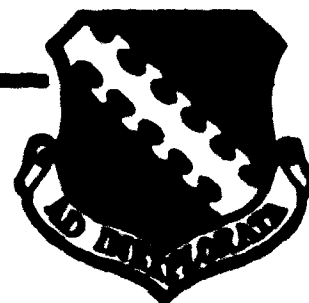


AFFTC-TN-96-02

AD-A286 920
[Barcode]



AIR FORCE FLIGHT TEST CENTER TOLAND USERS' GUIDE

KENT STANDLEY
Aerospace Engineer

OCTOBER 1996

TECHNICAL INFORMATION HANDBOOK

97 3 19 503

Approved for public release; distribution is unlimited.

DTIC QUALITY INSPECTED

AIR FORCE FLIGHT TEST CENTER
EDWARDS AIR FORCE BASE, CALIFORNIA
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

97-00467



A-1

A
F
F
T
C

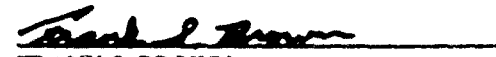
This handbook, AFPTC-TD-96-02, *Air Force Flight Test Center TOLAND Users Guide*, was prepared on behalf of the 412 TW/TSF, Flight Dynamics Division, Edwards Air Force Base, California 93524-6843

Prepared by:

This handbook has been reviewed and is approved for publication 23 October 1996



KENT STANDLEY
Aerospace Engineer



FRANK S. BROWN
Acting Chief, Performance Strategic Flight



for ROBERT D. EVANS
Chief, Flight Dynamics Division

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE September 1996		3. REPORT TYPE AND DATES COVERED Final
4. TITLE AND SUBTITLE AFFTC TOLAND Users' Guide			5. FUNDING NUMBERS N/A 99800000 PEC 658077	
6. AUTHOR(S) Standley, Kent				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Flight Test Center 412 Test Wing/TSFC 195 East Popson Avenue Bldg 2750 Edwards AFB CA 93524-6841			8. PERFORMING ORGANIZATION REPORT NUMBER AFFTC-TTH-96-02	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			10. SPONSORING / MONITORING AGENCY REPORT NUMBER N/A	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE A	
13. ABSTRACT (Maximum 200 words) This report describes a computer program that will model the takeoff and landing performance of aircraft. The program is written in Microsoft™ FORTRAN for execution on an IBM-compatible personal computer running with DOS 3.3 or better. Models of lift coefficient, drag coefficient, gross thrust, propulsive drag, and fuel flow within user-provided subroutines are required. The simulated aircraft can be jet or propeller powered, however, thrust must be passed from the user-provided subroutine. The intent of this handbook is to provide a guide to the programmer and operator of this program as well as to document the equations and assumptions used in the program. This program was adapted from the NASA takeoff and landing program documented in NASA Technical Memorandum X.622,333, (1973). This Air Force version of the program has been made to suit the needs of test and evaluation while the NASA program served as a design tool.				
14. SUBJECT TERMS simulators user manuals landings TOLAND			15. NUMBER OF PAGES 194	
simulation aircraft continued takeoffs			16. PRICE CODE	
computer programs takeoffs refused takeoffs				
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT SAR	



TABLE OF CONTENTS

BACKGROUND.....	1
INTRODUCTION.....	2
TAKEOFFS.....	3
LANDINGS.....	4
REFUSED TAKEOFFS.....	5
CONTINUED TAKEOFFS.....	6
NAMelist DESCRIPTIONS.....	7
Namelist DATA.....	7
Namelist DATA2.....	8
Namelist TKO.....	11
Namelist TKO2.....	11
Namelist TKOARY.....	13
Namelist LND.....	14
Namelist LND2.....	14
Namelist ROL.....	15
Namelist ROL2.....	15
OUTPUT DESCRIPTIONS.....	17
Takeoff and Continued Takeoff Output.....	17
Refused Takeoff Output.....	17
Landing Output.....	17
PROGRAM STRUCTURE.....	19
SUBROUTINE DESCRIPTIONS.....	21
User Provided Subroutines.....	22
Subroutine INICURV.....	22
Subroutine FORCEX.....	22
Subroutine 'FXXAERO'.....	22
Subroutine GEFFECT.....	23
Subroutine 'FXXENG'.....	23
Subroutine SPOOLUP.....	23
Subroutine SPOOLDNF.....	23
Subroutine SPOOLDNR.....	23
Subroutine GENMU.....	23
Program Subroutines.....	23
Subroutine INITIAL.....	23
Subroutine FRETRAC.....	24
Subroutine GRETRAC.....	24

TABLE OF CONTENTS (Continued)

Subroutine PITCH	24
Subroutine SPDBRAK	24
Subroutine SPOIL	24
Subroutine TVECTOR	24
Subroutine TAKOFF	24
Subroutine LANDNG	24
Subroutine STEDYST	24
Subroutine FLARENZ	27
Subroutine APPROCH	28
Subroutine FLARE	28
Subroutine ROLL	28
Subroutine INTX	28
Subroutine INTG	28
Subroutine DERIVGR	28
Subroutine DERIVAT	28
Subroutine DERIVAL	29
Subroutine ERROR	29
Subroutine HALT	29
Subroutine ATMOSPH	29
Subroutine SPEED	29
Subroutine ITRLND	29
Program Function Descriptions	30
Function DGGT	30
Function DVDI	30
Function DVTDH	30
Function DADH	30
Function DDELTDH	30
Function DSIGDH	30
Function INTERP	30
Function ZEROX	30
COMMON BLOCK DESCRIPTIONS.....	31
Common CTRL	31
Common AIRCRFT	33
Common AERO	34
Common ENGINE	35
Common AIRBORN	37
Common AIRSPED	38
Common RUNWAY	39
Common INTEG	41
Common FPINTEG	41
Common ATMOS	42
Common CONST	43
Common FLAGS	44

TABLE OF CONTENTS (Continued)

Common CHARV	46
Common FLAPDAT	48
Common GEARDAT	49
Common VECTDAT	50
Common CINDEX	51
Common RACURV	51
Common VALUES	51
Common TABLES	51
REFERENCES	53
BIBLIOGRAPHY	53
APPENDIX A: EQUATIONS OF MOTION	A-1
APPENDIX B: SOURCE CODE LISTING	B-1
Main Program	B-1
Program Subroutines	B-4
Subroutine INITIAL	B-4
Subroutine FRETRAC	B-9
Subroutine GRETRAC	B-10
Subroutine PITCH	B-12
Subroutine SPDBRAK	B-14
Subroutine SPOIL	B-16
Subroutine TVECTOR	B-18
Subroutine TAKOFF	B-20
Subroutine LANDNG	B-32
Subroutine STEDYST	B-37
Subroutine FLARENZ	B-45
Subroutine APPROCH	B-48
Subroutine FLARE	B-51
Subroutine ROLL	B-56
Subroutine INTX	B-66
Subroutine INTG	B-68
Subroutine DERIVGR	B-69
Subroutine DERIVAT	B-71
Subroutine DERIVAL	B-73
Subroutine ERROR	B-75
Subroutine HALT	B-76
Subroutine ATMOSPH	B-77
Subroutine SPEED	B-78
Subroutine ITRLND	B-79
Program Functions	B-81
Function DGDG	B-81
Function DVDT	B-81
Function DVTDH	B-82
Function DADH	B-84

TABLE OF CONTENTS (Continued)

Function DDELTDH.....	B-85
Function DSIGDH.....	B-86
Function INTERP.....	B-87
Function ZEROX.....	B-88
User Provided Subroutine Examples.....	B-90
Subroutine INICURV.....	B-90
Subroutine FORCEX.....	B-91
Subroutine FXXAERO.....	B-93
Subroutine GEFFECT.....	B-96
Subroutine FXXENG.....	B-97
Subroutine SPOOLUP.....	B-99
Subroutine SPOOLDNF.....	B-100
Subroutine SPOOLDNR.....	B-101
Subroutine GENMU.....	B-102
APPENDIX C: MICROSOFT FORTRAN COMPILER INFORMATION.....	C-1
APPENDIX D: NOMENCLATURE.....	D-1
APPENDIX E: INDEX.....	E-1

LIST OF ILLUSTRATIONS

Figure 1	Low Flare Landing Distance Definitions	17
Figure 2	High Flare Landing Distance Definitions	18
Figure 3	Takeoff and Landing Program Structural Layout	19
Figure 4	Takeoff and Landing Program Major Subroutine Associations	20
Figure 5	Steady-State Approach Solution	26
Figure 6	Initial Flare Load Factor Determination	27

LIST OF TABLES

Table 1	Takeoff Tracking Variables	3
Table 2	Landing Tracking Variables	4
Table 3	Namelist	7
Table 4	Subroutine Usage	21
Table C-1	Subroutine Locations	C-1

BACKGROUND

This handbook describes a general purpose computer program that will simulate the performance characteristics of aircraft takeoffs and landings. This program has been adapted from the original NASA Ames TOLAND program and documented in NASA Technical Memorandum X.62,333 (Reference 1). This original was written in FORTRAN IV in 1973 as a design tool by Jeff Bowles and Thomas Galloway. It was later modified by Dave Nesst and Wayne Olson circa 1979 to simulate refused takeoffs and rolling minimum interval takeoffs (MITOs), Reference 2. In 1980, the modified version was first used for performance simulation and flight test standardization on the F-15C flight test program and by the F-16A for landing performance tests in 1982.

Because the original NASA TOLAND was originally programmed as a design tool, it did not fully suit the needs of the flight test community. Program inputs, such as desired rate of climb, were needed as outputs; and program outputs, such as flare initiation height, were needed as inputs in a flight test oriented program. The original program ran on a large mainframe computer. As the engineers moved their flight test data processing to personal computers and workstations, a need was established to move the TOLAND program also. This new program has been completely rewritten to meet these flight test requirements. While the overall methods of numerical integration and the equations of motion have remained the same, the source code is all new.

INTRODUCTION

This document describes the inputs, the equations and computational techniques used, the subroutines and the main program developed to simulate the takeoff, landing and refused takeoff maneuvers of a given aircraft. The program can simulate takeoffs, continued takeoffs (with engine failure), refused takeoffs, and landings, and can be applied to conventional, thrust vectored and powered lift aircraft. This documentation describes the internal program subroutines, the namelist inputs, the equations and the computational techniques used.

The aircraft is treated as a point mass confined to motion in a vertical plane; the rotational dynamics have been neglected. This simplification requires an estimation of the angular rates and precludes the use of this program as a tool to evaluate minimum control speeds. These angular rates are approximated by a finite difference form or input by the user as commanded rates.

The user is required to provide at least two subroutines which pass lift coefficient (C_L), drag coefficient (C_D), net thrust (F_N), and fuel flow (W_f/dt). If random access curve files are used for aerodynamic or engine data, the initialization subroutine (INICURV) must be provided. Additional subroutines may be modified by the user for ground effect, engine transients, coefficient of friction (μ), flap retraction, gear retraction, pitch control, spoiler control, and thrust vectoring.

Detailed descriptions of the program inputs are provided in the Namelist Description section and program outputs in the Output Description section of this document. The program subroutines and functions are explained in the Subroutine Description section of this document. Detailed descriptions of the program variables contained in common blocks are provided in the Common Block Description section. The equations of motion used in the program are shown in Appendix A. Generic examples of all these subroutines are provided with the source code in Appendix B.

TAKEOFFS

The takeoff maneuver is divided into four basic segments: ground roll and rotation, liftoff and initial segment climb, constant calibrated airspeed climb, and second segment climb.

The three-point attitude ground roll is simulated at a user specified power setting, angle of attack and flap deflection. At the rotation speed (VKROTAT), the angle of attack (ALPHA) is increased linearly with time at the commanded angle of attack rate (DADTCMD) until the pitch attitude (THTR0T) is reached. If the pitch attitude is reached before liftoff (before lift and the vertical component of thrust is greater than the aircraft weight), the ground roll is continued until liftoff occurs or until the takeoff ground roll time limit (ROLLMAX) is reached.

During the airborne segments of the simulation, flightpath control is obtained by controlling two dynamic variables during each integration step: acceleration along the flightpath (FPACCEL) and pitch attitude (THETAP). These variables are controlled by modulating angle of attack (ALPHA). If the flightpath acceleration drops to less than zero, or the pitch attitude exceeds the maximum pitch attitude (THTMAX), the angle of attack is reduced. The angle of attack will continue to be reduced until all these constraints are satisfied or when the rate of climb drops below zero. The simulation will terminate with an error message if the rate of climb drops below zero.

Once the initial segment climb altitude (HCLIMB) is reached, the simulation switches from tracking a constant pitch attitude (THTCLM) to tracking a constant calibrated airspeed (VCLMOUT). Once the climbout altitude (HCLMOUT) is reached, the simulation switches from tracking a constant calibrated airspeed to tracking the constant pitch segment climb gradient attitude (THTFly). These pitch changes are performed at commanded angle of attack rate (DADTCMD) until the pitch angle is captured within the pitch angle tolerance (THTTOL). However, the simulation will terminate if the maximum altitude (HMAX), the end airspeed (VKEND), the time limit (TIMEMAX), or the ground distance limit (DISTMAX), is reached first. Table 1 shows the tracking variable (which variable the simulation is trying to match) and altitude ranges for each airborne segment of the takeoff maneuver.

Time, ground distance, and airspeed are provided when the aircraft reaches rotation speed and lifts off to the nearest integration step size time (DTIME) second, and reaches the clearance height (HCLEAR).

Table 1
TAKEOFF TRACKING VARIABLES

Climb Segment	Tracking Variable	FORTTRAN Name	Altitude Range
Initial	Pitch Angle	THTCLM	0 - HCLIMB
Constant Airspeed	Calibrated Airspeed	VCLMOUT	HCLIMB- HCLMOUT
Constant Pitch Angle	Pitch Angle	THTFly	HCLMOUT- HMAX

The "normal mode" for a takeoff is:

HCLIMB = HMAX = 50 Feet
HCLMOUT > 50 feet

in which case the program never executes the first or second segments of the climbout.

LANDINGS

The landing maneuver is divided into four basic segments: approach, flare, transition, and landing ground roll. A steady-state approach is initiated using one of two methods. In one method, the landing approach is simulated at the angle of attack and net thrust required for zero acceleration along and normal to the flightpath, e.g. at an input constant calibrated airspeed. Using the other method, the landing approach is simulated at the calibrated airspeed and net thrust required for zero acceleration along and normal to the flightpath, e.g. at an input angle of attack. These values for angle of attack or calibrated airspeed and net thrust are calculated in subroutine STEDYST using inputs of angle of attack (ALPHA) or approach speed (VKAPP), and flightpath angle (GAMMAPP). The approach starts at the obstacle clearance altitude, (HCLEAR) and ends at the flare altitude (HFLARE).

At the flare altitude, a pull-up is simulated at a normal load factor calculated based on a circular arc trajectory and the thrust is changed from the steady-state condition to idle. This trajectory is defined by the approach flightpath angle (GAMMAPP), the sink rate at touchdown (SINKTD), and the approach speed, (VKAPP). Because airspeed can change during the flare, touchdown airspeed is not an input variable. Table 2 shows the tracking variables and altitude ranges for the various segments of the landing maneuver.

Table 2
LANDING TRACKING VARIABLES

Segment	Tracking Variable	Altitude Range
Approach	Flightpath Angle and either Angle of Attack or Calibrated Airspeed	HCLEAR - HFLARE
Flare	Sink Rate at Touchdown	HFLARE - 0
Transition	Pitch Angle	0
Ground Roll	Not Applicable	0

After touchdown, aerobraking can be accomplished by pitching the aircraft to the angle of attack for aerobraking (AOAABRK), and then achieving a three-point attitude after reaching the minimum speed for aerobraking (VKABRK). The angle of attack for a three-point attitude is defined by the constant (AOA3PT). Pitch changes are simulated at a constant rate (DADTCMD) the commanded alpha rate.

During the landing ground roll, the simulation coasts the aircraft to the Wheel Brake Airspeed (VKBRAKE). In addition, individual time delays can be set for brake application (TIMEBRK), flap retraction initiation (TIMEFLP), thrust reverser deployment initiation (TIMEREV), speedbrake deployment initiation (TIMESBK) and spoiler deployment initiation (TIMESPL).

REFUSED TAKEOFFS

The refused takeoff maneuver is identical to the takeoff maneuver with the following exceptions. An engine failure is simulated at the engine failure speed (VKFAIL). The engine failure speed must be less than the liftoff speed. After a time delay from engine failure (TIMEIDL), the simulation initiates the reduction of thrust on the remaining engines to idle. After a time delay from engine failure (TIMEBRK), the simulation applies brakes by changing the current coefficient of friction (XMU) from the rolling coefficient of friction (ROLLMU) to the braking coefficient of friction (BRAKMU). If the rotation speed has been reached, the simulation will either return the angle of attack to zero or change the angle of attack to the angle of attack for aerobraking (AOAABRK). The angle of attack for aerobraking will be used for calibrated airspeeds above the aerobrake limit airspeed (VKABRK). Changes in angle of attack are made at the commanded angle of attack rate (DADTCMD). The minimum control groundspeed (VKMCG), can be used as a trigger speed to adjust engine thrust if desired. The (VKMCG) variable might be used to trigger the variable XENG to be changed to XENG + 1, on a four-engine aircraft performing a three-engine takeoff. Two engines would be used from brake release to the ground minimum control speed and then changed to three engines above that speed. This option was used on the B-1B TOLAND program to aid in developing a three-engine takeoff capability and is an uncommon use of the program.

Wheel braking can also be initiated at the wheel brake airspeed (VKBRAKE). In addition, individual time delays can be set for flap retraction initiation (TIMEFLP), thrust reverser deployment initiation (TIMEREV), speedbrake deployment initiation (TIMESBK) and spoiler deployment initiation (TIMESPL).

Engine failures can be simulated with one of three methods. The simplest method is an abrupt engine failure where the thrust loss is instantaneous. This is accomplished by setting the engine failure mode (FAILMOD) to 'SEIZE' and the delta time for engine failure (DTFAIL). These are the default settings for the program. The second method is a linear loss of thrust over a time interval. This is accomplished by setting the engine failure mode (FAILMOD) to 'SPOOL' and the delta time for engine failure (DTFAIL) to desired value in seconds.

The third method is a loss of thrust using a user-provided subroutine SPOOLDNF. This is accomplished by setting the engine failure mode (FAILMOD) to 'SPOOL' and the delta time for engine failure (DTFAIL) to 0.0. Subroutine SPOOLDNF might consist of a curve or set of curves or an exponential equation or set of exponential equations. A set of curves or a set of equations might be used to distinguish between different failure states and different initial thrust levels. SPOOLDNF is used for the spool down of the failed engine whereas an additional and identical subroutine, SPOOLDNR, is used for the spool down of the remaining engines.

