPRP computational routine.

5.28.7 Diagnostics and Debug

None.

5.28.8 Special Comments

None.

5.28.9 References

None.

TABLE 5.28-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine GLPRP

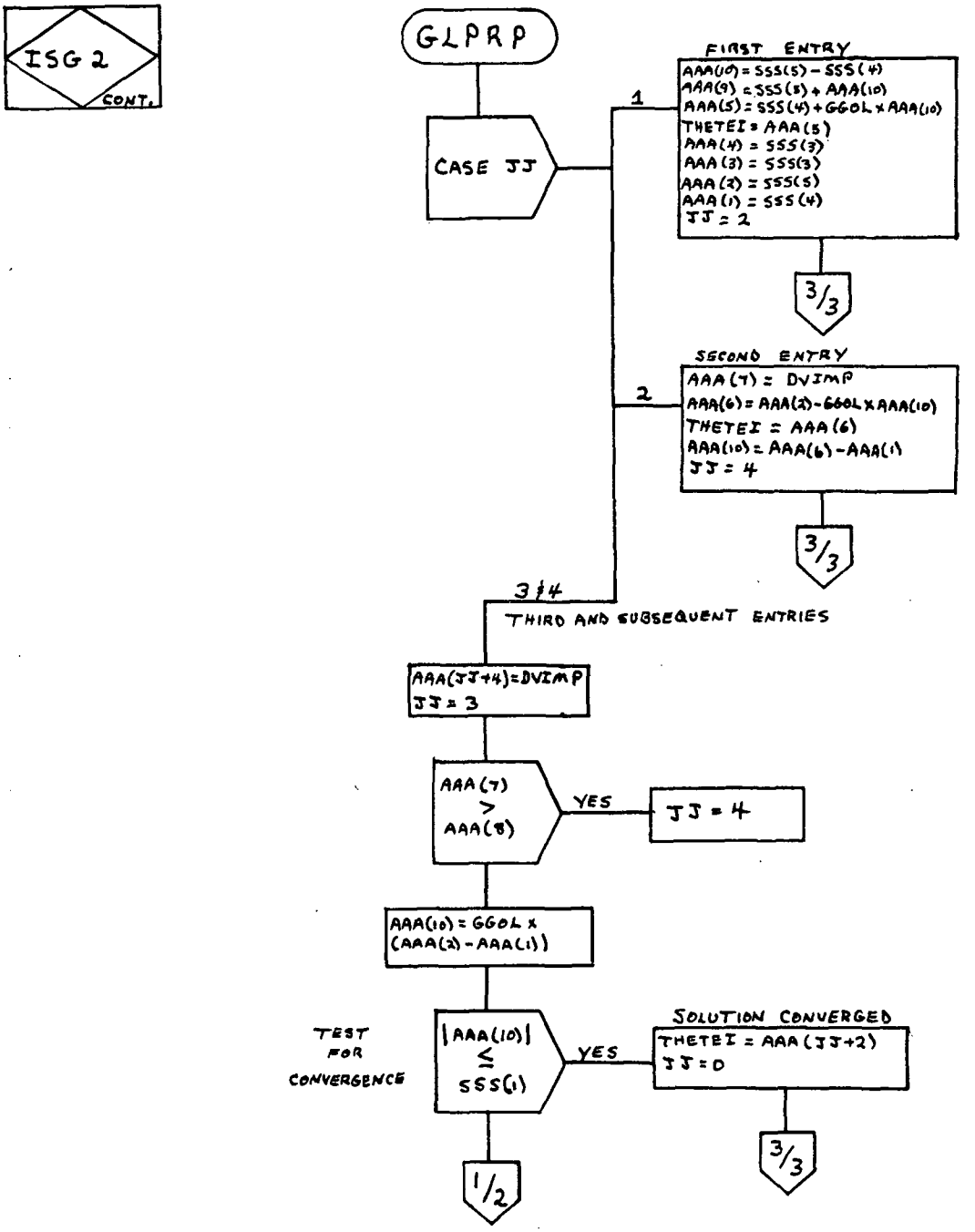
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AAA		Real	I/O		C		Refer to table 5.1-I for all code symbol definitions.
DVIMP		Real	I		C		
SSS		Real	I		C	DTMCON(13)	
GGOL		Real	I		C	DTMCON(26)	
JJ		Real	O		C		
THETEI		Real	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory



Page 1 of 3

Figure 5.28-1.- Functional logic flow for the GLPRP computational routine.

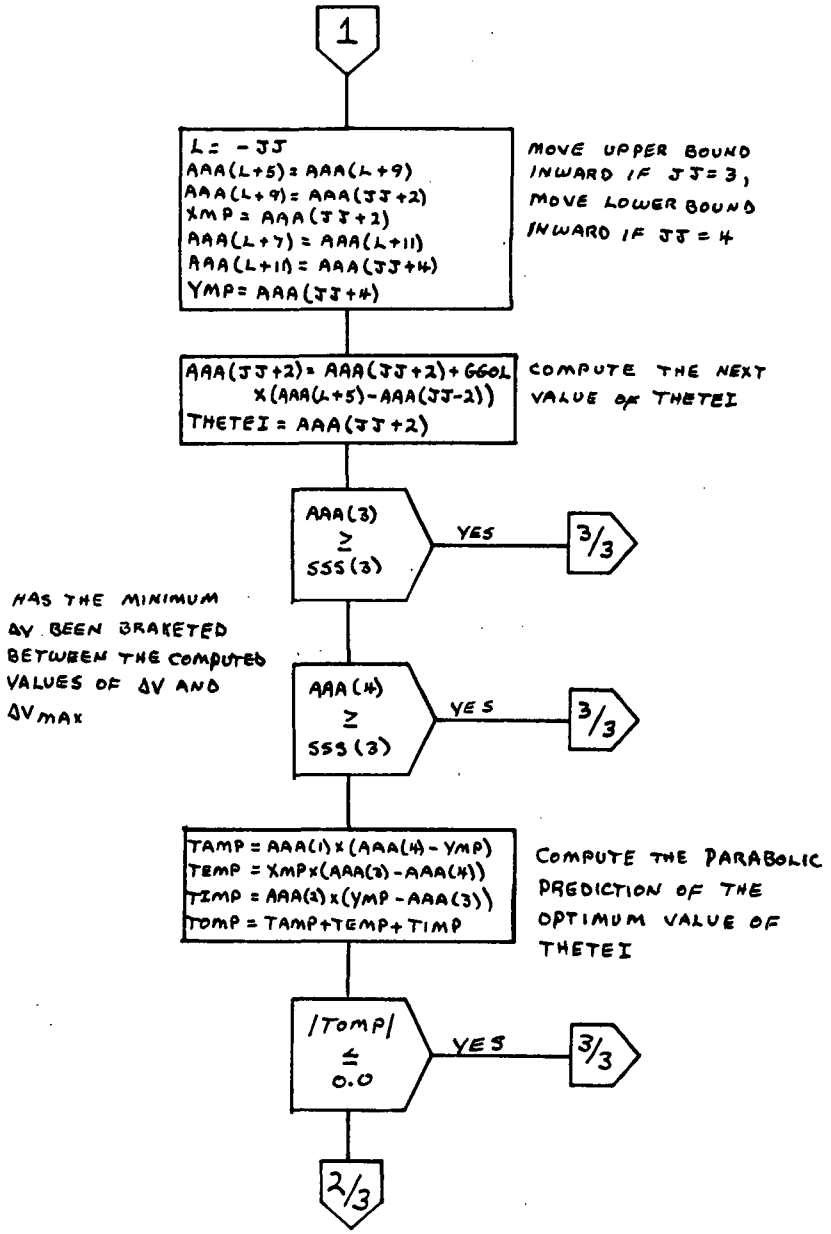


Figure 5.28-1.- Continued.

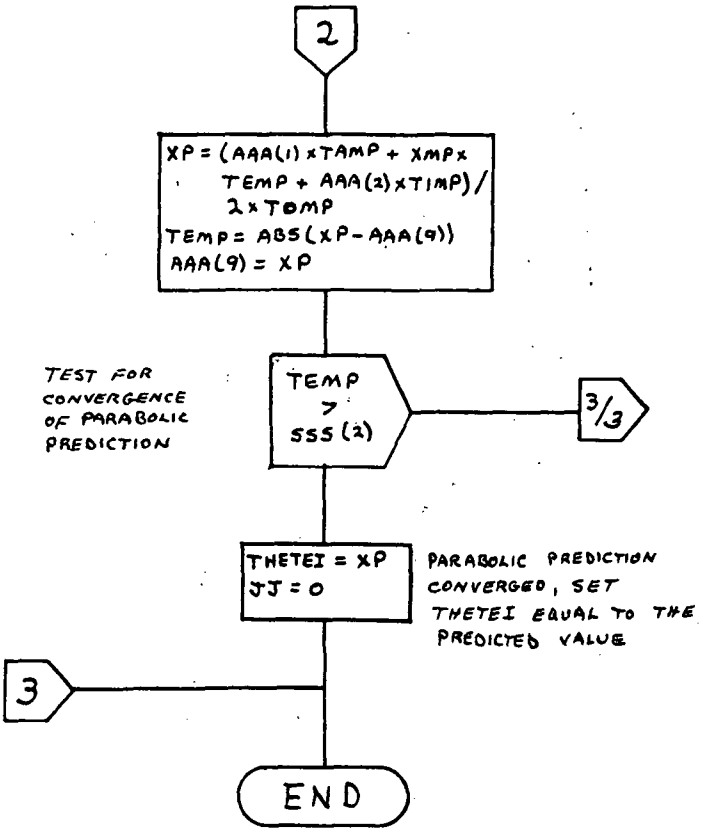


Figure 5.28-1.- Concluded.

5.29 ROUTINE NAME - DTMER MESSAGE ROUTINE - SEGMENT 6

5.29.1 Purpose

The DTMER routine writes error messages to the user terminal. It is the computational segment 6.

5.29.2 Functional Description

The error messages defined in table 4-V are output to the user's terminal when an error in processing has been detected. If the error results in a termination of processing, the return parameter is set prior to exit.

5.29.3 Assumptions and Limitations

None.

5.29.4 Method

None.

5.29.5 Routine Input/Output Variables

Table 5.29-I contains the definitions of all the input/output variables for the DTMER computational routine.

5.29.6 Functional Logic Flow

Figure 5.29-1 contains the functional logic flow for the DTMER computational routine.

5.29.7 Diagnostics and Debug

None.

5.29.8 Special Comments

None.

5.29.9 References

None.

TABLE 5.29-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTMER

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IMSG		Intg	I		C		Refer to table 5.1-I for all code symbol definitions.
IPARM		Intg	I		C		
MITER		Intg	I/O		C,T		
IRPARM		Intg	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

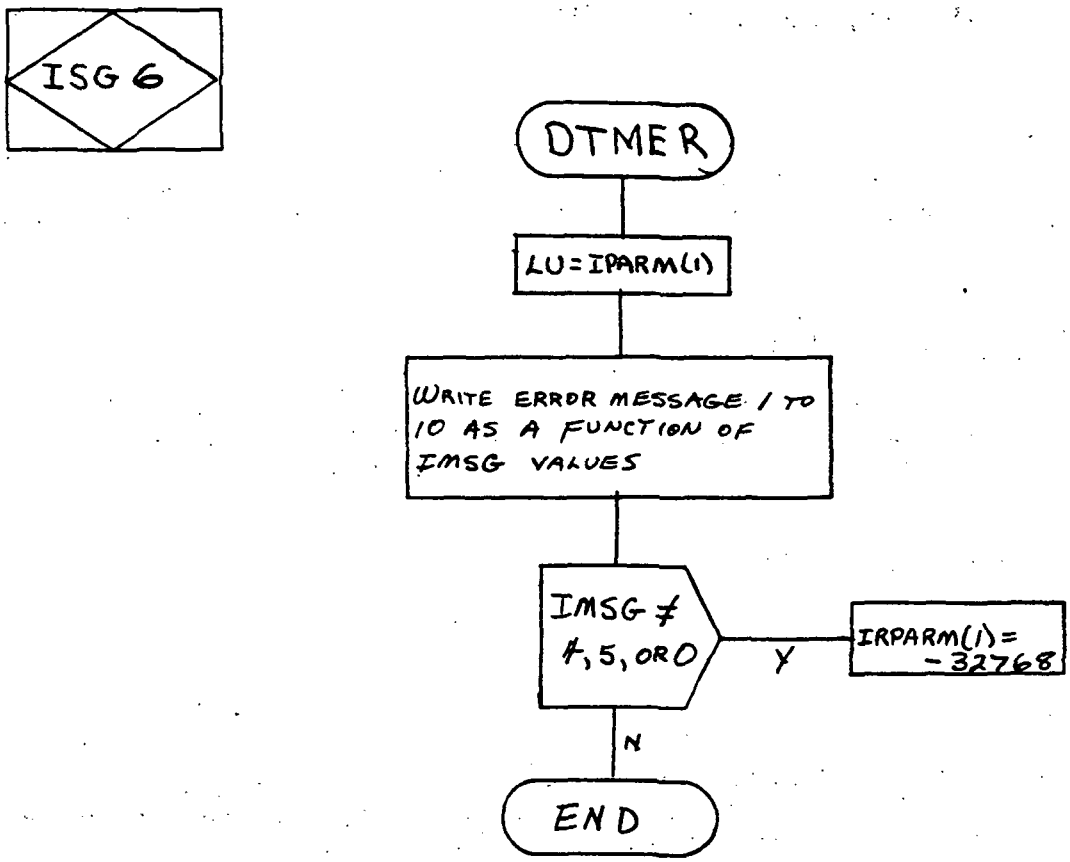


Figure 5.29-1.- Functional logic flow for the DTMER computational routine.

5.30 ROUTINE NAME - DTT7 COMPUTATIONAL ROUTINE - SEGMENT 7

5.30.1 Purpose

The DTT7 routine is used to propagate a state vector to the ignition time or to entry interface. Along with the routine UPDTV it is the computational segment 7.

5.30.2 Functional Description

The DTT7 routine calls the UPDTV routine to set up the inputs to the AEG propagator. DTT7 supports a propagation from the current state to a time of ignition or a propagation from deorbit burnout to the entry altitude.

5.30.3 Assumptions and Limitations

None.

5.30.4 Method

The DTT7 routine has an internal loop to locate the time of entry interface. The radius error at the entry time is nulled by adjusting the entry interface time until the propagation terminates on the input entry altitude.

5.30.5 Routine Input/Output Variables

Table 5.30-I contains the definitions of all the input/output variables for the DTT7 computational routine.

5.30.6 Functional Logic Flow

Figure 5.30-1 contains the functional logic flow for the DTT7 computational routine.

5.30.7 Diagnostics and Debug

None.

5.30.8 Special Comments

None.

5.30.9 References

None.

TABLE 5.30-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT7

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AREA		Real	I		C	DTMVEC(12)	Refer to table 5.1-I for all code symbol definitions.
CD		Real	I		C	DTMVEC(11)	
EQRAD		Real	I		C	GLOCON(23)	
IFFPTM		Intg	I		C		
POLRAD		Real	I		C	GLOCON(25)	
RDA		Real	I		C		
REIS		Real	I		C		
RENK		Real	I/O		C		
TIG		Real	I		C		
TIMEC		Real	I		C	DTMVEC(1)	
TP		Real	I		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.30-I.- Continued

Routine DTTZ

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TT		Real	I/O		C		
VDA		Real	I		C		
WBO		Real	I		C		
WT		Real	I		C	DTMVEC(13)	
XYZI		Real	I		C	DTMVEC(2)	
XYZID		Real	I		C	DTMVEC(5)	
DTCOST		Real	O		C		
HEINK		Real	O		C		
HEIS		Real	O		C		
RAAA		Real	O		C		
REISM		Real	O		C		
RTA		Real	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.30-I.- Concluded

Routine DTT7

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TEINK		Real	O		C		
VA		Real	O		C		
VENK		Real	O		C		
VTAA		Real	O		C		
XOFCN		Real	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

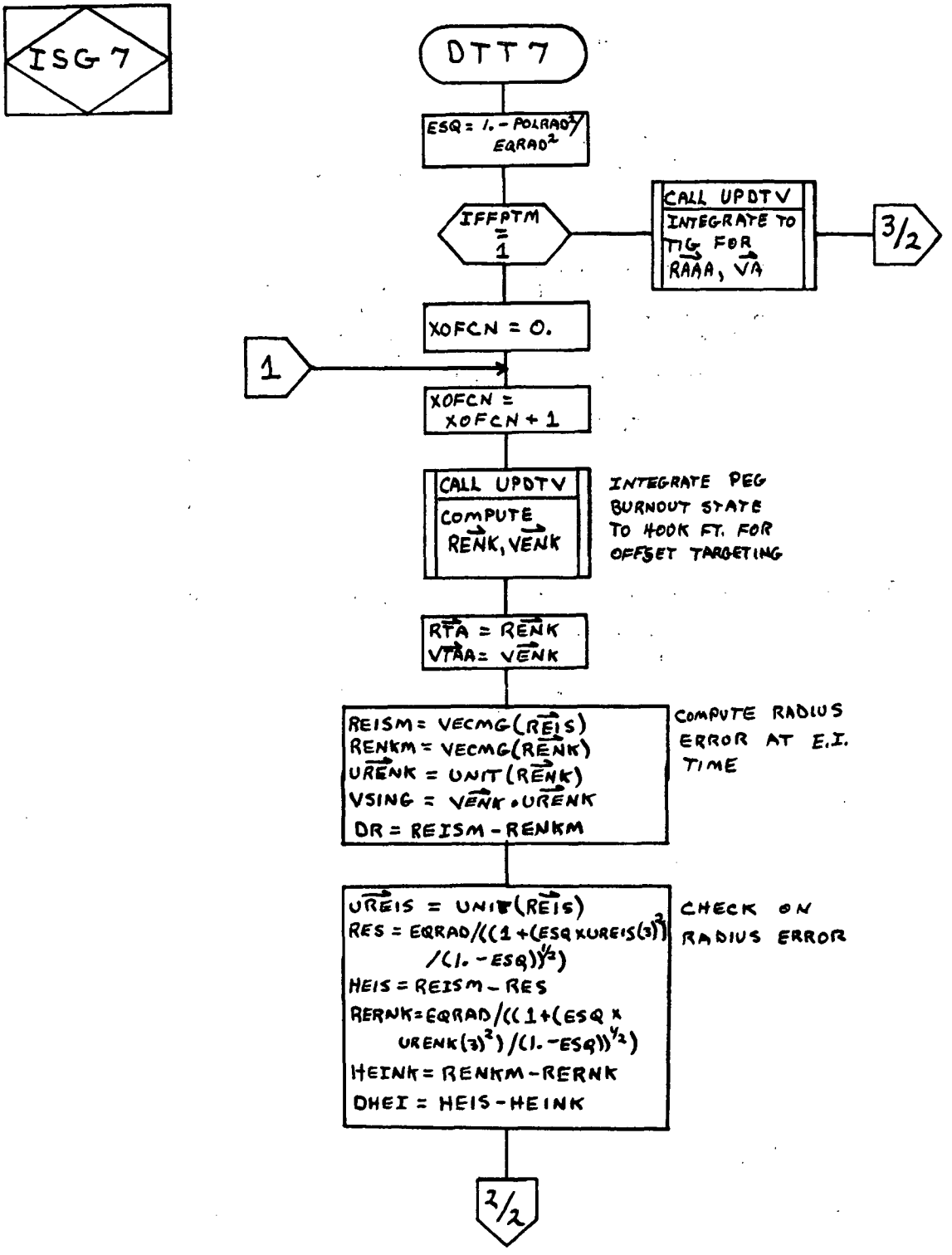


Figure 5.30-1.- Functional logic flow for the DTT7 computational routine.

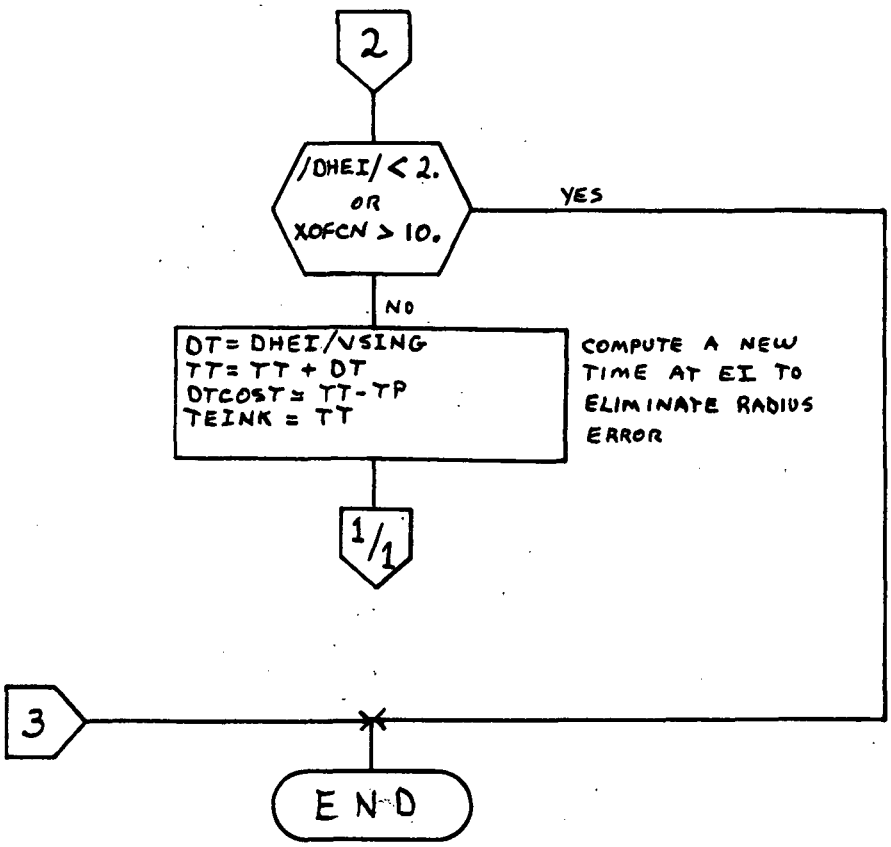


Figure 5.30-1.- Concluded.

5.31 ROUTINE NAME - UPDTV COMPUTATIONAL ROUTINE

5.31.1 Purpose

The UPDTV routine sets up the inputs to and calls the AEG propagator.

5.31.2 Functional Description

UPDTV constructs an array of inputs called AEGCOM containing the current state, the time to propagate to, and all necessary constants for the AEG propagator. It then calls the AEG propagator and passes back the final state when completed.

5.31.3 Assumptions and Limitations

None.

5.31.4 Method

None.

5.31.5 Routine Input/Output Variables

Table 5.31-I contains the definitions of all the input/output variables for the UPDTV computational routine.

5.31.6 Functional Logic Flow

Figure 5.31-1 contains the functional logic flow for the UPDTV computational routine.

5.31.7 Diagnostics and Debug

When the user selects the debug print option the initial and final state is output as R, V, T, RF, VF, and TF.

5.31.8 Special Comments

None.

5.31.9 References

None.

TABLE 5.31-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine UPDTV

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AMILE		Real	I		C	GLOCON(89)	Refer to table 5.1-I for all code symbol definitions.
AREA		Real	I		A		Vehicle area
CD		Real	I		A		Vehicle drag coefficient
EQRAD		Real	I		C	GLOCON(23)	Refer to table 1.2-III of volume VI for all code symbol definitions.
HALFPI		Real	I		C	GLOCON(63)	Refer to table 1.2-III of volume VI for all code symbol definitions.
PI		Real	I		C	GLOCON(59)	Refer to table 1.2-III of volume VI for all code symbol definitions.
RADIAN		Real	I		C	GLOCON(83)	Refer to table 1.2-III of volume VI for all code symbol definitions.
RE		Real	I		C	GLOCON(21)	Refer to table 1.2-III of volume VI for all code symbol definitions.
RS		Real	I		A		*Position state vector*
SRMU		Real	I		C	GLOCON(3)	
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.31-I.- Continued

Routine UPDTV

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TS		Real	I		A		Initial state time
TF		Real	I/O		A/A,T		Final state time
TWOPI		Real	I		C	GLOCON(61)	
VS		Real	I		A		Velocity state vector
WE		Real	I		C	GLOCON(13)	
WHT		Real	I		A		Vehicle weight
XJ2		Real	I		C	GLOCON(31)	Refer to table 5.2.11.5-I.
XJ3		Real	I		C	GLOCON(33)	Refer to table 5.2.11.5-I.
XJ4		Real	I		C	GLOCON(35)	Refer to table 5.2.11.5-I.
XJ22		Real	I		C	GLOCON(37)	Refer to table 5.2.11.5-I.
XJ31		Real	I		C	GLOCON(39)	Refer to table 5.2.11.5-I.
XLAM22		Real	I		C	GLOCON(41)	Refer to table 5.2.11.5-I.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.31-I.- Concluded

Routine UPDTV

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
XLAM31		Real	I		C	GLOCON(43)	Refer to table 5.2.11.5-I.
XMU		Real	I		C	GLOCON(1)	Refer to table 5.2.11.5-I.
RF		Real	O		A,T		Final position state
VF		Real	O		A,T		Final velocity state
IPOUT		6CH	I		C		
IOUNIT		Intg	I		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

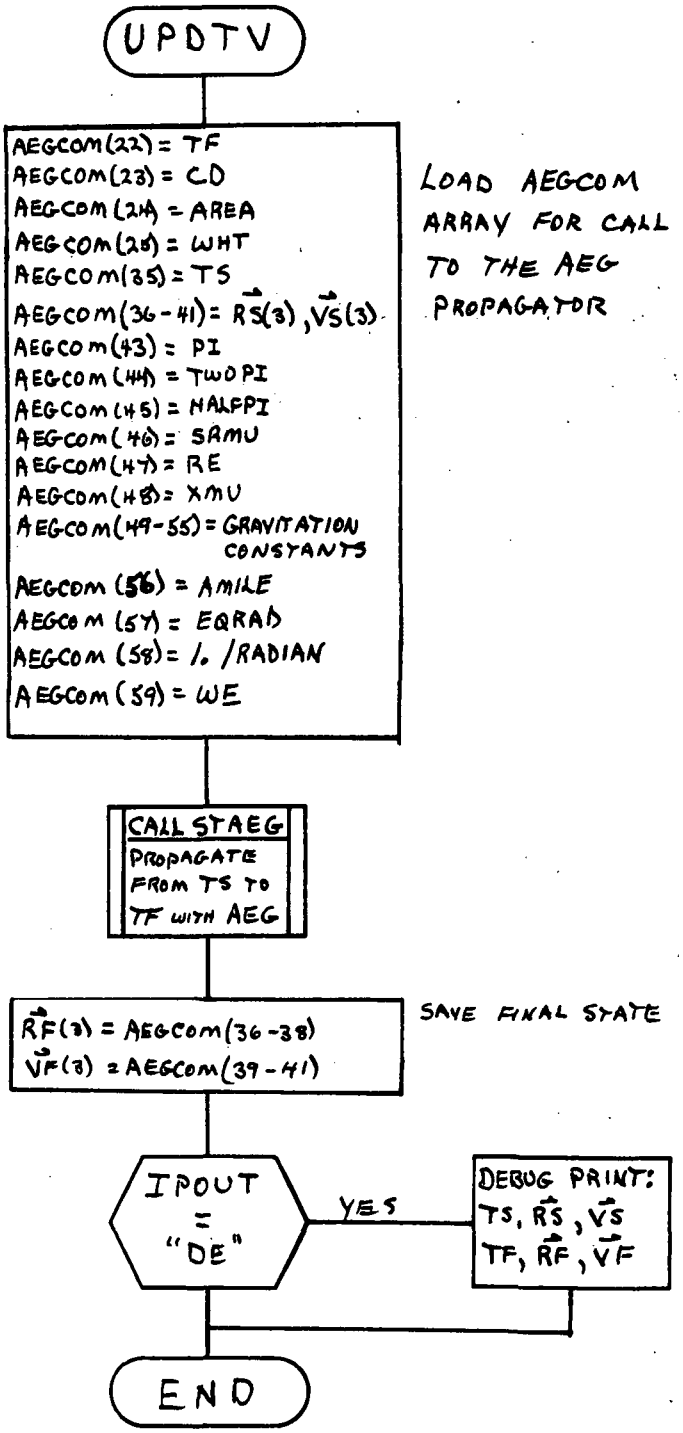


Figure 5.31-1.- Functional logic flow for the UPDATV computational routine.

5.32 ROUTINE NAME - DTMPR PRINT ROUTINE - SEGMENT 9

5.32.1 Purpose

The DTMPR routine produces the user display output for DTM. Along with the ST routine it is computational segment 9.

5.32.2 Functional Description

DTMPR produces the primary and backup thrust system output displays. It uses a call to the ST routine to compute orbital elements at ignition and entry interface times, converts the ignition, burnout, and entry interface times to hours, minutes, and seconds, converts ranges to nautical miles, prints the output block, and then converts all parameter units back to the internal units set.

5.32.3 Assumptions and Limitations

None.

5.32.4 Method

None.

5.32.5 Routine Input/Output Variables

Table 5.32-I contains the definitions of all the input/output variables for the DTMPR computational routine.

5.32.6 Functional Logic Flow

Figure 5.32-1 contains the functional logic flow for the DTMPR computational routine.

5.32.7 Diagnostics and Debug

None.

5.32.8 Special Comments

None.

5.32.9 References

None.

TABLE 5.32-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTMPR

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AMILE		Real	I		C	GLOCON(89)	Refer to table 5.1-I for all code symbol definitions.
CA		Real	I/O		C,T	C1, C2	
CRSRNG		Real	I/O	n. mi.	C,T		
DTCOST		Real	I/O		C,T	TFF	
DVBU		Real	I/O		C		
DVPR		Real	I/O		C		
DVIMP		Real	I/O		C,T		
DVOBBU		Real	I/O		C,T	DVOB	
DVOBPR		Real	I/O		C,T	DVOB	
DWBU		Real	I/O		C,T	DW	
DWPR		Real	I/O		C,T	DW	

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.32-I.- Continued

Routine DTMPR

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
EIALT		Real	I		C	GLOCON(57)	
ETGRNG		Real	I/O	n.mi.	C,T		
GAMETG		Real	I/O		C,T		
IABT		Intg	I/O		C,T		
IBTR		Intg	I/O		C,T		
ICALLL		Intg	I/O		C,T	ICALL	
IFINAL		Intg	I/O		C,T		
IFUEL		Intg	I/O		C,T		
IGAMFR		Intg	I/O		C,T		
IMPULS		Intg	I/O		C,T		
INGBU		Intg	I/O		C,T		
INGPR		Intg	I/O		C,T		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.32-I.- Continued

Routine DTMPR

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IOUNIT		Intg	I		C		
IPOUT		6CH	I		C		
IPARM		Intg	I		C		
IPRNG		Intg	I/O		C,T		
IPLACE		Intg	I/O		C,T	IPLCE	
ITIGFR		Intg	I/O		C,T		
ITFF		Intg	I/O		C,T		
ITIGG		Intg	I/O		C,T		
JJ		Intg	I/O		C,T		
JNGPR		6CH	I/O		C,T	JNG	
JNGBU		6CH	I/O		C,T	JNG	
J2FLG		Intg	I/O		C,T		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.32-I.- Continued

Routine DTMPR

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
KDGUID		Intg	I/O		C,T		
KDSTER		Intg	I/O		C,T		
NCNT		Intg	I/O		C,T		
NCNT1		Intg	I/O		C,T		
NCNT2		Intg	I/O		C,T		
NCNT3		Intg	I/O		C,T		
NCNT4		Intg	I/O		C,T		
NSYS		Intg	I		C		
MITER		Intg	I/O		C,T		
PSIBU		Real	I/O		C,T	PSI	
PSIPR		Real	I/O		C,T	PSI	
RAAA		Real	I/O		C,T		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.32-I.- Continued

Routine DTMPR

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
RDA		Real	I/O		C		
REIS		Real	I/O		C		
RENK		Real	I/O		C		
RTA		Real	I/O		C		
SESCON		Free	I		C		
TIG		Real	I/O		C		
TGO		Real	I/O		C,T	TBURN	
THETEI		Real	I/O		C,T		
THETLS		Real	I/O		C		
THELSD		Real	I/O		C		
TP		Real	I/O		C		
TPCHB		Real	I/O		C,T	TPCH	

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.32-I.- Continued

Routine DTMPR

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TPCHP		Real	I/O		C,T	TPCH	
TRANG		Real	I/O	n. mi.	C,T	TRANGE	
TT		Real	I/O		C		
TYAWB		Real	I/O		C,T	TYAW	
TYAWP		Real	I/O		C,T	TYAW	
VA		Real	I/O		C		
VDA		Real	I/O		C		
VEIS		Real	I/O		C		
VENK		Real	I/O		C		
VEIM		Real	I/O		C,T	VEIETG	
VG		Real	I/O		C,T		
VGMAG		Real	I/O		C,T		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.32-I.- Continued

Routine DTMPR

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
VGOX		Real	I/O		C,T		
VGOY		Real	I/O		C,T		
VGOZ		Real	I/O		C,T		
VTAA		Real	I/O		C		
WBO		Real	I/O		C,T		
WCG		Real	I/O		C,T		
WT		Real	I		C	DTMVEC(13)	
XYZI		Real	I		C	DTMVEC(2)	
XYZID		Real	I		C	DTMVEC(5)	
ZEROT		Real	I/O		C		
PVNAME		6CH	O		T	PVTAB	Position/velocity phase table name
STNAME		6CH	O		T	SUMTAB	Summary table name

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.32-I.- Continued

Routine DTMPR

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
EIGAM		Real	O		T	GAMEI	Entry interface flightpath angle
EILAT		Real	O		T		Entry interface latitude
EILONG		Real	O		T		Entry interface longitude
TGLAT		Real	O		T		TIG latitude
TGLONG		Real	O		T		TIG longitude
VEIMG		Real	O		T		Entry interface velocity magnitude
IHTIGG		Intg	O		T	TIGGET	Hours, TIG GET
IMTIGG		Intg	O		T	TIGGET	Minutes, TIG GET
STIGG		Real	O		T	TIGGET	Seconds, TIG GET
IHTBOO		Intg	O		T	TBOGET	Hours, burnout GET
IMTBOO		Intg	O		T	TBOGET	Minutes, burnout GET
STBOO		Real	O		T	TBOGET	Seconds, burnout GET
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.32-I.- Concluded

Routine DTMPR

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
IHTEII		Intg	O		T	TEIGET	Entry interface GET
IMTEII		Intg	O		T	TEIGET	Minutes, entry interface GET
STEII		Real	O		T	TEIGET	Seconds, entry interface GET

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

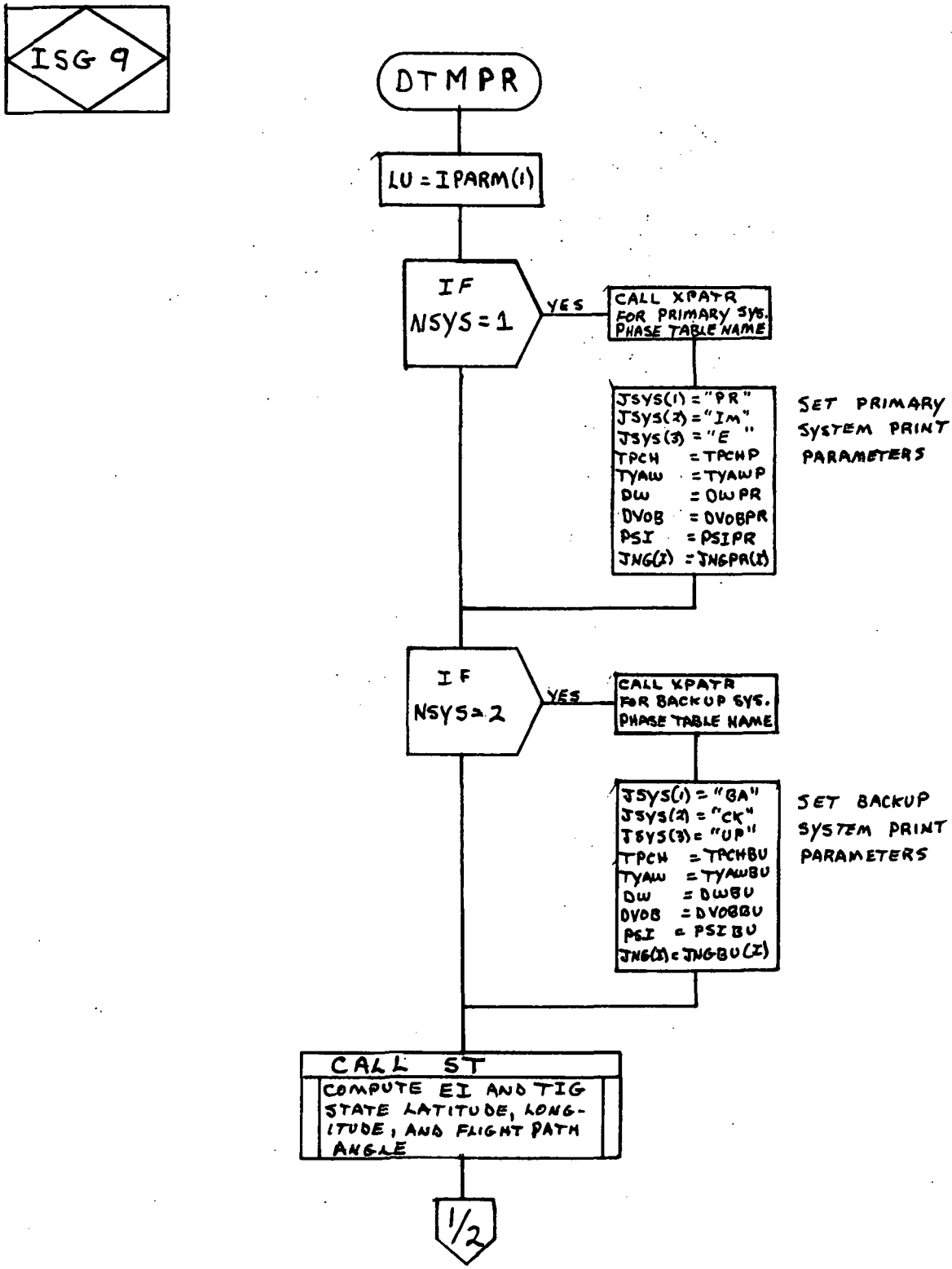


Figure 5.32-1.- Functional logic flow for the DTMPR computational routine.

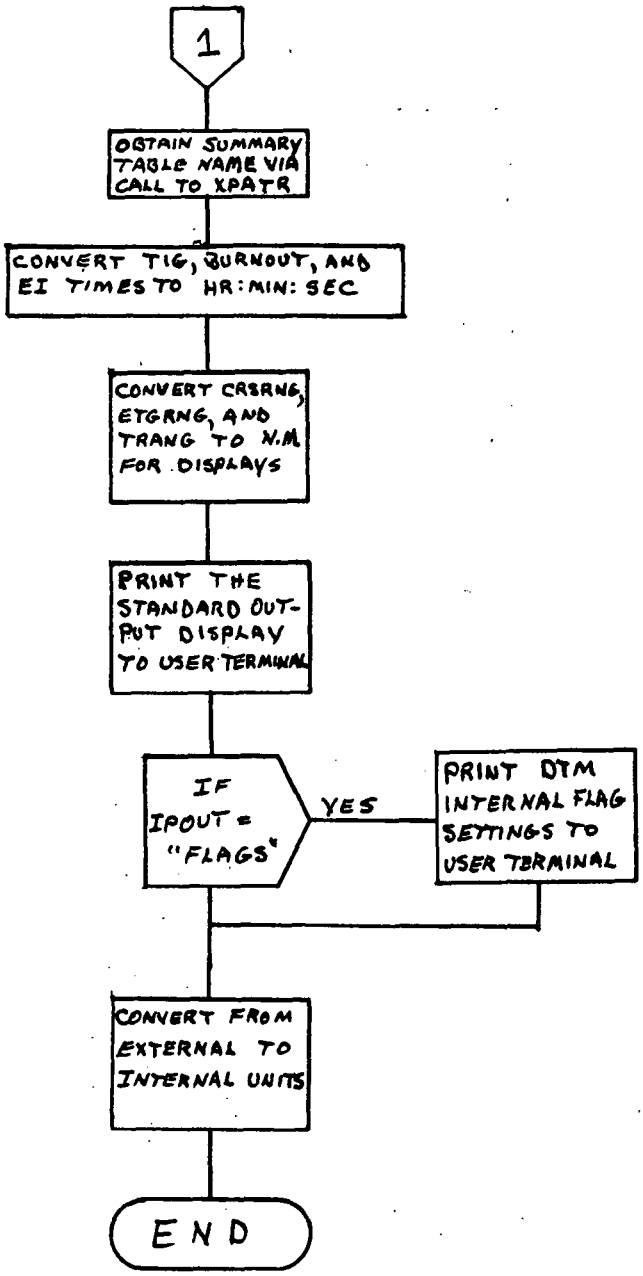


Figure 5.32-1.- Concluded.

5.33 ROUTINE NAME - ST COMPUTATIONAL ROUTINE

5.33.1 Purpose

The ST routine computes an orbital element vector from an XYZ vector.

5.33.2 Functional Description

The ST routine computes the radius, velocity, flightpath angle, latitude, longitude, and PSI angle from an input XYZ vector.

5.33.3 Assumptions and Limitations

None.

5.33.4 Method

None.

5.33.5 Routine Input/Output Variables

Table 5.33-I contains the definitions of all the input/output variables for the ST computational routine.

5.33.6 Functional Logic Flow

Figure 5.33-1 contains the functional logic flow for the ST computational routine.

5.33.7 Diagnostics and Debug

None.

5.33.8 Special Comments

None.

5.33.9 References

None.

TABLE 5.33-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine ST

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ERAI		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
PI		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
TWOPI		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
WE		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
XR		Real	I		A		Position vector
XT		Real	I		A		Vector time
XV		Real	I		A		Velocity vector
XRM		Real	O		A		Radius
XVM		Real	O		A		Velocity
XLAT		Real	O		A		Latitude
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.33-I.- Concluded

Routine ST

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
XLON		Real	O		A		Longitude
XGAM		Real	O		A		Flightpath angle
XPSI		Real	O		A		PSI angle
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

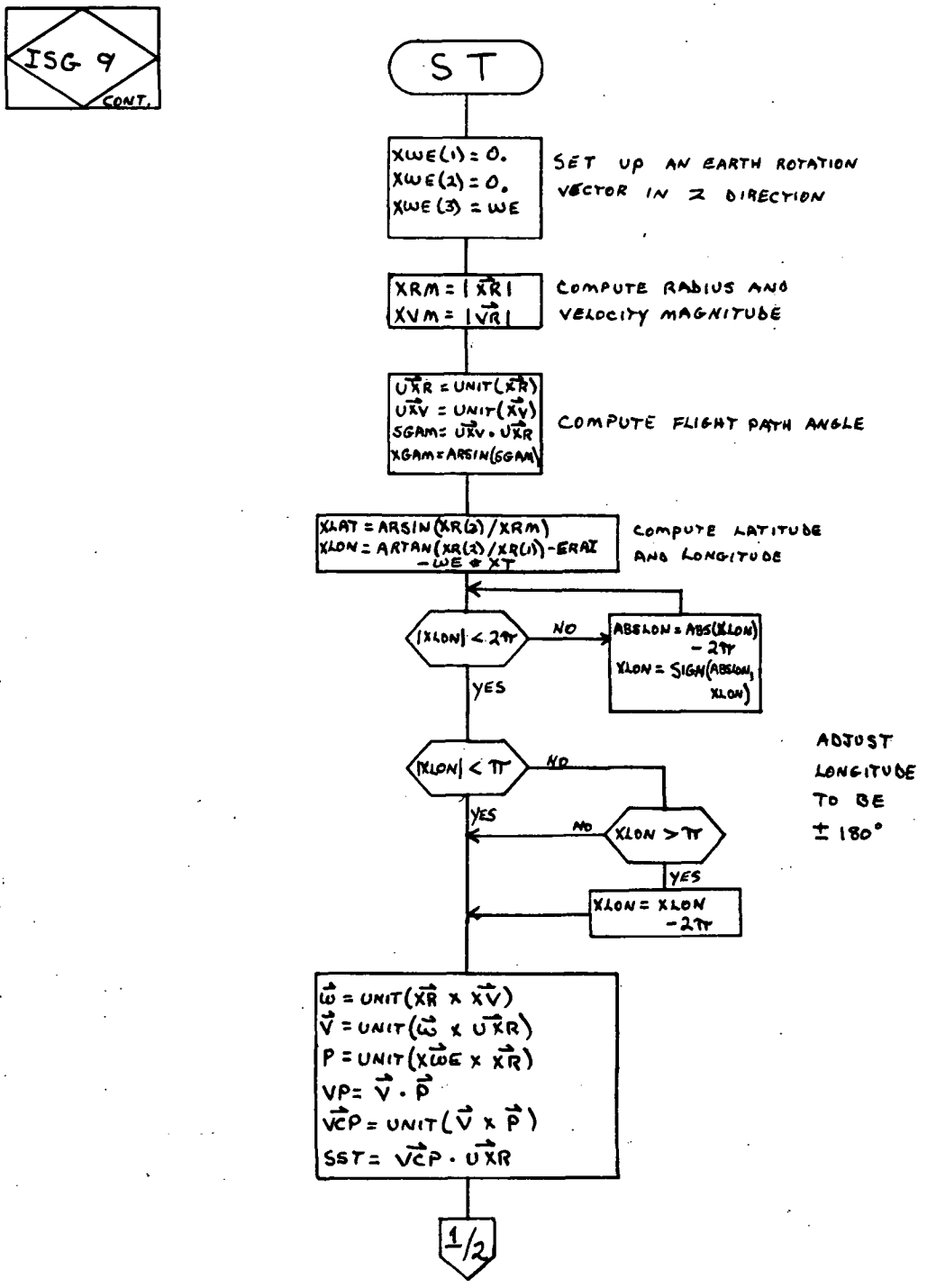


Figure 5.33-1.- Functional logic flow for the ST computational routine.

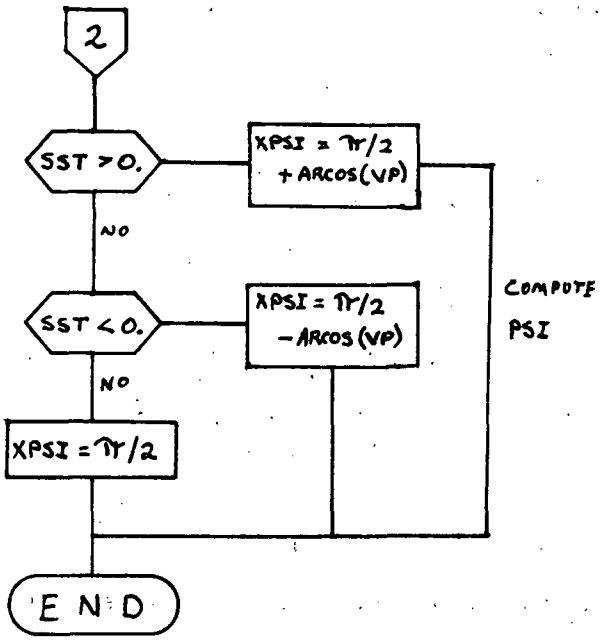


Figure 5.33-1.- Concluded.

5.34 ROUTINE NAME - DTT11 COMPUTATIONAL ROUTINE - SEGMENT 11

5.34.1 Purpose

The DTT11 routine uses LTVCN to compute an impulsive burn deorbit solution. Along with LTVCN it is computational segment 11.

5.34.2 Functional Description

The DTT11 routine first computes a unit vector normal to the orbit plane for use in doing a conic transfer. The LTVCN routine is called to compute an impulsive burn for the conic deorbit solution. Finally, the primary system OMS fuel expended is computed along with the time of entry interface.

5.34.3 Assumptions and Limitations

The DTT11 produces an impulsive solution for a conic transfer deorbit to a target entry velocity.

5.34.4 Method

None.

5.34.5 Routine Input/Output Variables

Table 5.34-I contains the definitions of all the input/output variables for the DTT11 computational routine.

5.34.6 Functional Logic Flow

Figure 5.34-1 contains the functional logic flow for the DTT11 computational routine.

5.34.7 Diagnostics and Debug

When the user selects the debug output option the following parameters are printed; LTVCN arguments RAAA, RTA, UIN, CA, VTGIMP, VTAA, DTCOST; DTT11 quantities DWPR, THEBO, TT, DVIMP, VGOIMP, VDA, and RDA.

5.34.8 Special Comments

None.

5.34.9 References

None.

TABLE 5.34-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT11

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
CA		Real	I/O		C,T		Refer to table 5.1-I for all code symbol definitions.
DTCOST		Real	I/O		C,T		
FPR		Real	I		C		
IPOUT		6CH	I		C		
IOUNIT		Intg	I		C		
RAAA		Real	I/O		C,T		
RTA		Real	I/O		C,T		
STP		Real	I		C	GLOCON(117)	
TIG		Real	I		C		
VA		Real	I		C		
WDPR		Real	I		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.34-I.- Concluded

Routine DT11

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
WT		Real	I		C	DTMVEC(13)	
DVIMP		Real	O		C,T		
DWPR		Real	O		C,T		
IABT		Intg	O		C		
RDA		Real	O		C,T		
THEBO		Real	O		C,T		
TT		Real	O		C,T		
VDA		Real	O		C,T		
UFIMP		Real	O		C		
VTAA		Real	O		C,T		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

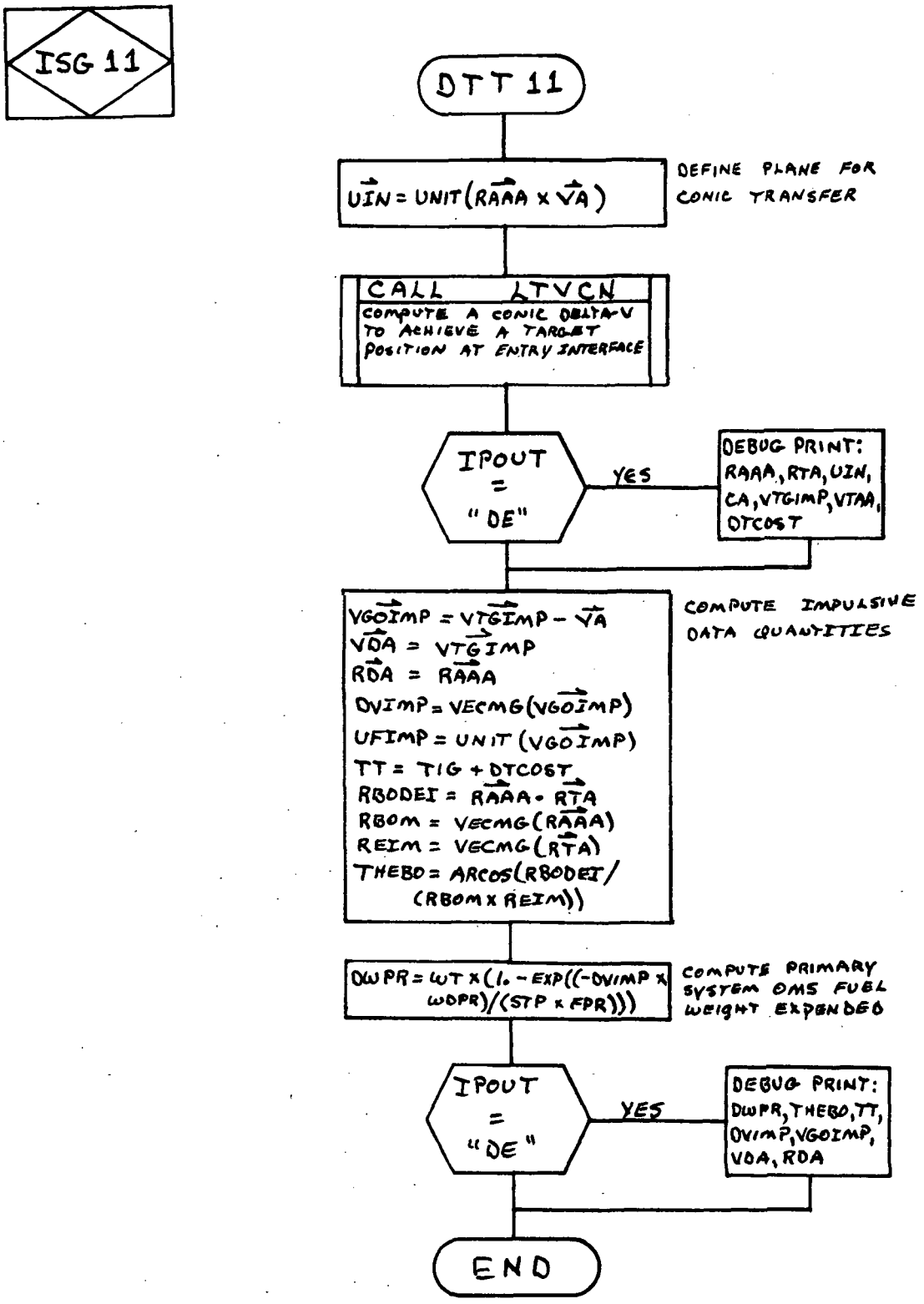


Figure 5.34-1.- Functional logic flow for the DTT11 computational routine.

5.35 ROUTINE NAME - LTVCN COMPUTATIONAL ROUTINE

5.35.1 Purpose

The LTVCN routine computes the velocity vector required at an initial time and position to intercept the target position vector and velocity vector at entry.

5.35.2 Functional Description

LTVCN determines the velocity required at an initial position to intercept a target position such that a specified linear relationship between the radial and horizontal velocity components is met. The linear relationship is given by Vradial = C1 + C2 * Vhorizontal. C1 and C2 are input targets and the solution is obtained from a set of two-body differential equations of motion.

5.35.3 Assumptions and Limitations

The LTVCN routine computations assume a point mass in an inverse squared force field.

5.35.4 Method

None.

5.35.5 Routine Input/Output Variables

Table 5.35-I contains the definitions of all the input/output variables for the LTVCN computational routine.

5.35.6 Functional Logic Flow

Figure 5.35-1 contains the functional logic flow for the LTVCN computational routine.

5.35.7 Diagnostics and Debug

None.

5.35.8 Special Comments

None.

5.35.9 References

None.