Different curves might be needed to simulate a failure to idle thrust and a failure to off. The failure state (FAILST) would be used to distinguish between the different curves. Different curves might be needed if an engine failure started from military thrust or maximum continuous thrust instead of maximum thrust or takeoff rated thrust. The power code (PWRCODE), engine group (ENGGRP) or engine multiplicative factor (XENG) variables could be used to accomplish these simulation variations.

Engine failures occurring in different positions can be simulated using the engine failure group (FAILGRP) variable. This would allow different engine curve lookups for an outboard engine inoperative or an inboard engine inoperative. This variable is user-definable.

CONTINUED TAKEOFFS

The continued takeoff maneuver is identical to the takeoff maneuver with the following exceptions. An engine failure occurs at the engine failure speed (VKFAIL). The engine failure speed must be less than the liftoff speed. The minimum control groundspeed (VKMCG) can be used as a trigger speed to adjust engine thrust if desired. The VKMCG variable might be used to trigger the variable XENG to be changed to $XENG + 1$ on a four-engine aircraft performing a three-engine takeoff. Two engines would be used from brake release to the ground minimum control speed and then changed to three engines above that speed. This option was used on the B-1B TOLAND program to aid in developing a three-engine takeoff capability and is an uncommon use of the program.

Engine failures can be simulated with one of three methods. The simplest method is an abrupt engine failure where the thrust loss is instantaneous. This is accomplished by setting the engine failure mode (FAILMOD) to 'SEIZE' and the delta time for engine failure (DTFAIL). These are the default settings for the program. The second method is a linear loss of thrust over a time interval. This is accomplished by setting the engine failure mode (FAILMOD) to 'SPOOL' and the delta time for engine failure (DTFAIL) to desired value in seconds.

The third method is a loss of thrust using a user-provided subroutine SPOOLDNF. This is accomplished by setting the engine failure mode (FAILMOD) to 'SPOOL' and the delta time for engine failure (DTFAIL) to 0.0. Subroutine SPOOLDNF might consist of a curve or set of curves or an exponential equation or set of exponential equations. A set of curves or a set of equations might be used to distinguish between different failure states and different initial thrust levels. SPOOLDNF is used for the spool down of the failed engine whereas an additional and identical subroutine, SPOOLDNR, is used for the spool down of the remaining engines. SPOOLDNR would not be executed for a continued takeoff.

Different curves might be needed to simulate a failure to idle thrust and a failure to off. The failure state (FAILST) would be used to distinguish between the different curves. Different curves might be needed if an engine failure started from military thrust or maximum continuous thrust instead of maximum thrust or takeoff rated thrust. The power code (PWRCODE), engine group (ENGGRP) or engine multiplicative factor (XENG) variables could be used to accomplish these simulation variations.

Engine failures occurring in different positions can be simulated using the engine failure group (FAILGRP) variable. This would allow different engine curve lookups for an outboard engine inoperative or an inboard engine inoperative. This variable is user-definable.



NAMelist DESCRIPTIONS

This section describes the namelists and variables for input to the TOLAND program. Namelist DATA is used for the primary aircraft program inputs whereas namelists TKO and LND are used for the primary inputs to takeoff and landing maneuvers respectively. Namelist ROI is used for the primary inputs to the decelerating ground roll segment of a refused takeoff or a landing maneuver. Namelists DATA2, TKO2, LND2 and ROL2 are used for secondary or auxiliary inputs. These namelists are used to separate the primary inputs from the secondary inputs for ease of use and clarity. These secondary inputs would contain variables which are used infrequently or not at all by most users. Namelist TKOARY is used for inputs to flap and thrust vectoring schedules during a takeoff maneuver. Table 3 indicates the required namelist inputs for each maneuver.

Table 3
NAMELISTS

Namelist	Takeoff	Landing	Refused Takeoff	Continued Takeoff
DATA	X	X	X	X
DATA2	X	X	X	X
TKO	X		X	X
TKO2	X		X	X
TKOARY	X		X	X
LND		X		
LND2		X		
ROL		X	X	
ROL2		X	X	

Namelist DATA

CGPCT is the longitudinal position of the center of gravity as a percentage distance of the mean aerodynamic chord aft from the leading edge of the wing. The default for CGPCT is user defined in subroutine FORCEX. This input is required only if the aerodynamics are provided as a function of longitudinal position of the center of gravity.

DADTCMD is the commanded angle of attack rate. It is the rate at which the aircraft rotates during the takeoff maneuver. DADTCMD affects the rate of convergence to target pitch attitudes and target airspeeds. The default for DADTCMD is 2.5 degrees per second.

DTEMPF is the delta temperature from standard day in degrees Fahrenheit. The default for DTEMPF is 0.0 degree Fahrenheit.

FLAP is the current flap deflection in degrees during program execution. The default for FLAP is -1.0 degree. This default value allows FLAP to be set from FLPPCT. FLAP is a required input if FLPPCT is not used.

GAMMARW is the slope of the runway in degrees. The default for GAMMARW is 0.0 degree.

GWT0 is the initial gross weight in pounds. GWT0 is a required variable; there is no default.

HCLEAR is the obstacle clearance height in feet. The default for HCLEAR is 50.0 feet.

HRUNWAY is the pressure altitude of the runway in feet. The default for **HRUNWAY** is 0.0 feet.

VKWIND is the headwind component of wind speed in knots. Tailwinds are input using negative headwind components. The default is 0.0 knot (no wind).

Namelist DATA2

AOA3PT is a angle of attack for a 3-point attitude in degrees. The default for **AOA3PT** is 0.0. This variable is not required.

CONFIG is a number representing the configuration of the aircraft. This floating point variable is a user definable input. This variable is not required.

DCDX is the user drag coefficient increment. The user may alter the drag coefficient (**CD**) by adding or subtracting a delta drag coefficient increment. The default for **DCDX** is 0.0000.

DCLX is the user lift coefficient increment. The user may alter the lift coefficient (**CL**) by adding or subtracting a delta lift coefficient increment. The default for **DCLX** is 0.0000.

DTFAIL is the delta time for the failed engine to lose thrust by the thrust level defined by **XENGFLD**. Thrust and fuel flow decrease linearly over the time interval **DTFAIL**. The default for **DTFAIL** is 0.0 second.

DTIME is the Runge-Kutta numerical integration step size in seconds. The default for **DTIME** is 0.10 second. This variable controls the accuracy of the integrated variables.

ENGGRP is the operating engine group. It is a user definable three-character string variable whose contents define which engines are operating during a maneuver. Examples of **ENGGRP** are 'AEO', 'OEI', 'IEI', and 'AEI' which correspond to all engines operating, outboard engine inoperative, inboard engine inoperative and all engines idling respectively. The default for **ENGGRP** is 'AEO' for all takeoffs and 'AEI' for all landings.

EPR is engine pressure ratio. This parameter can optionally be used to input engine thrust levels to the program or can be used as a variable which contains the current value of Engine Pressure Ratio. The default for **EPR** is 0.0.

FAILGRP is the fail engine group. It is a user definable three-character string variable whose contents define the state of the engines after an engine failure or failures. The default for **FAILGRP** is 'OEI', an outboard engine inoperative.

FAILMOD is the engine failure mode. It is a five-character string variable whose contents define the mode of the failed engine. Usually, only two states are used, 'SEIZE' and 'SPOOL'. The default for **FAILMOD** is 'SEIZE'.

FAILST is the engine failure state. It is a four-character string variable whose contents define the state of the failed engine. Usually, only two states are used, 'IDLE' and 'OFF', however, for aircraft with afterburners the state 'MIL' is available. These states are quantified by the variables **XIDLE**, 0.0, and **XMIL**, respectively. The values of **XIDLE** and **XMIL** are initialized in subroutine **FORCEX**. The default for **FAILST** is 'IDLE'.

FGPCT is the gross thrust percentage increment. The user may increase the net thrust by a given percentage using this input variable. The default for FGPCT is 0.0 percent. For example, with FGPCT set to -3.5, the variable THRUST is multiplied by 0.965.

FLPPCT is the current flap percentage setting during program execution. This variable provides the user an additional method of entering flap deflection into the program. The default is user defined in subroutine FORCEX. For example, FLPPCT could be set to 50.0. The user would provide a method of initializing the FLAP variable to the corresponding value for 50 percent flaps. This variable is not required.

FLT is the flight number. It is a floating point variable available to the user for management and tracking of data. The default for FLT is 0.0. This variable is not required.

FLTNDX is the flight index. This floating point variable is a user definable input. It is not passed through any program supplied common blocks. This variable is not required.

IDBUG is the UFTAS debug code. It can provide debug information concerning the UFTAS random access curve file subroutines. The default for IDBUG is 0.

JDEBUG is the TOLAND debug code. This code is user definable; it can be used to output debug data from the user provided subroutines. Two values for JDEBUG are already used within the program. JDEBUG = 9999 for full debug output and JDEBUG = 6666 for special landing and ground roll debug output for refused takeoffs. The default for JDEBUG is 0.

LOADING is a number representing the external store loading of the aircraft. This integer variable is a user definable input. This variable is not required.

LUMSG is the logical unit for message output. It is an integer variable which determines which file certain program messages are written to. The messages that can be re-routed come from the debug output as well as UFTAS random access curve file subroutines. The default for LUMSG is LUOUT.

PWRCODE is the aircraft power code. It can describe the thrust settings to which all the aircraft engines are set. The default for PWRCODE is user defined in subroutine FORCEX.

RC is the engine thrust rating code. It can describe the thrust setting to which an engine deck or a curve file based on an engine deck is set during a simulation. The default for RC is user defined in subroutine FORCEX.

REVFLAG is the reverse thrust flag. This flag is set to .TRUE. when reverse thrust is to be enabled for a simulation. The default for REVFLAG is .FALSE.

ROLLMU is the rolling coefficient of friction. The default for ROLLMU is 0.025 and is based on MIL-C-005011B. (Reference 3)

SPDBRK0 is the initial speed brake deflection in degrees. The default for SPDBRK0 is 0.0.

THRCRV is the thrust curve type. It is a three-character input variable that is used to describe the type of Thrust Curve to be utilized. This input variable allows the program to distinguish between Thrust Curves containing different independent (Input) variables. Examples of THRCRV are 'RC', (rating code), 'PLA', (power lever angle) and 'EPR', (engine pressure ratio). The default for THRCRV is user defined in subroutine FORCEX.

VKMCG is the minimum control airspeed on the ground. It is an input variable used to trigger a spoolup of an operating but asymmetric engine. For example, it would be used in the case of a three-engine takeoff of a four-engine aircraft whose nose wheel steering was not sufficient to maintain directional control with full

or partial asymmetric thrust. VKMCG is not used to fail an engine; use VKFAIL. The default for VKMCG is 0.0 knot calibrated airspeed.

XENG is the engine multiplicative factor. Total engine parameters (e.g. net thrust, fuel flow) are defined by their corresponding per engine values multiplied by XENG. This variable is not an integer variable and is not the number of engines. It is a floating point variable that is used with per engine values of thrust and fuel flow. This allows the program to represent values of partial engine thrust to transition from one thrust setting to another. The default for XENG is user defined in subroutine FORCEX.

XENGFLD is the failed engines multiplicative factor. This variable is not an integer variable and is not the number of failed engines. It is a floating point variable that is used with per engine values of thrust and fuel flow. This allows the program to represent engine values of a partial failed engine (e.g. afterburner). If XENGFLD is not equal to 1.0, FAILST must be set to 'NULL'. FAILST overrides XENGFLD. The default for XENGFLD is 1.0.

Namelist TKO

HGEAR is landing gear retraction height in feet above the takeoff point. The default for HGEAR is 205 feet.

HMAX is the termination height limit in feet above the takeoff point for the takeoff simulation. The simulation will end if this limit is reached. The default for HMAX is 1000 feet.

THTC LM is the pitch attitude the simulation tracks after liftoff but before reaching HCLIMB feet altitude above the takeoff point. The default for THTC LM is 10 degrees.

THTR OT is the pitch attitude the simulation tracks after rotation but before liftoff. In general, the aircraft will liftoff before reaching THTR OT and will continue to rotate to THTC LM. The default for THTR OT is 10 degrees.

VKFAIL is the engine failure airspeed in knots. The default for VKFAIL is 0.0 knot calibrated airspeed which is defined as no engine failure.

VKROTAT is the rotation airspeed in knots. The default for VKROTAT is 0.0 knot calibrated airspeed. The simulation will rotate the aircraft without regard to actual aircraft horizontal tail authority; TOLAND does not define moments about the aircraft center of gravity.

Namelist TKO2

DISTMAX is ground distance limit for the takeoff simulation. The simulation will end if this limit is reached. The default for DISTMAX is 60,760 feet, or ten nautical miles.

HCLIMB is the height in feet above the takeoff point at which the takeoff simulation switches from tracking a constant pitch attitude (THTC LM) to tracking a constant calibrated airspeed (VCLMOUT). The default for HCLIMB is 100 feet.

HCLMOUT is the height in feet above the takeoff point at which the takeoff simulation switches from tracking a constant calibrated airspeed (VCLMOUT) to tracking a constant pitch attitude (THTF LY). The default for HCLMOUT is 200 feet.

ROLLMAX is the takeoff ground roll time limit. The simulation will end if this limit is reached. The default for ROLLMAX is 120 seconds.

THTF LY is the pitch attitude the simulation tracks after reaching HCLMOUT feet altitude above the takeoff point. The default for THTF LY is 8 degrees.

THTTOL is the pitch attitude tolerance the simulation tracks to. The simulation tracks to the target pitch angle within plus or minus THTTOL degrees; after which no pitch adjustment is made. The default for THTTOL is 0.10 degree.

TIMEMAX is the time limit for the takeoff simulation. The simulation will end if this limit is reached. The default for TIMEMAX is 300 seconds.

TKOTYPE is the takeoff type. It is a seven character input variable that is used to describe the type of takeoff as either 'STATIC' or 'ROLLING'. This input variable allows the program to distinguish between the thrust settings of either a static or rolling takeoff independently of the VKSTART input variable. The default for TKOTYPE is 'STATIC'.

VCLMOUT is the calibrated airspeed (in knots) the simulation tracks after reaching HCLIMB feet altitude above the takeoff point but before reaching HCLMOUT feet altitude above the takeoff point. The default for VCLMOUT is the calibrated airspeed reached at HCLIMB feet altitude above the takeoff point.

VKEND is the simulation end airspeed in knots. The default for VKEND is 250 knots calibrated airspeed.

VKFLAP is the calibrated airspeed at which the flaps are retracted. The default for VKFLAP is the flap limit airspeed, VKFLPMX. The flap limit airspeed, VKFLPMX, is a hard coded value user defined in subroutine 'FXXAERO'.

VKSTART is the simulation start groundspeed in knots. This input variable allows the user to simulate takeoffs where there is a given groundspeed at the time the aircraft is aligned on the centerline of the runway. For example, rolling minimum interval takeoffs (MITOs) frequently roll onto the centerline of the runway without stopping. The default for VKSTART is 0.0 knot.

Namelist TKOARY

This namelist contains parameters that are not required.

FLPARY is the flap deflection array. This array contains five elements. This array allows the user to enter in a flap deflection schedule as a function of calibrated airspeed. At each airspeed contained in the flap deflection airspeed array (VFLPARY), the takeoff simulation changes the flap deflection, FLAP, to the corresponding flap deflection contained in the flap deflection array (FLPARY) at a rate of DFLAPDT degrees per second. The flap deflection rate (DFLAPDT) is provided by subroutine 'FXXAERO' which is the user provided subroutine called by subroutine FORCEX. If the last element of FLPARY is not 0.0, the simulation will retract the flaps to 0.0 degree at the calibrated airspeed contained in the last element of VFLPARY or at VKFLPMX which ever is less. The default for the elements of FLPARY are 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation does not change the flap deflection until the flap retraction speed (VKFLAP) or the maximum flap deflection airspeed (VKFLPMX) is reached.

VFLPARY is the flap deflection airspeed array. This array contains five elements. This array allows the user to enter in a flap deflection schedule as a function of calibrated airspeed. At each airspeed contained in the flap deflection airspeed array (VFLPARY) the takeoff simulation changes the flap deflection (FLAP) to the corresponding flap deflection contained in the flap deflection array (FLPARY) at a rate of DFLAPDT degrees per second. The flap deflection rate (DFLAPDT) is provided by subroutine 'FXXAERO' which is the user provided subroutine called by subroutine FORCEX. If the last element of FLPARY is not 0.0, the simulation will retract the flaps to 0.0 degree at the calibrated airspeed contained in the last element of VFLPARY or at VKFLPMX which ever is less. The default for the elements of VFLPARY are 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation does not change the flap deflection until the flap retraction speed (VKFLAP) or the maximum flap deflection airspeed (VKFLPMX) is reached.

HVCTARY is the vectored thrust altitude array. This array contains five elements. At each altitude contained in the vectored thrust altitude array, HVCTARY, the takeoff simulation changes the vectored thrust angle (VTANGLE), to the corresponding vectored thrust angle contained in the vectored thrust angle array (XNUARY) at a rate of DVECTDT degrees per second. The vectored thrust angle rate (DVECTDT) is provided by subroutine 'FXXENG' which is the user provided subroutine called by subroutine FORCEX. If both the HVCTARY and VVCTARY arrays are used, the HVCTARY array takes precedence. The default for the elements of HVCTARY are 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation does not change the vectored thrust angle.

VVCTARY is the vectored thrust airspeed array. This array contains five elements. At each calibrated airspeed contained in the vectored thrust airspeed array (VVCTARY) the takeoff simulation changes the vectored thrust angle (VTANGLE) to the corresponding vectored thrust angle contained in the vectored thrust angle array (XNUARY) at a rate of DVECTDT degrees per second. The vectored thrust angle rate (DVECTDT) is provided by subroutine 'FXXENG' which is the user provided subroutine called by subroutine FORCEX. If both the HVCTARY and VVCTARY arrays are used, the HVCTARY array takes precedence. The default for the elements of VVCTARY are 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation does not change the vectored thrust angle.

XNUARY is the vectored thrust angle array. This array contains five elements. At each altitude contained in the vectored thrust altitude array (HVCTARY) or at each calibrated airspeed contained in the vectored thrust airspeed array (VVCTARY) the takeoff simulation changes the vectored thrust angle (VTANGLE) to the corresponding vectored thrust angle contained in the vectored thrust angle array (XNUARY) at a rate of DVECTDT degrees per second. The vectored thrust angle rate (DVECTDT) is provided by subroutine 'FXXENG' which is the user provided subroutine called by subroutine FORCEX. If both the HVCTARY and VVCTARY arrays are used, the HVCTARY array takes precedence. The default for the elements of VVCTARY are 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation does not change the vectored thrust angle.

Namelist LND

ALPHA is the angle of attack in degrees during approach. The program will determine the calibrated airspeed (VKAPP) for the approach if ALPHA is input. ALPHA or VKAPP are required variables; there are no defaults.

GAMMAPP is the flightpath angle in degrees (negative for a descent) during approach. The default for GAMMAPP is -3.0 degrees (descending).

HFLARE is the height in feet above the touchdown point of the runway at which the simulation initiates a flare. The default for HFLARE is 0.0 feet (no flare).

SINKTD is the sink rate at touchdown. The default for SINKTD is 10.0 feet per second.

VKAPP is the calibrated airspeed in knots during the approach. The program will determine the angle of attack (ALPHA) for the approach if VKAPP is input. VKAPP or ALPHA are required variables; there are no defaults.

Namelist LND2

DTDTMX is the maximum pitch rate during the flare simulation in degrees per second. The default for DTDTMX is user defined in subroutine FORCEX.

SPLFLAG is the spoiler flag. The flag is set to .TRUE. to initialize the simulation with the spoilers deployed. The default for SPLFLAG is .FALSE. (spoilers retracted).

WRITTR is the write flare iterations flag. This flag is set to .TRUE. when the user desires output on each flare iteration executed from subroutine FLARENZ. The default for WRITTR is .FALSE.

Namelist ROL

BRAKMU is the braking coefficient of friction. This input parameter would be used if the braking coefficient selector (**IMU**) is set to 0 (corresponding to a constant **BRAKMU**). The default for **BRAKMU** is 0.250.

IMU is the braking coefficient selector. The default selections for **IMU** are constant braking coefficient (**BRAKMU**), 0; user supplied curve file lookup, 1; generic dry runway braking coefficient, 2; and generic wet runway braking coefficient, 3. The default for **IMU** is 0, which corresponds to a constant braking coefficient of friction. For **IMU**=1, **BRAKMU** might be looked up as a function of weight on wheels, groundspeed and runway condition reading (**RCR**). The default for **BRAKMU** is 0.250.

RCR is the runway condition reading. It can be used as an input to determine the variable **BRAKMU**. The default for **RCR** is user defined in subroutine **GENMU**.

Namelist ROL2

AOAABRK is the angle of attack for aerobraking during the landing ground roll if the calibrated airspeed is greater than **VKABRK** knots calibrated airspeed. The default for **AOAABRK** is 0.0 degree (no aerobraking).

BRKFCTR is the braking factor. It is a multiplicative factor applied to **BRAKMU**, the braking coefficient of friction. It allows the user to adjust the coefficient of friction during braking without recompiling code. This input parameter would normally be used if the braking coefficient selector (**IMU**) was not set to 0. The default for **BRKFCTR** is 1.0.

TIMEBRK is the braking time delay in seconds between the time touchdown has occurred and the time the brakes are applied for a landing maneuver. It is also the braking time delay between the time an engine has failed and the time the brakes are applied for a refused takeoff maneuver. The default for **TIMEBRK** is 3.0 seconds.

TIMEFLP is the flap retraction time delay in seconds between the time touchdown has occurred and the time the flap retraction initiation has occurred retracting for a landing maneuver. It is also the flap retraction time delay between the time an engine has failed and the time the flap retraction initiation has occurred for a refused takeoff. The default for **TIMEFLP** is 999.0 seconds; essentially the flaps are not retracted.

TIMEIDL is the idle thrust spool down time delay in seconds between the time an engine has failed and the time a spool down to idle has been initiated on the remaining engine(s) for a refused takeoff. This parameter has no effect during a landing because the engines are in idle during the flare. The default for **TIMEIDL** is 3.0 seconds.

TIMEREV is the reverse thrust time delay in seconds between the time touchdown has occurred and the time the thrust reverser deployment initiation has occurred for a landing maneuver. It is also the reverse thrust time delay between the time the remaining engine(s) has spooled down to idle and the time the thrust reverser deployment initiation has occurred for a refused takeoff maneuver. The default for **TIMEREV** is 0.0 second.

TIMESBK is the speedbrake deployment time delay in seconds between the time touchdown has occurred and the time the speedbrake deployment initiation has occurred for a landing maneuver. It is also the speedbrake time delay between the an engine has failed and the time the speedbrake deployment initiation has occurred for a refused takeoff maneuver. The default for **TIMESBK** is 0.0 second.

TIMESPL is the spoiler deployment time delay in seconds between the time touchdown has occurred and the time the spoiler deployment initiation has occurred for a landing maneuver. It is also the spoiler time delay between the an engine has failed and the time the spoiler deployment initiation has occurred for a refused takeoff maneuver. The default for **TIMESPL** is 0.0 second.

VKABRK is the calibrated airspeed during the landing ground roll that aerobraking is stopped and the angle of attack (which is the same as pitch angle during the ground roll) is lowered from **AOAABRK** to 0.0 degree. The default for **VKABRK** is 0.0 knot calibrated airspeed (no aerobraking).

VKBRAKE is the wheel braking airspeed in knots. Wheel braking is initiated after the calibrated airspeed is less than **VKABRK**. The default for **VKBRAKE** is 999.0 knots calibrated airspeed.

OUTPUT DESCRIPTIONS

This section describes the output provided by the TOLAND program.

Takeoff and Continued Takeoff Output

For each set of namelist inputs provided, the program echoes many of the input variables back to the output file. Then the core of the program output is written. This core consists of the following variables listed in a table from left to right: elapsed time (TIME), ground distance (GDIST), gross weight (GWT), altitude above ground level (HAGL), knots calibrated airspeed (VKCAS), knots true airspeed (VKTAS), knots true groundspeed (VKTGS), acceleration along flightpath or ground track (FPACCEL or ACCEL), lift coefficient (CL), drag coefficient (CD), pitch attitude (THETAF), angle of attack (ALPHA), flightpath angle (GAMMA), time rate of change of angle of attack (DADT), rate of climb (ROC), normal load factor (XLF), net thrust (THRUST), and the engine multiplicative factor (XENG). The core data is repeated once every 10 DTIME seconds. However at rotation, liftoff, and the obstacle clearance height the data is also listed.

After the core output is written, summary information about the takeoff is output. This summary information consists of flight number (FLT), initial gross weight (GWT0), user drag coefficient increment, (DCDX), net thrust percentage increment (FGPCT), ground minimum control airspeed (VKMCG), original flap setting (FLAP0), delta temperature from standard day (DTEMPF), user lift coefficient increment (DCLX), the failure state (FAILST), and the operating engine group (ENGGRP).

Refused Takeoff Output

The output for refused takeoffs is identical to that for takeoffs and continued takeoffs with the following exception: the coefficient of friction (XMU), is substituted for the engine multiplicative factor (XENG), during the decelerating portion of the ground roll.

Landing Output

The core output for landings is identical to that for takeoffs and continued takeoffs with the following exceptions: the time rate of change in pitch attitude (DTDIT), is substituted for the time rate of change in angle of attack (DADT), during the approach and flare portion of the landing; and the coefficient of friction (XMU), is substituted for the engine multiplicative factor (XENG), during the decelerating portion of the landing ground roll.

After the core output is written, summary information about the landing is output. This summary

information consists of the distance over an HCLEAR foot obstacle, air distance, ground roll distance, pre-flare distance, flare distance, and average deceleration. Figure 1 defines these distances for a landing with the obstacle clearance height (HCLEAR), greater than the flare height (HFLARE); a low flare landing.

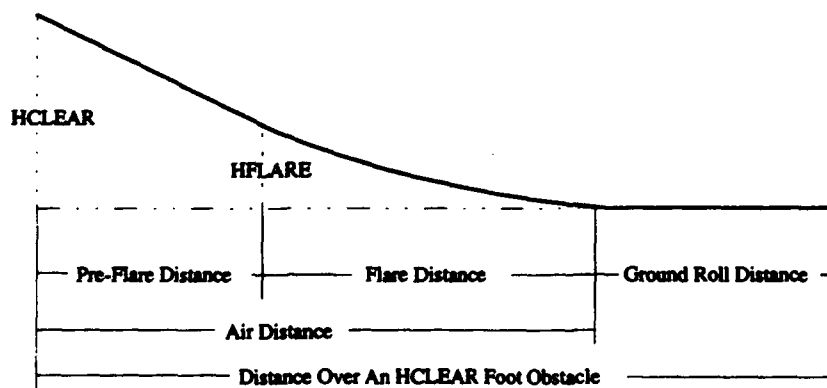


Figure 1 Low Flare Landing Distance Definitions

Figure 2 defines these distances for a landing with the obstacle clearance height (HCLEAR), less than the flare height (HFLARE); a high flare landing. In addition, flight number (FLT), user drag coefficient increment (DCDX), delta temperature from standard day (DTEMPF), braking multiplicative factor (BRKFCTR), user lift coefficient increment (DCLX), and original flap setting (FLAP0) are also output.

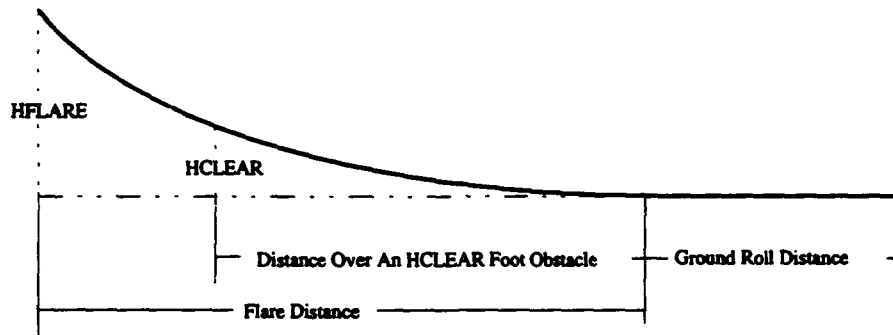


Figure 2 High Flare Landing Distance Definitions

PROGRAM STRUCTURE

This section describes the structural layout of the TOLAND program.

All execution begins and ends with the main program. Namelist inputs are read and parameter initialization are performed first. Curve file initialization is then performed (if applicable). Program execution then branches to one of two control subroutines, TAKOFF or LANDNG, as shown in Figure 3. For takeoffs, refused takeoffs or continued takeoffs, execution branches to subroutine TAKOFF. For landings, execution branches to subroutine LANDNG.

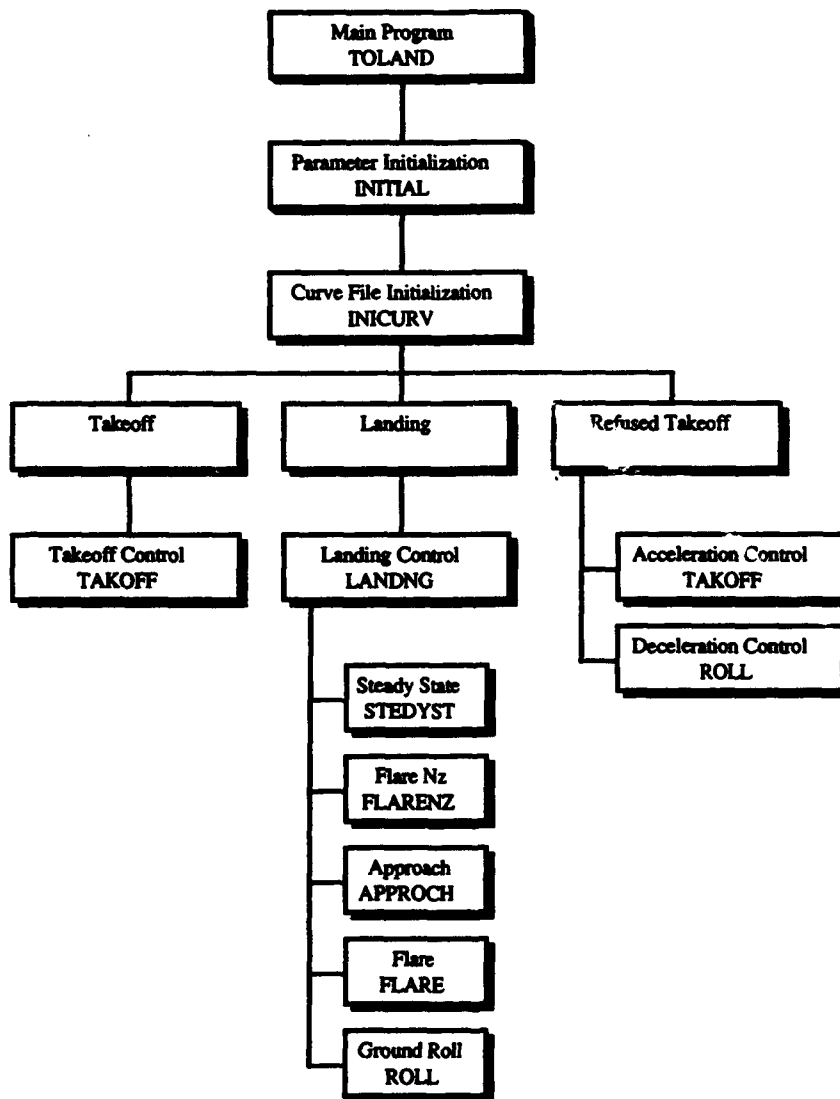


Figure 3 Takeoff and Landing Program Structural Layout

Control is passed to subroutine TAKOFF for takeoffs and to subroutine LANDNG for landings. Control is first passed to subroutine TAKOFF for refused takeoffs, until engine failure when control is passed to

subroutine ROLL. After the maneuver is complete, program execution returns to the main program. The main program then either reads another set of namelist inputs or terminates execution.

Within subroutine LANDNG, program execution passes to subroutine STEDYST to determine the steady-state conditions for the landing approach. After that is accomplished, program execution is passed to subroutine FLARENZ, where the constant normal load factor for the flare (if any) is determined through multiple calls to subroutine FLARE. Program execution then passes to subroutine APPROCH (if the obstacle clearance height (HCLEAR) is greater than the flare initiation height (HFLARE)) or subroutine FLARE (if the obstacle clearance height (HCLEAR) is less than or equal to the flare initiation height (HFLARE)). After touchdown, program execution passes to subroutine ROLL until the end of the ground roll. Program execution then passes back to subroutine LANDNG and then to the main program. The main program either reads another set of namelist inputs or terminates execution.

Runge-Kutta numerical integration is performed from only subroutines TAKOFF, FLARE, and ROLL. The integrator subroutine INTX determines the acceleration from one of equations of motion subroutines, DERIVGR, DERIVAT, or DERIVAL. The force coefficients for the equations of motion are provided by subroutine FORCEX. FORCEX calculates the force coefficients from the lift and drag coefficients provided by subroutine 'FXXAERO' and from thrust provided by subroutine 'FXXENG'. The associations of these subroutines are shown in Figure 4.

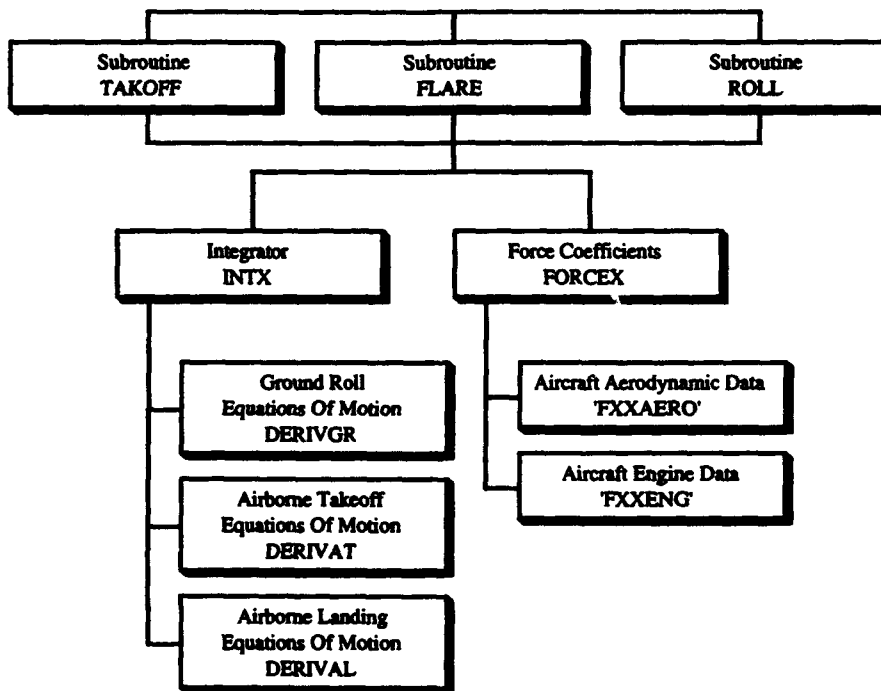


Figure 4 Takeoff and Landing Program Major Subroutine Associations



SUBROUTINE DESCRIPTIONS

This section describes the subroutines and functions of the TOLAND program.

Table 4
SUBROUTINE USAGE

Subroutine	Takeoff	Landing	Refused Takeoff	Continued Takeoff
INITIAL	P	P	P	P
FRETRAC ¹	P,C	P,C	P,C	P,C
GRETRAC ²	P,C			P,C
PITCH	P	P	P	P
SPOIL ³		P	P	
TVECTOR	O	O	O	O
TAKOFF	P		P	P
LANDNG		P		
STEDYST		P		
FLARENZ ⁴		P		
APPROCH ⁵		P,C		
FLARE		P		
ROLL		P	P	
INTX	P	P	P	P
DERIVGR	P	P	P	P
DERIVAT	P			P
DERIVAL		P		
ERROR ⁶	P,C			P,C
HALT	P	P	P	P
ATMOSPH	P	P	P	P
ITRLND		P		
DGDT		P		
DVDT		P		
DVTDH		P		
DADH		P		
DDELTDH		P		
INTERP	P	P	P	P
ZEROX		P		
INICURV	U	U	U	U
FORCEX	U	U	U	U
'FXXAERO'	U	U	U	U
GEFFECT	U	U	U	U
'FXENG'	U	U	U	U
SPOOLUP ⁷	U,C	U,C	U,C	U,C
SPOOLDNF ⁷		U,C	U,C	U,C
SPOOLDNR ⁷		U,C	U,C	U,C
GENMU		U,C	U,C	

- C Conditional execution of subroutine
- O Optional sub-routine--not required
- P Program provided subroutine
- U User provided subroutine

NOTES:

1. FRETRAC is called only when the flap setting is changed.
2. GRETRAC is called only when gear is retracted.
3. SPOIL is called only when the spoiler setting is changed.
4. FLARENZ is called when HFLARE is greater than 0.
5. APPROCH is called when HCLEAR is greater than HFLARE.
6. ERROR is called when the equations of motion subroutine is unable to find a solution.
7. SPOOLUP, SPOOLDNF, or SPOOLDNR is called when a throttle transient is performed.

Table 4 shows which subroutines are executed for which maneuver. Conditional subroutines are executed only if the certain conditions are met. See notes (1-7). The optional subroutine TVECTOR is not required if thrust vectoring is not used. Program provided subroutines are intended to be used without modifications. User provided subroutines have program supplied templates for easy modification.