TABLE 5.35-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine LTVCN

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
C		Real	I		A		C1 and C2 target line coefficients
RO		Real	I		A		Initial position vector
R1		Real	I		A		Entry interface position vector
XMU		Real	I		C	GLOCON(1)	Refer to table 5.1-I for all code symbol definitions.
XNUNIT		Real	I		A		Unit vector normal to orbit plane
DELTAT		Real	O		A		Time from burnout to entry interface
KFLAG		Real	O		A		PEG error flag
VO		Real	O		A		Impulsive delta-V vector
V1		Real	O		A		Entry interface velocity vector
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

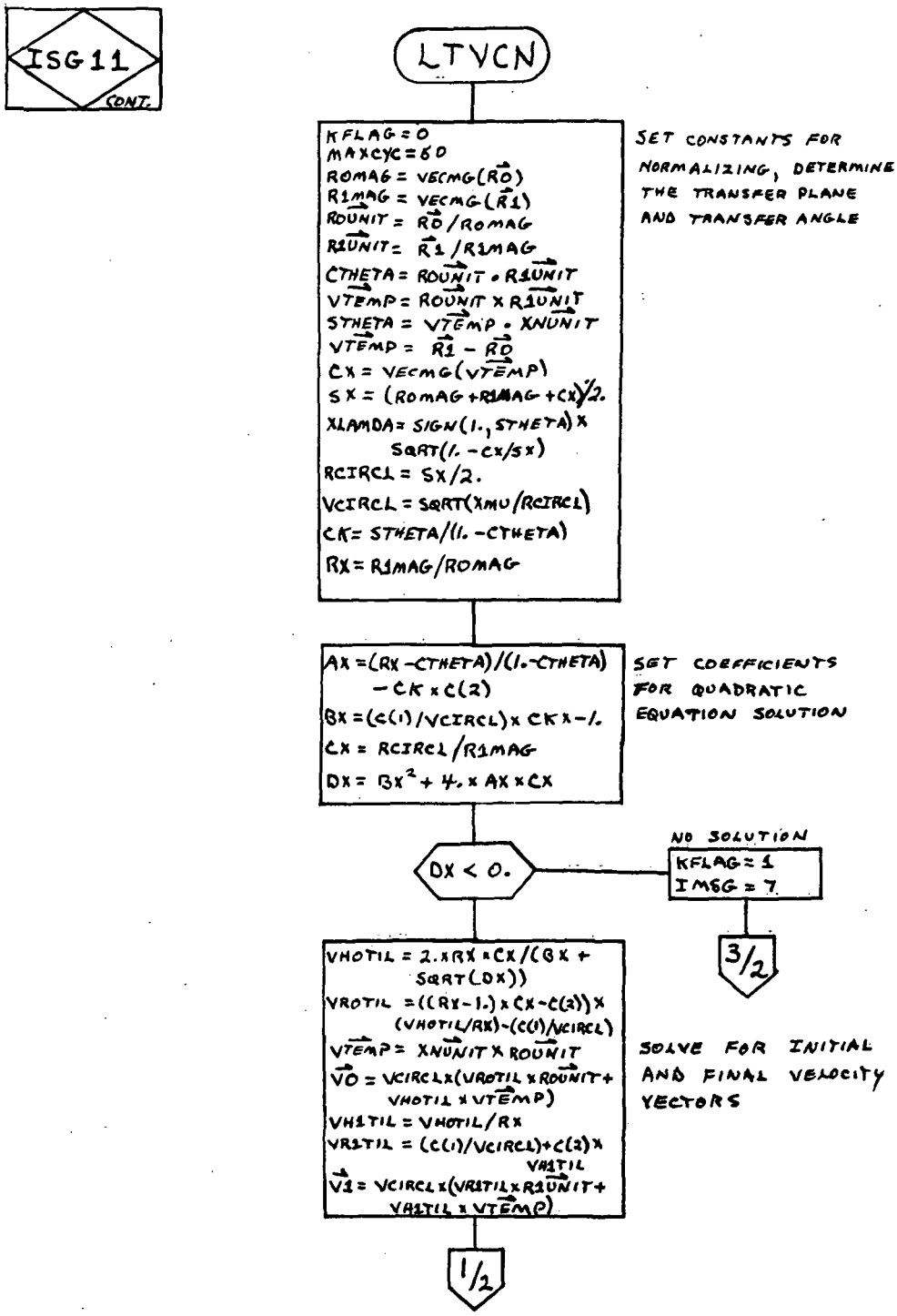


Figure 5.35-1.- Functional logic flow for the LTVCN computational routine.

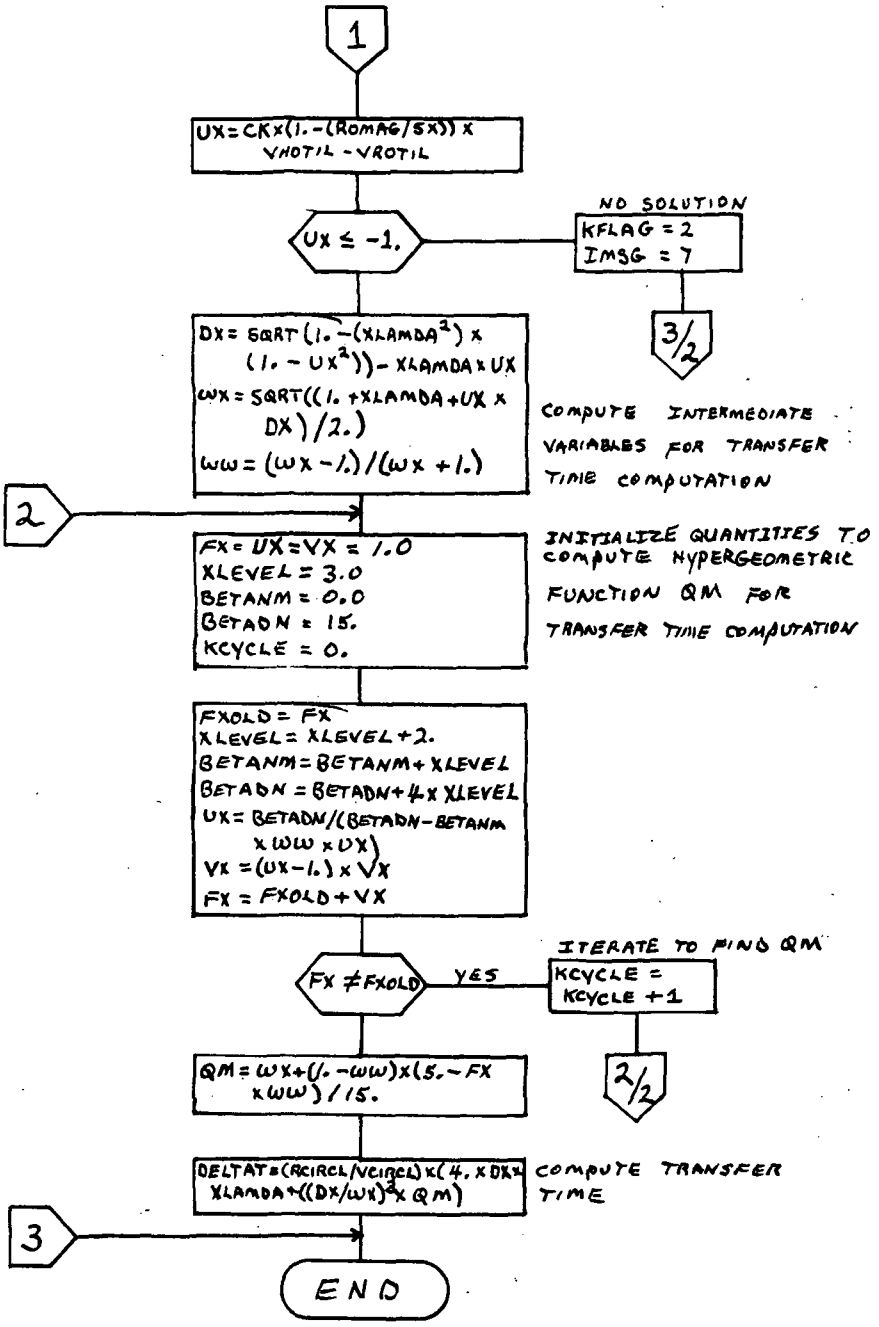


Figure 5.35-1.- Concluded.

5.36 ROUTINE NAME - DTT12 COMPUTATIONAL ROUTINE - SEGMENT 12

5.36.1 Purpose

The DTT12 routine sets up inputs for and calls the PGSUP routine to compute a targeted finite burn deorbit maneuver.

5.36.2 Functional Description

DTT12 computes thrust system quantities for input to the PEG guidance routines. It calls PGSUP to compute the finite burn deorbit maneuver using the PEG4 linear terminal velocity constraint guidance. Upon completion of the solution, DTT12 computes OMS usage and other thrust system quantities.

5.36.3 Assumptions and Limitations

None.

5.36.4 Method

None.

5.36.5 Routine Input/Output Variables

Table 5.36-I contains the definitions of all the input/output variables for the DTT12 computational routine.

5.36.6 Functional Logic Flow

Figure 5.36-1 contains the functional logic flow for the DTT12 computational routine.

5.36.7 Diagnostics and Debug

When the user specifies the debug output option a set of input and outputs to the PEG guidance is printed. The PEG inputs printed are RTIG, VTIG, WT, C1, C2, THETAT, TIG, POLE, and WCG. The PEG outputs printed are TBO, RBO, VBO, TEI, REI, VEI, WCG, VGO, TGO, FWYAW, DTCOST, DTAVG, DVBU, DYPR, DWBU, DWPR, PSIBU, PSIPR, TFFBU, TFFPR, THETEI, TIG, VMISS, VGOMAG, IMSG, IYAW, NCYC, NMAX, KDGUID, KFLAG, and KCONV.

5.36.8 Special Comments

Along with the PGSUP, PGOP3, INI1, PRDT6, CORT7, H2M50, SUPRG, and LTVC2 routines it is computational segment 12.

5.36.9 References

Ives, D.: Level C Requirements for the Mission Control Center Orbital Guidance Software. JSC IN 76-FM-100 dated December 1, 1976.

TABLE 5.36-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT12

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ATP		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
DTCOST		Real	I		C	GLOCON(57)	
EIALT		Real	I		C		
FT		Real	I		C		
FWYAW		Real	I/O		C,T		
INGBU		Intg	I		C		
INGPR		Intg	I		C		
IPOUT		6CH	I		C		
IOUNIT		Intg	I		C		
NSYS		Intg	I		C		
SBD		Real	I		C		
STP		Real	I		C	GLOCON(117)	
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.36-I.- Continued

Routine DTT12

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
THETEI		Real	I		C		
TP		Real	I		C		
VG		Real	I		C		
WCGOMS		Real	I		C		
WDBU		Real	I		C		
WDPR		Real	I		C		
WT		Real	I/O		C,T	DTMVEC(13)	
ATPBU		Real	O		C		
ATPPR		Real	O		C		
DVBU		Real	O		C,T		
DVPR		Real	O		C,T		
DWBU		Real	O		C,T		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.36-I.- Continued

Routine DTT12

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DWPR		Real	O		C,T		
HTGT		Real	O		C		
KDTHR		Intg	O		C		
KINIT		Intg	O		C		
KSTEER		Intg	O		C		
PSIBU		Real	O		C,T		
PSIPR		Real	O		C,T		
TFFBU		Real	O		C,T		
TFFPR		Real	O		C,T		
THETAT		Real	O		C		
TT		Real	O		C,T		
VGMAG		Real	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.36-I.- Continued

Routine DTT12

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
WCG		Real	O		C,T		
COM(510)		Real	O		T	TBO	Time of burnout
COM(421)		Real	O		T	RBO	Burnout position vector
COM(540)		Real	O		T	VBO	Burnout velocity vector
COM(515)		Real	O		T	TEI	Time of entry interface
COM(432)		Real	O		T	REI	E.I. position vector
COM(564)		Real	O		T	VEI	E.I. velocity vector
COM(553)		Real	O		T	VGO	Intertial delta-V
COM(495)		Real	O		T	TGO	Delta-T of burn
COM(52)		Real	O		T	DTCOST	Coast time - burnout to E.I.
COM(567)		Real	O		T	DTAVG	Gravity prediction stepsize
COM(501)		Real	O		T	THETEI	TIG to E.I. transfer angle
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.36-I.- Continued

Routine DTI12

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
COM(488)		Real	O		T	TIG	Ignition time
PO3COM(18)		Real	O		T	VMISS	Delta between predicted and actual time
COM(557)		Real	O		T	VGOMAG	Magnitude of inertial delta-V
ICOM(520)		Intg	O		T	IMSG	Error message flag
IPO3CM(1)		Intg	O		T	IYAW	FWYAW flag
IPGCOM(32)		Intg	O		T	NCYC	Number of PEG calls per guidance cycle
IPGCOM(33)		Intg	O		T	NMAX	Maximum number of guidance iterations
ICOM(786)		Intg	O		T	KDGUID	PEG guidance mode flag
IPGCOM(29)		Intg	O		T	KFLAG	PEG error flag
IPGCOM(31)		Intg	O		T	KCONV	PEG convergence flag
COM(407)		Real	O		T	RTIG	TIG position vector
COM(531)		Real	O		T	VTIG	TIG velocity vector

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.36-I.- Concluded

Routine DDT12

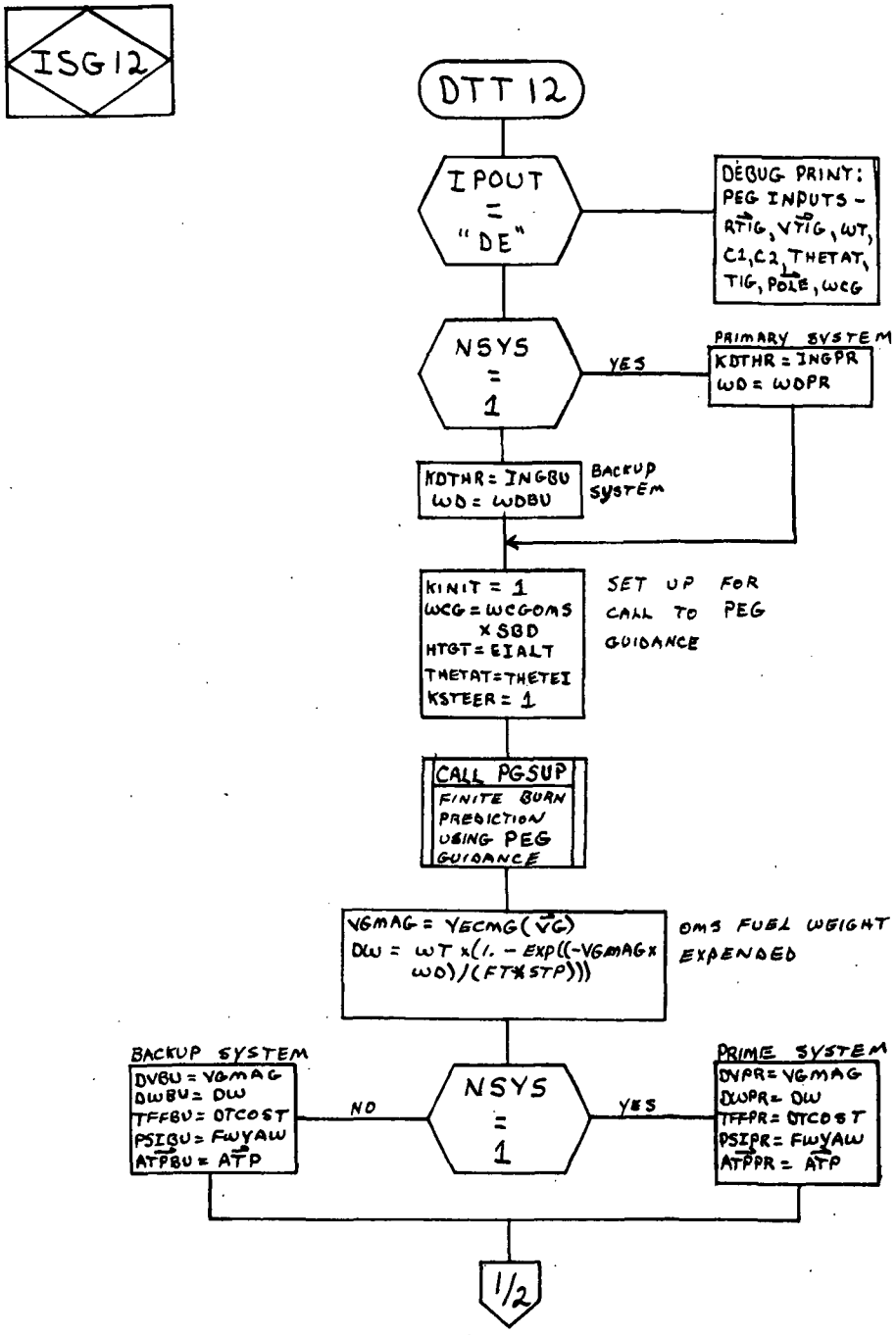
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
COM(25)		Real	O		T	C1	Target line intercept
COM(26)		Real	O		T	C2	Target line slope
COM(498)		Real	O		T	THETAT	Desired angle - TIG to E.I.
COM(92)		Real	O		T	POLE	Earth rotation axis unit vector

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory



Page 1 of 2

Figure 5.36-1.- Functional logic flow for the DTT12 computational routine.

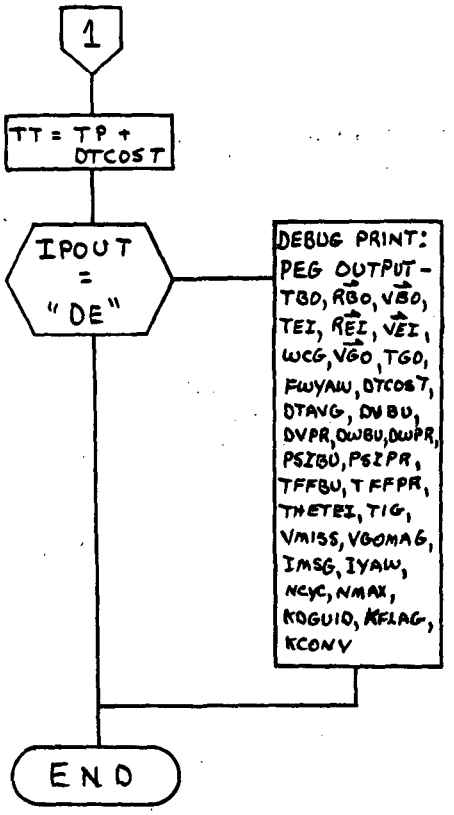


Figure 5.36-1.- Concluded.

5.37 ROUTINE NAME - PGSUP COMPUTATIONAL ROUTINE

5.37.1 Purpose

The PGSUP routine is the supervisor for the PEG guidance software that is a functional simulation of the onboard Shuttle guidance.

5.37.2 Functional Description

PGSUP establishes initial guidance flags and thrust system characteristics. It computes the guidance turning rate, vehicle acceleration, and vehicle mass characteristics. It calls H2M50 to compute the target position vector and then calls PGOP3 to compute the thrust vector for the guided deorbit. Finally, it checks for guidance convergence and recalls PGOP3 as required until the solution is converged.

5.37.3 Assumptions and Limitations

None.

5.37.4 Method

None.

5.37.5 Routine Input/Output Variables

Table 5.37-I contains the definitions of all the input/output variables for the PGSUP computational routine.

5.37.6 Functional Logic Flow

Figure 5.37-1 contains the functional logic flow for the PGSUP computational routine.

5.37.7 Diagnostics and Debug

None.

5.37.8 Special Comments

None.

5.37.9 References

None.

TABLE 5.37-1.- ROUTINE INPUT/OUTPUT VARIABLES

Routine PGSUP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DVSS		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
FOMS		Real	I		C		
FRCS		Real	I		C		
FTS		Real	I/O		C		
HTGT		Real	I		C		
IPS		Intg	I/O		C		
KSTOP		Intg	I		C		
KCUTOF		Intg	I/O		C		
KDSTER		Intg	I		C		
KDTHR		Intg	I		C		
KINIT		Intg	I		C		
KITER		Intg	I		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.37-I.- Continued

Routine PGSUP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
KMODE		Intg	I		C		
NMAX		Intg	I/O		C		
NMAX1		Intg	I		C	DTMCON(18)	
NMAX2		Intg	I		C		
RTA		Real	I/O		C		
STP		Real	I		C	GLOCON(117)	
TBRCS		Real	I		C		
TGDP		Real	I/O		C		
TGOMIN		Real	I		C	DTMCON(22)	
TTT		Real	I/O		C		
TGO		Real	I/O		C		
TIG		Real	I		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.37-I.- Continued

Routine PGSUP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TIGA		Real	I		C		
VEXS		Real	I/O		C		
VG		Real	I/O		C		
WCG		Real	I		C		
WT		Real	I		C	DTMVEC(13)	
XMDRCS		Real	I		C		
XMDOMS		Real	I		C		
XLD		Real	I		C		
ATR		Real	O		C		
CLMDXZ		Real	O		C		
FT		Real	O		C		
IMSG		Intg	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.37-I.- Concluded

Routine PGSUP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
KFLAG		Intg	O		C		
KFUELD		Intg	O		C		
NCYC		Intg	O		C		
NOMS		Intg	O		C		
NPHASE		Intg	O		C		
TRCSOF		Real	O		C		
VEXX		Real	O		C		
VGOP		Real	O		C		
VSP		Real	O		C		
XMASSE		Real	O		C		
XMBO		Real	O		C		
XMDOT		Real	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

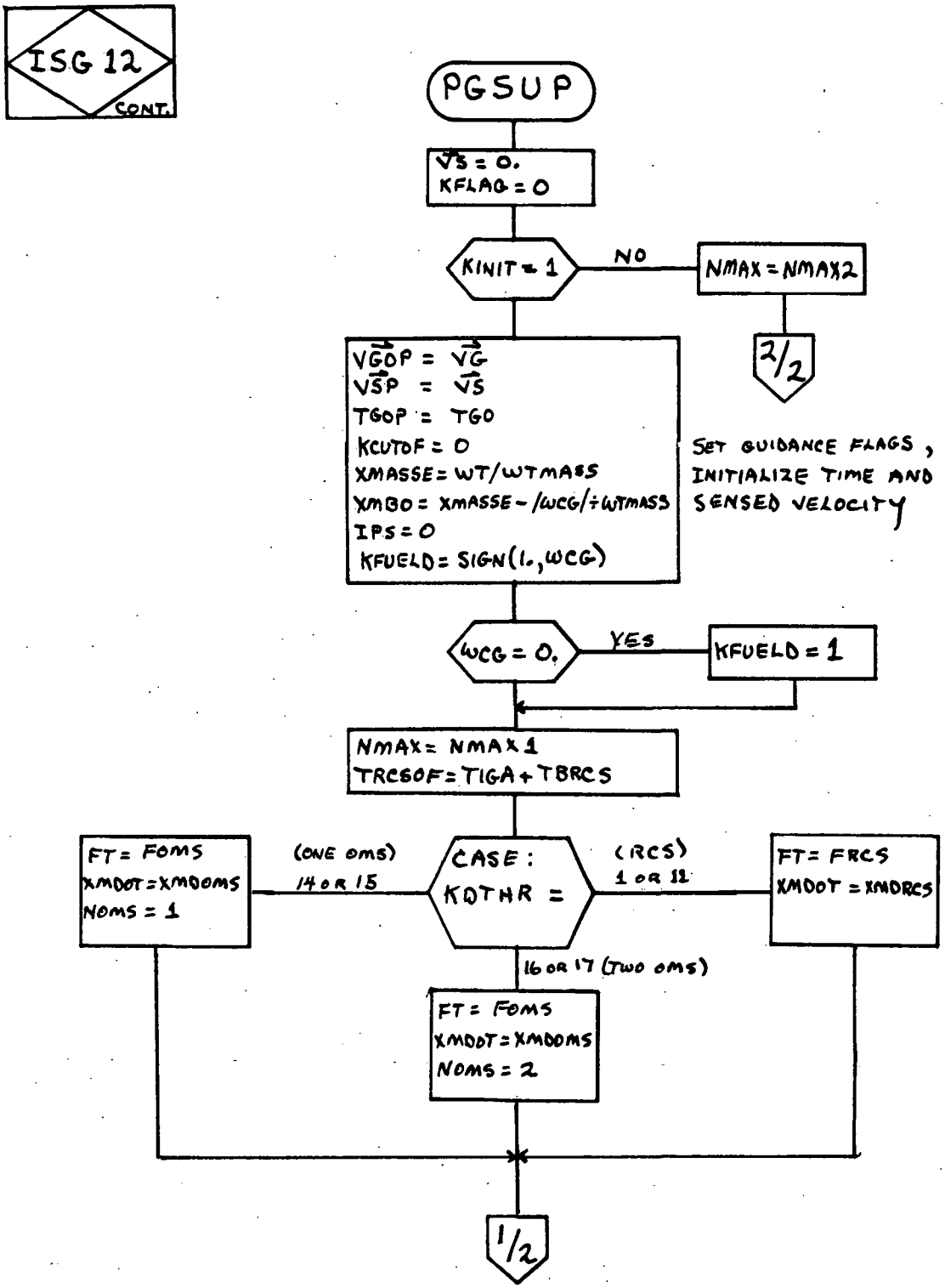


Figure 5.37-1.- Functional logic flow for the PGSUP computational routine.

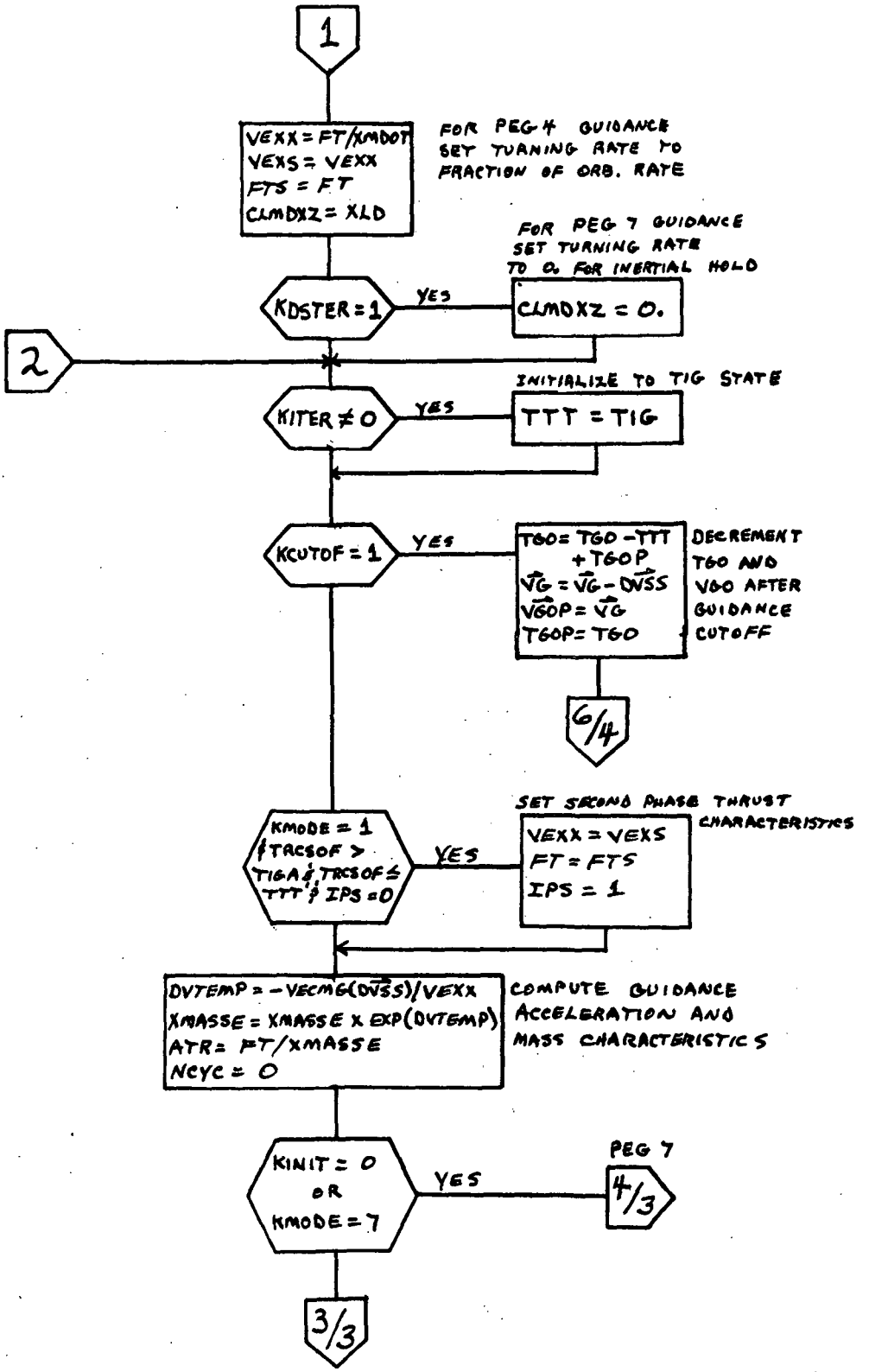


Figure 5.37-1.- Continued.

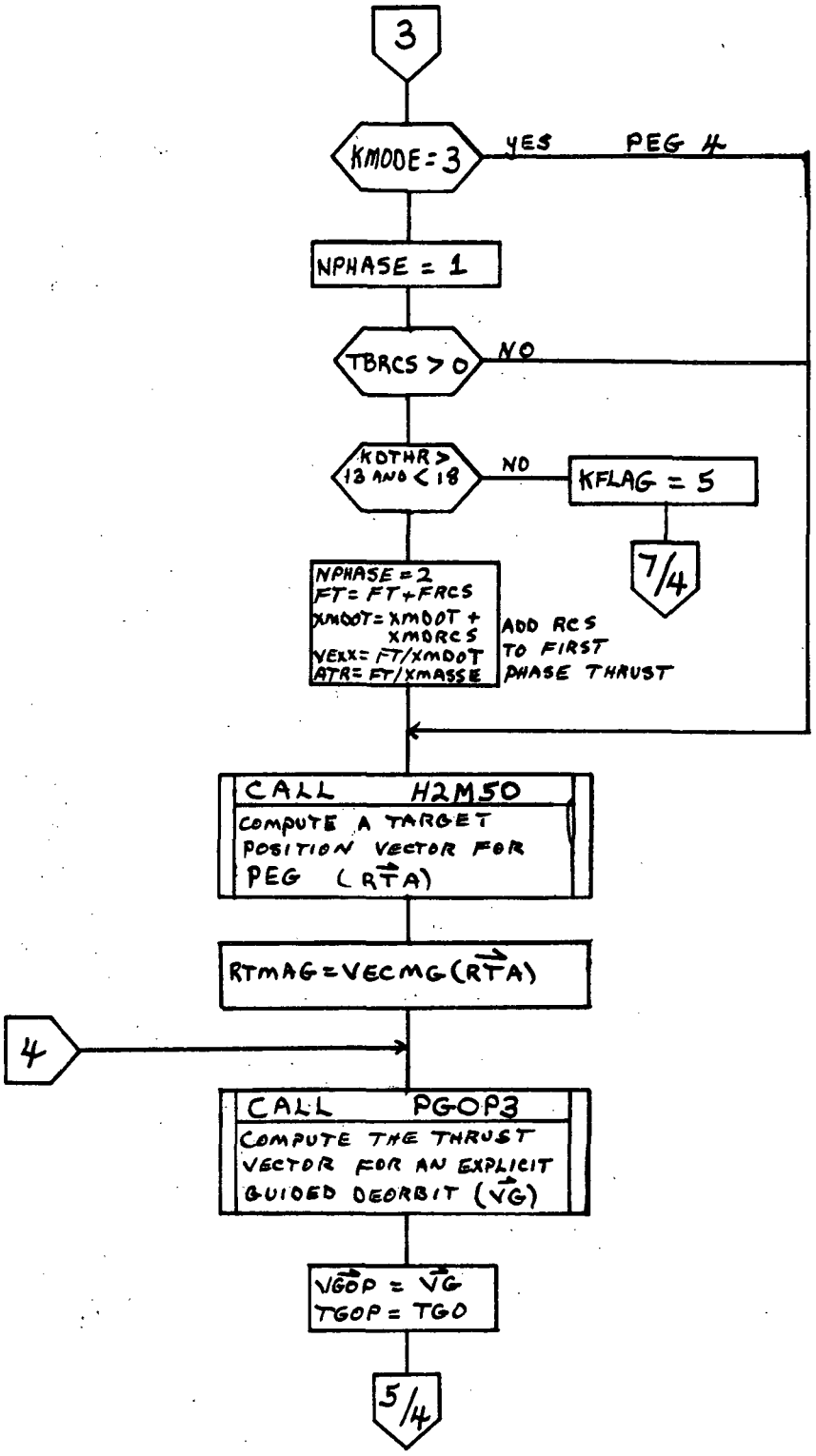


Figure 5.37-1.- Continued.

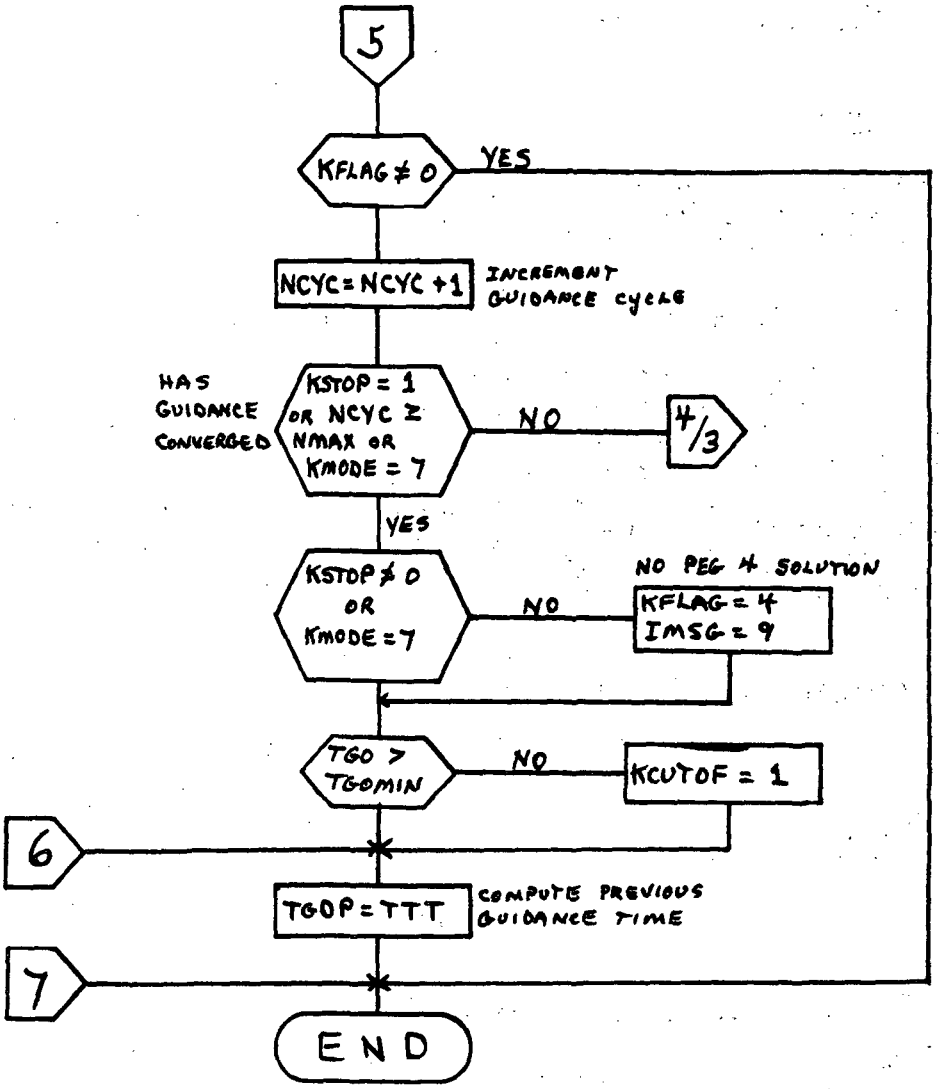


Figure 5.37-1.- Concluded.

5.38 ROUTINE NAME - H2M50 COMPUTATIONAL ROUTINE

5.38.1 Purpose

The H2M50 routine computes a target position vector from an input target entry altitude and range angle from the launch site to the target.

5.38.2 Functional Description

The H2M50 routine uses the Tig position and velocity vector to compute a vector normal to the orbit plane from which the down range and radial unit vectors are computed. The down range and radial unit vectors are then used to compute a unit position vector located down range by the inplane range angle theta. The HELIP function is used to compute the magnitude of the entry interface radius with the input entry altitude established relative to the Fischer ellipsoid. Finally, the target XYZ position vector is computed from the position unit vector and the radius magnitude.

5.38.3 Assumptions and Limitations

None.

5.38.4 Method

None.

5.38.5 Routine Input/Output Variables

Table 5.38-I contains the definitions of all the input/output variables for the H2M50 computational routine.

5.38.6 Functional Logic Flow

Figure 5.38-1 contains the functional logic flow for the H2M50 computational routine and the HELIP function routine.

5.38.7 Diagnostics and Debug

None.

5.38.8 Special Comments

5.38.9 References

None.

TABLE 5.38-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine H2M50

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ELLIPT		Real	I		C	GLOCON(27)	Refer to table 5.1-I for definition not supplied here.
EQRAD		Real	I		C	GLOCON(23)	
HTGT		Real	I		A		E.I. target altitude
KABORT		Intg	I		C		
KMODE		Intg	I		C		
POLE		Real	I		C	DTMCON(39)	
RAAA		Real	I		C		
THETA		Real	I		A		Angle - launch site to target
VA		Real	I		C		
DTHETA		Real	O		A		Angle - current position to target
RT		Real	O		A		Target position vector
UYD		Real	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.38-I.- Concluded

Routine HELIP

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ELLIPT		Real	I		A		Earth flattening coefficient
EQRAD		Real	I		A		Earth equatorial radius
POLE		Real	I		A		Earth rotation axis unit vector
R		Real	I		A		Position vector
HELIP		Real	O		A		Altitude above Fischer ellipsoid
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

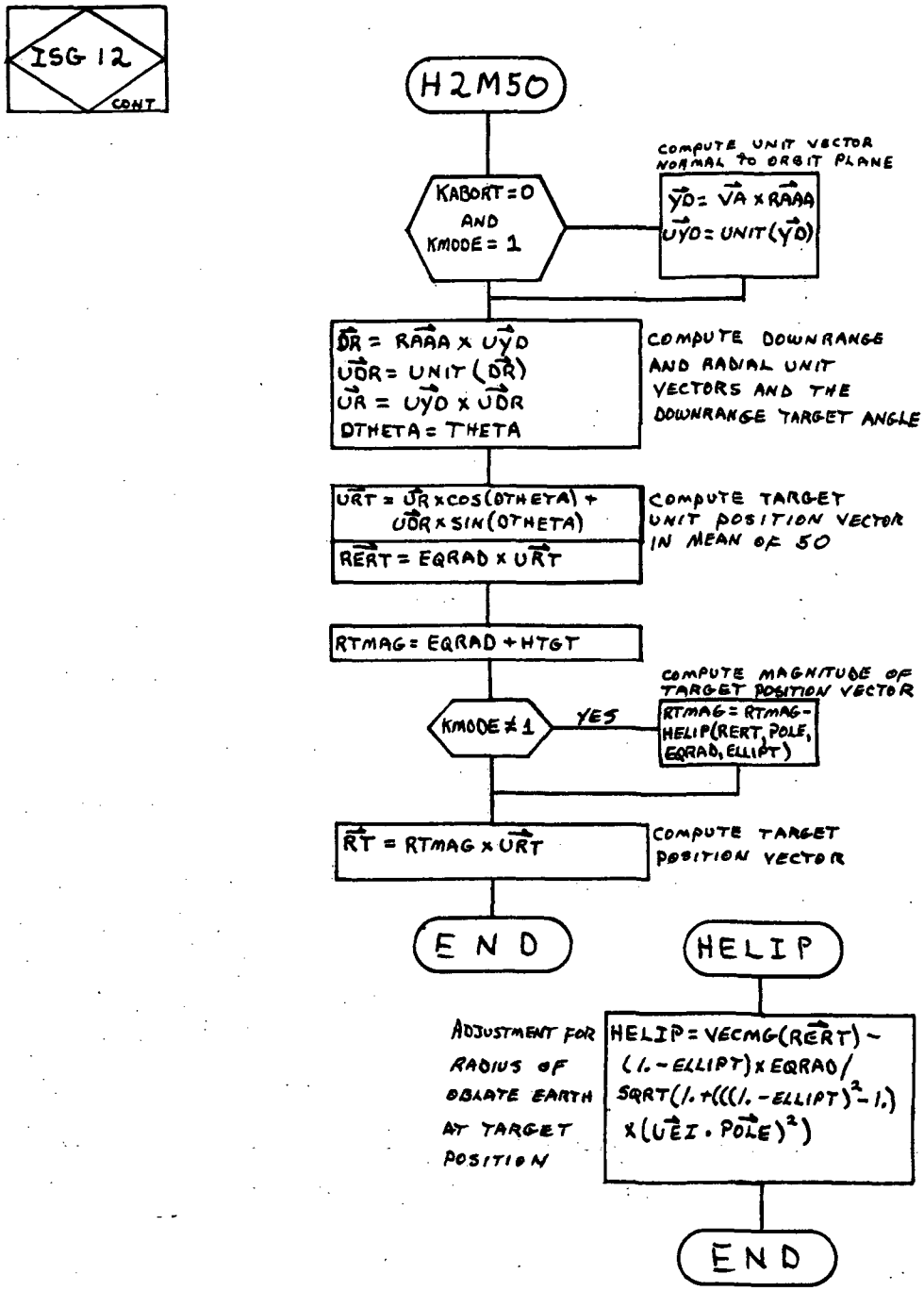


Figure 5.38-1.- Functional logic flow for the H2M50 computational routine and the HELIP function routine.

5.39 ROUTINE NAME - PGOP3 COMPUTATIONAL ROUTINE

5.39.1 Purpose

The PGOP3 routine sequences through the calls to INI1, PRDT6, and CORT7 to simulate the OPS-3 onboard PEG guidance.

5.39.2 Functional Description

PGOP3 calls INI1 for initialization of guidance parameters. The velocity to go and time to go are computed, and if a PEG4 solution is desired the thrust integrals and turning rate are computed. A unit vector in the thrust direction is then obtained followed by calls to PRDT6 and CORT7 to compute the end of burn position vector.

5.39.3 Assumptions and Limitations

For a PEG7 solution the turning rate is held to zero for an inertial hold simulation.

5.39.4 Method

None.

5.39.5 Routine Input/Output Variables

Table 5.39-I contains the definitions of all the input/output variables for the PGOP3 computational routine.

5.39.6 Functional Logic Flow

Figure 5.39-1 contains the functional logic flow for the PGOP3 computational routine.

5.39.7 Diagnostics and Debug

When the user selects the debug print option the following PEG quantities are printed, XLAMC, XLAMDC, TTT, TLAMC, ATP, VGMAG, VEXX, ATR, TGO, and VG.

5.39.8 Special Comments

None.

5.39.9 References

None.

TABLE 5.39-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine PGOP3

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ATP		Real	I/O		C,T		Refer to table 5.1-I for all code symbol definitions.
DVSS		Real	I/O		C		
IPOUT		6CH	I		C		
IOUNIT		Intg	I		C		
KFLAG		Intg	I/O		C		
KINIT		Intg	I		C		
KMODE		Intg	I		C		
KSTEER		Intg	I		C		
RP		Real	I		C		
TTT		Real	I/O		C,T		
UYY		Real	I		C		
VEXX		Real	I		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.39-I.- Continued

Routine PGOP3

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
VG		Real	I/O		C		
XLDXZ		Real	I		C		
IMSG		Intg	O		C		
QPRIME		Real	O		C		
RDA		Real	O		C		
TAU		Real	O		C		
TGO		Real	O		C,T		
TIJOL		Real	O		C		
TIS		Real	O		C		
TP		Real	O		C		
TPREV		Real	O		C		
TLAMC		Real	O		C,T		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.39-I.- Concluded

Routine PGOP3

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
VGMAG		Real	O		C,T		
XLAM		Real	O		C		
XLDA		Real	O		C		
XLAMC		Real	O		C,T		
XLAMDC		Real	O		C,T		
ATR		Real	O		T		
VEXX		Real	O		T		
VG		Real	O		T		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

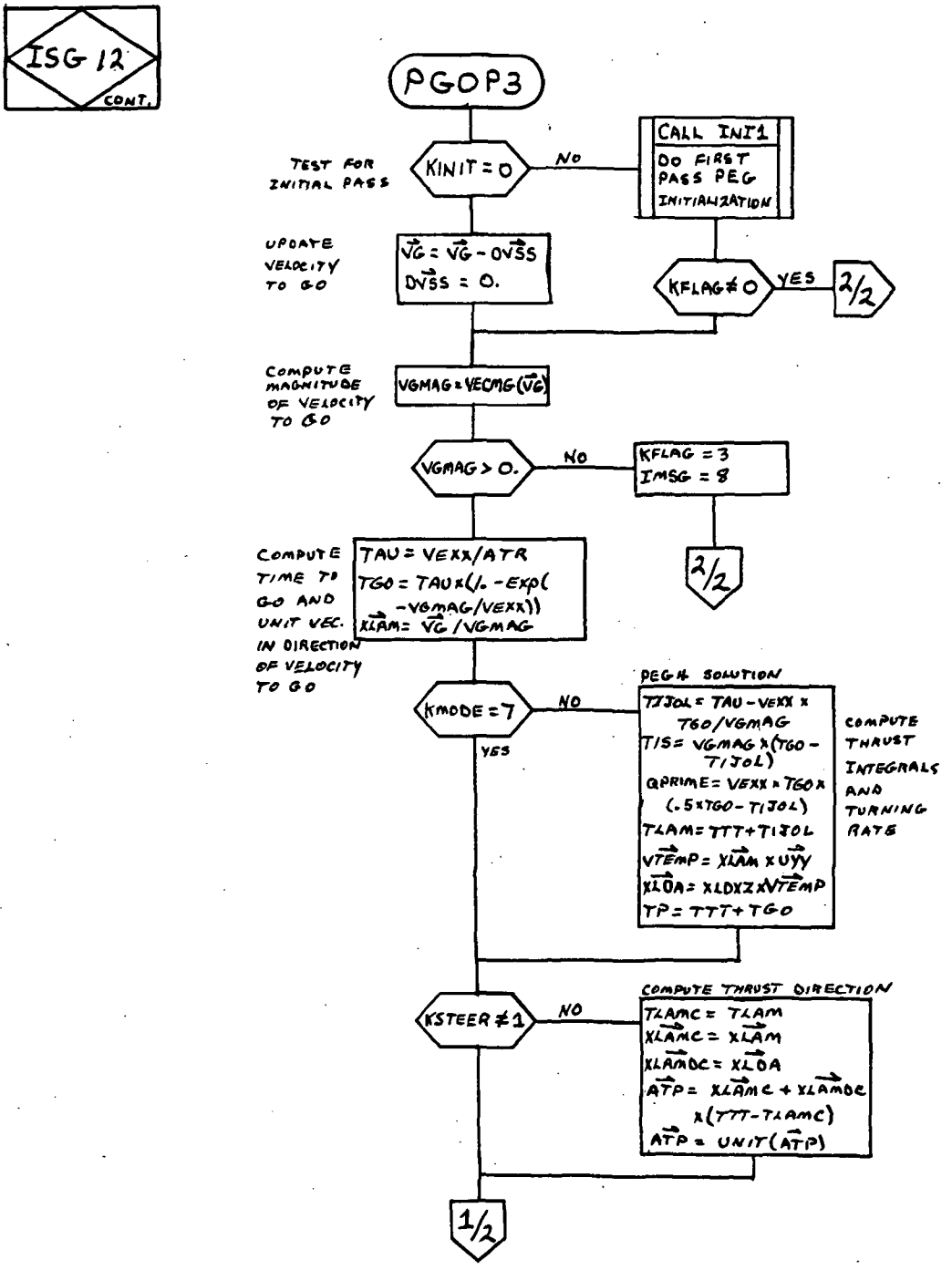


Figure 5.39-1.- Functional logic flow for the PGOP3 computational routine.

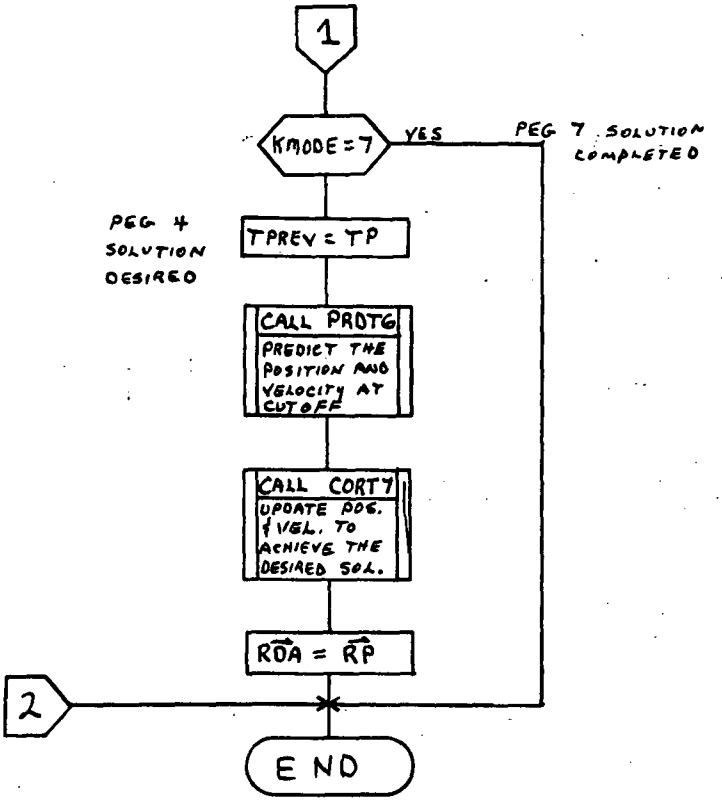


Figure 5.39-1.- Concluded.

5.40 ROUTINE NAME - INI1 INITIALIZATION ROUTINE

5.40.1 Purpose

The INI1 routine initializes the state and guidance parameters for the PGOP3 routine.

5.40.2 Functional Description

INI1 sets the VGO unit vector and the guidance reference time to zero for a PEG7 solution initialization. For a PEG4 solution initialization it initializes the predicted burnout state and calls CORT7 to obtain an initial VGO value.

5.40.3 Assumptions and Limitations

None.

5.40.4 Method

None.

5.40.5 Routine Input/Output Variables

Table 5.40-I contains the definitions of all the input/output variables for the INI1 initialization routine.

5.40.6 Functional Logic Flow

Figure 5.40-1 contains the functional logic flow for the INI1 initialization routine.

5.40.7 Diagnostics and Debug

None.

5.40.8 Special Comments

None.

5.40.9 References

None.

TABLE 5.40-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine INI1

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
CLMDXZ		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
GM		Real	I		C	GLOCON(1)	
KMODE		Intg	I		C		
RAAA		Real	I		C		
TTT		Real	I		C		
VA		Real	I		C		
IYAW		Intg	O		C		
KINIT		Intg	O		C		
RP		Real	O		C		
TGO		Real	O		C		
TP		Real	O		C		
VP		Real	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.40-I.- Concluded

Routine INI1

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TLAM		Real	O		C		
XLDA		Real	O		C		
XLDXZ		Real	O		C		

NOTES:

TYPE

Free	Dubl	18CH	Mix
Intg	2CH	36CH	Char
Real	6CH	72CH	Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

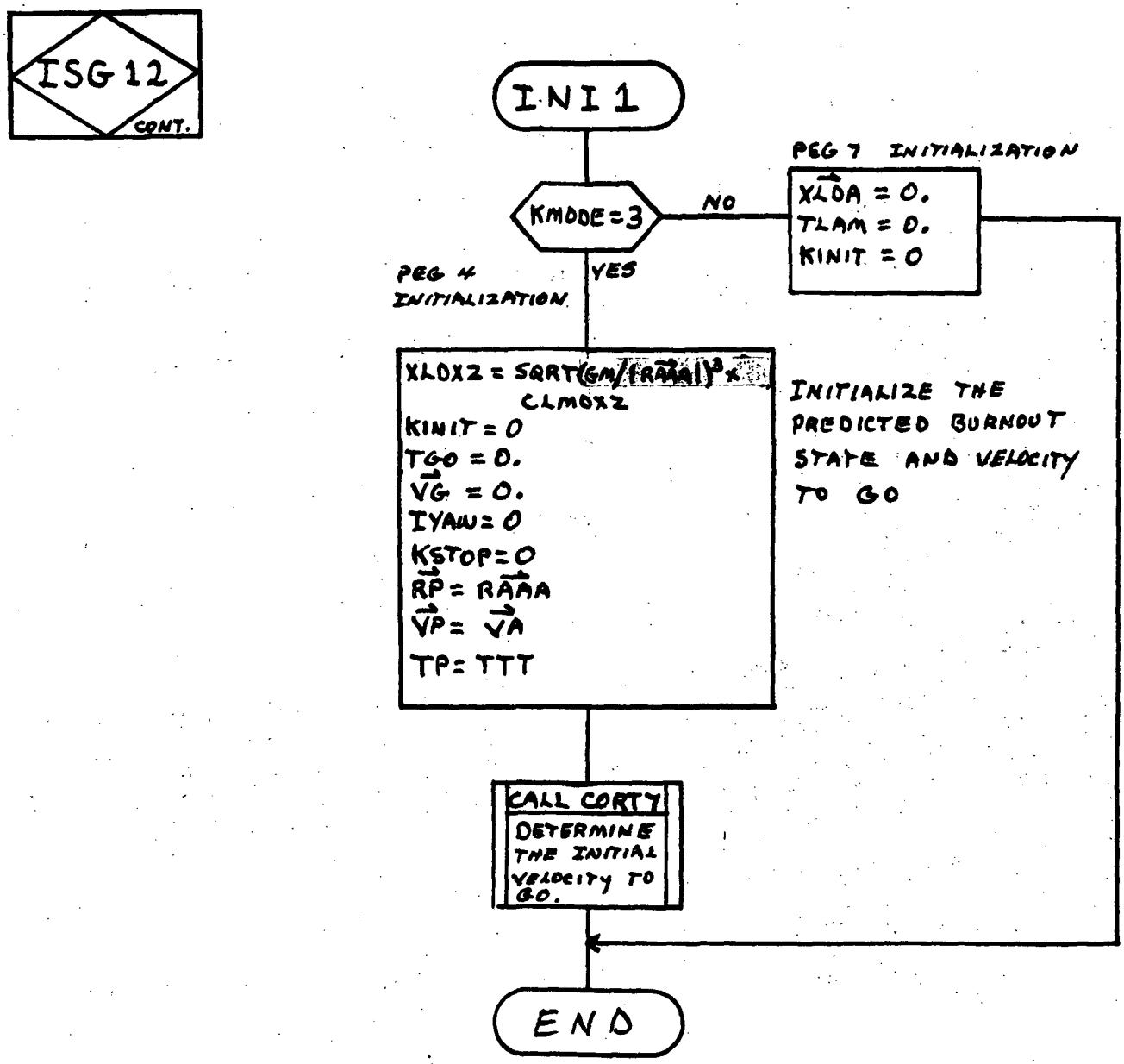


Figure 5.40-1.- Functional logic flow for the INT1 initialization routine.

5.41 ROUTINE NAME - PRDT6 COMPUTATIONAL ROUTINE

5.41.1 Purpose

The PRDT6 routine is used to compute the predicted burnout state vector.

5.41.2 Functional Description

The position to be gained vector $\overrightarrow{RGO}$ is computed along with the initial position and velocity vectors. The SUPRG routine is called to compute the gravity acceleration effects upon the final state. Finally the burnout state is computed by adding the gravity effects, the velocity to go, and the position to go to the position and velocity vectors at Tig.

5.41.3 Assumptions and Limitations

The computation for the burnout state uses only the first integrals of the thrust acceleration.

5.41.4 Method

The thrust time integral $\overrightarrow{RGO}$ is computed as a function of the unit thrust vectors in the direction of VGO and the force over mass integrals computed in PGOP3. The gravity effects are approximated by propagating the initial state over an arc approximating the powered trajectory. The initial state for this propagation is computed from the actual initial state, the thrust integrals, and time to go. The actual state propagation with gravity effects is done by the SUPRG routine.

5.41.5 Routine Input/Output Variables

Table 5.41-I contains the definitions of all the input/output variables for the PRDT6 computational routine.

5.41.6 Functional Logic Flow

Figure 5.41-1 contains the functional logic flow for the PRDT6 computational routine.

5.41.7 Diagnostics and Debug

None.

5.41.8 Special Comments

None.

5.41.9 References

None.