User Provided Subroutines

Some of the following subroutines must be modified for a particular aircraft. For aircraft simulations using random access curve files, subroutine INICURV must be modified to contain the architecture of the user's curves and initializes the appropriate arrays in common blocks VALUES and TABLES. Subroutine FORCEX requires initializing certain aircraft unique constants within a DATA statement and using the proper calling arguments in calls to subroutines 'FXXAERO' and 'FXXENG'

Subroutine 'FXXAERO' must return lift coefficient (C_L), and drag coefficient (C_D) to subroutine FORCEX. At least the lift coefficient must be a function of angle of attack (α); this program is angle of attack driven. Subroutine GEFFECT can be used for ground effect increments to the lift and drag coefficients. A sample subroutine is provided that includes Dr. E K Parks increment equations.

Subroutine 'FXXENG' must return gross thrust (F_G), propulsive drag (F_E), and fuel flow (W_f/dt) to subroutine FORCEX. Samples of subroutines SPOOLDNF, SPOOLDNR, and SPOOLUP are provided as guides to the user for simulating throttle transients.

Subroutine GENMU must return the braking coefficient (BRAKMU). A sample is provided as a guide to the user showing a variety of methods which produce braking coefficient.

Subroutine INICURV

This subroutine initializes the names, tables and value names of the random access curve files. This subroutine opens the appropriate curves as needed. This subroutine has no calling arguments and contains common blocks: FLAGS, CHARV, CINDEX, RACURV, VALUES, and TABLES.

Subroutine FORCEX

This subroutine calculates the total force coefficients along and normal to the flightpath. Equations of motion defining the force coefficients are shown in Appendix A. The force coefficients are calculated with the lift and drag coefficients supplied by subroutine 'FXXAERO' and with thrust supplied by subroutines 'FXXENG', SPOOLDNF, SPOOLDNR and SPOOLUP.

Subroutine FORCEX should require only the following modifications:

- 1) Initialization of the DATA statement providing values for AIT, AOA3PT, AR, B, CGPCT, CLALPH, CONFIG, DTDTMX, FLAP, FLPPCT, HZ, LOADING, NENG, PWRCODE, RC, SWING, THRCRV, THTMAX, XIDLE, and XLFMAX. DTDTMX and XLFMAX is only required for the flare portion of landings. AR, B, CLALPH, and HZ are variables used only in determining ground effect increments to the aerodynamic models. CONFIG and LOADING are optional user-definable variables. PWRCODE, RC, and THRCRV are optional user-definable variables for distinguishing power settings and switching between different engine curves.
- 2) Changing the calling statement for subroutines 'FXXAERO' and 'FXXENG' to the user provided names with the appropriate calling arguments.

This subroutine has the calling arguments: ALPHA, CD, and CL; and contains common blocks: CTRL, AIRCRFT, AERO, ENGINE, AIRBORN, AIRSPED, FLAGS, and CHARV.

Subroutine 'FXXAERO'

This subroutine calculates or performs a table lookup to determine the lift and drag coefficients for the aircraft. This subroutine and its name are provided by the user. For example, this subroutine might be called F22AERO for the F-22. The calling arguments are user defined but might be ALPHA, AMACH, CL, CD, FLAP, SPOILER, VKCAS and XENGOUT. The subroutine might contain common blocks: CTRL,

AIRCRAFT, AIRSPED, RUNWAY, FPINTEG, ATMOS, FLAGS, CHARV, RACURV, VALUES and TABLES.

Subroutine GEFFECT

This subroutine supplies the ground effect increments to the aerodynamic coefficients. The generic subroutine provides predicted ground effect equations which can be modified by the user for a particular aircraft. The calling arguments are user defined but might be CL, HAGL, DCLGE and DCDGE. The subroutine might contain common blocks: AIRCRAFT, AERO, CONST, and CHARV.

Subroutine 'FXXENG'

This subroutine calculates or performs a table lookup to determine the net thrust and fuel flow for the aircraft. This subroutine and its name are provided by the user. For example, this subroutine might be called F22ENG for the F-22. The calling arguments are user defined but might be ALPHA, AMACH, PWRCODE, QS, VKCAS, XENG, FE, FG, THRUST, WFUEL, and EPR. The subroutine might contain the common blocks: CTRL, VECTOR, RUNWAY, FPINTEG, ATMOS, CONST, FLAGS, CHARV, RACURV, VALUES and TABLES.

Subroutine SPOOLUP

This subroutine determines the ratio of thrust to takeoff rated thrust during a rolling takeoff using subroutine TABINT and data array SPOOLA. This is accomplished by passing the engine multiplicative factor, XENG, back to the calling subroutine. This subroutine is just an example and does not apply to all aircraft. This subroutine has the calling arguments: ENGNDX, TIME, XENGTRN, XENG0, SPOOL and XENG; and contains no common blocks.

Subroutine SPOOLDNF

This subroutine determines the engine multiplicative factor for a failed engine during a throttle chop to idle or a fuel cut using subroutine TABINT and data array SPOOLA. This subroutine is just an example and does not apply to all aircraft. This subroutine has the calling arguments: TIME, XENGEND, XENGTRN, SPOOL, XENG, and LUMSG and contains the common block: CHARV.

Subroutine SPOOLDNR

This subroutine determines the engine multiplicative factor for the remaining engines during a throttle chop to idle or a fuel cut using subroutine TABINT and data array SPOOLA. This subroutine is just an example and does not apply to all aircraft. This subroutine has the calling arguments: TIME, XENGEND, XENGTRN, SPOOL, XENG, and LUMSG and contains the common block: CHARV.

Subroutine GENMU

This subroutine provides the braking coefficient of friction. This subroutine has the calling arguments: VKTGS, WTML, XMAIN, XNOSE and YCG; and contains the common blocks: CTRL, RUNWAY, CONST, RACURV, VALUES, and TABLES. The user must provide values for XMAIN, XNOSE, and YCG. These are all positive values which correspond to distances from the aircraft's center of gravity. XMAIN is the horizontal distance from the main gear to the center of gravity. XNOSE is the horizontal distance from the nose gear to the center of gravity. YCG is the vertical distance from the ground to the center of gravity.

Program Subroutines

Subroutine INITIAL

This subroutine initializes groups of variables. This subroutine has the calling arguments: GROUP, ALPHA, FLAPO, GWTO, and SPDBRK0; and contains common blocks: CTRL, AIRCRAFT, AERO, ENGINE, VECTOR, AIRBORN, AIRSPED, RUNWAY, INTEG, FPINTEG, ATMOS, CONST, FLAGS, CHARV, FLAPDAT, GEARDAT, VECTDAT, and RACURV.

Subroutine FRETRAC

This subroutine controls the flap setting during flap retraction. This subroutine has the calling arguments: FLAP, VKFLAP, and DFLAPDT; and contains common blocks: CTRL, FLAGS, and FLAPDAT.

Subroutine GRETRAC

This subroutine returns the value of the incremental drag of the landing gear, DCDLGR. The flag LGRFLAG is available to add the incremental drag in subroutine 'FXXAERO'. This subroutine has the calling arguments: DCDLGR, DTGEAR and HAGL; and contains common blocks: CTRL, FLAGS, and GEARDAT.

Subroutine PITCH

This subroutine modulates angle of attack to match an input pitch attitude or climb speed if limit conditions are met. The rate at which angle of attack is increased is based on normal load factor (XLF) and the commanded alpha rate (DADTCMD), a user input. This subroutine has the calling arguments: MANUVR, ALPHA, DADTCMD, DTIME, DTDGEX, and LUOUT; and contains common blocks: AIRCRFT, AIRBORN, FPINTEG, CONST and FLAGS.

Subroutine SPDBRAK

This subroutine controls speed brake deployment and retraction. This subroutine has the calling arguments: ACTION, SPDBRK, SBKEND, and DSBKDT and contains common blocks: CTRL and FLAGS. The possible values for action are 'RESET', 'DEPLOY', and 'RETRACT'.

Subroutine SPOIL

This subroutine controls spoiler deployment and retraction. This subroutine has the calling arguments: ACTION, SPOILER, SPLREND, and DSPLRDT and contains common blocks: CTRL and FLAGS. The possible values for action are 'RESET', 'DEPLOY', and 'RETRACT'.

Subroutine TVECTOR

This subroutine controls the vectored thrust angle (VTANGLE). This subroutine has the calling arguments: VTANGLE, HVECT, VVECT, and DVECTDT; and contains common blocks: CTRL, FLAGS, and VECTDAT.

Subroutine TAKOFF

This subroutine controls the execution of the takeoff maneuver from brake release through climb out. Subroutine TAKOFF calls INTX to perform numerical integration of the resultants of the equations of motion and calls FORCEX to obtain the force coefficients for the equations of motion. This subroutine has the calling argument: ALPHA; and contains common blocks: CTRL, AIRCRFT, AERO, ENGINE, AIRBORN, AIRSPED, RUNWAY, INTEG, FPINTEG, ATMOS, CONST, FLAGS, CHARV, FLAPDAT, GEARDAT, and VECTDAT.

Subroutine LANDNG

This subroutine controls the execution of the landing maneuver. LANDNG calls STEDYST to obtain required values of thrust and angle of attack for a steady-state approach. LANDNG calls subroutine FLARENZ to determine the normal load factor required to execute a flare at the desired sink rate at touchdown. LANDNG call subroutine APPROCH to execute an approach down to the flare height (HFLARE) if the obstacle clearance height (HCLEAR) is greater than HFLARE. LANDING calls subroutine FLARE to execute a flare and calls subroutine ROLL for the ground portion of the landing. This subroutine has the calling argument: ALPHA; and contains common blocks: CTRL, AIRCRFT, AERO, ENGINE, AIRBORN, AIRSPED, RUNWAY, INTEG, FPINTEG, ATMOS, CONST, and FLAGS.

Subroutine STEDYST

This subroutine calculates the required values of net thrust and angle of attack for zero acceleration along and normal to the flightpath, dV_c/dt and $V_c(dy/dt)$ respectively when the flag, FINDV is set to .FALSE. It calculates the required values of net thrust and calibrated airspeed for zero acceleration along and normal to

the flightpath, dV_c/dt and $V_c(dy/dt)$ respectively when the flag, FINDV is set to .TRUE. Finding one of these two sets of values is accomplished by driving two functions of two independent variables to zero simultaneously

$$\begin{aligned} f(F_N, \alpha) &= dV_c/dt = 0 \\ g(F_N, \alpha) &= V_c(dy/dt) = 0 \\ g(F_N, \alpha) &= (dy/dt) = 0 \end{aligned} \quad \text{for all non-zero positive values of } V_c, \text{ this reduces to}$$

where:

F_N is net thrust
 α is angle of attack
 V_c is calibrated airspeed
 dV_c/dt is the time rate of change of velocity along flightpath
 dy/dt is the time rate of change of flightpath angle

Subroutine STEDYST starts generating the locus of $f(F_N, \alpha) = 0$ and $g(F_N, \alpha) = 0$, and then searches for values of net thrust and angle of attack, or net thrust and calibrated airspeed, which satisfy the intersection of the two loci within a tolerance bound. The output values of net thrust and angle of attack, or net thrust and calibrated airspeed, are the required values for a steady-state approach.

Figure 5 shows a typical plot of the dV_c/dt and dy/dt functions, f and g , respectively. The values α_{app} and F_{Napp} are the desired values of angle of attack and net thrust. The upper bound on thrust is the gross weight (GWT) divided by the engine multiplicative factor (XENG). The search for α_1 and α_2 is made on the interval $-\alpha_{max} \leq \alpha_i \leq \alpha_{max}$, $i=1,2$. The error is then defined to be the difference between α_1 and α_2 divided by α_1 . Subroutine STEDYST then varies the value of net thrust until the error is driven to less than the subroutine tolerance, or $\alpha_1 - \alpha_2 \leq \epsilon$. This tolerance value for α , ϵ_α , is 0.005 degrees until the number of iterations within the subroutine exceeds 25; then ϵ_α is raised to 0.010 degrees. The initial value of angle of attack is estimated based on the lift coefficient where lift equals gross weight. The initial value of net thrust is estimated to be 1.06 times the sum of the drag at the initial angle of attack and the excess thrust (net thrust). Net thrust is varied each iteration, until bounded, by the following equation: $F_{N_{i+1}} = F_{N_i}/1.05$. Making this initial 1.1 times the net thrust prevents the program searching in areas where no solution to $dV_c/dt = 0$ exists. For finding calibrated airspeed the process is identical. Calibrated airspeed is substituted for angle of attack. However, the tolerance value for V_c , ϵ_v , is 0.001 knot until the number of iterations within the subroutine exceeds 25; then ϵ_v is raised to 0.005 knot.

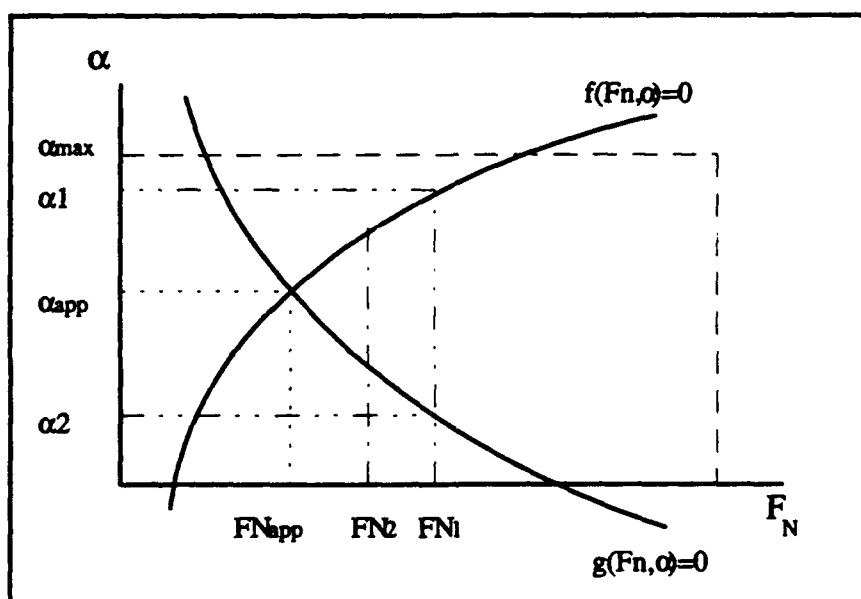


Figure 5 Steady-State Approach Solution

Subroutine STEDYST has calling arguments: ALPHA, FPVTAS, GAMMAR, and FINDV; and contains common blocks: CTRL, AIRCRAFT, AERO, ENGINE, ATMOS, CONST and FLAGS. The mechanization of subroutine STEDYST for determining THRUST and ALPHA is as follows. The outer loop varies THRUST, while the two inner loops vary ALPHA from ALPHMX to -ALPHMX. For a fixed value of THRUST (outer loop), functions DVDT (acceleration along the flightpath) and DGD (acceleration normal to the flightpath), are called with the values of ALPHA (inner loops). These two functions call subroutine FORCEX. Subroutine ITRLND and function ZEROX are both zero finders called from STEDYST. Once the values of net thrust are bounded, function ZEROX locates the values within the boundaries to within a specified tolerance. The tolerance for net thrust is based on 0.1 percent of the aircraft gross weight, or 5 pounds, whichever is greater.

The loop containing the call to ITRLND is exited successfully when search status variable, JFLAG, is set internally by the subroutine to 3, or when the magnitude of the error is less than the tolerance value (TOLRNCE). Variable JCOUNT stores the number of iterations which is limited to 25.

Subroutine FLARENZ

This subroutine determines the normal load factor, n_z , of the flare maneuver. Subroutine FLARENZ calls FLARE at various normal load factors until the sink rate at touchdown (SINKTD) is matched to within

EPSROC feet/minute. EPSROC is hard-coded at 0.5 feet/minute. When FLARENZ is first called, the initial guess for normal load factor is determined by assuming the flare is using the following simplified trajectory. The flare trajectory is estimated by a constant normal load factor (n_z) constant airspeed circular arc starting at the flare height (HFLARE), flightpath angle (GAMMAPP), and approach speed (VKAPP). The end of the arc is defined by the sink rate at touchdown (SINKTD), and the approach speed (VKAPP).

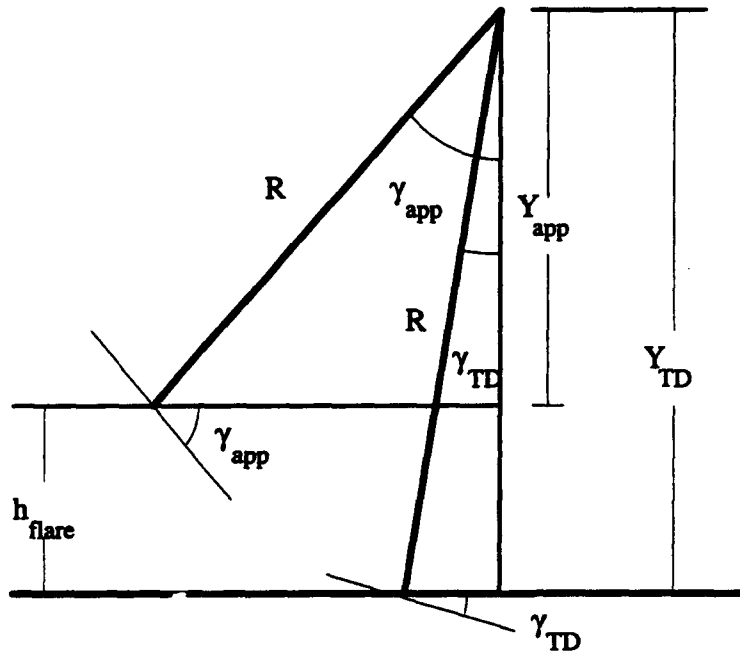


Figure 6 Initial Flare Load Factor Determination

Figure 6, shows a diagram of the flare. The flare height can be defined as a function of the flare radius (R) and the flare radius can be represented using the circular centripetal acceleration equation:

$$h_{\text{flare}} = Y_{\text{TD}} - Y_{\text{app}} = R \cos \gamma_{\text{TD}} - R \cos \gamma_{\text{app}} = R (\cos \gamma_{\text{TD}} - \cos \gamma_{\text{app}}) \quad (1)$$

$$a_n = V_{\text{app}}^2 / R = g(n_z - 1.0) \quad (2)$$

Substituting equation 2 for R into equation 1,

$$h_{\text{flare}} = [V_{\text{app}}^2 (\cos \gamma_{\text{TD}} - \cos \gamma_{\text{app}}) / g] / (n_z - 1.0) \quad (3)$$

Solving for normal load factor as a function of the two flightpath angles and approach speed,

$$n_z = [V_{\text{app}}^2 (\cos \gamma_{\text{TD}} - \cos \gamma_{\text{app}}) / g h_{\text{flare}}] + 1.0 \quad (4)$$

This gives subroutine FLARENZ a starting point for iterating on normal load factor to match the sink rate at touchdown.