TABLE 5.41-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine PRDT6

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DTMAXX		Real	I		C	DTMCON(24)	Refer to table 5.1-1 for all code symbol definitions.
DTMINN		Real	I		C	DTMCON(25)	
J2FLG		Intg	I		C		
NSEG		Intg	I		C	DTMCON(19)	
QPRIME		Real	I		C		
RAAA		Real	I		C		
TIS		Real	I		C		
TTT		Real	I		C		
VA		Real	I		C		
VG		Real	I		C		
XLAM		Real	I		C		
XLDA		Real	I		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.41-I.- Concluded

Routine PRDT6

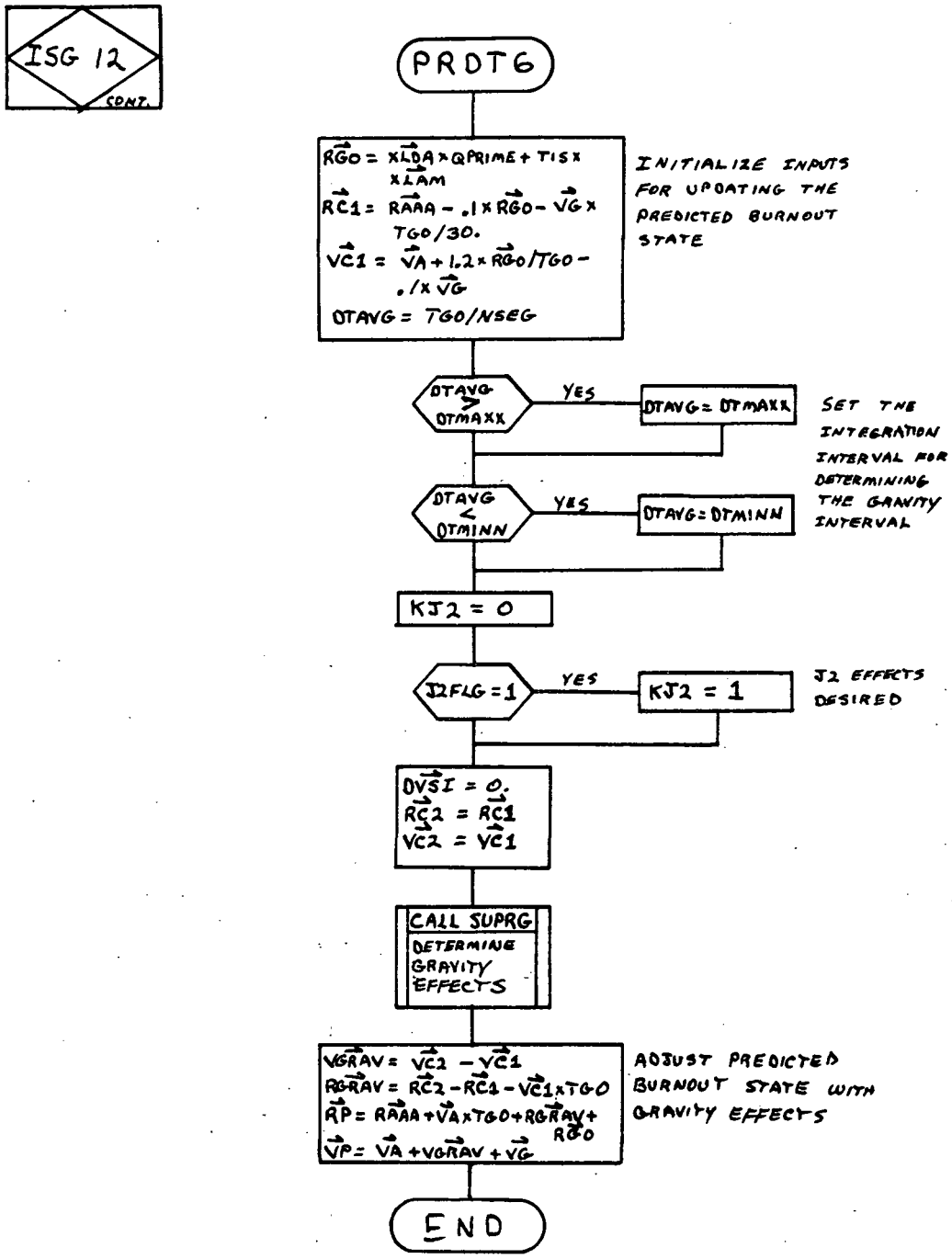
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DTAVG		Real	O		C		
DVSI		Real	O		C		
KJ2		Intg	O		C		
RC1		Real	O		C		
RC2		Real	O		C		
RGO		Real	O		C		
RGRAV		Real	O		C		
RP		Real	O		C		
VC1		Real	O		C		
VC2		Real	O		C		
VGRAV		Real	O		C		
VP		Real	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory



Page 1 of 1

Figure 5.41-1.- Functional logic flow for the PRDT6 computational routine.

5.42 ROUTINE NAME - SUPRG COMPUTATIONAL ROUTINE

5.42.1 Purpose

The SUPRG routine does a state extrapolation to account for the gravitational effects on the deorbit burnout vector prediction in PEG.

5.42.2 Functional Description

The initial vector from PEG is propagated to the burnout time by an iterative series of calls to GRAVJ to compute the current gravity acceleration vector at the input vector position. The state vector is extrapolated through a computed number of steps using the gravity acceleration vector that is updated each step. With each cycle the position and velocity vectors are updated as a function of the current velocity vector and the gravity acceleration vector until the burnout time is reached.

5.42.3 Assumptions and Limitations

None.

5.42.4 Method

None.

5.42.5 Routine Input/Output Variables

Table 5.42-I contains the definitions of all the input/output variables for the SUPRG computational routine.

5.42.6 Functional Logic Flow

Figure 5.42-1 contains the functional logic flow for the SUPRG computational routine.

5.42.7 Diagnostics and Debug

None.

5.42.8 Special Comments

None.

5.42.9 References

None.

TABLE 5.42-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine SUPRG

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DT		Real	I		A		Step size
DVS		Real	I		A		Sensed velocity
J2		Intg	I		A		J2 effects flag
RF		Real	I/O		A		Position vector
TF		Real	I		A		Vector time
TI		Real	I		A		Initial guidance time
VF		Real	I/O		A		Velocity vector

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

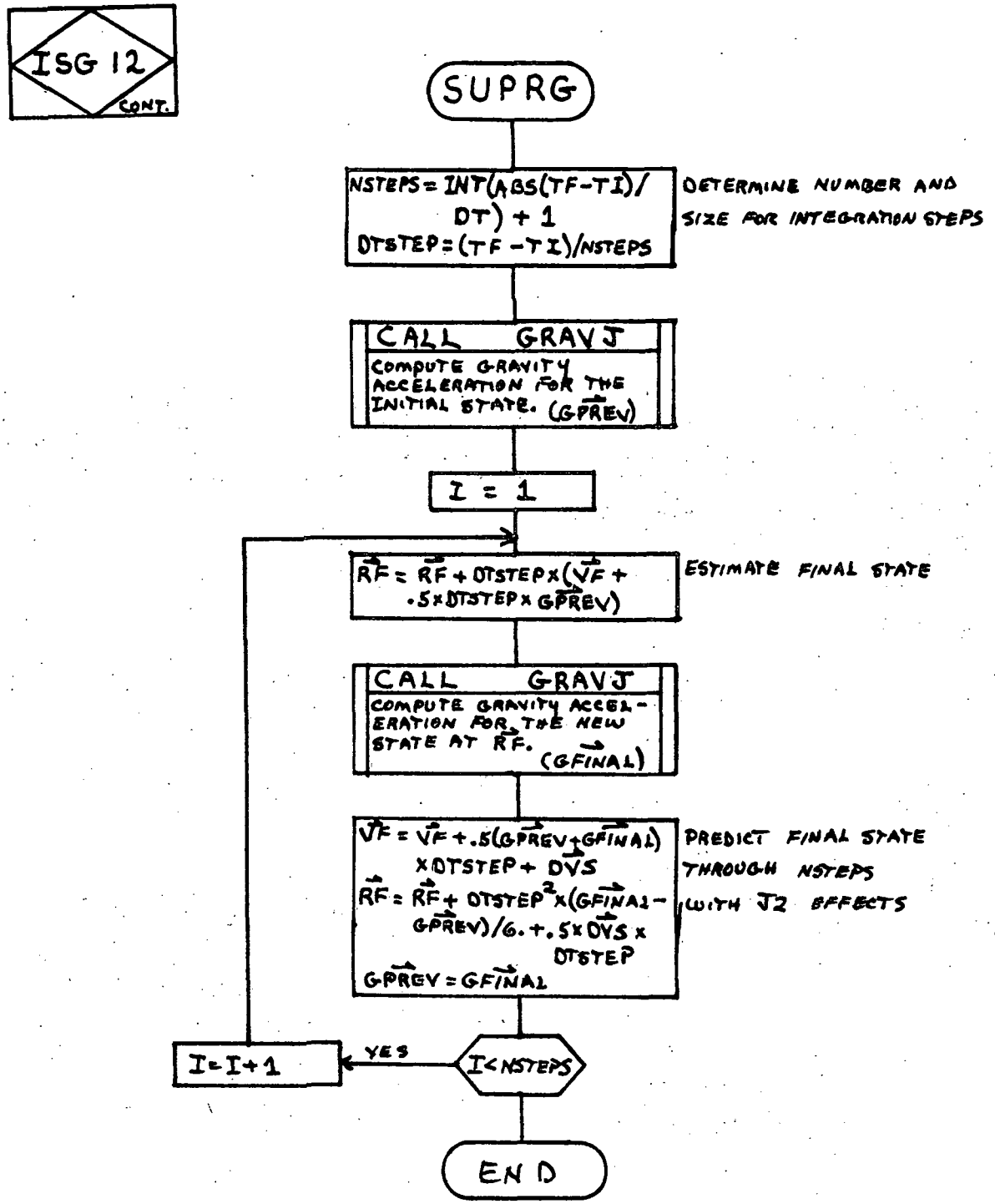


Figure 5.42-1.- Functional logic flow for the SUPRG computational routine.

5.43 ROUTINE NAME - CORT7 COMPUTATIONAL ROUTINE

5.43.1 Purpose

The CORT7 routine computes the velocity to go to achieve the predicted state and tests for a converged solution.

5.43.2 Functional Description

The CORT7 first adjusts the target radius $\overrightarrow{RTA}$ to be in the flight plane and to ensure that the input entry interface altitude is established above the Fischer ellipsoid. LTVC2 is then called to compute the cutoff velocity that will result in meeting the target radius and velocity at entry interface. The velocity to go is then updated by subtracting a portion of VMISS (the difference between the predicted and actual cutoff velocity). For fuel wasting solutions the out-of-plane velocity required is computed and the velocity-to-go vector updated to reflect the out-of-plane burn. The VMISS magnitude is tested for convergence within a small input tolerance and when converged the out-of-plane yaw angle is computed.

5.43.3 Assumptions and Limitations

None.

5.43.4 Method

None.

5.43.5 Routine Input/Output Variables

Table 5.43-I contains the definitions of all the input/output variables for the CORT7 computational routine.

5.43.6 Functional Logic Flow

Figure 5.43-1 contains the functional logic flow for the CORT7 computational routine.

5.43.7 Diagnostics and Debug

None.

5.43.8 Special Comments

None.

5.43.9 References

None.

TABLE 5.43-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine CORT7

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
CA		Real	I		C		Refer to table 5.1-I for all code symbol definitions.
DTCOST		Real	I		C		
ELLIPT		Real	I		C	GLOCON(27)	
EQRAD		Real	I		C	GLOCON(23)	
HTGT		Real	I		C		
IYAW		Intg	I/O		C		
KFLAG		Intg	I		C		
KFUELD		Intg	I		C		
POLE		Real	I		C	DTMCON(39)	
RHOI		Real	I		C	DTMCON(20)	
RP		Real	I		C		
RTA		Real	I/O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.43-I.- Continued

Routine CORTZ

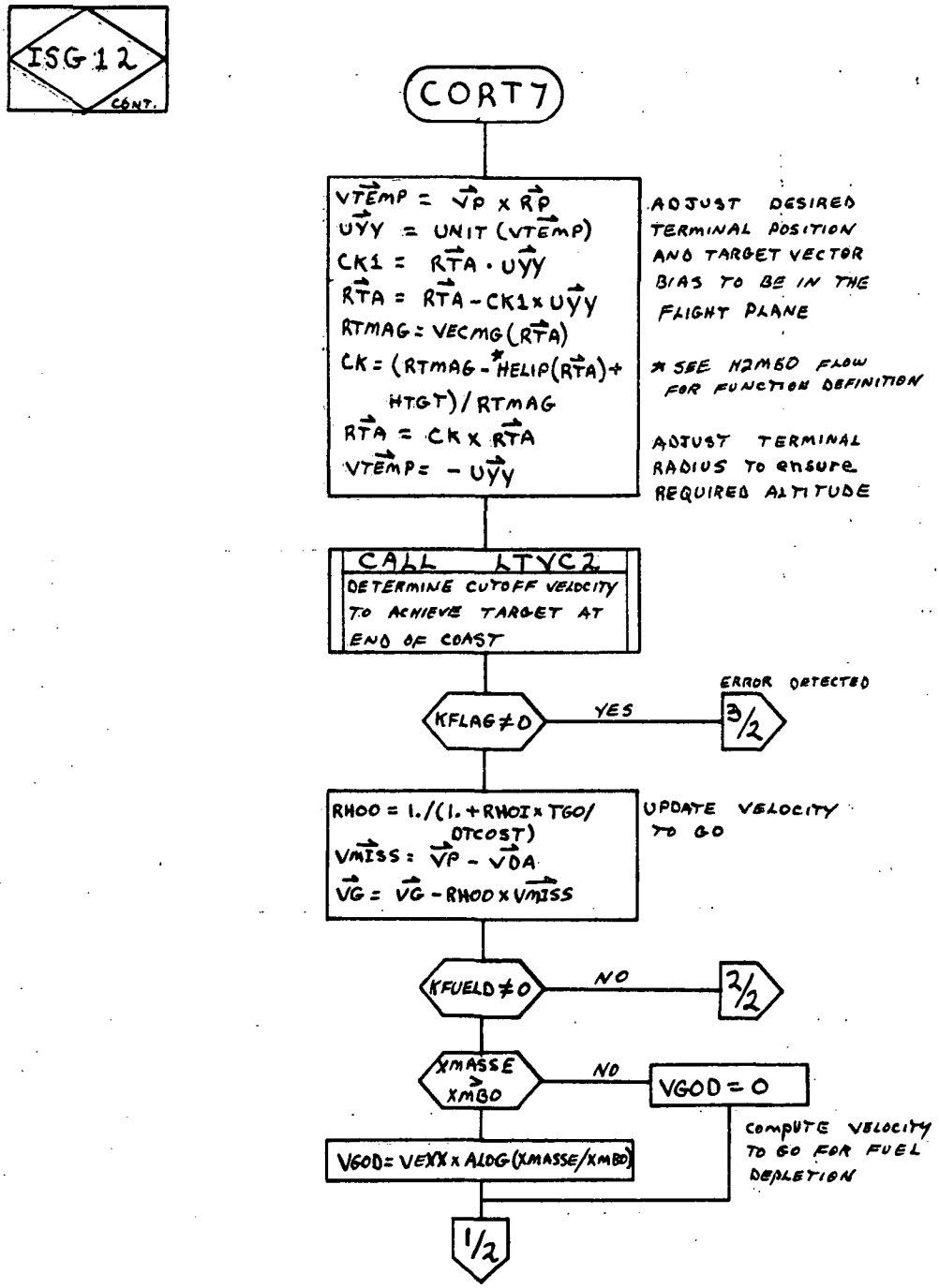
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TGO		Real	I		C		
VDA		Real	I		C		
VEXX		Real	I		C		
VG		Real	I/O		C		
VP		Real	I		C		
XMASSE		Real	I		C		
XMBO		Real	I		C		
FWYAW		Real	O		C		
KSTOP		Intg	O		C		
RHOO		Real	O		C		
UYY		Real	O		C		
VGMAG		Real	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.43-I.- Concluded

Routine CORTZ

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
VGOD		Real	O		C		
VGOYS		Real	O		C		
VMISS		Real	O		C		

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory



Page 1 of 2

Figure 5.43-1.- Functional logic flow for the CORT7 computational routine.

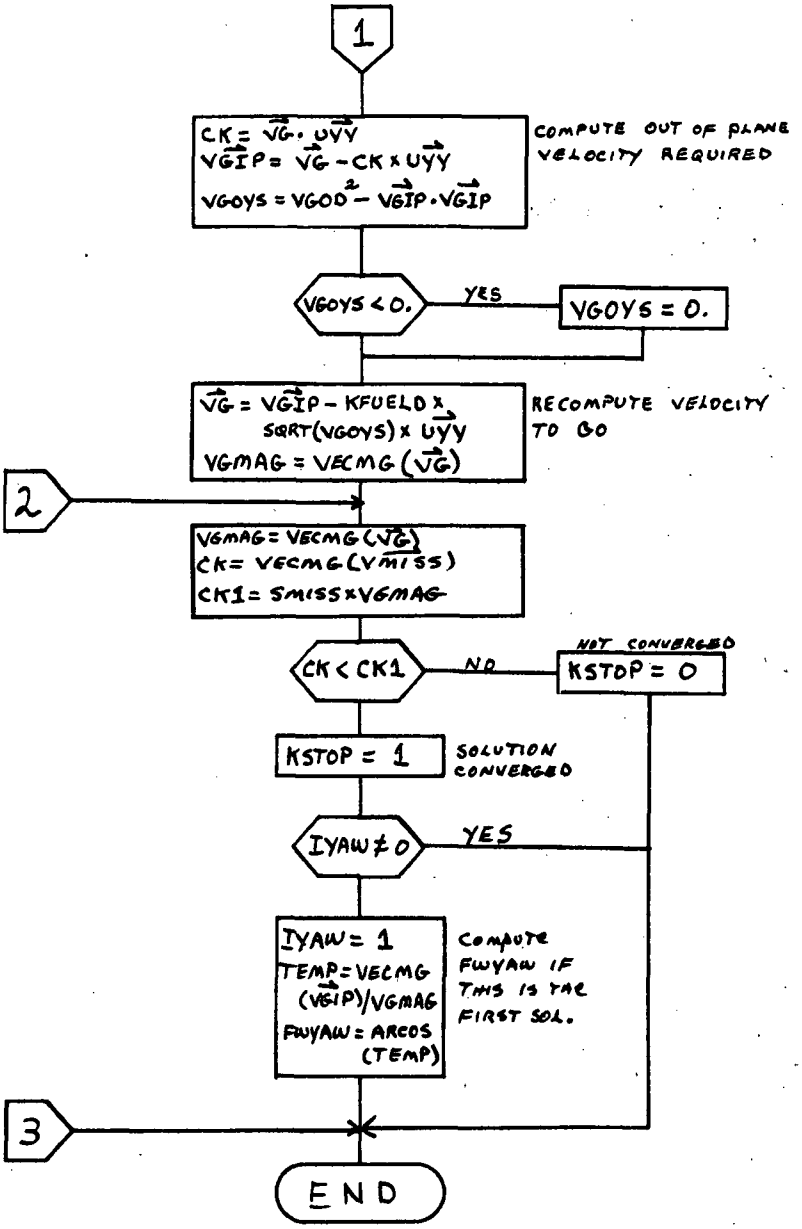


Figure 5.43-1.- Concluded.

5.44 ROUTINE NAME - LTVC2 COMPUTATIONAL ROUTINE

5.44.1 Purpose

The LTVC2 routine is the same as the LTVCN routine. It computes the velocity vector required at an initial time and position to intercept the target position vector and velocity vector at entry. Two separate routines exist because of the segmentation scheme used with the FDS-1 implementation. LTVCN is used for the impulsive burn solution and LTVC2 is used in the PEG finite burn solutions.

5.44.2 Functional Description

LTVC2 determines the velocity required at an initial position to intercept a target position such that a specified linear relationship between the radial and horizontal velocity components is met. The linear relationship is given by Vradial = C1 + C2 * Vhorizontal. C1 and C2 are input targets and the solution is obtained from a set of two body differential equations of motion.

5.44.3 Assumptions and Limitations

The LTVC2 routine computations assume a point mass in an inverse squared field.

5.44.4 Method

None.

5.44.5 Routine Input/Output Variables

Table 5.44-I contains the definitions of all the input/output variables for the LTVC2 computational routine.

5.44.6 Functional Logic Flow

Figure 5.44-1 contains the functional logic flow for the LTVC2 computational routine.

5.44.7 Diagnostics and Debug

None.

5.44.8 Special Comments

None.

5.44.9 References

None.

TABLE 5.44-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine LTVC2

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
C		Real	I		A		Target line coefficients
R0		Real	I		A		TIG position state
R1		Real	I		A		E.I. target position state
XMU		Real	I		C	GLOCON(1)	Refer to table 5.1-I.
XNUNIT		Real	I		A		Unit vector normal to orbit plane
DELTAT		Real	O		A		Time from burnout to E.I.
KFLAG		Intg	O		A		PEG error flag
V0		Real	O		A		Impulsive delta-V vector
V1		Real	O		A		E.I. velocity state

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

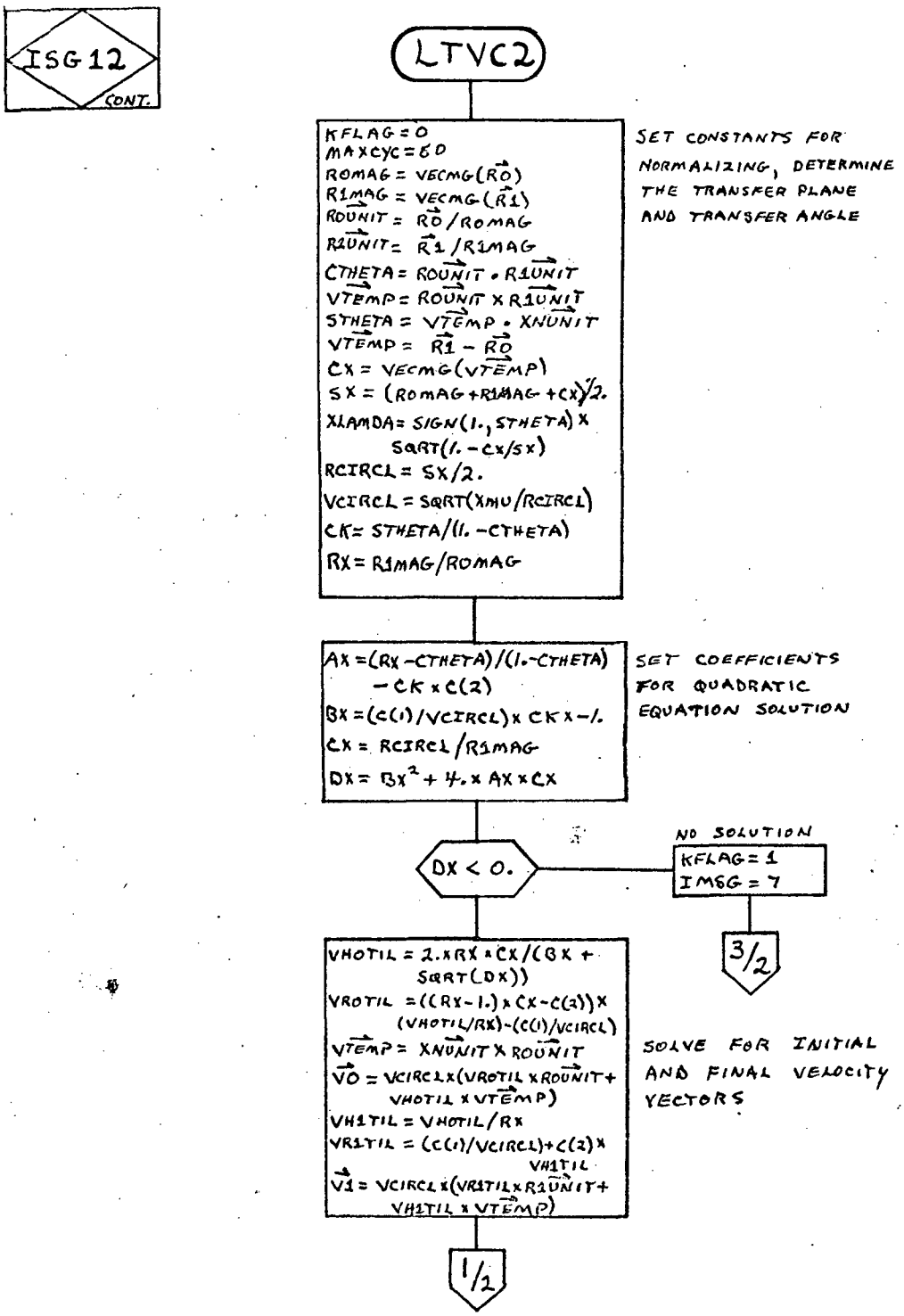


Figure 5.44-1.- Functional logic flow for the LTVC2 computational routine.

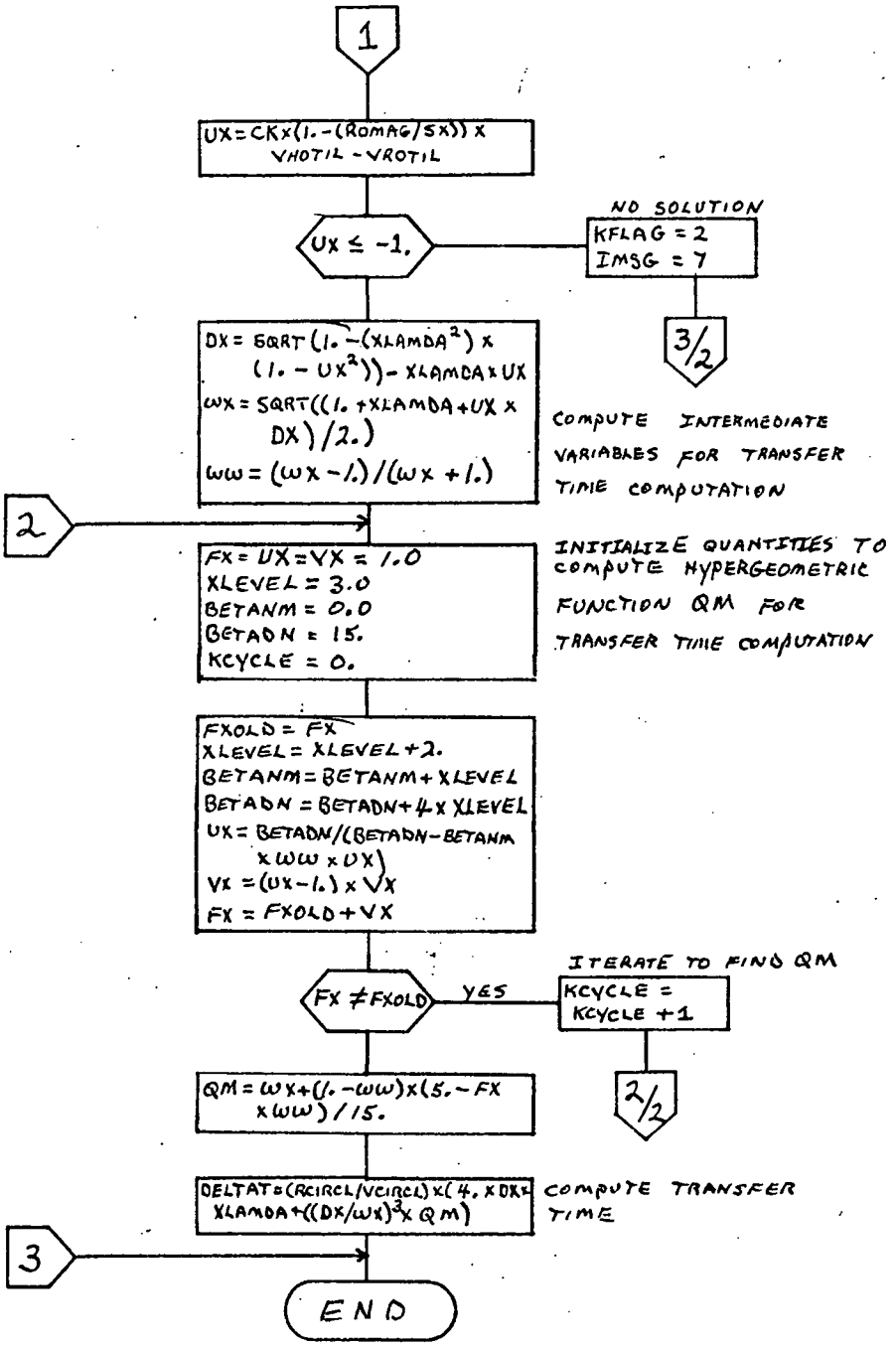


Figure 5.44-1.- Concluded.

5.45 ROUTINE NAME - DTT13 COMPUTATIONAL ROUTINE - SEGMENT 13

5.45.1 Purpose

The DTT13 routine is the supervisor for computation of the entry range from entry interface to landing. Along with the EGRT, GD2EF, EF2TD, ROTMX, VREL, EF2MF, and EF2GD routines it is the computational segment 13.

5.45.2 Functional Description

DTT13 calls a series of vector and matrix transformation routines to obtain the Earth fixed landing site position ($\vec{RLS}$), the Earth fixed to topodetic transformation matrix with the Z-axis alined with the runway heading (REC), the Earth fixed entry interface position vector ($\vec{XYZEN}$), and the Earth relative velocity vector in topodetic coordinates ($\vec{VRNOTD}$). These are all used as inputs to the EGRT routine that is then called to compute the entry range.

5.45.3 Assumptions and Limitations

None.

5.45.4 Method

None.

5.45.5 Routine Input/Output Variables

Table 5.45-I contains the definitions of all the input/output variables for the DTT13 computational routine.

5.45.6 Functional Logic Flow

Figure 5.45-1 contains the functional logic flow for the DTT13 computational routine.

5.45.7 Diagnostics and Debug

When the user selects the debug print option the following variables are output TRANGE, THETLS, ERAEI, DELAZ, and RCHMAG.

5.45.8 Special Comments

None.

TABLE 5.45-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT13

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AMILE		Real	I		C	GLOCON(89)	Refer to table 5.1-I for all code symbol definitions.
DELAZ		Real	I/O		C,T		
ERAI		Real	I		C		
IPOUT		6CH	I		C		
IOUNIT		Intg	I		C		
PI		Real	I		C	GLOCON(59)	
RAZ		Real	I		C		
RCHMAG		Real	I/O		C,T		
RTA		Real	I		C		
TALTD		Real	I		C		
TLATD		Real	I		C		
TLONG		Real	I		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.45-I.- Concluded

Routine DTT13

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TRANG		Real	I/O	n. mi.	C,T		
TM502I		Real	I		C	SESCON(45)	
TT		Real	I		C		
VTAA		Real	I		C		
EARPOL		Real	O		C		
ERAEI		Real	O		C,T		
REC		Real	O		C		
RLS		Real	O		C		
THETLS		Real	O		C,T		
VRHOTD		Real	O		C		
XYZEN		Real	O		C		
XYZEDN		Real	O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

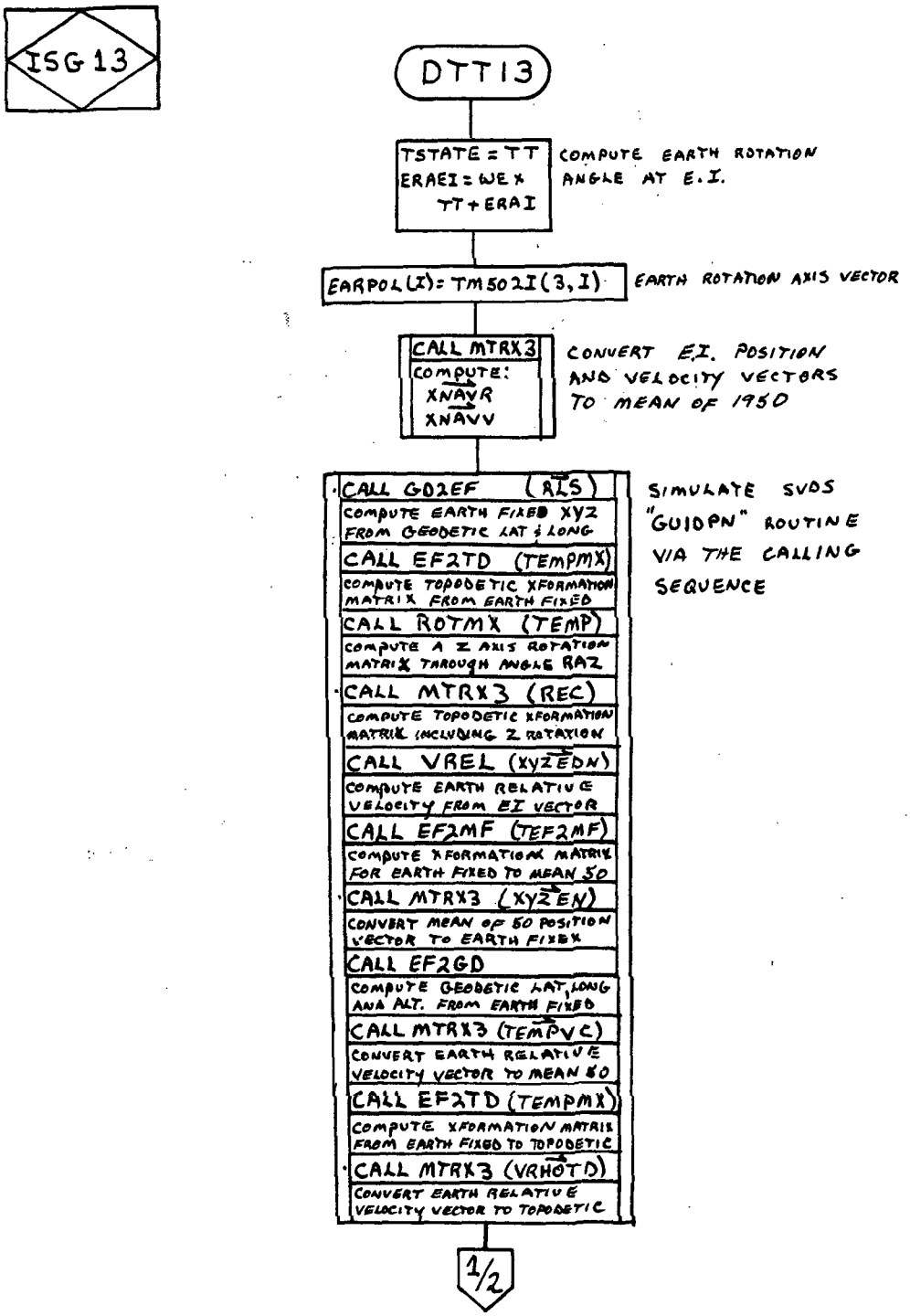


Figure 5.45-1.- Functional logic flow for the DTT13 computational routine.

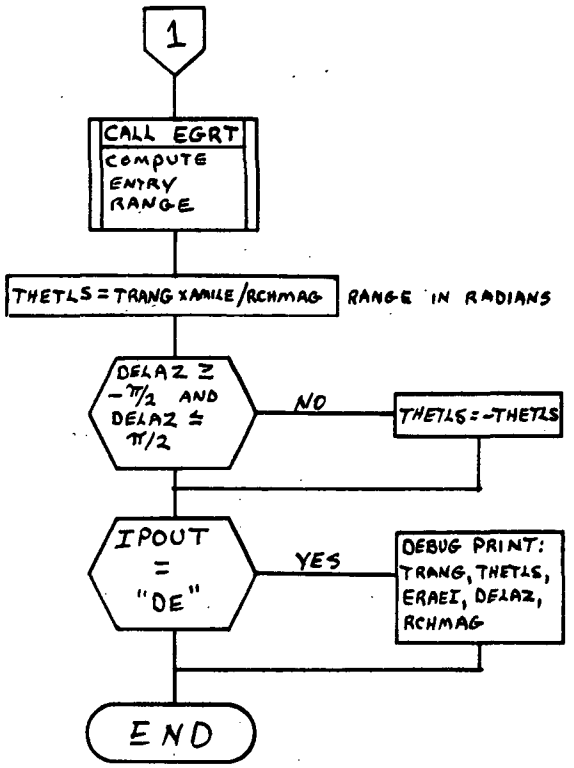


Figure 5.45-1.- Concluded.

5.46 ROUTINE NAME - GD2EF TRANSFORMATION ROUTINE

5.46.1 Purpose

The GD2EF routine converts an input geodetic latitude, longitude, and altitude to an Earth fixed XYZ position vector.

5.46.2 Functional Description

None.

5.46.3 Assumptions and Limitations

None.

5.46.4 Method

None.

5.46.5 Routine Input/Output Variables

Table 5.46-I contains the definitions of all the input/output variables for the GD2EF tranformation routine.

5.46.6 Functional Logic Flow

Figure 5.46-1 contains the functional logic flow for the GD2EF transformation routine.

5.46.7 Diagnostics and Debug

None.

5.46.8 Special Comments

None.

5.46.9 References

None.

TABLE 5.46-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine GD2EF

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ELLIPT		Real	I		C	GLOCON(27)	Refer to table 5.1-I for code symbol definition.
EQRAD		Real	I		C	GLOCON(23)	Refer to table 5.1-I for code symbol definition.
GLAT		Real	I		A		Geodetic latitude
GLON		Real	I		A		Longitude
ALT		Real	I		A		Geodetic altitude
GD2E		Real	O		A		Earth-fixed position vector
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

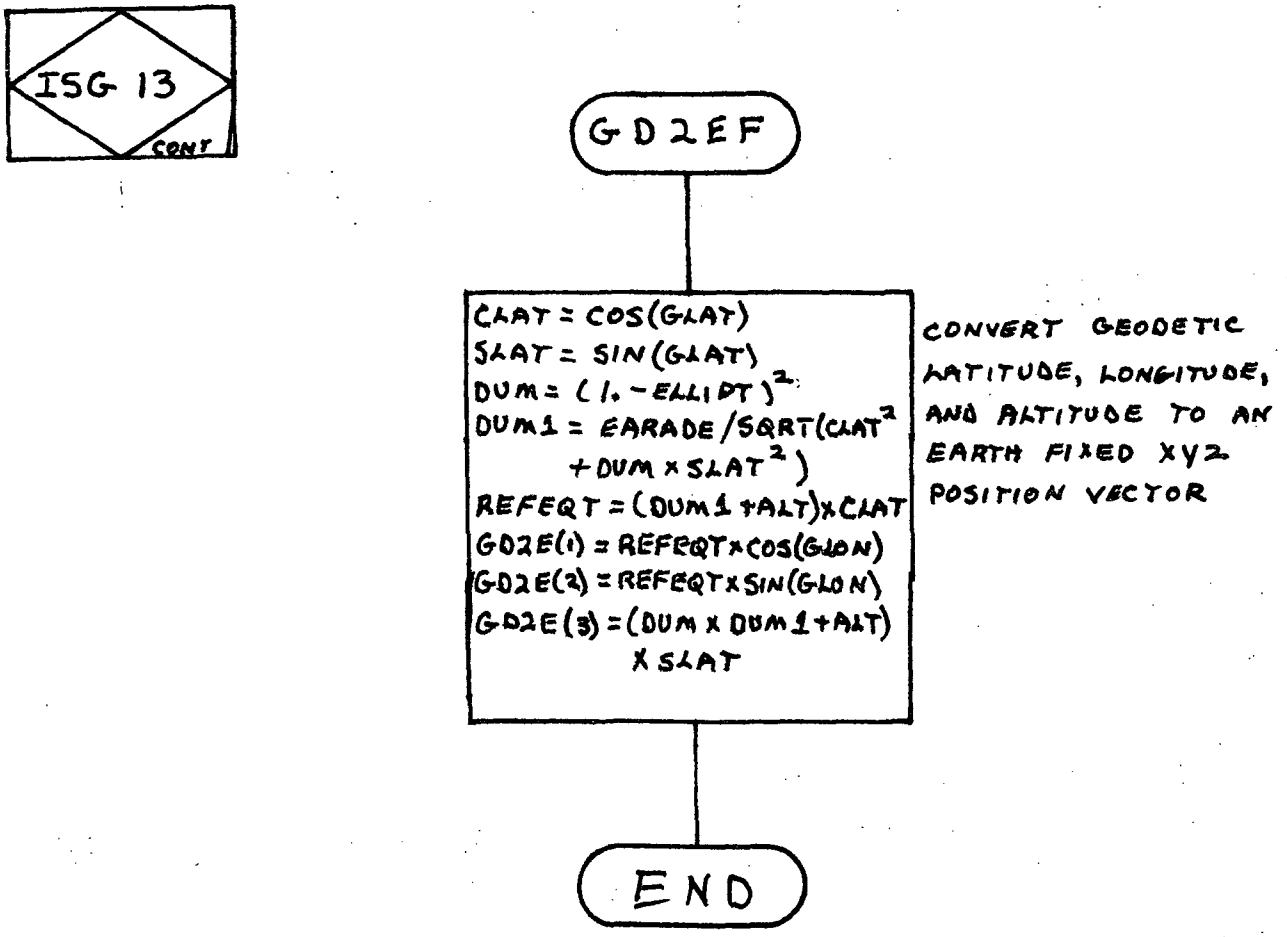


Figure 5.46-1.- Functional logic flow for the GD2EF transformation routine.

5.47 ROUTINE NAME - EF2TD TRANSFORMATION ROUTINE

5.47.1 Purpose

The EF2TD routine computes a transformation matrix to go from Earth fixed to a topodetic coordinate system at a specified latitude and longitude.

5.47.2 Functional Description

None.

5.47.3 Assumptions and Limitations

None.

5.47.4 Method

None.

5.47.5 Routine Input/Output Variables

Table 5.47-I contains the definitions of all the input/output variables for the EF2TD transformation routine.

5.47.6 Functional Logic Flow

Figure 5.47-1 contains the functional logic flow for the EF2TD transformation routine.

5.47.7 Diagnostics and Debug

None.

5.47.8 Special Comments

None.

5.47.9 References

None.

TABLE 5.47-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine EF2TD

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
GLAT		Real	I		A		Geodetic latitude
XLON		Real	I		A		Longitude
TM		Real	O		A		Earth-fixed to topodetic matrix
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

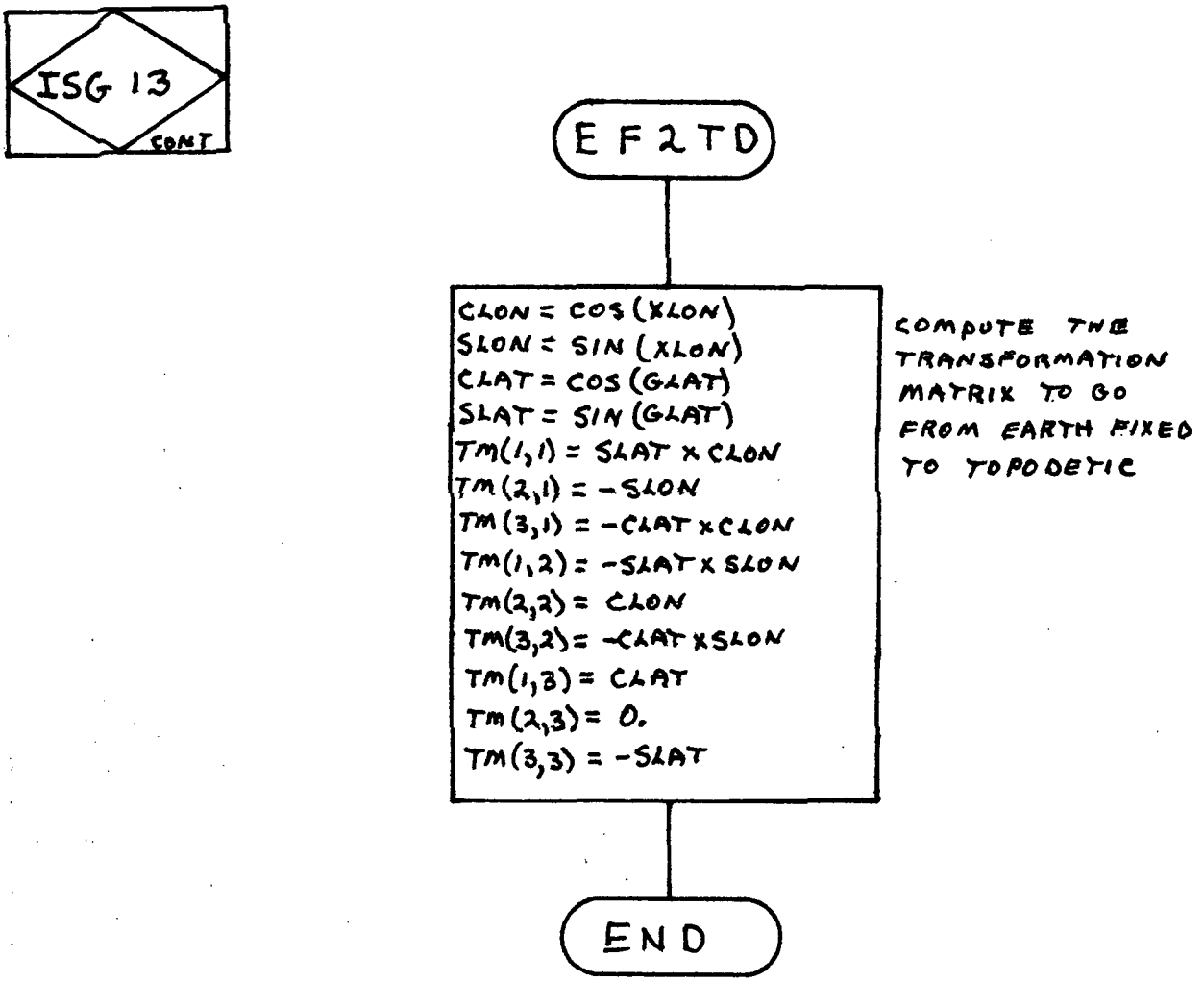


Figure 5.47-1.- Functional logic flow for the EF2TD transformation routine.

5.48 ROUTINE NAME - ROTMX TRANSFORMATION ROUTINE

5.48.1 Purpose

The ROTMX routine computes a matrix that describes a speci fied number and order of rotations of a specified magnitude about specified axes.

5.48.2 Functional Description

ROTMX will accept up to four ordered rotations of specified magnitudes and will compute a nine-element matrix that expresses these rotations. Each rotation may be about any of the three orthogonal coordinate system axes.

5.48.3 Assumptions and Limitations

The ROTMX routine assumes a standard right-handed three-axis orthogonal coordinate system.

5.48.4 Method

None.

5.48.5 Routine Input/Output Variables

Table 5.48-I contains the definitions of all the input/output variables for the ROTMX transformation routine.

5.48.6 Functional Logic Flow

Figure 5.48-1 contains the functional logic flow for the ROTMX transformation routine.

5.48.7 Diagnostics and Debug

None.

5.48.8 Special Comments

None

5.48.9 References

None.

TABLE 5.48-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine ROTMX

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ANGLE		Real	I		A		Rotation angles
IAXIS		Intg	I		A		Rotation axis
NUMBER		Real	I		A		Number of rotations
ROT		Real	O		A		Transformation matrix

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

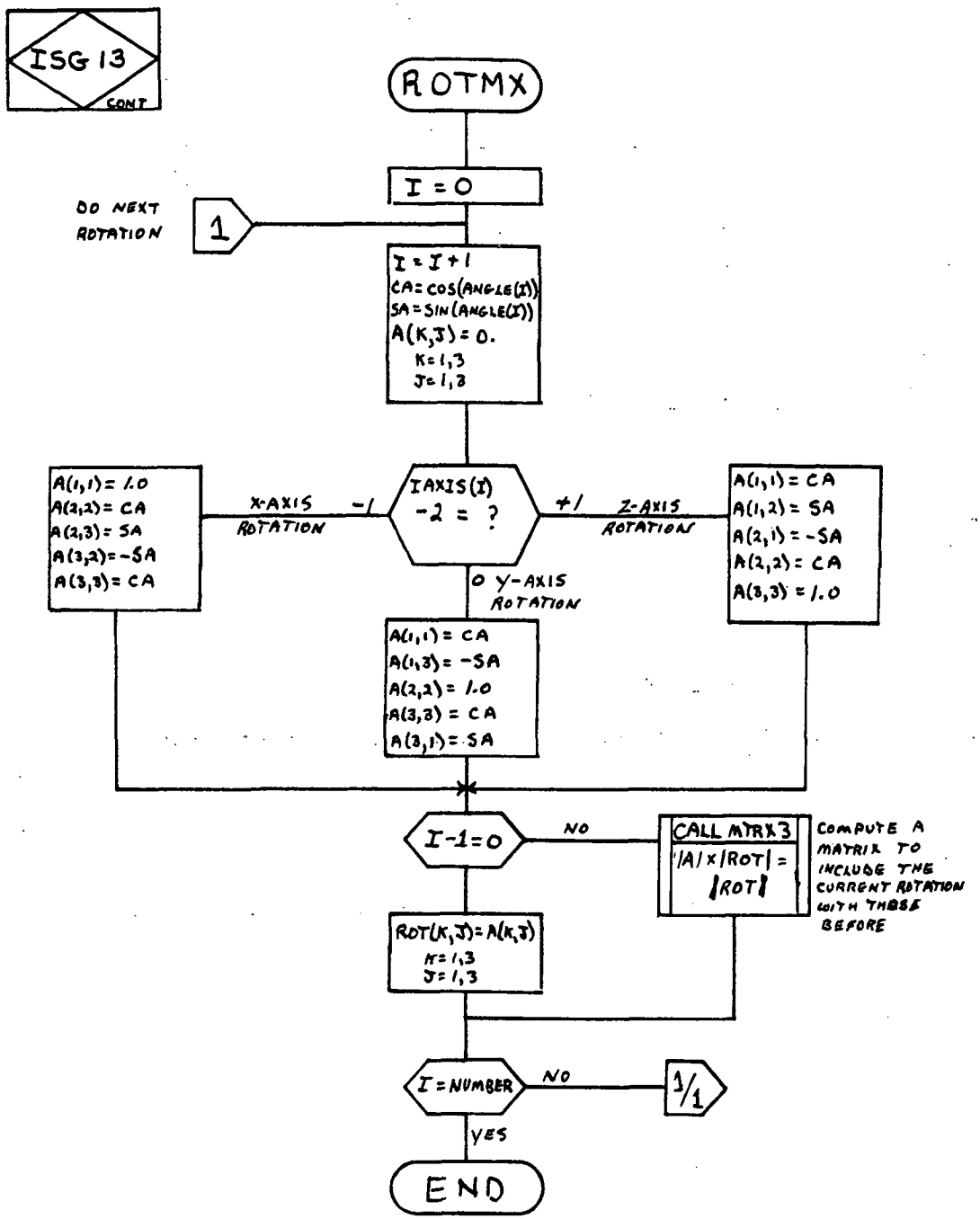


Figure 5.48-1.- Functional logic flow for the ROTMX transformation routine.

5.49 ROUTINE NAME - VREL COMPUTATIONAL ROUTINE

5.49.1 Purpose

The VREL routine computes an Earth relative velocity vector from an input Earth centered inertial vector by subtracting the Earth rotation components.

5.49.2 Functional Description

None.

5.49.3 Assumptions and Limitations

None.

5.49.4 Method

None.

5.49.5 Routine Input/Output Variables

Table 5.49-I contains the definitions of all the input/output variables for the VREL computational routine.

5.49.6 Functional Logic Flow

Figure 5.49-1 contains the functional logic flow for the VREL computational routine.

5.49.7 Diagnostics and Debug

None.

5.49.8 Special Comments

None.

5.49.9 References

None.

TABLE 5.49-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine VREL

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
EARPOL		Real	I		C		Refer to table 5.1-I for code symbol definition.
EARTHR		Real	I		C	GLOCON(13)	Refer to table 5.1-I for code symbol definition.
R		Real	I		A		Position vector.
V		Real	I		A		Velocity vector.
VR		Real	O		A		Relative velocity vector.

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

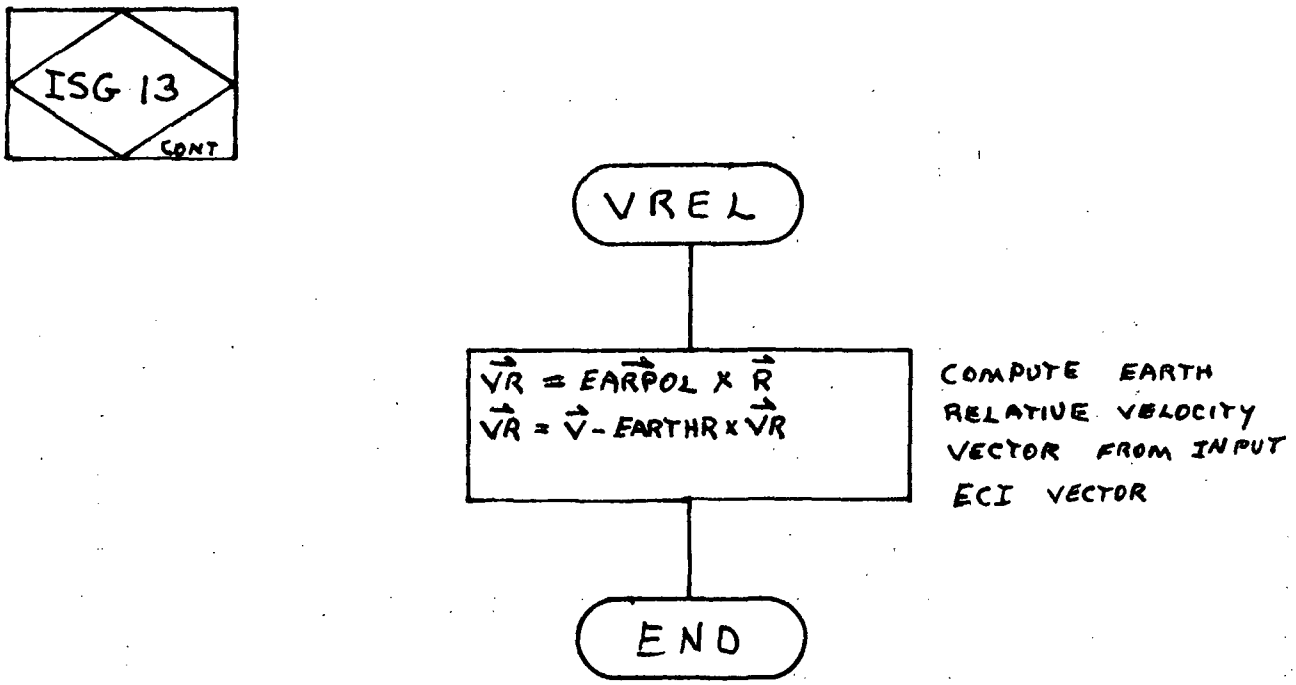


Figure 5.49-1.- Functional logic flow for the VREL computational routine.

5.50 ROUTINE NAME - EF2MF TRANSFORMATION ROUTINE

5.50.1 Purpose

The EF2MF routine computes a transformation matrix to go from an Earth-fixed coordinate system at an input time to an inertial mean-of-1950 coordinate system.

5.50.2 Functional Description

None.

5.50.3 Assumptions and Limitations

None.

5.50.4 Method

None.

5.50.5 Routine Input/Output Variables

Table 5.50-I contains the definitions of all the input/output variables for the EF2MF transformation routine.

5.50.6 Functional Logic Flow

Figure 5.50-1 contains the functional logic flow for the EF2MF transformation routine.

5.50.7 Diagnostics and Debug

None.

5.50.8 Special Comments

None.

5.50.9 References

None.