Subroutine FLARENZ uses ITRLND to determine new values of normal load factor to use in calling FLARE. FLARENZ can call subroutine FLARE up to MAXITER times. MAXITER is hard-coded at 31. This subroutine has the calling arguments: ALPHA, GAMMAPP, GAMMATD, HAGL, HCLEAR, HFLARE, HRUNWAY, SINKTD, THRUST, VKAPP, XLFLARE, and XLFMAX; and contains common blocks: CTRL, FPINTEG, CONST, and FLAGS.

Subroutine APPROCH

This subroutine controls execution of the approach above the flare height, HFLARE. This subroutine is called only if the obstacle clearance height HCLEAR is greater than HFLARE. APPROCH calculates

values for output every APPDTIM seconds. APPDTIM is hard coded to 0.5 second. This subroutine has the calling arguments: ALPHA, GAMMAPP, GDIST, HAGL, HCLEAR, HFLARE, and HRUNWAY; and contains common blocks: CTRL, AIRCRFT, AERO, ENGINE, AIRSPED, FPINTEG, ATMOS, CONST, and FLAGS.

Subroutine FLARE

This subroutine controls execution of the flare maneuver. Subroutine FLARE calls INTX to perform numerical integration of the resultants of the equations of motion and calls FORCEX to obtain the force coefficients for the equations of motion. This subroutine has the calling arguments: ALPHA, GDIST, ODIST, ROCTD, and VKTGS; and contains common blocks: CTRL, AIRCRFT, AERO, ENGINE, AIRBORN, AIRSPED, RUNWAY, FPINTEG, ATMOS, CONST, and FLAGS.

The variable DTDGEX is provided to the user as a means to account for a loss in the pitch rate capability due to ground effect. DTDGEX may be a function of wing height/wingspan; total lift coefficient (CL), or circulation lift coefficient; et cetera. DTDGEX would equal 1.0 out of ground effect and less than 1.0 in ground effect. DTDGEX should be calculated in subroutine 'FXXAERO' and passed through the calling statement to subroutine FORCEX. The user may ignore DTDGEX without affecting program execution.

Subroutine ROLL

This subroutine controls the execution of the ground roll of the landing maneuver or the deceleration portion of a refused takeoff maneuver. Subroutine ROLL calls INTX to perform numerical integration of the resultants of the equations of motion and call FORCEX to obtain the force coefficient for the equations of motion. The calling arguments are, angle of attack (ALPHA) and the ground distance covered by the approach and flare or the takeoff ground roll (GDIST); ROLL contains common blocks: CTRL, AIRCRFT, AERO, ENGINE, AIRSPED, RUNWAY, INTEG, ATMOS, CONST, FLAGS and CHARV.

Subroutine INTX

This subroutine performs the numerical integration using a fourth order Runge-Kutta scheme with the Adams-Bashforth-Moulton predictor-corrector method (Reference 4). The calling arguments are NEQ, TIME, DTIME, T, DERIV and ALPHA. NEQ is the number of equation of motion to be integrated. T is either the values of common block INTEG or FPINTEG. DERIV is a subroutine that must be made external in the calling subroutine. DERIV corresponds to either DERIVGR, DERIVAT, or DERIVAL; INTX contains no common blocks.

Subroutine INTG

This subroutine calculates incremental distance, DDIST; total ground distance, GDIST; and gross weight, GWT. This subroutine has the calling arguments: DIST, DISTJ, DTIME, GAMMAR, VWIND, WFUEL, DDIST, GDIST, and GWT; and contains no common blocks.

Subroutine DERIVGR

This subroutine calculates the acceleration, ACCEL, for the ground roll of a takeoff, landing or refused takeoff. This subroutine has the calling argument: ALPHA; and contains common blocks: CTRL, AIRCRFT, AERO, AIRSPED, RUNWAY, INTEG, CONST, and FLAGS.

Subroutine DERIVAT

This subroutine calculates the time derivatives for the airborne portion of the takeoff and manages the flightpath control. This subroutine has the calling argument: ALPHA; and contains common blocks: AIRCRFT, AERO, AIRBORN, FPINTEG, CONST, and FLAGS.

Subroutine DERIVAL

This subroutine calculates the time derivatives for the airborne portion of the landing and manages the flightpath control. This subroutine has the calling argument: ALPHA; and contains common blocks: CTRL, AIRCRFT, AERO, AIRBORN, FPINTEG, CONST, and FLAGS.

Subroutine ERROR

This subroutine is called during a takeoff simulation when the flag ERRFLAG is set to .TRUE. Program execution is terminated for the present namelist inputs and will continue if additional namelist inputs are to be processed. This subroutine has the calling arguments: LUOUT and ROCFPM; and contains no common blocks.

Subroutine HALT

This subroutine terminates program execution and writes a termination message, TERMMSG. This subroutine has the calling arguments: LUIN, LUMSG, LUOUT, and TERMMSG; and contains no common blocks.

Subroutine ATMOSPH

This subroutine calculates atmospheric pressure, temperature, density, speed of sound, kinematic viscosity, as well as pressure, temperature and density ratios. The inputs are pressure altitude (PRESALT) and the temperature increment from standard day (DTEMPF). This subroutine has the calling arguments: PRESALT and ARRAY; and contains common block ATMOS.

Subroutine SPEED

This subroutine calculates dynamic pressure, Mach number, calibrated airspeed, equivalent airspeed, true airspeed, and groundspeed. This subroutine has the calling arguments: GAMMAR, VTASX, VWIND, AMACH, QS, VKCAS, VKEAS, VKTAS, VKTGS, and VTGS; and contains common blocks: AIRCRFT, ATMOS, and CONST.

Subroutine ITRLND

This subroutine is a zero-finding routine which varies the independent variable based on the sign and size of the error. The loop containing the call to ITRLND is exited successfully when JFLAG = 3 or when the magnitude of the error is less than the tolerance value, TOLRNCE. This subroutine has the calling arguments: ERROR, ERRORJ, DRIVER, FACTOR, TOLRNCE, and JFLAG; and contains no common blocks. ERROR and ERRORJ are the present and previous errors, respectively. JFLAG is a variable which indicates the status of the search. JFLAG is set to 0 before ITRLND is called. After the first call to ITRLND, JFLAG is set to 1. JFLAG is set to 2 once the zero is isolated between the upper and lower boundaries, BOUNDU and BOUNDL, respectively. JFLAG is set to 3 and the zero is considered found when the absolute value of the difference between BOUNDU and BOUNDL is less than TOLRNCE, the tolerance passed to subroutine ITRLND.

During the first call to ITRLND DRIVERJ, the previous DRIVER, is set to DRIVER and then DRIVER is multiplied by FACTOR. This forms a range with boundaries, DRIVERJ and DRIVER. On subsequent calls to ITRLND, these two driver values are set to either an upper or lower boundary or a multiplication or division by FACTOR is applied to DRIVER. Once the location of the zero is determined to be on one side of the DRIVER or the other, DRIVER is set to either the upper or lower boundary as the search is narrowed.

Program Function Descriptions

Function DGGT

This function returns the acceleration normal to the flightpath. This function has the calling arguments: ALPHA, FPVTAS, GAMMAR, and GWT; and contains common blocks: AERO, AIRSPED and CONST.

Function DVDT

This function returns the acceleration along the flightpath, dV_c/dt . Function DVDT uses functions, DVTDH, DADH, and DDELTDH to account for acceleration from flying at a constant calibrated airspeed. This function has the calling arguments: ALPHA, FPVTAS, GAMMAR, and GWT; and contains common blocks: AERO, AIRSPED and CONST.

Function DVTDH

This function returns the climb speed derivative with respect to pressure altitude (dV_c/dH) for a constant calibrated airspeed, equivalent airspeed or Mach number. Using this function for a calibrated airspeed (by setting CONSTANT = 'VC') allows function DVDT to return a value dV_c/dt instead of dV_t/dt . This function has the calling arguments: VTAS and CONSTANT; and contains common blocks: CTRL, FPINTEG, ATMOS, CONST and RACURV.

Function DADH

This function returns the speed of sound derivative with respect to pressure altitude (da/dH). This function has the calling argument: HC; and contains common block: CTRL.

Function DDELTDH

This function returns the pressure ratio derivative with respect to pressure altitude ($d\delta/dH$). This function has the calling argument: HC; and contains common block: CTRL.

Function DSIGDH

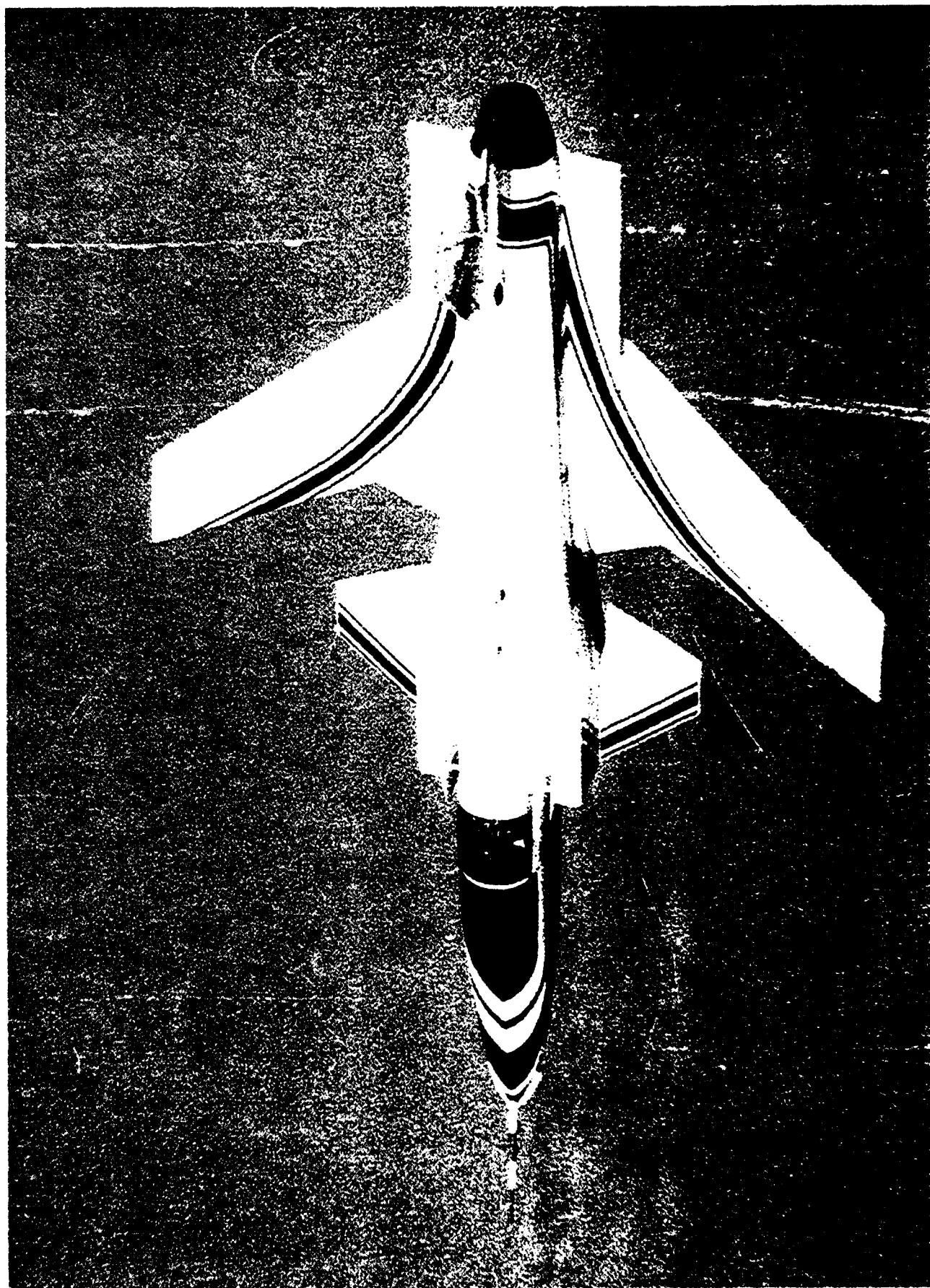
This function returns the density ratio derivative with respect to pressure altitude, $d\sigma/dH$. This function has the calling argument: HC; and contains common block: CTRL.

Function INTERP

This function returns an interpolated value between two ordinate inputs based on the additional inputs of three abscissas. To prevent an arithmetic indefinite this function resets the difference of the two maxima abscissas to one if the difference equals zero. This function has the calling arguments: YNOW, YPAST, XNOW, XPAST and XANS; and contains no common blocks.

Function ZEROX

This function finds the location (in X) of a zero of a function FUNCT(X) on an interval $X = A$ to $X = B$. The function ZEROX assumes that F(A) and F(B) are of opposite sign, implying that FUNCT(X) is equal to zero somewhere within that interval. Function ZEROX finds this zero by recursively calling function FUNCTN. ZEROX uses the logical variable, FINDV, to transfer the proper calling arguments to function FUNCTN. If FINDV is .TRUE., ZEROX passes calibrated airspeeds for A and B; otherwise, angles of attack are passed. The argument "FUNCTN" must be declared external in the calling subroutine and have four calling arguments, A or B, VAR2, VAR3, and VAR4. Functions DGGT and DVDT meet this criteria. ZEROX has the calling arguments: A, B, FUNCTN, VAR2, VAR3, VAR4, TOLRNCE, and FINDV; and contains no common blocks.



COMMON BLOCK DESCRIPTIONS

This section describes the common blocks of the TOLAND program and their variables.

Common CTRL

Common CTRL contains variables concerning program control.

DTIME is the current Runge-Kutta Numerical Integration Step Size in seconds. The default for DTIME is 0.10 seconds; DTIME is entered through namelist DATA2.

DTIMEJ is the previous Runge-Kutta numerical integration step size in seconds. This variable is used to store the original value of DTIME while the current value is lowered to allow the output to be printed at an even interval of time. This variable is used at the interfaces of subroutines APPROCH, FLARE, and ROLL.

ICOUNT is the integration loop initializing variable. After switching from one integration loop to another, ICOUNT stores the integer variable which initializes the integration loop. Therefore, the program will write the output variables after executing both integration loops a total of 10 times. This could be two iterations of the first loop and eight iterations of the next loop or three iterations of the first loop and seven iterations of the next loop, et cetera. Placing ICOUNT in Common CTRL allows the program to execute different integration loops in different subroutines while still retaining writing the output variables once every 10 iterations. With DTIME, the Runge-Kutta Integration Step Size set to 0.1 seconds, output will occur once per second of simulation time.

JDEBUG is the TOLAND debug code. This code is user definable; it can be used to output debug data from the user provided subroutines. When JDEBUG is set to 6666, debug output for the steady-state iterations is printed. The JDEBUG setting of 9999 is reserved and used for full debug output. The default for JDEBUG is 0; JDEBUG is entered through namelist DATA2.

KENG is the subroutine 'FXXENG' output switch. It allows the program to switch between the types of output subroutine 'FXXENG' can provide subroutine 'FXXENG' outputs gross thrust and fuel flow from a power setting input if KENG is set equal to 1. Subroutine 'FXXENG' outputs power setting from an input of thrust if KENG is set equal to 2. This subroutine is provided by the user; KENG can be just a null or inactive variable. Currently all the calls to subroutine 'FXXENG' require outputs of thrust and fuel flow from a power setting input.

LINENUM is the current line number of the current page of output. When LINENUM exceeds NPAGE, a new page is started.

LUIN is the logical unit for input. It is an integer variable which determines which file namelist inputs are read from. The default for LUIN is 3, but can be user defined; LUIN is initialized in the main program.

LUMSG is the logical unit for message output. It is an integer variable which determines which file certain program messages are written to. The messages that can be re-routed come from the debug output as well as UFTAS random access curve file subroutines. The default for LUMSG is LUOUT; LUMSG is entered through namelist DATA2.

LUOUT is the logical unit for output. It is an integer variable which determines which file program output is written to. The default for LUOUT is 4, but can be user defined; LUOUT is initialized in the main program.

NCOUNT is the integration loop counter. After executing an integration loop 10 times, the program will write the output variables. Placing NCOUNT in common CTRL allows the program to execute different integration loops in different subroutines while still retaining writing the output variables once every

10 iterations. With DTIME, the Runge-Kutta integration step size set to 0.1 seconds, output will occur once per second of simulation time.

NEQ is the number of equations of motion to be integrated. NEQ is set to 2 for the takeoff ground roll and landing roll and is set to 4 for the airborne portions of the takeoffs and landings.

NPAGE is the number of lines of output per page. When LINENUM exceeds NPAGE, a new page is started. The default for NPAGE is 64 lines per page; NPAGE is hard coded in the main program.

TIME is the elapsed time in seconds.

TIMEROL is the elapsed time of the ground roll. This elapsed time corresponds to the landing ground roll or the portion of the ground roll of a refused takeoff after engine failure.

Common AIRCRFT

Common AIRCRFT contains many of the aircraft specific constants and variables.

AOA3PT is the angle of attack for the aircraft in a three-point attitude. This aircraft constant is used to initialize angle of attack at the beginning of a takeoff and to set the zero angle of attack flag (AOA0FLG) to true for a landing. AOA3PT is initialized in subroutine FORCEX.

AR is the aspect ratio of the wing. This aircraft constant is used to calculate ground effect variables. AR is initialized in subroutine FORCEX.

B is the wingspan in feet. This aircraft constant is used to calculate ground effect variables. B is initialized in subroutine FORCEX.

CGPCT is the longitudinal position of the center of gravity as a percentage distance of the mean aerodynamic chord aft from the leading edge of the wing. The default for CGPCT is user defined in subroutine FORCEX; CGPCT is entered through namelist DATA.

CLALPH is the lift curve slope out of ground effect. This aircraft constant is used to calculate ground effect variables. Because this constant is used only in the user provided subroutine, GEFFECT, it can have units of either 1/degrees or 1/radians. CLALPH is initialized in subroutine FORCEX.

CONFIG is a number representing the configuration of the aircraft. This floating point variable is a user definable input. CONFIG is initialized in subroutine FORCEX. This variable is not required; CONFIG is entered through namelist DATA2.

DTDTMX is the maximum pitch rate during the landing simulation in degrees per second. The default for DTDTMX is user defined in subroutine FORCEX; DTDTMX is entered through namelist LND2.

FLT is the flight number. It is a floating point variable available to the user for management and tracking of data. The default for FLT is 0.0; FLT is entered through namelist DATA2.

GWT is the gross weight during program execution in pounds. GWT is a required variable; there is no default. GWT is initialized as GWT0. GWT0 is entered through namelist DATA.

HZ is the distance from the bottom of the wing to the bottom of the tires in feet. This aircraft constant is used to calculate ground effect variables. HZ is initialized in subroutine FORCEX.

LOADING is a number representing the external store loading of the aircraft. This integer variable is a user definable input. LOADING is initialized in subroutine FORCEX. This variable is not required; LOADING is entered through namelist DATA2.

SWING is the wing surface area in square feet. SWING is initialized in subroutine FORCEX.