TABLE 5.50-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine EF2MF

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ERAI		Real	I		C		Refer to table 5.1-I for code symbol definition.
EARTHR		Real	I		C	GLOCON(13)	Refer to table 5.1-I for code symbol definition.
TM502I		Real	I		C	SESCON(45)	Refer to table 5.1-I for code symbol definition.
TIME		Real	I		A		Time of desired matrix.
TEF2MF		Real	O		A		True Earth-fixed to mean of 50 matrix.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

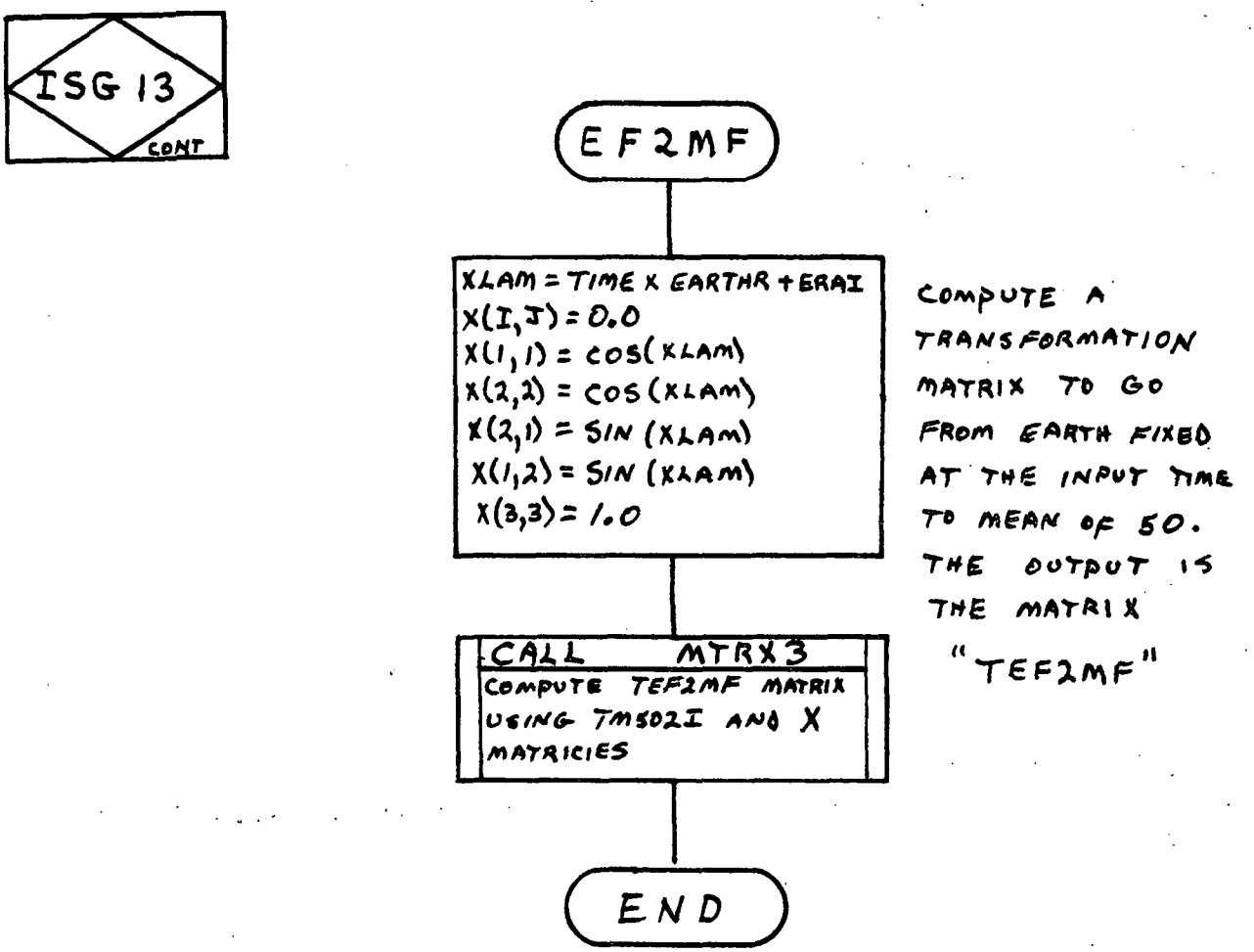


Figure 5.50-1.- Functional logic flow for the EF2MF transformation routine.

5.51 ROUTINE NAME - EF2GD TRANSFORMATION ROUTINE

5.51.1 Purpose

The EF2GD routine computes the geodetic latitude, longitude, and altitude from an Earth-fixed XYZ vector.

5.51.2 Functional Description

None.

5.51.3 Assumptions and Limitations

None.

5.51.4 Method

None.

5.51.5 Routine Input/Output Variables

Table 5.51-I contains the definitions of all the input/output variables for the EF2GD transformation routine.

5.51.6 Functional Logic Flow

Figure 5.51-1 contains the functional logic flow for the EF2GD transformation routine.

5.51.7 Diagnostics and Debug

None.

5.51.8 Special Comments

None.

5.51.9 References

None.

TABLE 5.51-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine EF2GD

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
ELLIPT		Real	I		C	GLOCON(27)	Refer to table 5.1-I for code symbol definition.
EQRAD		Real	I		C	GLOCON(23)	Refer to table 5.1-I for code symbol definition.
R		Real	I		A		Earth-fixed position vector.
GLAT		Real	O		A		Geodetic latitude.
XLON		Real	O		A		Longitude.
ALT		Real	O		A		Geodetic altitude.
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

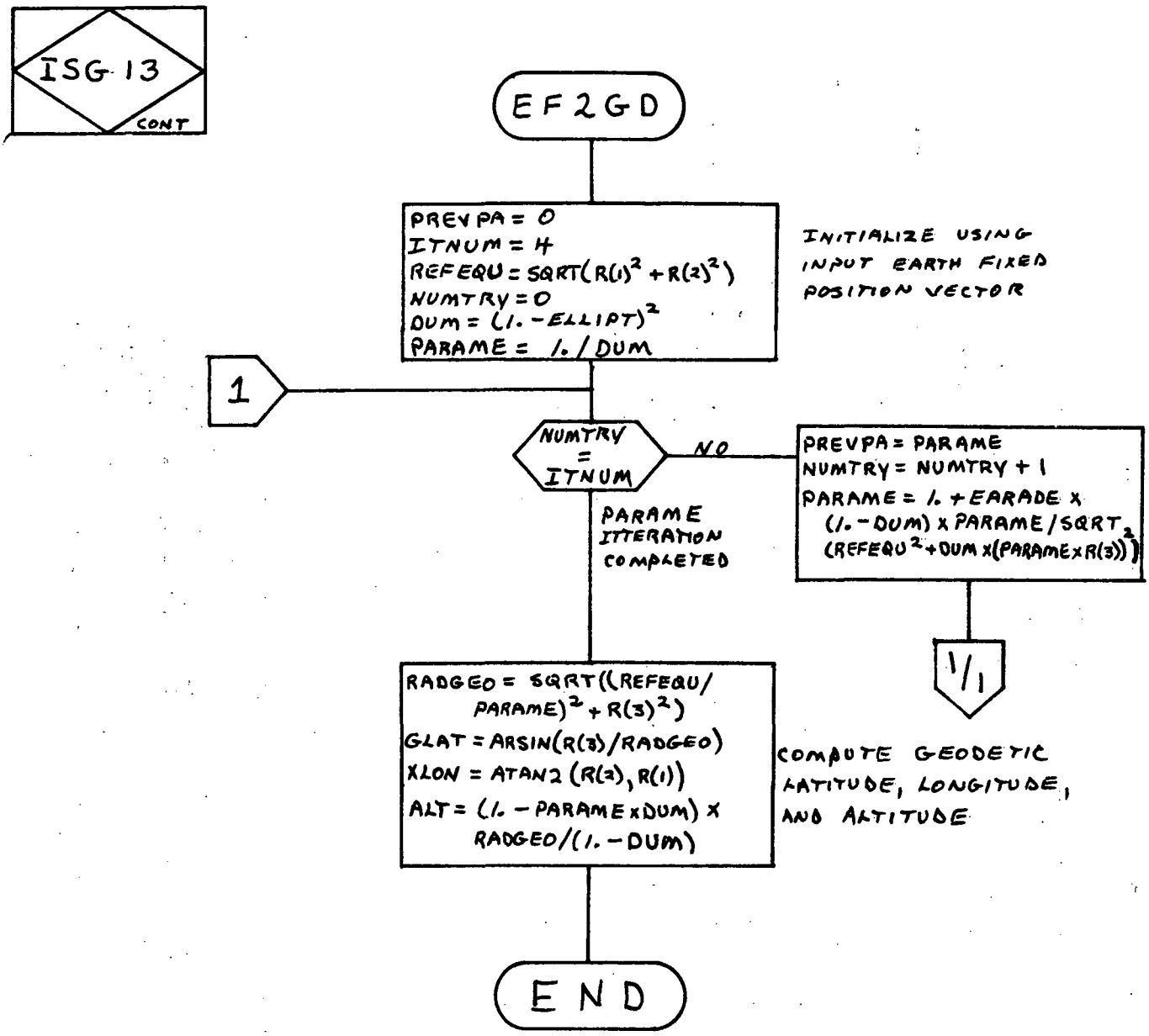


Figure 5.51-1.- Functional logic flow for the EF2GD transformation routine.

5.52 ROUTINE NAME - EGRT COMPUTATIONAL ROUTINE

5.52.1 Purpose

The EGRT routine computes the range angle from entry interface to landing.

5.52.2 Functional Description

The range computed by EGRT is a great circle arc range from the entry interface point to the tangent of the TAEM alinement circles plus the distance around the circles plus the distance from the circles to the runway threshold. The heading error, DELAZ, is also computed.

5.52.3 Assumptions and Limitations

None.

5.52.4 Method

None.

5.52.5 Routine Input/Output Variables

Table 5.52-I contains the definitions of all the input/output variables for the EGRT computational routine.

5.52.6 Functional Logic Flow

Figure 5.52-1 contains the functional logic flow for the EGRT computational routine.

5.52.7 Diagnostics and Debug

When the user selects the debug print option, the variables TRANG and DELAZ are printed. These are the range and range error, respectively.

5.52.8 Special Comments

None.

5.52.9 Reference

SMCC Level B Formulation Requirement, Entry Guidance and Entry Autopilot. JSC IN 76-FM-77 dated September 1976.

TABLE 5.52-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine EGRT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AMILE		Real	I		C	GLOCON(89)	Refer to table 5.1-I for all code symbol definitions.
DBARR		Real	I		C	DTMCON(1)	
IPOUT		6CH	I		C		
IOUNIT		Intg	I		C		
PI		Real	I		C	GLOCON(59)	
RTURNN		Real	I		C	DTMCON(2)	
RAZ		Real	I		C		
RCHMAG		Real	I		C		
REC		Real	I		C		
RLS		Real	I		C		
XYZEN		Real	I		C		
VRHOTD		Real	I		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.52-I.- Concluded

Routine EGRT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DELAZ		Real	O		C,T		
TRANG		Real	O	n. mi.	C,T		

NOTES:	TYPE				USE	SOURCE
	Free	Dubl	18CH	Mix	I = Input	IT = Interface Table
	Intg	2CH	36CH	Char	O = Output	T = Terminal
	Real	6CH	72CH	Bin	I/O = Input/Output	A = Calling Argument
						C = Common
						F = Disk File
						SAM = System Available Memory

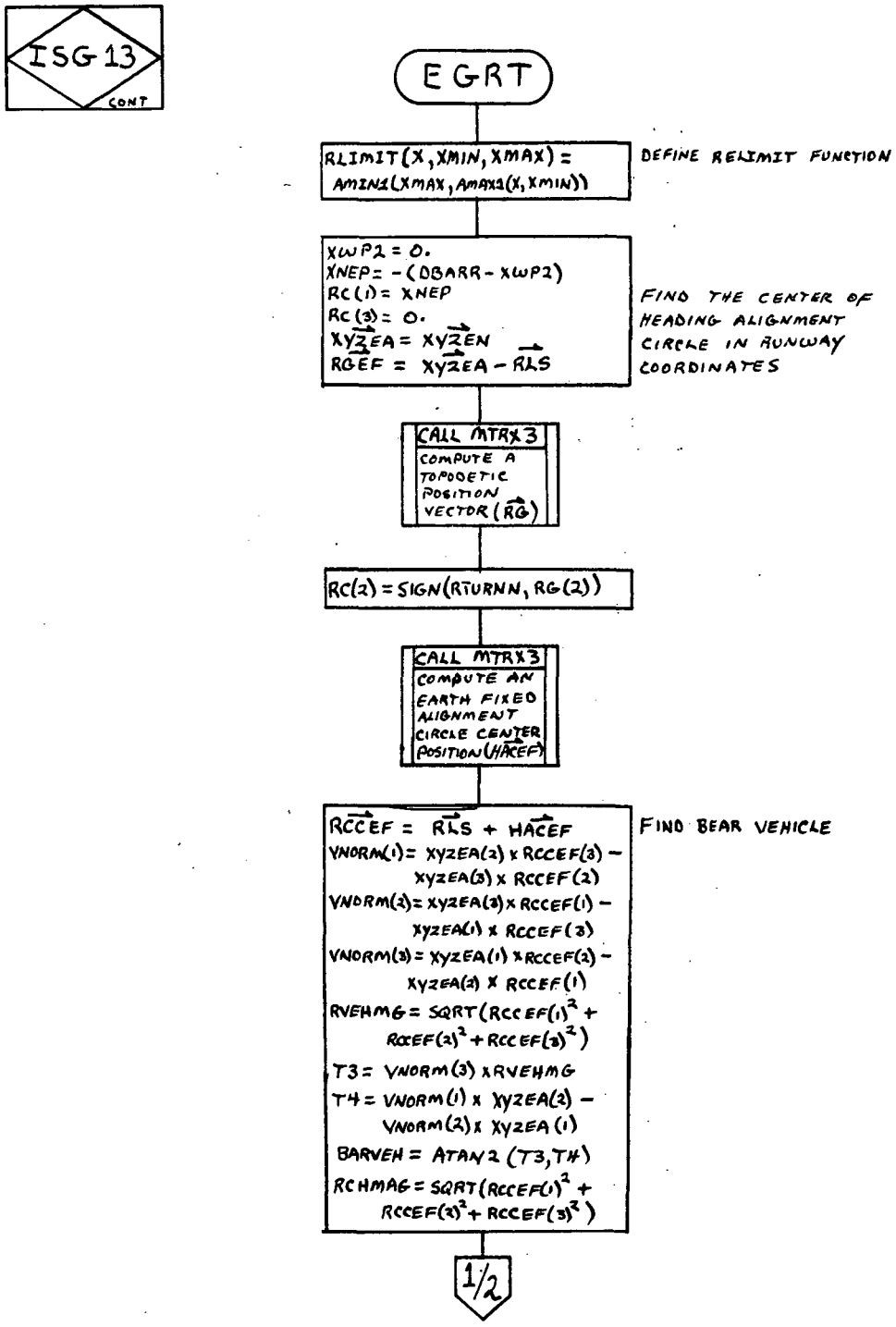
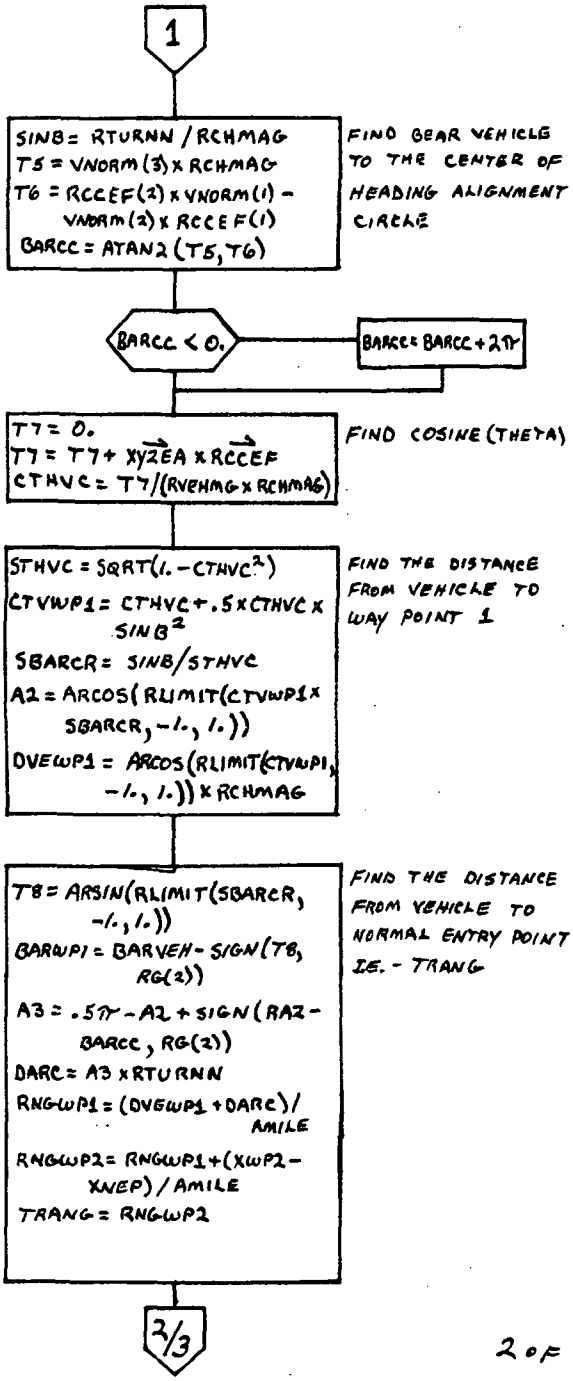


Figure 5.52-1.- Functional logic flow for the EGRT computational routine.



2 of 3

Figure 5.52-1.- Continued.

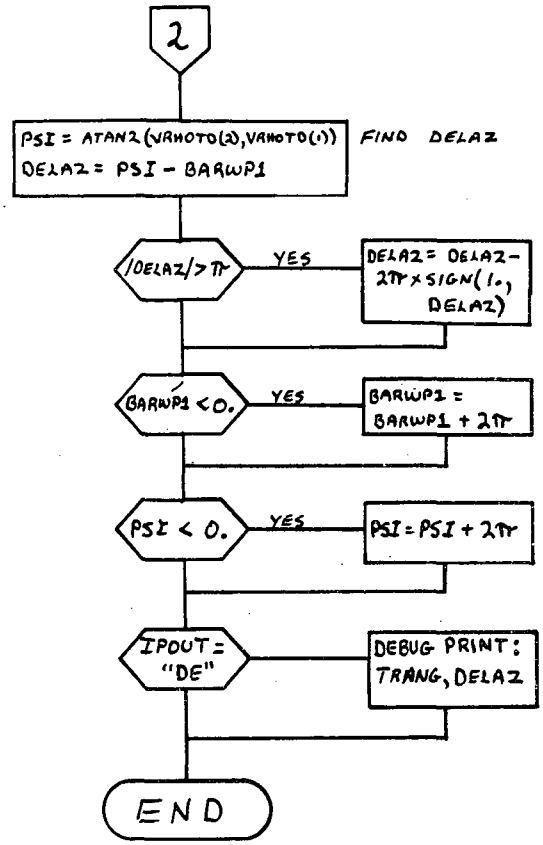


Figure 5.52-1.- Concluded.

5.53 ROUTINE NAME - DTT14 COMPUTATIONAL ROUTINE - SEGMENT 14

5.53.1 Purpose

The DTT14 routine does set up for and calls the FVE routine to obtain a compatible velocity, flightpath angle, and range target at entry interface. Along with the FVE routine, it is computational segment 14.

5.53.2 Method

DTT14 calls the FVE routine in the approximate mode to compute the flightpath angle and range which are compatible with the input entry velocity. The C1 target line intercept for PEG is then computed from the entry velocity and flightpath angle using the input C2 target line slope.

5.53.3 Assumptions and Limitations

The use of the FVE simulation of the entry target generator does not have a precise solution mode for which the C_2 slope constant is also computed. On the final pass the FVE simply produces another approximate solution. However, the extra pass does aid in the convergence to a good solution.

5.53.4 Method

None.

5.53.5 Routine Input/Output Variables

Table 5.53-I contains the definitions of all the input/output variables for the DTT14 computational routine.

5.53.6 Functional Logic Flow

Figure 5.53-1 contains the functional logic flow for the DTT14 computational routine.

5.53.7 Diagnostics and Debug

When the user selects the debug print option the following variables are printed, IAME, IFINAL, ICALLL, IGAMFR, IPRETG, IPRNG, VEIM, GAMETG, THELSD, C1, C2, and TEIFVE.

5.53.8 Special Comments

None.

5.53.9 References

None.

TABLE 5.53-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT14

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AMILE		Real	I		C	GLOCON(89)	Refer to table 5.1-1 for all code symbol definitions.
CA		Real	I/O		C,T	C1,C2	
DC1		Real	I		C		
DWPR		Real	I		C		
GAMETG		Real	I/O		C,T		
IAME		Intg	I/O		C,T		
ICALLL		Intg	I/O		C,T		
IFINAL		Intg	I/O		C,T		
IGAMFR		Intg	I/O		C,T		
IPRNG		Intg	I/O		C,T		
IPOUT		6CH	I		C		
IOUNIT		Intg	I		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.53-I.- Concluded

Routine DTT14

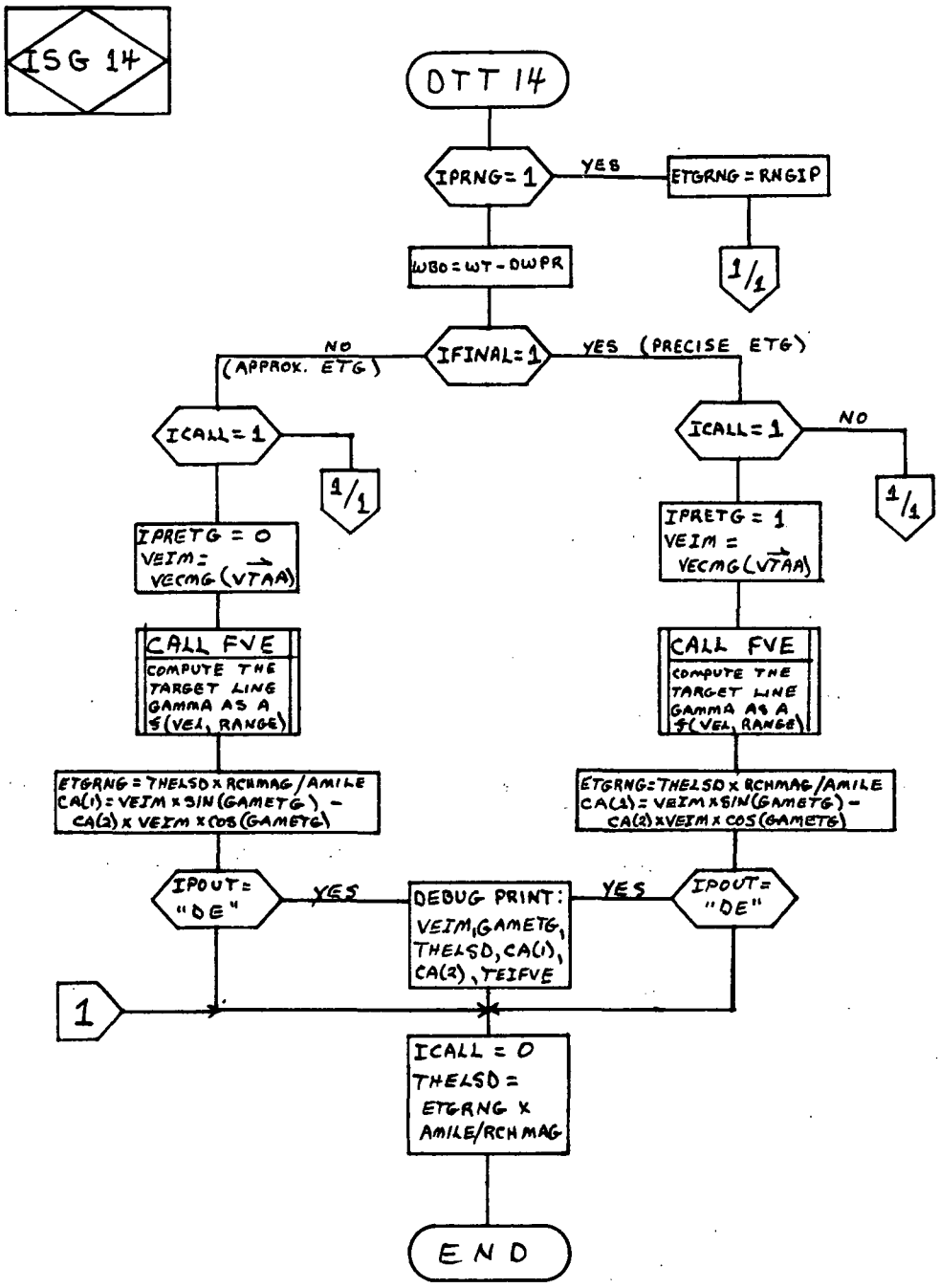
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
RCHMAG		Real	I		C		
RNGIP		Real	I	n. mi.	C		
TEIFVE		Real	I/O		C,T		
THELSD		Real	I/O		C,T		
VTAA		Real	I		C		
WT		Real	I		C	DTMVEC(13)	
ETGRNG		Real	O	n. mi.	C		
IPRETG		Intg	O		C,T		
VEIM		Real	O		C,T		
WBO		Real	O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory



Page 1 of 1

Figure 5.53-1.- Functional logic flow for the DTT14 computational routine.

5.54 ROUTINE NAME - FVE COMPUTATIONAL ROUTINE

5.54.1 Purpose

The FVE routine is a simulation of the entry target generator (ETG) using an input target line.

5.54.2 Functional Description

FVE extracts the entry flightpath angle from the input entry target line as a function of entry velocity and entry range desired.

5.54.3 Assumptions and Limitations

The input target line used by FVE must be a valid ETG-generated target line for the characteristic deorbit being planned in order to assure an accurate solution.

5.54.4 Method

None.

5.54.5 Routine Input/Output Variables

Table 5.54-I contains the definitions of all the input/output variables for the FVE computational routine.

5.54.6 Functional Logic Flow

Figure 5.54-1 contains the functional logic flow for the FVE computational routine.

5.54.7 Diagnostics and Debug

If the entry velocity is greater than the maximum target line velocity or less than the minimum target line velocity an error message is printed. However, processing will continue with the target line maximum or minimum velocity used as the target velocity. The user should compare the actual entry velocity to the target line inputs and supply a new target line as required.

5.54.8 Special Comments

None.

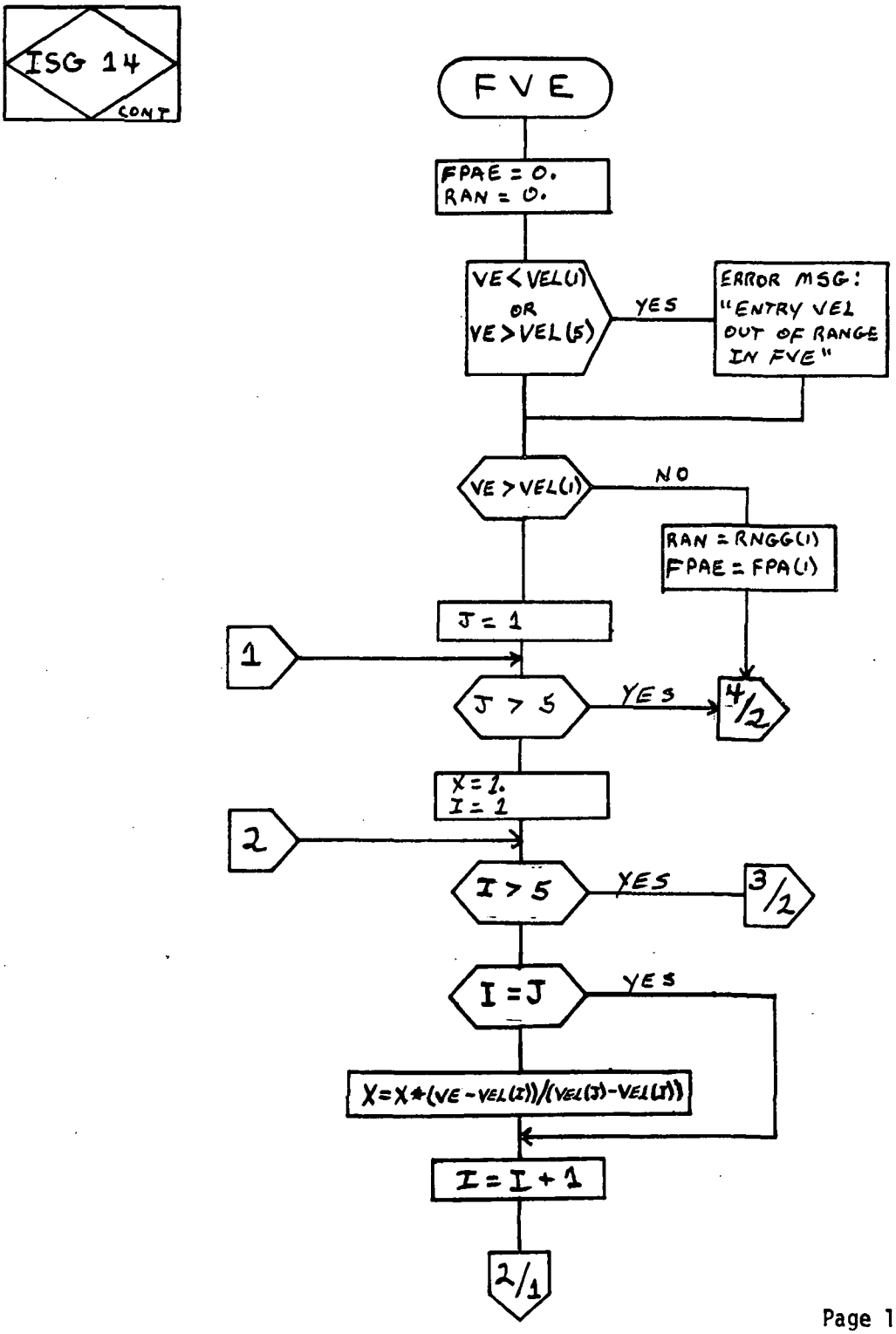
5.54.9 References

None.