THTMAX is the maximum pitch attitude limit in degrees. THTMAX is initialized in subroutine FORCEX.

WNGLOD is the wing loading in pounds per square foot. It is calculated in the main program and output at the beginning of a takeoff simulation in subroutine TAKOFF.

XLFMAX is the simulation maximum normal load factor in g's. XLFMAX is initialized in subroutine FORCEX. XLFMAX is only used for the flare portion of a landing.

Common AERO

Common AERO contains the major aerodynamic parameters.

CX is the force coefficient along the flightpath or ground roll. This parameter is further defined in Appendix A, Equations of Motion.

CY is the force coefficient normal to the flightpath or ground roll. This parameter is further defined in Appendix A, Equations of Motion.

DADTCMD is the commanded angle of attack rate. It is the rate at which the aircraft rotates during the takeoff maneuver. DADTCMD affects the rate of convergence to target pitch attitudes and target airspeeds. The default for DADTCMD is 2.5 degrees per second; DADTCMD is entered through namelist DATA.

DCDX is the user drag coefficient increment. The user may alter the drag coefficient (CD) by adding a delta drag coefficient increment. The default for DCDX is 0.0000; DCDX is entered through namelist DATA2.

DCLX is the user lift coefficient increment. The user may alter the lift coefficient (CL) by adding a delta lift coefficient increment. The default for DCLX is 0.0000; DCLX is entered through namelist DATA2.

DTDTGEX is the loss in pitch rate capability due to ground effect. The user may lower the pitch rate during the flare in ground effect by using this multiplicative factor applied to pitch rate. The default for DTDTGEX is 1.0; DTDTGEX can be calculated in subroutine 'FXXAERO' and passed through the calling statement to subroutine FORCEX.

FLAP is the current flap deflection in degrees during program execution. The default for FLAP is -1.0 degree. This default value allows FLAP to be set from FLPPCT. FLAP is a required input if FLPPCT is not used. FLAP is entered through namelist DATA.

FLPPCT is the current flap percentage setting during program execution. This optional variable provides the user an additional method of entering flap deflection into the program. The default is user defined in subroutine FORCEX. FLPPCT is entered through namelist DATA2.

QS is the product of current dynamic pressure (Q) and wing surface area (SWING). The units for QS are in pounds.

SPDBRK is the current speed brake deflection in degrees.

SPOILER is the current spoiler deflection in degrees.

VKCAS is the current calibrated airspeed in knots.

VKTAS is the current true airspeed in knots.

Common ENGINE

Common ENGINE contains the major engine parameters.

AIT is the thrust incidence angle in degrees. This constant is initialized in subroutine FORCEX.

AMACH is the current Mach number.

DTFAIL is the delta time for the failed engine to lose thrust by the thrust level defined by XENGFLD. Thrust and fuel flow decrease linearly over the time interval DTFAIL. The default for DTFAIL is 0.0 second; DTFAIL is entered through namelist DATA2.

EPR is engine pressure ratio. This parameter can optionally be used to input engine thrust levels to the program or can be used as a variable which contains the current value of Engine Pressure Ratio. The default for EPR is 0.0; EPR is entered through namelist DATA2.

FE is the current propulsive drag of all the engines in pounds.

FG is the current gross thrust of all the engines in pounds.

FGPCT is the gross thrust percentage increment. The user may increase the gross thrust by a given percentage using this input variable. The default for FGPCT is 0.0 percent; FGPCT is entered through namelist DATA2.

NENG is the total number of engines the aircraft has. This constant is initialized in subroutine FORCEX.

PWRCODE is the aircraft power code. It can describe the thrust settings to which all the aircraft engines are set. The default for PWRCODE is user defined in subroutine FORCEX; PWRCODE is entered through namelist DATA2.

REVNDX is the reverse engine spool-up index. It can describe the spool-up curve in subroutine SPOOLUP which corresponds to thrust reversers. This constant is user defined in subroutine 'FXXENG' and must be passed through as a calling argument to subroutine FORCEX; REVNDX is not required if thrust reversers are not to be simulated.

THRUST is the current total net thrust of all the engines in pounds.

VTANGLE is the current vectored thrust angle in degrees.

WFUEL is the current total fuel flow of all the engines in pounds per hour.

XENG is the current engine multiplicative factor. Total engine parameters (e.g. net thrust, fuel flow) are defined by their corresponding per engine values multiplied by XENG. The default for XENG is user defined in subroutine FORCEX; XENG is entered through namelist DATA2.

XENGFLD is the failed engines multiplicative factor. This variable is not an integer variable and is not the number of failed engines. It is a floating point variable that is used with per engine values of thrust and fuel flow. This allows the program to represent engine values of a partial failed engine (e.g. afterburner). If XENGFLD is not equal to 1.0, FAILST must be set to 'NULL'. FAILST overrides XENGFLD. The default for XENGFLD is 1.0; XENGFLD is entered through namelist DATA2.

XENGOUT is the engine multiplicative factor after an engine failure of XENGFLD engines. It is defined as the XENG - XENGFLD.

XIDLE is the idle engine multiplicative factor for a single engine. The default for **XIDLE** is 0.06; **XIDLE** is user defined in subroutine **FORCEX**.

XMIL is the military thrust engine multiplicative factor for a single engine. The default for **XMIL** is 0.50; **XMIL** is user defined in subroutine **FORCEX**.

ZFN is a user definable variable.

Common AIRBORN

Common AIRBORN contains airborne control and status parameters.

ALPHAJ is the previous angle of attack calculated from the last integration in degrees.

ALPHMX is the maximum angle of attack limit in degrees. ALPHMX is set to THTMAX in subroutine LANDNG.

DTDT is the current pitch rate in degrees per second.

GAMMAPP is the flightpath angle in degrees during approach. The default for GAMMAPP is -3.0 degrees (descending); GAMMAPP is entered through namelist LND.

ROCFPM is the current rate of climb in feet per minute.

THETAF is the current pitch attitude in degrees.

XLF is the current normal load factor in g's.

XLFJ is the previous normal load factor calculated from the last numerical integration in g's.

Common AIRSPED

Common AIRSPED contains the airspeed constants used for program control.

VKABRK is the calibrated airspeed during the landing ground roll that aerobraking is stopped and the angle of attack is lowered from AOAABRK to 0.0 degree. The default for VKABRK is 0.0 knot calibrated airspeed (no aerobraking); VKABRK is entered through namelist ROL2.

VKAPP is the calibrated airspeed in knots during the approach. VKAPP is entered through namelist LND.

VKBRAKE is the wheel braking airspeed in knots. Wheel braking is initiated after the calibrated airspeed is less than VKBRAKE. The default for VKBRAKE is 999 knots calibrated airspeed; VKBRAKE is entered through namelist ROL2.

VKEND is the simulation end airspeed in knots. The default for VKEND is 250 knots calibrated airspeed; VKEND is entered through namelist TKO2.

VKFAIL is the engine failure airspeed in knots. The default for VKFAIL is 0.0 knot calibrated airspeed which is defined as no engine failure; VKFAIL is entered through namelist TKO.

VKFLAP is the calibrated airspeed at which the flaps are retracted. The default for VKFLAP is the flap limit airspeed (VKFLPMX); VKFLAP is entered through namelist TKO2. The flap limit airspeed (VKFLPMX) is user defined in subroutine 'FXXAERO'.

VKFLPMX is the maximum flap calibrated airspeed in knots. This constant is used as the default for VKFLAP; it is user defined in subroutine 'FXXAERO'.

VKMCG is the minimum control airspeed on the ground. It is an input variable used to trigger a spool-up of an operating but asymmetric engine. It would be used in the case of (for example) a three-engine takeoff of a four-engine aircraft whose gross weight was not sufficient for nose wheel steering to account for the full or partial asymmetric thrust. VKMCG is not used to fail an engine; use VKFAIL. The default for VKMCG is 0.0 knot calibrated airspeed; VKMCG is entered through namelist DATA2.

VKROTAT is the rotation airspeed in knots. The default for VKROTAT is 0.0 knot calibrated airspeed. The simulation will rotate the aircraft without adequate tail authority; TOLAND runs with only engine and aerodynamic models which do not define moments about the aircraft center of gravity; VKROTAT is entered through namelist TKO.

VKSTART is the simulation start groundspeed in knots. This input variable allows the user to simulate takeoffs where there is a given groundspeed at the time the aircraft is aligned on the centerline of the runway. The default for VKSTART is 0.0 knot; VKSTART is entered through namelist TKO2.

VKWIND is the headwind component of wind speed in knots. Tailwinds are input using negative wind speeds. The default for VKWIND is 0.0 knot (no wind); VKWIND is entered through namelist DATA.

VWIND is the headwind component of wind speed in feet per second. Tailwinds use negative wind speeds.

Common RUNWAY

Common RUNWAY contains the major parameters related to the runway.

ABARG is the average deceleration in g's

AOAABRK is the angle of attack for aerobraking during the landing ground roll if the calibrated airspeed is greater than **VKABRK** knots calibrated airspeed. The default for **AOAABRK** is 0.0 degree (no aerobraking); **AOAABRK** is entered through namelist **ROL2**.

BRAKMU is the braking coefficient of friction. This input parameter would be used if **IMU**, the Braking Coefficient Selector, is set to 0 (corresponding to a constant **BRAKMU**). The default for **BRAKMU** is 0.250; **BRAKMU** is entered through namelist **ROL**.

BRKFCTR is the braking factor. It is a multiplicative factor applied to **BRAKMU**, the Braking Coefficient of Friction. It allows the user to adjust the coefficient of friction during braking without recompiling code. This input parameter would normally be used if **IMU**, the Braking Coefficient Selector, was not set to 0. The default for **BRKFCTR** is 1.0; **BRKFCTR** is entered through namelist **ROL2**.

GAMMARW is the slope of the runway in degrees. The default for **GAMMARW** is 0.0 degree; **GAMMARW** is entered through namelist **DATA2**.

GRW is the slope of the runway in radians.

HAGL is the current altitude of the aircraft above the liftoff point on the runway for takeoff simulations or the current altitude of the aircraft above the runway for landing simulations.

HCLEAR is the obstacle clearance height in feet. The default for **HCLEAR** is 50.0 feet; **HCLEAR** is entered through namelist **DATA**.

HFLARE is the height in feet above the touchdown point of the runway at which the simulation initiates a flare. The default for **HFLARE** is 0.0 feet (no flare); **HFLARE** is entered through namelist **LND**.

HGEAR is landing gear retraction height in feet above the takeoff point. The default for **HGEAR** is 205 feet; **HGEAR** is entered through namelist **TKO**.

HRUNWAY is the pressure altitude of the runway in feet. The default for **HRUNWAY** is 0.0 feet; **HRUNWAY** is entered through namelist **DATA**.

IMU is the braking coefficient selector. The default selections for **IMU** are constant braking coefficient, **BRAKMU**, 0; user supplied curve file lookup, 1; generic dry runway braking coefficient, 2; and generic wet runway braking coefficient, 3. The default for **IMU** is 0, which corresponds to a constant braking coefficient of friction, **BRAKMU**; **IMU** is entered through namelist **ROL**. The default for **BRAKMU** is 0.250; **BRAKMU** is entered through namelist **ROL**.

RCR is the runway condition reading. It can be used as an input to determine the variable **BRAKMU**. The default for **RCR** is user defined in subroutine **GENMU**; **RCR** is entered through namelist **ROL**.

ROLLMU is the rolling coefficient of friction. The default for **ROLLMU** is 0.025; **ROLLMU** is entered through namelist **DATA2**.

TIMEBRK is the braking time delay in seconds between the time touchdown has occurred and the time the brakes are applied for a landing maneuver. It is also the braking time delay between the time an engine

has failed and the time the brakes are applied for a refused takeoff maneuver. The default for TIMEBRK is 3.0 seconds; TIMEBRK is entered through namelist ROL2.

TIMEFLD is the elapsed time from engine failure. TIMEFLD is initialized in subroutine TAKOFF when the engine failure speed (VKFAIL) is attained.

TIMEFLP is the flap retraction time delay in seconds between the time touchdown has occurred and the time the flap retraction initiation has occurred retracting for a landing maneuver. It is also the flap retraction time delay between the time an engine has failed and the time the flap retraction initiation has occurred for a refused takeoff. The default for TIMEFLP is 999.0 seconds; essentially the flaps are not retracted. TIMEFLP is entered through namelist ROL2.

TIMESBK is the speedbrake deployment time delay in seconds between the time touchdown has occurred and the time the speedbrake deployment initiation has occurred for a landing maneuver. It is also the speedbrake time delay between the an engine has failed and the time the speedbrake deployment initiation has occurred for a refused takeoff maneuver. The default for TIMESBK is 0.0 second; TIMESBK is entered through namelist ROL2.

TIMESPL is the spoiler deployment time delay in seconds between the time touchdown has occurred and the time the spoiler deployment initiation has occurred for a landing maneuver. It is also the spoiler time delay between the an engine has failed and the time the spoiler deployment initiation has occurred for a refused takeoff maneuver. The default for TIMESPL is 0.0 second; TIMESPL is entered through namelist ROL2.

XMU is the current coefficient of friction. It is set equal to either ROLLMU or BRAKMU. ROLLMU is a constant entered through namelist DATA. BRAKMU is a variable set in subroutine GENMU or entered as a constant through namelist ROL.

Common INTEG

Common INTEG contains the ground roll integration variables.

VTAS is the true airspeed in feet per second; it is the integral of ACCEL with respect to time.

DIST is the air distance (ground distance uncorrected for wind) in feet; It is the integral of VTAS with respect to time.

ACCEL is the ground roll acceleration in feet per second squared. ACCEL is provided by the equation of motion executed in subroutine DERIVGR.

VTASJ is the previous true airspeed in feet per second. It is used to continually initialize the Runge-Kutta array for the ground roll iterations.

RKGRND is an array containing previous values of the ground roll integration variables. These values are used in the fourth order Runge-Kutta numerical integration calculations to determine future values.

Common FPINTEG

Common FPINTEG contains the flightpath integration variables.

FPVTAS is the flightpath true airspeed in feet per second; it is the integral of FPACCEL with respect to time.

GAMMAR is the flightpath angle in radians; it is the integral of DGDTR with respect to time.

FPDIST is the flightpath air distance in feet; it is the integral of FPVTAS with respect to time.

PRESALT is the pressure altitude in feet; it is the sum of HRUNWAY and the integral of ROC with respect to time.

FPACCEL is the flightpath acceleration in feet per second squared. FPACCEL is provided from an equation of motion executed in either subroutine DERIVAT or subroutine DERIVAL.

DGDTR is the rate of change in flightpath angle with respect to time in radians per second. DGDTR is provided by an equation of motion executed in either subroutine DERIVAT or subroutine DERIVAL.

VHAS is the horizontal airspeed in feet per second. VHAS is calculated from FPVTAS and GAMMAR in either subroutine DERIVAT or subroutine DERIVAL.

ROC is the rate of climb in feet per second. ROC is calculated from FPVTAS and GAMMAR in either subroutine DERIVAT or subroutine DERIVAL.

RKAIR is an array containing previous values of the flightpath integration variables. These values are used in the fourth order Runge-Kutta numerical integration calculations to determine future values.

Common ATMOS

Common ATMOS contains the atmospheric parameters.

TEMPR is the current temperature in degrees Rankine.

PRESS is the current ambient pressure in pounds per square foot.

RHO is the current density is slugs per cubic feet.

AFPS is the current speed of sound in feet per second.

VISCOSK is the current kinematic viscosity in feet squared per second.

DELTA is the current pressure ratio.

SIGMA is the current density ratio.

THETA is the current temperature ratio.

DTEMPF is the delta temperature from standard day in degrees Fahrenheit. The default for DTEMPF is 0.0 degree Fahrenheit; it is entered through namelist DATA.

Common CONST

Common CONST contains the major program constants.

ASL is the speed of sound at sea level in knots. This constant is equal to 661.48 and is initialized in the main program.

ASLSQR5 is the speed of sound at sea level multiplied by the square root of 5.0, = 1479.114245. This constant is used in the calculation for calibrated airspeed.

FPSKTS is the conversion factor from knots to feet per second. This constant is equal to 1.687806 and is initialized in the main program.

G is the reference acceleration at sea level, 45 degrees latitude, due to gravity. This constant is equal to 32.174 and is initialized in the main program.

RX is the conversion factor from radians to degrees. This constant is equal to 57.2957759130824 and is initialized in the main program.

TSLF is the standard day temperature at sea level in degrees Fahrenheit. This constant is equal to 59.0 degrees and is initialized in the main program.

TWOVR7 is the ratio of 2.0 over 7.0. This constant is used in the calculation for calibrated airspeed; it is equal to 0.285714285714285 and is initialized in the main program.

ZERO is the constant 0.0; it is initialized in the main program.

Common FLAGS

Common FLAGS contains the major variables of type LOGICAL.

AOA0FLG is the zero angle of attack flag. This flag is .TRUE. when the angle of attack (ALPHA) is equal to 0.0.

BRKFLAG is the brake flag. This flag is set to .TRUE. when the conditions for brake application are met. The conditions are: current calibrated airspeed (VKCAS) less than brake application speed (VKBRAKE) and elapsed time from engine failure or touchdown (TIMEROL) is greater than the braking time delay (TIMEBRK).

CLRHTG is the clearance height flag. This flag is set to .TRUE. after the current altitude (HAGL) exceeds the clearance height (HCLEAR).

ERRFLAG is the error flag. This flag is set to .TRUE. by any of the equation of motion subroutines, (DERIVGR, DERIVAT, or DERIVAL) when certain flightpath constraints are not satisfied and the condition is not recoverable. The program will terminate processing with the current set of namelist inputs.

FAILFLG is the engine failure flag. This flag is set to .TRUE. when a refused takeoff or a continued takeoff simulation is to be performed. At the engine failure speed (VKFAIL) the engine failure speed flag (VFFLAG) is set to .TRUE. and the program switches the operating engine group (ENGGRP) to the fail engine group (FAILGRP). If the engine failure state (FAILST) is set to 'OFF' the current engine multiplicative factor (XENG) is set to the engine multiplicative factor after engine failure (XENGOUT). If the time for a failed engine to lose thrust, DTFAIL, is greater than zero, the current engine multiplicative factor is linearly reduced to the engine multiplicative factor after engine failure, XENGOUT, over DTFAIL seconds.

FLAPFLG is the flap flag. This flag is .TRUE. when the flap deflection, FLAP, is greater than zero.

FPCTFLG is the flap percentage flag. This flag is set to .TRUE. when the flap deflection is initialized at -1. With the flap percentage flag set to .TRUE., the flap percentage variable (FLPPCT) is used to determine flap deflection (FLAP).

GEFLAG is the ground effect flag. This flag is set to .TRUE. when the aircraft is in ground effect. The conditions which determine when this flag should be .TRUE. are to be user defined.

LGRFLAG is the landing gear flag. This flag is .TRUE. when the landing gear is not fully retracted.