TABLE 5.54-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine FVE

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AMILE		Real	I		C	GLOCON(89)	Refer to table 5.1-I for definitions not found here.
FPA		Real	I	deg	C	FVECON(6)	
LU		Intg	I		C		
RCHMAG		Real	I		C		
RNGG		Real	I	n. mi.	C	FVECON(11)	
RADIAN		Real	I		C	GLOCON(83)	
VE		Real	I		A		Entry velocity
VEL		Real	I		C	FVECON(1)	
DTAE		R	O		A		Entry range
DTE		Real	O		A		Time to entry interface
FPAE		Real	O		A		Entry flightpath angle
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory



Page 1 of 2

Figure 5.54-1.- Functional logic flow for the FVE computational routine.

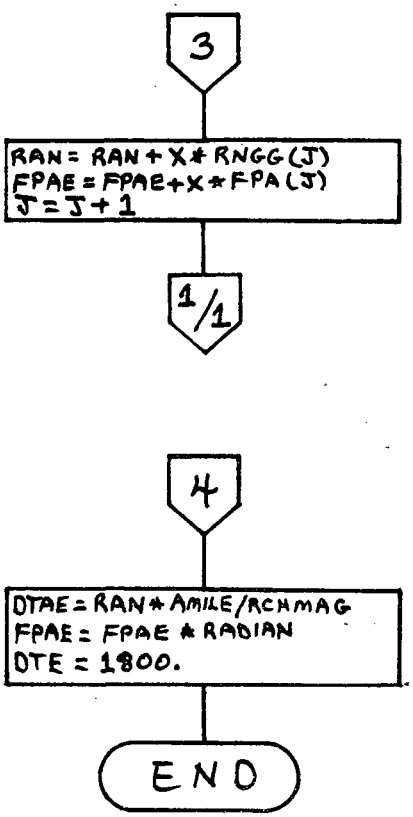


Figure 5.54-1.- Concluded.

5.55 ROUTINE NAME - DTMOT OUTPUT ROUTINE - SEGMENT 19

5.55.1 Purpose

The DTMOT routine performs the array output operations. It produces position/velocity phase tables for the primary and backup system solutions and also the summary table output array.

5.55.2 Functional Description

DTMOT computes the thrust, pitch, and yaw angles for the primary and backup solutions. It then converts from internal units to the user selected external units set. The primary system and backup system phase table arrays are then loaded with the appropriate quantities and the summary table array is also constructed.

5.55.3 Assumptions and Limitations

None.

5.55.4 Method

None.

5.55.5 Routine Input/Output Variables

Table 5.55-I contains the definitions of all the input/output variables for the DTMOT output routine.

5.55.6 Functional Logic Flow

Figure 5.55-1 contains the functional logic flow for the DTMOT output routine.

5.55.7 Diagnostics and Debug

None.

5.55.8 Special Comments

None.

5.55.9 References

None.

TABLE 5.55-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTMOT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AMILE		Real	I		C	GLOCON(89)	Refer to table 5.1-I for all code symbol definitions.
ATPBU		Real	I		C		
ATPPR		Real	I		C		
CA		Real	I/O		C		
CRSRNG		Real	I/O		C		
DTCOST		Real	I/O		C		
DVBU		Real	I/O		C		
DVPR		Real	I/O		C		
DVIMP		Real	I/O		C		
DVOBBU		Real	I/O		C		
DVOBPR		Real	I/O		C		
DWBU		Real	I/O		C		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory

TABLE 5.55-I.- Continued

Routine DTMOT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
DWPR		Real	I/O		C		
EIALT		Real	I		C	GLOCON(57)	
ETGRNG		Real	I/O		C		
GAMETG		Real	I/O		C		
IFUEL		Intg	I		C		
INGBU		Intg	I		C		
INGPR		Intg	I		C		
IPARM		Intg	I		C		
JDGUID		6CH	I		C		
NSYS		Intg	I		C		
PSIBU		Real	I/O		C		
PSIPR		Real	I/O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.55-I.- Continued

Routine DTMOT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
RAAA		Real	I/O		C		
RDA		Real	I/O		C		
REIS		Real	I/O		C		
RENK		Real	I/O		C		
RTA		Real	I/O		C		
SESCON		Free	I		C		
TIG		Real	I/O		C		
TGO		Real	I/O		C		
THETEI		Real	I/O		C		
THETLS		Real	I/O		C		
THELSD		Real	I/O		C		
TP		Real	I/O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.55-I.- Continued

Routine DTMOT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TPCHB		Real	I/O		C		
TPCHP		Real	I/O		C		
TRANG		Real	I/O		C		
TT		Real	I/O		C		
TYAWB		Real	I/O		C		
TYAWP		Real	I/O		C		
UXDTM		Real	I		C		
UYDTM		Real	I		C		
UZDTM		Real	I		C		
VA		Real	I/O		C		
VDA		Real	I/O		C		
VEIS		Real	I/O		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.55-I.- Continued

Routine DTMOT

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
VEIM		Real	I/O		C		
VENK		Real	I/O		C		
VG		Real	I/O		C		
VGMAG		Real	I/O		C		
VGOX		Real	I/O		C		
VGOY		Real	I/O		C		
VGOZ		Real	I/O		C		
VTAA		Real	I/O		C		
WBO		Real	I/O		C		
WCG		Real	I/O		C		
WT		Real	I/O		C	DTMVEC(13)	
XYZI		Real	I/O		C	DTMVEC(2)	
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.55-I.- Concluded

Routine DTMOT

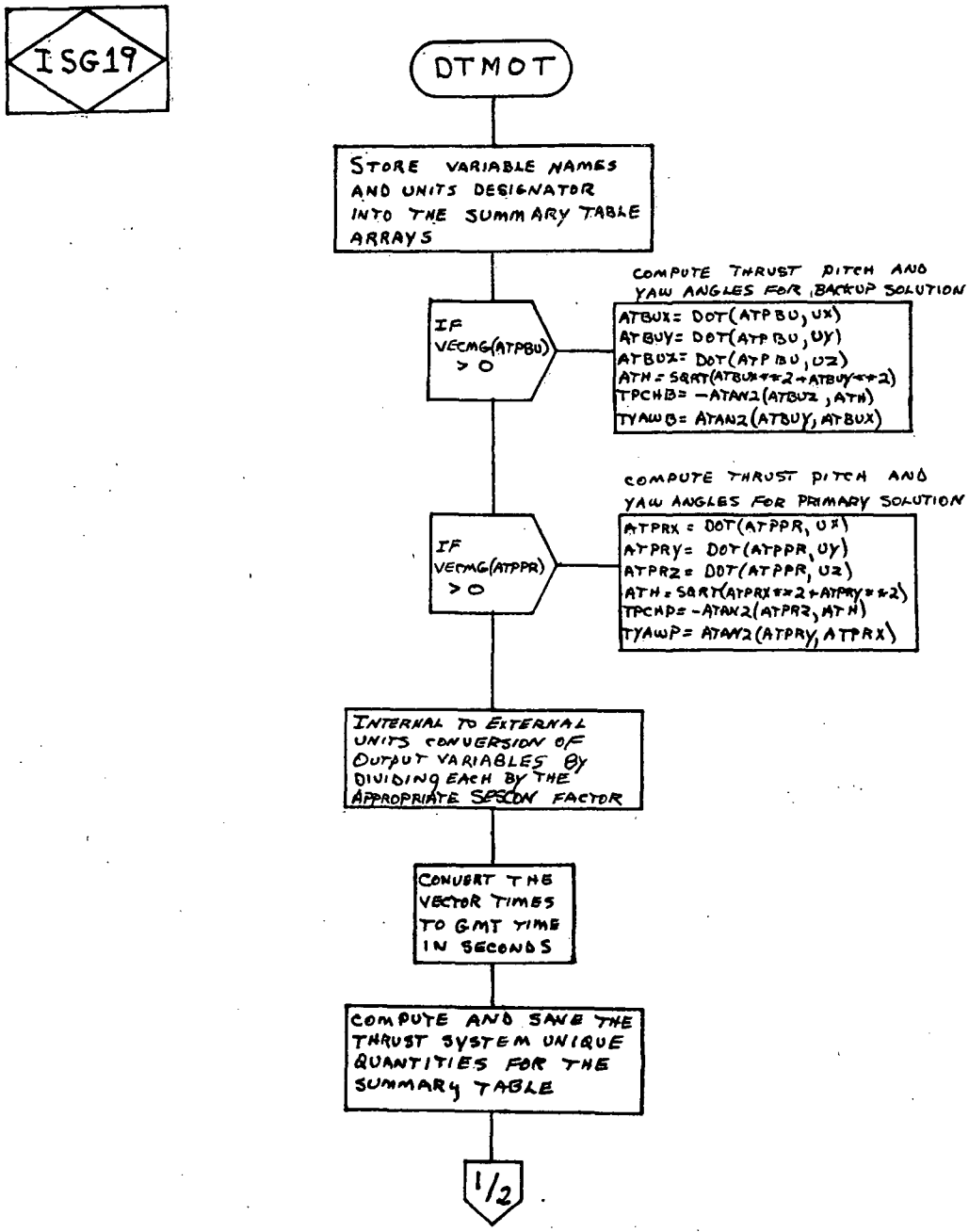
Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
XYZID		Real	I/O		C	DTMVEC(5)	
ZEROT		Real	I/O		C		
PVTABB		Real	O		F		
PVTABP		Real	O		F		
SUMTAB		Free	O		F		

NOTES:

TYPE
Free Dubl 18CH Mix
Intg 2CH 36CH Char
Real 6CH 72CH Bin

USE
I = Input
O = Output
I/O = Input/Output

SOURCE
IT = Interface Table
T = Terminal
A = Calling Argument
C = Common
F = Disk File
SAM = System Available Memory



Page 1 of 2

Figure 5.55-1.- Functional logic flow for the DTMOT output routine.

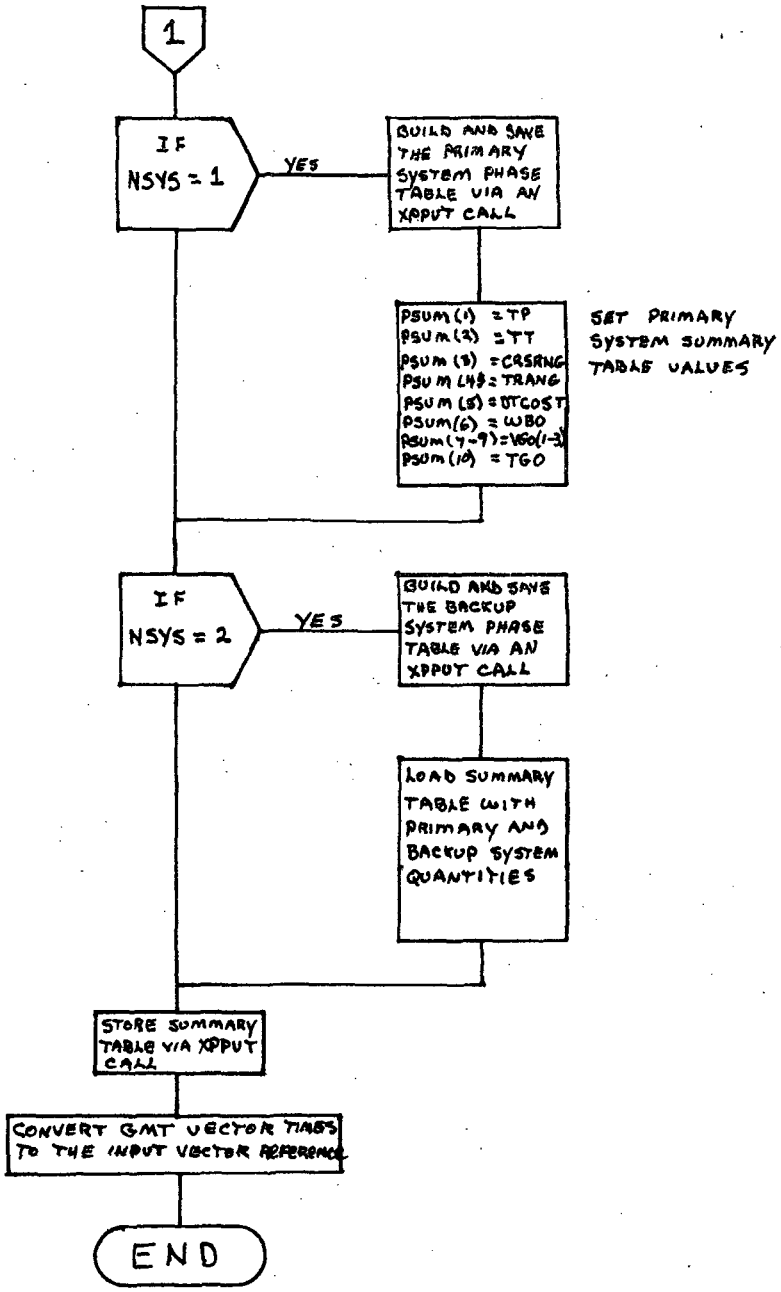


Figure 5.55-1.- Concluded.

5.56 ROUTINE NAME - DTT24 COMPUTATIONAL ROUTINE - SEGMENT 24

5.56.1 Purpose

The DTT24 routine computes the cross range distance from the trajectory plane to the landing site at the time of closest approach of the entry vector.

5.56.2 Functional Description

DTT24 uses the entry vector after the deorbit burn to compute the orbital rate and predict the time of closest approach. The landing site location at the predicted time of closest approach is then computed. The time of closest approach is adjusted until the unit vector to the landing site is normal to local horizontal within a small input tolerance. When this condition is met the cross range distance from the orbital closest approach point and the landing site is computed.

5.56.3 Assumptions and Limitations

The cross range is a great circle distance from the closest approach point to the landing site based on the entry orbit.

5.56.4 Method

An iterative loop is used to adjust the time of closest approach until a unit vector to the landing site from the closest approach point is normal to the local horizontal at the time of closest approach. The cross range computed is positive in the direction of the negative angular momentum vector. Thus, the cross range is positive when the orbital groundtrack is north of the landing site.

5.56.5 Routine Input/Output Variables

Table 5.56-I contains the definitions of all the input/output variables for the DTT24 computational routine.

5.56.6 Functional Logic Flow

Figure 5.56-1 contains the functional logic flow for the DTT24 computational routine.

5.56.7 Diagnostics and Debug

When the user selects the debug print option the following variables are output, TC, THETLS, THETDT, DELTAT, DTCRO, T, and CRSRNG.

5.56.8 Special Comments

None.

5.56.9 References

None.

TABLE 5.56-I.- ROUTINE INPUT/OUTPUT VARIABLES

Routine DTT24

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
AMILE		Real	I		C	GLOCON(89)	Refer to table 5.1-I for all code symbol definitions.
CRTOL		Real	I		C	DTMCON(30)	
DTCRO		Real	I/O		C,T	DTMCON(32)	
ERAI		Real	I		C		
IOUNIT		Intg	I		C		
IPOUT		6CH	I		C		
RCHMAG		Real	I		C		
RTA		Real	I		C		
TDENMN		Real	I		C	DTMCON(31)	
THETLS		Real	I/O		C,T		
TLATC		Real	I		C		
TLONG		Real	I		C		
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

TABLE 5.56-I.- Concluded

Routine DTT24

Code symbol	Math symbol	Type	Use	Units	Source	External label	Definition
TT		Real	I		C		
VTAA		Real	I		C		
WE		Real	I		C	GLOCON(13)	
XMU		Real	I		C	GLOCON(1)	
DNRNG		Real	O	n. mi.	C		
CRSRNG		Real	O	n. mi.	C,T		
TC		Real	O		C,T		
NCMAX		Intg	I		C	DTMCON(38)	
DELTAT		Real	O		T		Predicted coast time to closest approach
T		Real	O		T		Time to coast from E.I. to landing
THETDT		Real	O		T		Orbital rate
NOTES:		TYPE Free Dubl 18CH Mix Intg 2CH 36CH Char Real 6CH 72CH Bin				USE I = Input O = Output I/O = Input/Output	SOURCE IT = Interface Table T = Terminal A = Calling Argument C = Common F = Disk File SAM = System Available Memory

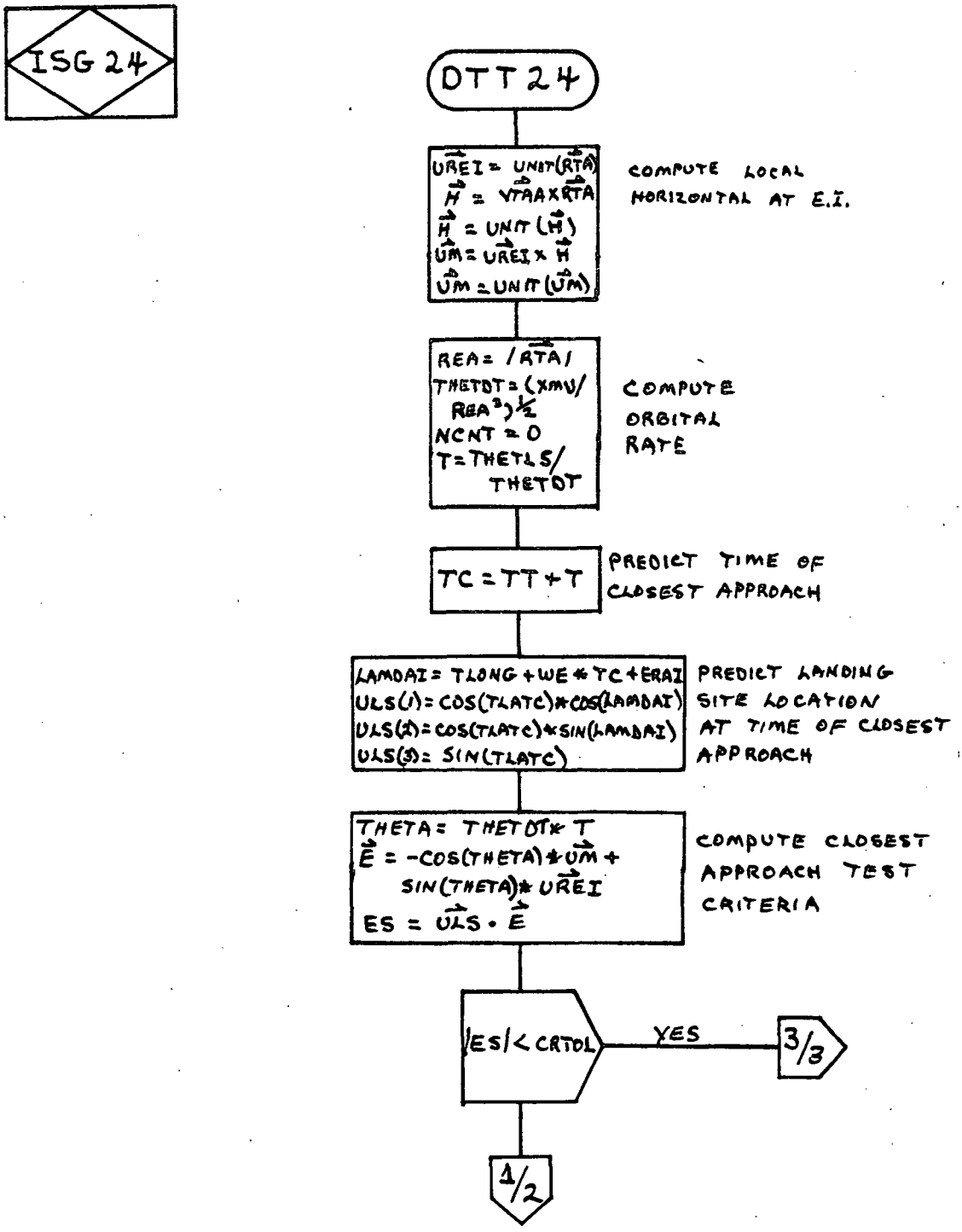


Figure 5.56-1.- Functional logic flow for the DTT24 computational routine.

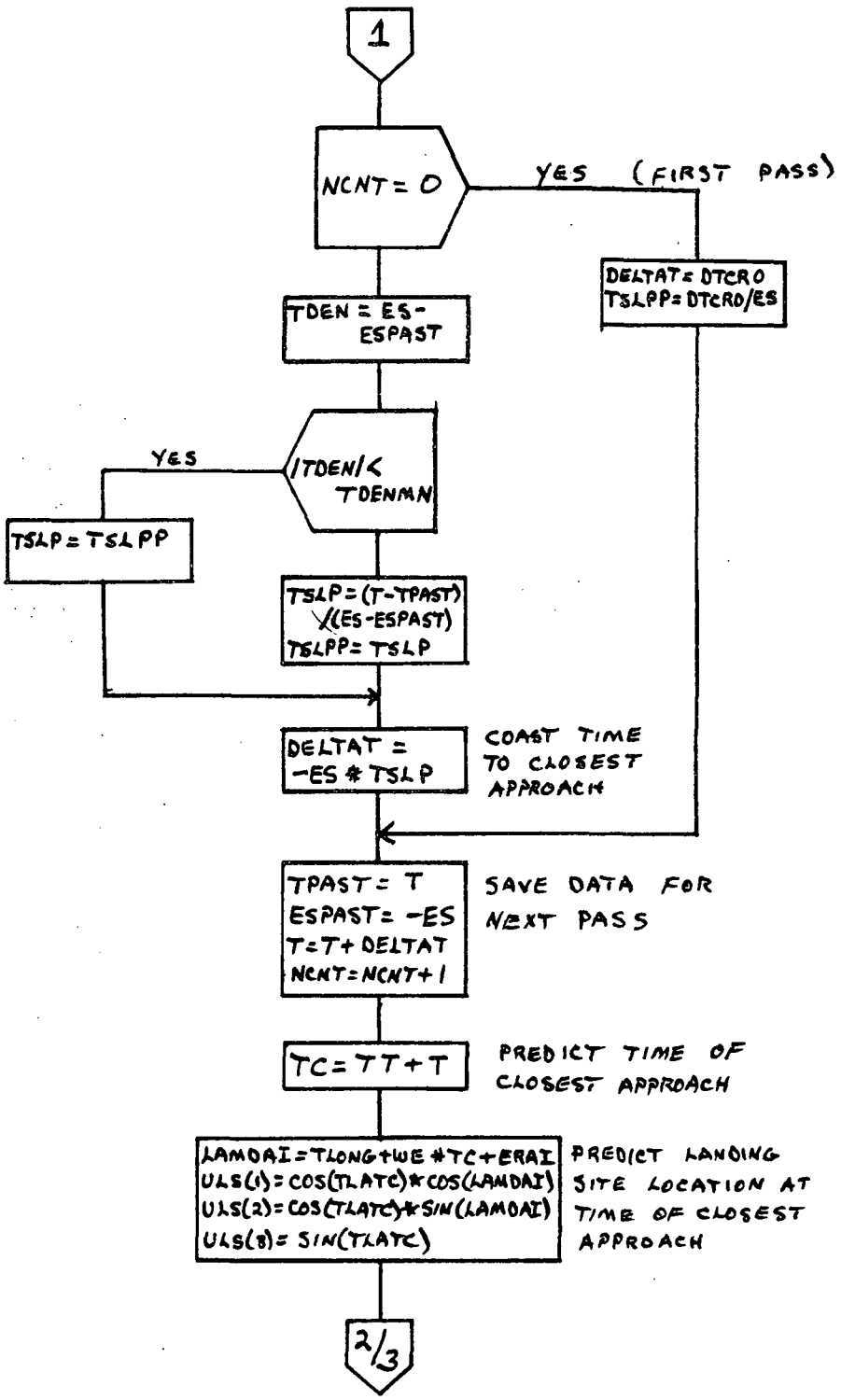


Figure 5.56-1.- Continued.

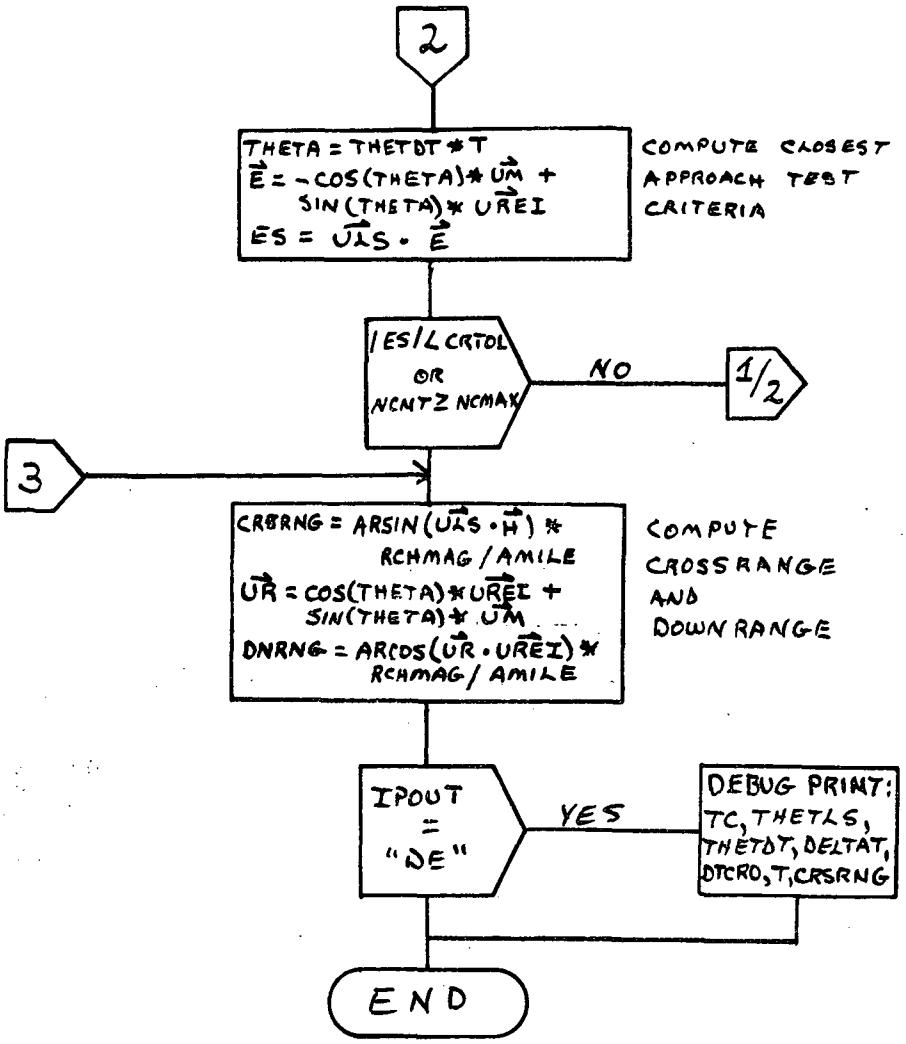


Figure 5.56-1.- Concluded.

☆ U.S. GOVERNMENT PRINTING OFFICE: 1979—671-099/2345

DISTRIBUTION FOR JSC IN 77-FM-18, REVISION 1, VOLUME III

JM6/Technical Library (2)
JM61/Center Data Management (3)
FD7/J. Fisher
FM/Chief
FM/E. Davis
FM2/Chief
    Section Heads (4)
    D. Alexander
FM4/Chief
    C. Graves
FM6/Chief
    A. Nolting
    D. Braley
    R. Davis (2)
    E. Fridge
    J. Martin
    G. Martinez
    R. Merriam
    W. Pruett
    W. Reini
    R. Reynolds
    R. Rogan
    G. Roush
    G. Weisskopf
FM8/Chief
FM14/Report Control Files (25)
     S. Cole
     B. Woodland
FM15/Chief
FM17/Chief
CSC/M30/O. Dial
        R. Herder
IBM/MC56/R. Turner (2)
MDTSCO/B. Brown (10)

Deletions or additions to this distribution must be coordinated through Gloria Martinez/FM6; 483-4491.