LIFTOFF is the liftoff flag. This flag is set to .TRUE. after liftoff is achieved.

OVERFLG is the flare height over obstacle clearance height flag. This flag is set to .TRUE. if the flare height (HFLARE) is greater than the obstacle clearance height (HCLEAR).

REVFLAG is the reverse thrust flag. This flag is set to .TRUE. when reverse thrust is to be enabled for a simulation.

REVRSE is the thrust reverse active flag. This flag is set to .TRUE. when reverse thrust is simulated.

ROTATE is the rotation flag. This flag is set to .TRUE. after rotation is initiated.

RTOFLAG is the refused takeoff flag. This flag is initialized to .TRUE. when the simulation performs a refused takeoff. It causes subroutine TAKOFF to return control to the main program which subsequently calls subroutine ROLL.

SBKFLAG is the speed brake flag. This flag is set to **.TRUE.** after the speed brake deflection has changed.

SPLFLAG is the spoiler flag. This flag is **.TRUE.** when the spoiler deflection (**SPOILER**) is greater than 0.0; **SPLFLAG** is entered through namelist **LND2**.

SPOOL is the engine spool flag. This flag is **.TRUE.** when the engine spooling for a throttle transient is completed.

STEADY is the steady-state flag. This flag is set to **.TRUE.** during a steady-state approach when engine data is not to be determined by subroutine **'FXKENG'**. Thrust and fuel flow do not change during the steady-state approach; only during the flare can thrust, fuel flow, and throttle setting be changed.

TERMFLG is the termination flag. This flag is set to **.TRUE.** when the any termination criterion is satisfied. This flag is currently used for takeoff and refused takeoff simulations.

VECTFLG is the thrust vectoring flag. This flag is set to **.TRUE.** when thrust vectoring is to be enabled for a simulation.

VFFLAG is the engine failure velocity flag. This flag is set to **.TRUE.** when the engine failure speed (**VKFAIL**) is reached.

WRITTTT is the write flare iterations flag. This flag is set to **.TRUE.** when the user desires output on each flare iteration executed from subroutine **FLARENZ**. The default for **WRITTTT** is **.FALSE.**; **WRITTTT** is entered through namelist **LND2**.

Common CHARV

Common CHARV contains the major variables of type CHARACTER.

ENGGRP is the operating engine group. It is a user definable three-character string variable whose contents define which engines are operating during a maneuver. Examples of ENGGRP are 'AEO', 'OET', 'TET', and 'AET' which correspond to all engines operating, outboard engine inoperative, inboard engine inoperative and all engines idling respectively. The default for ENGGRP is 'AEO' for all takeoffs and 'AET' for all landings; ENGGRP is entered through namelist DATA2.

FAILGRP is the fail engine group. It is a user definable three-character string variable whose contents define the state of the engines after an engine failure or failures. The default for FAILGRP is 'OET'; FAILGRP is entered through namelist DATA2.

FAILMOD is the engine failure mode. It is a five-character string variable whose contents define the mode of the failed engine. Usually, only two states are used: 'SEIZE' and 'SPOOL'. The default for FAILMOD is 'SEIZE'; FAILMOD is entered through namelist DATA2.

FAILST is the engine failure state. It is a four-character string variable whose contents define the state of the failed engine. Usually, only two states are used, 'IDLE' and 'OFF'. The default for FAILST is 'IDLE'; FAILST is entered through namelist DATA2.

MANUVR is the maneuver input. It is a three- or six-character string variable whose contents define which maneuver is to be simulated during execution. Valid three-character values of MANUVR are: TKO, RTO, and LND, for takeoffs, refused takeoffs, and landings respectively. Valid six-character values of MANUVR are: TOLTKO, TOLRTO, and TOLLND; for takeoffs, refused takeoffs, and landings respectively. The three-character values indicate that the program is to read MANUVR once for the entire input file. This allows the user to run one type of maneuver repetitively. For example, a single input file containing many namelist entries for landings would use MANUVR = LND. The input file would look like this:

```
LND
&DATA DADTCMD=2.50, FLAP=30.0, GWTO= 200000., .../
&DATA2.../
&LND HFLARE = 20., SINKTD = 10.0, VKAPP= 150.0 /
&LND2.../
&ROL IMU=1, RCR=16. /
&ROL2 VKBRAKE=110.0 /
&DATA DADTCMD=2.50, FLAP=30.0, GWTO= 200000., .../
&DATA2.../
&LND HFLARE = 15., SINKTD = 10.0, VKAPP= 150.0 /
&LND2.../
&ROL IMU=1, RCR=16. /
&ROL2 VKBRAKE=110.0 /
&DATA DADTCMD=2.50, FLAP=30.0, GWTO= 200000., .../
&LND HFLARE = 10, SINKTD = 10.0, VKAPP= 150.0 /
&LND2.../
&ROL IMU=1, RCR=16. /
&ROL2 VKBRAKE=110.0 /
```

The six character values indicate that the program is to read MANUVR for each set of namelist inputs within the input file. This allows the user to run different types of maneuvers in one input file. For example, a single input file containing many namelist entries for determine critical field lengths would look like this:

```
TOLTKO
  &DATA DADTCMD=2.50, FLAP=15.0, GWT= 250000., .../
  &TKO HMAX=50., THTROT=8.0, THTCLM= 8.0, VKFAIL=110.0, VKROTAT=120.0 /
  &TKO2 /
  &TKOARY /
TOLRTO
  &DATA DADTCMD=2.50, FLAP=15.0, GWT= 250000., .../
  &TKO HMAX=50., THTROT=8.0, THTCLM= 8.0, VKFAIL=110.0, VKROTAT=120.0 /
  &TKO2 /
  &TKOARY /
  &ROL IMU=1, RCR=16 /
  &ROL2 VKBRAKE=100.0 /
```

Continued takeoffs use the same MANUVR as takeoffs. There is no default for MANUVR; MANUVR is input in the main program before namelist DATA in the input file.

MVR is the first three characters of MANUVR. When MVR is set equal to 'TOL', the parameter MANUVR is read for each set of namelist inputs. (See description of MANUVR in the previous paragraphs.) There is no default for MVR; MVR is defined by the first three characters of MANUVR.

THRCRV is the thrust curve type. It is a three-character input variable that is used to describe the type of thrust curve to be utilized. This input variable allows the program to distinguish between thrust curves containing different independent (Input) variables. Examples of THRCRV are 'RC' and 'EPR'. The default for THRCRV is user defined in subroutine FORCEX; THRCRV is entered through namelist DATA2.

TKOTYPE is the takeoff type. It is a seven character input variable that is used to describe the type of takeoff as either 'STATIC' or 'ROLLING'. This input variable allows the program to distinguish between the thrust settings of either a static or rolling takeoff independently of the VKSTART input variable. The default for TKOTYPE is 'STATIC'; TKOTYPE is entered through namelist TKO2.

Common FLAPDAT

Common FLAPDAT contains the major flap variables.

IFLAP is the current element of arrays FLPARY and VFLPARY.

MAXSIZF is the maximum size of arrays FLPARY and VFLPARY. The default for MAXSIZF is 5; MAXSIZF is hard coded in subroutine TAKOFF. If MAXSIZF is increased, the dimensions for arrays FLPARY and VFLPARY must be increased accordingly.

FLPARY is the flap deflection array. This array contains five elements. This array allows the user to enter in a flap deflection schedule as a function of calibrated airspeed. At each airspeed contained in the flap deflection airspeed array (VFLPARY) the takeoff simulation changes the flap deflection (FLAP) to the corresponding flap deflection contained in the flap deflection array (FLPARY) at a rate of DFLAPDT degrees per second. The flap deflection rate (DFLAPDT) is provided by subroutine 'FXXAERO' which is a user provided subroutine called by subroutine FORCEX. If the last element of FLPARY is not 0.0, the simulation will retract the flaps to 0.0 degree at the calibrated airspeed contained in the last element of VFLPARY or at VKFLPMX which ever is less. The default for the elements of FLPARY are 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation does not change the flap deflection until the flap retraction speed, VKFLAP, or the maximum flap deflection airspeed (VKFLPMX) is reached. The elements of FLPARY are entered through namelist TKOARY.

VFLPARY is the flap deflection airspeed array. This array contains five elements. This array allows the user to enter in a flap deflection schedule as a function of calibrated airspeed. At each airspeed contained in the flap deflection airspeed array (VFLPARY) the takeoff simulation changes the flap deflection (FLAP) to the corresponding flap deflection contained in the flap deflection array (FLPARY) at a rate of DFLAPDT degrees per second. The flap deflection rate (DFLAPDT) is provided by subroutine 'FXXAERO' which is a user provided subroutine called by subroutine FORCEX. If the last element of FLPARY is not 0.0, the simulation will retract the flaps to 0.0 degree at the calibrated airspeed contained in the last element of VFLPARY or at VKFLPMX which ever is less. The default for the elements of VFLPARY are 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation does not change the flap deflection until the flap retraction speed, VKFLAP, or the maximum flap deflection airspeed, VKFLPMX, is reached. The elements of VFLPARY are entered through namelist TKOARY.

Common GEARDAT

Common GEARDAT contains the major landing gear variables.

IGEAR is the current element of array LGRARY.

MAXSIZG is the maximum size of array LGRARY. The default for MAXSIZG is 6; MAXSIZG is hard coded in subroutine TAKOFF. If MAXSIZG is increased, the dimension for array LGRARY must be increased accordingly.

LGRARY is the landing gear drag array. This array contains six elements. This array allows the user to enter in a landing gear drag coefficient schedule as a function of time from the start of landing gear retraction. When the landing gear retraction height (HGEAR) is reached the subroutine GRETRAC returns the gear drag coefficient increment (DCDLGR) as the value contained in the first element of LGRARY. This value remains current for one second. The value contained in the second element of the array replaces the gear drag coefficient increment previously for another 1.0 second interval. The gear drag coefficient increment (DCDLGR) must be added where the drag coefficient (CD) is summed from its components. This will usually be accomplished in the user provided subroutine 'FXXAERO'. An example table is shown below. If the last element of LGRARY is not 0.0000, the simulation will set DCDLGR to 0.0000 after drag coefficient increment. The default for the elements of LGRARY are 0.0, 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation will reduce the drag coefficient linearly over the time for gear retraction (DTGEAR) in seconds. The elements of LGRARY are initialized in subroutine 'FXXAERO' which is the user provided subroutine called by subroutine FORCEX.

Array Element	C_D (without gear increment)	+	LGRARY	= C_D (with gear increment)
1	0.0280		0.0220	0.0500
2	0.0280		0.0240	0.0520
3	0.0280		0.0280	0.0560
4	0.0280		0.0160	0.0440
5	0.0280		0.0070	0.0350
6	0.0280		0.0000	0.0280

Notice the total landing gear increment is removed between the 5th and 6th second after the initiation of landing gear retraction.

Common VECTDAT

Common VECTDAT contains the major thrust vectoring variables.

IVECT is the current element of arrays XNUARY, HVCTARY, and VVCTARY.

MAXSIZV is the maximum size of arrays XNUARY, HVCTARY, and VVCTARY. The default for MAXSIZV is 5; MAXSIZV is hard coded in subroutine TAKOFF. If MAXSIZV is increased, the dimensions for arrays XNUARY, HVCTARY, and VVCTARY must be increased accordingly.

XNUARY is the vectored thrust angle array. This array contains five elements. At each altitude contained in the vectored thrust altitude array (HVCTARY) or at each calibrated airspeed contained in the vectored thrust airspeed array (VVCTARY) the takeoff simulation changes the vectored thrust angle (VTANGLE) to the corresponding vectored thrust angle contained in the vectored thrust angle array (XNUARY) at a rate of DVECTDT degrees per second. The vectored thrust angle rate (DVECTDT) is provided by subroutine 'FXKENG' which is a user provided subroutine called by subroutine FORCEX. If both the HVCTARY and VVCTARY arrays are used, the HVCTARY array takes precedence. The default for the elements of VVCTARY are 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation does not change the vectored thrust angle. The elements of XNUARY are entered through namelist TKOARY.

HVCTARY is the vectored thrust altitude array. This array contains five elements. At each altitude contained in the vectored thrust altitude array (HVCTARY) the takeoff simulation changes the vectored thrust angle (VTANGLE) to the corresponding vectored thrust angle contained in the vectored thrust angle array (XNUARY) at a rate of DVECTDT degrees per second. The vectored thrust angle rate (DVECTDT) is provided by subroutine 'FXKENG' which is a user provided subroutine called by subroutine FORCEX. If both the HVCTARY and VVCTARY arrays are used, the HVCTARY array takes precedence. The default for the elements of HVCTARY are 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation does not change the vectored thrust angle. The elements of HVCTARY are entered through namelist TKOARY.

VVCTARY is the vectored thrust airspeed array. This array contains five elements. At each calibrated airspeed contained in the vectored thrust airspeed array (VVCTARY) the takeoff simulation changes the vectored thrust angle (VTANGLE) to the corresponding vectored thrust angle contained in the vectored thrust angle array (XNUARY) at a rate of DVECTDT degrees per second. The vectored thrust angle rate (DVECTDT) is provided by subroutine 'FXKENG' which is a user provided subroutine called by subroutine FORCEX. If both the HVCTARY and VVCTARY arrays are used, the HVCTARY array takes precedence. The default for the elements of VVCTARY are 0.0, 0.0, 0.0, 0.0, 0.0. At this default setting, the takeoff simulation does not change the vectored thrust angle. The elements of VVCTARY are entered through namelist TKOARY.

Common CINDEX

Common CINDEX contains major random access curve file constants.

MSTRNDX is the random access curves master index. It is an array of NDXLEN (3), integers initialized by subroutine OPENMS, a CDC Cyber specific FORTRAN subroutine.

NDXLEN is the length of the master index. NDXLEN is an integer constant equal to three; NDXLEN is initialized in subroutine INICURV.

LUCURV is the logical unit for the curve file. It is an integer variable which determines the local file name of the random access curve file. The default for LUCURV is 2 (which corresponds to file TAPE2); LUCURV is initialized in subroutine INICURV. Currently, no other value is valid for this parameter.

Common RACURV

Common RACURV contains the major random access curve file control variables as well as the random access curve file names.

IDBUG is the UFTAS debug flag. It can provide debug information concerning the UFTAS random access curve file subroutines. The default for IDBUG is 0; IDBUG is entered through namelist DATA2.

IEXTOR is the extrapolation override code. When set to a non-zero value, IEXTOR overrides the curve extrapolation code. This flag is intended to be used in only specialized cases. The default for IEXTOR is 0; IEXTOR is initialized in subroutine INICURV.

KURNAM is the random access curve file name array. It contains the names of up to 99 random access curve file names as Hollerith strings stored in an array of type INTEGER.

Common VALUES

Common VALUES contains the arrays which contain the values passed to and from the random access curve files. Examples for array names are VALIN, VAERO, and VENGINE for the input values array, the aerodynamic values output array, and the engine values output array respectively. This common block is user defined and largely determined by the curve file structure.

Common TABLES

Common TABLES contains the table arrays used to store raw values from the random access curve files. The values which are passed to the output values arrays are interpolated from these raw values. Examples for array names are TAERO and TENGINE for the aerodynamic output table array and the engine output table array respectively. This common block is user defined and determined by the curve file structure. The size of each of the table arrays is provided to the user when the random access curve file is generated.



REFERENCES

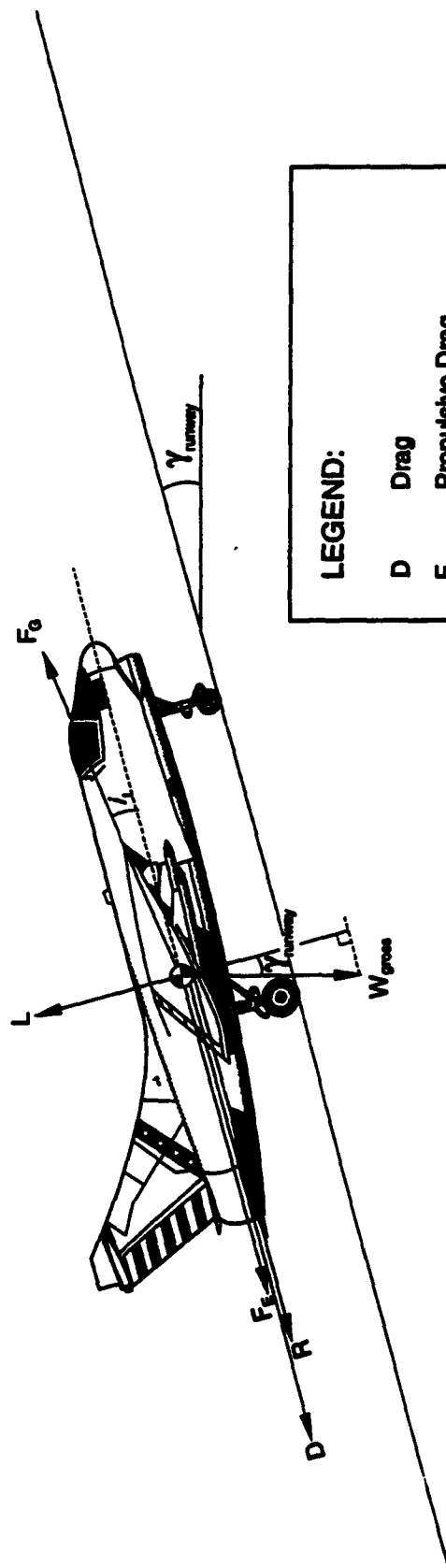
1. Bowles, Jeff V., and Galloway, Thomas L., *Computer Programs for Estimating Aircraft Takeoff and Landing Performance*, NASA Technical Memorandum X.62,333, 1973.
2. Olson, Wayne, NASA Ames Takeoff and Landing Simulation Computer Program Supplementary Documentation, circa 1979.
3. MIL-C-005011B (USAF), *Charts: Standard Aircraft Characteristics And Performance, Pilot Aircraft (Fixed Wing)*, 21 June 1977.
4. Morris, John L.L., *Computational Methods in Elementary Numerical Analysis*, John Wiley and Sons, LTD, 1983, ISBN 0 471 10419 1.

BIBLIOGRAPHY

1. Batt, James T., F-15C Limited Takeoff and Landing Evaluation, AFFTC-TR-81-18, Air Force Flight Test Center, Edwards AFB, California, January 1982.
2. Jabour, William J.; Jones, Lyle W.; and Friedman, Lee M., F-16A Braking Evaluation, AFFTC-TR-85-05, Air Force Flight Test Center, Edwards AFB, California, June 1985.



FREE - BODY DIAGRAM



$$R = \mu[W_{gross} \cos \gamma_{runway} - L - F_G \sin (\alpha + i_i)]$$

$$\frac{dV_L}{dt} = \frac{g}{W_{gross}} [F_G \cos (\alpha + i_i) - F_E - D - R - W_{gross} \sin \gamma_{runway}]$$

ASSUMPTIONS:

- Wings Level
- Zero Sideslip
- Point Mass
- Three-point Attitude

LEGEND:

D	Drag
F _E	Propulsive Drag
F _G	Gross Thrust
g	Acceleration due to gravity
i _i	Thrust Incidence Angle
L	Lift
R	Runway Friction
t	time
V _L	True Airspeed
W _{gross}	Gross Weight
α	Angle of Attack
γ _{runway}	Runway Slope
μ	Coefficient of Friction

APPENDIX A: EQUATIONS OF MOTION

This section shows the equations of motion used in Subroutines FORCEX, DERIVGR, DERIVAT, and DERIVAL. The aircraft is treated as a point mass confined to motion in a vertical plane, and the rotational dynamics have been neglected. This simplification requires the input of the angular rates and precludes the use of this program as a tool to simulate minimum control speeds. These rates are approximated by a finite difference form or input by the user as commanded rates.

Force coefficient equations:

$$C_X = C_D - [F_G (\cos(\alpha + i_D) - F_E)/qS_{wing} \quad (\text{positive aft}) \quad (1)$$

$$C_Y = C_L + [F_G (\sin(\alpha + i_D) \quad]/qS_{wing} \quad (\text{positive up}) \quad (2)$$

Equation of motion during ground roll:

$$dV/dt = (g/W_{gross}) \{-W_{gross}[\sin(\gamma_{runway}) + \mu_{runway}\cos(\gamma_{runway})] + qS_{wing}(C_Y\mu_{runway} - C_X)\} \quad (3)$$

Equation of motion along flightpath:

$$dV/dt = (g/W_{gross}) \{-W_{gross}\sin(\gamma) - qS_{wing}(C_X)\} \quad (4)$$

Equation of motion normal to flightpath:

$$V_t(dy/dt) = (g/W_{gross}) \{-W_{gross}\cos(\gamma) + qS_{wing}(C_Y)\} \quad (5)$$

Load factor equation:

$$n_z = qS_{wing}C_Y/(W_{gross}) + [1.0 - \cos(\gamma)] \quad (6)$$

Rate of Climb equation:

$$R/C = V_t\sin(\gamma) \quad (7)$$

where:

- C_D = aerodynamic drag coefficient
- C_L = aerodynamic lift coefficient
- C_X = force coefficient along flightpath
- C_Y = force coefficient normal to flightpath
- F_E = ram drag
- F_G = gross thrust
- dt = delta time
- g = acceleration due to gravity
- n_z = load factor
- q = dynamic pressure
- R/C = rate of climb
- S_{wing} = reference wing area
- V_t = true airspeed
- W_{gross} = gross weight
- α = angle of attack
- γ = flightpath angle
- γ_{runway} = runway slope
- μ_{runway} = runway friction coefficient (either braking or rolling)

Rotational rate approximations by finite difference form

$$\theta = \gamma + \alpha \quad (8)$$

Differentiating with respect to time

$$d\theta/dt = d\gamma/dt + d\alpha/dt \quad (9)$$

Divide equation 5 by V_i

$$d\gamma/dt = [g/(V_i W_{gross})] \{-W_{gross} \cos(\gamma) + q S_{wing} (C_Y)\} \quad (10)$$

Finite difference approximation

$$d\alpha/dt \equiv (\alpha - \alpha_j)/\Delta t \quad (11)$$

Substituting equations 10 and 11 into equation 9

$$d\theta/dt = [g/(V_i W_{gross})] \{-W_{gross} \cos(\gamma) + q S_{wing} (C_Y)\} + (\alpha - \alpha_j)/\Delta t \quad (12)$$

where:

dt = delta time

Δt = integration step size

α = current angle of attack

α_j = previous angle of attack

γ = flightpath angle

θ = pitch attitude

Landing equations

The flare height can be defined as a function of the flare radius (R) and the flare radius can be represented using the circular centripetal acceleration equation. For constant normal load factor (n_z) and constant airspeed ($V_{t_{app}}$)

$$h_{flare} = Y_{TD} - Y_{app} = R \cos \gamma_{TD} - R \cos \gamma_{app} = R (\cos \gamma_{TD} - \cos \gamma_{app}) \quad (13)$$

$$a_n = V_{t_{app}}^2 / R = g(n_z - 1.0) \quad (14)$$

Substituting equation 14 for R into equation 13,

$$h_{flare} = [V_{t_{app}}^2 (\cos \gamma_{TD} - \cos \gamma_{app}) / g] / (n_z - 1.0) \quad (15)$$

Solving for normal load factor as a function of the two flightpath angles and approach speed,

$$n_z = [V_{t_{app}}^2 (\cos \gamma_{TD} - \cos \gamma_{app}) / g h_{flare}] + 1.0 \quad (16)$$

where:

a_n = acceleration normal to flightpath	γ_{app} = approach flightpath angle
g = acceleration due to gravity	γ_{TD} = touchdown flightpath angle
h_{flare} = flare initiation altitude (AGL)	
n_z = normal load factor	
R = flare radius	
$V_{t_{app}}$ = approach true airspeed	

This gives subroutine FLARENZ a starting point for iterating on normal load factor to match the sink rate at touchdown.

Using the following equation for kinetic energy, brake energy is calculated with equation 18;

$$E_{kinetic} = \frac{1}{2} (W_{gross} / g) V_{t_g}^2 \quad (17)$$

$$E_{brake} = E_{kinetic} + \Sigma (F_N - C_D q S_{wing}) V_{t_g} \Delta t \quad (18)$$

where:

C_D = aerodynamic drag coefficient
F_N = net thrust
E_{brake} = brake energy
$E_{kinetic}$ = kinetic energy
g = acceleration due to gravity
q = dynamic pressure
S_{wing} = reference wing area
V_{t_g} = true ground speed
W_{gross} = gross weight
Δt = delta time

Main Gear Normal Force equations during braking

The normal force on the main gear can be determined by summing the forces in both the x- and y-axes and summing the moments about the center of gravity. The rolling coefficient of friction is applied to the nose gear normal force and the braking coefficient of friction is applied to the main gear normal force. The following equations assume no aerodynamic loading of the tail, no moments from spoilers or other control surfaces, and the center of gravity is located forward of the braking gear (the main).

Summing the forces in the x-axis,

$$-(W_{gross}/g) dV/dt = C_X (q S_{wing}) + W_{gross} \sin \gamma_{runway} + \mu_{rolling} N_{nose} + \mu_{brake} N_{main} \quad (19)$$

Summing the forces in the y-axis,

$$0 = C_Y (q S_{wing}) - W_{gross} \cos \gamma_{runway} + N_{nose} + N_{main} \quad (20)$$

Summing the moments about the center of gravity,

$$0 = N_{nose} X_{nose} - N_{main} X_{main} - \mu_{rolling} N_{nose} Y_{cg} - \mu_{brake} N_{main} Y_{cg} \quad (21)$$

Solving for N_{main} ,

$$N_{main} = \frac{[(W_{gross}/g) dV/dt + C_X (q S_{wing}) + W_{gross} \sin \gamma_{runway}] Y_{cg} + [W_{gross} \cos \gamma_{runway} - C_Y (q S_{wing})] X_{nose}}{(X_{main} + X_{nose})} \quad (22)$$

where:

- C_X = force coefficient along flightpath
- C_Y = force coefficient normal to flightpath
- dV/dt = acceleration
- g = acceleration due to gravity
- N_{main} = main gear normal force
- N_{nose} = nose gear normal force
- q = dynamic pressure
- S_{wing} = reference wing area
- W_{gross} = gross weight
- X_{main} = horizontal distance from main gear to center of gravity
- X_{nose} = horizontal distance from nose gear to center of gravity
- Y_{cg} = vertical distance from ground to center of gravity
- γ_{runway} = runway slope
- μ_{brake} = braking friction coefficient
- $\mu_{rolling}$ = rolling friction coefficient

Distances are all positive values. Any units for distance, if applied consistently, will work in these equations.



APPENDIX B: SOURCE CODE LISTING

This section provides the user with a complete source code listing of the program and calling subroutines. Curve file subroutines are not included.

Main Program

PROGRAM TOLAND	TOLAND	1
C****	TOLAND	2
C**** SUBROUTINES-- INITIAL SPOIL FLARE SPEED INTG	TOLAND	3
C**** PITCH TVECTOR ROLL DERIVGR STEDYST	TOLAND	4
C**** FRETRAC TAKEOFF ERROR DERIVAT FLARENZ	TOLAND	5
C**** GRETRAC LANDING HALT DERIVAL ITRLND	TOLAND	6
C**** SPDBRAK APPROCH ATMOSPH INTX	TOLAND	7
C****	TOLAND	8
C**** FUNCTIONS-- DGD T DVTDH DDELTDH INTERP ZEROX	TOLAND	9
C**** DVT DADH DSIGDH	TOLAND	10
C****	TOLAND	11
C**** USER PROVIDED SUBROUTINES--	TOLAND	12
C**** INICURV 'FXXAERO' 'FXXENG' SPOOLDNF GENMU	TOLAND	13
C**** FORCEX GEFFECT SPOOLUP SPOOLDNR	TOLAND	14
C****	TOLAND	15
C**** TAKEOFF, LANDING AND REFUSED TAKEOFF SIMULATION	TOLAND	16
C****	TOLAND	17
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCFT/ AO3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKCAS,VKTAS	AERO	2
COMMON/ENGINE/ AIT,AMACH,DTFAIL,EPR,FE,FG,FGPCT,NENG,PWRCODE,	ENGINE	1
& REVNDX,RC,THRUST,VTANGLE,WFUEL,XENG,XENGFLD,	ENGINE	2
& XENGOUT,XIDLE,XMIL,XFN	ENGINE	3
COMMON/VECTOR/ HVECT,VVECT	VECTOR	1
COMMON/AIRBORN/ ALPHAJ,ALPHMX,DTDT,GAMMAPP,ROCFPM,THETAJ,XLF,XLFJ	AIRBORN	1
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
COMMON/RUNWAY/ ABARG,AOABRK,BRAKMU,BRKFCR,GAMMARW,GRW,HAGL,	RUNWAY	1
& HCLEAR,HFLARE,HGEAR,HRUNWAY,IMU,RCR,ROLLMU,TIMEFLD,	RUNWAY	2
& TIMEBRK,TIMEFLP,TIMESBK,TIMESPL,XMU	RUNWAY	3
DOUBLEPRECISION VTAS,DIST,ACCEL,VTASJ,RKGRND	INTEG	1
COMMON/INTEG/ VTAS,DIST,ACCEL,VTASJ,RKGRND(20)	INTEG	2
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
COMMON/ATMOS/ TEMPR,PRESS,RHO,AFPS,VISCOSE,DELTA,SIGMA,THETA,DTEMPF	ATMOS	1
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3

LOGICAL	AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
&	GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
&	RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
&	VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/	AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
&	FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
&	REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
&	TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
CHARACTER	ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
&	THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/	ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
&	TKOTYPE	CHARV	4
COMMON/FLAPDAT/	IFLAP,MAXSIZF,FLPARY(5),VFLPARY(5)	FLAPDAT	1
REAL	LGRARY(6)	GEARDAT	1
COMMON/GEARDAT/	IGEAR,MAXSIZG,LGRARY	GEARDAT	2
COMMON/VECTDAT/	IVECT,MAXSIZV,XNUARY(5),HVCTARY(5),VVCTARY(5)	VECTDAT	1
COMMON/CINDEX/	MSTRNDX(3),NDXLEN,LUCURV	CINDEX	1
COMMON/RACURV/	IDBUG,IEXTOR,KURNAM(99)	RACURV	1
LOGICAL	MOREFLG	TOLAND	36
NAMelist/DATa/	CGPCT,DADTCMD,DTEMPF,FLAP,GAMMARW,GWT0,HCLEAR,	TOLAND	37
&	HRUNWAY,VKWINd	TOLAND	38
NAMelist/DATa2/	AOA3PT,CONFIG,DCDX,DCLX,DTIME,DTFAIL,ENGGRP,EPR,	TOLAND	39
&	FAILGRP,FAILMOD,FAILST,FLPPCT,FLT,FLTNDX,FGPCT,	TOLAND	40
&	IDBUG,JDEBUG,LOADING,LUMSG,PWRCODE,RC,REVFLAG,	TOLAND	41
&	ROLLMU,SPDBRK0,THRCRV,VKMCG,XENG,XENGFLD	TOLAND	42
C****		TOLAND	43
C****	OPEN INPUT AND OUTPUT FILE	TOLAND	44
	OPEN (UNIT=LUIN,FILE='')	TOLAND	45
	OPEN (UNIT=LUOUT,FILE='')	TOLAND	46
C****		TOLAND	47
C****	MANUVR TKO,TOLTKO TAKEOFF	TOLAND	48
C****	LND,TOLLND LANDING	TOLAND	49
C****	RTO,TOLRTO REFUSED TAKEOFF	TOLAND	50
15	READ(LUIN,1001,END=99)MANUVR	TOLAND	51
1001	FORMAT(A6)	TOLAND	52
C****		TOLAND	53
C****	RESET SOME INPUT PARAMETERS FROM ANY PREVIOUS RUN (IF ANY), THEN	TOLAND	54
C****	READ NAMelist DATA FROM INPUT FILE (LUIN). PARAMETERS NOT RESET	TOLAND	55
C****	HERE RETAIN THEIR VALUE FOR THE SUBSEQUENT RUN IF NOT EXPLICITLY	TOLAND	56
C****	DECLARED IN NAMelist DATA.	TOLAND	57
25	CALL INITIAL('ZERO',ALPHA,FLAP0,GWT0,SPDBRK0)	TOLAND	58
	READ(LUIN,DATA,END=99)	TOLAND	59
	READ(LUIN,DATA2)	TOLAND	60
	CALL INITIAL('MAIN',ALPHA,FLAP0,GWT0,SPDBRK0)	TOLAND	61
	IF(MOREFLG) WRITE(LUOUT,2001)	TOLAND	61
2001	FORMAT('I')	TOLAND	62

IF	(MANUVR.EQ.'TKO' .OR. MANUVR.EQ.'TOLTKO') THEN	TOLAND	63
C****	TAKEOFF ROUTINE	TOLAND	64
	CALL TAKOFF(ALPHA,GDIST)	TOLAND	65
	WRITE(LUOUT,2002) FLT,GWT0,DCDX,FGPCT,VKMCG,FLAP0,DTEMPF,DCLX,	TOLAND	66
&	FAILST,ENGGRP	TOLAND	67
2002	FORMAT(' FLT =',F8.4,' WT =',F8.0,' DCD =',F7.4,	TOLAND	68
&	' FGPCT =',F7.1,' VKMCG =',F7.1,' FLAP =',F6.1/,	TOLAND	69
&	' DTEMP =',F7.1,' DEGREES F ',' DCL =',F7.4,	TOLAND	70
&	' FAILST =',A5,',', ENGGRP =',A4,/)	TOLAND	71
	ELSEIF(MANUVR.EQ.'RTO' .OR. MANUVR.EQ.'TOLRTO') THEN	TOLAND	72
C****	REFUSED TAKEOFF ROUTINE	TOLAND	73
	CALL TAKOFF(ALPHA,GDIST)	TOLAND	74
	CALL ROLL(ALPHA,GDIST)	TOLAND	75
	WRITE(LUOUT,2002) FLT,GWT0,DCDX,FGPCT,VKMCG,FLAP0,DTEMPF,DCLX,	TOLAND	76
&	FAILST,ENGGRP	TOLAND	77
	ELSEIF(MANUVR.EQ.'LND' .OR. MANUVR.EQ.'TOLLND') THEN	TOLAND	78
C****	LANDING ROUTINE	TOLAND	79
	CALL LANDNG(ALPHA)	TOLAND	80
	WRITE(LUOUT,2003) FLT,DCDX,DTEMPF,BRKFCCTR,DCLX,FLAP0	TOLAND	81
2003	FORMAT(' FLT =',F12.4,' DCD =',F7.4,' DTEMP =',F6.1,' DEGREES',	TOLAND	82
&	' F/,', BRKFCCTR =',F8.3,' DCL =',F7.4,' FLAP =',F6.1,/)	TOLAND	83
	ENDIF	TOLAND	84
	MOREFLG = .TRUE.	TOLAND	85
	IF(MVR.EQ.'TOL') THEN	TOLAND	86
	GO TO 15	TOLAND	87
	ELSE	TOLAND	88
	GO TO 25	TOLAND	89
	ENDIF	TOLAND	90
99	CALL HALT(LUIN,LUMSG,LUOUT,' TOLAND NORMAL TERMINATION')	TOLAND	91
	END	TOLAND	92

Program Subroutines

Subroutine INITIAL

SUBROUTINE INITIAL(GROUP,ALPHA,FLAP0,GWT0,SPDBRK0)	INITIAL	1
C**** THIS SUBROUTINE INITIALIZES PROGRAM VARIABLES.	INITIAL	2
C****	INITIAL	3
CHARACTER*4 GROUP	INITIAL	4
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKCAS,VKTAS	AERO	2
COMMON/ENGINE/ AIT,AMACH,DTFAIL,EPR,FE,FG,FGPCT,NENG,PWRCODE,	ENGINE	1
& REVNDX,RC,THRUST,VTANGLE,WFUEL,XENG,XENGFLD,	ENGINE	2
& XENGOUT,XIDLE,XMIL,ZFN	ENGINE	3
COMMON/VECTOR/ HVECT,VVECT	VECTOR	1
COMMON/AIRBORNI/ ALPHAJ,ALPHMX,DTDT,GAMMAPP,ROCFPM,THETAJ,XLF,XLFJ	AIRBORNI	1
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
COMMON/RUNWAY/ ABARG,AOAABRK,BRAKMU,BRKFCR,GAMMARW,GRW,HAGL,	RUNWAY	1
& HCLEAR,HFLARE,HGEAR,HRUNWAY,IMU,RCR,ROLLMU,TIMEFLD,	RUNWAY	2
& TIMEBRK,TIMEFLP,TIMESBK,TIMESPL,XMU	RUNWAY	3
DOUBLEPRECISION VTAS,DIST,ACCEL,VTASJ,RKGRND	INTEG	1
COMMON/INTEG/ VTAS,DIST,ACCEL,VTASJ,RKGRND(20)	INTEG	2
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
COMMON/ATMOS/ TEMPR,PRESS,RHO,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMPF	ATMOS	1
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRS,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITTR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRS,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	8
CHARACTER ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
& THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/ ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
& TKOTYPE	CHARV	4
COMMON/FLAPDAT/ IFLAP,MAXSIZF,FLPARY(5),VFLPARY(5)	FLAPDAT	1
REAL LGRARY(6)	GEARDAT	1
COMMON/GEARDAT/ IGEAR,MAXSIZG,LGRARY	GEARDAT	2
COMMON/VECTDAT/ IVECT,MAXSIZV,XNUARY(5),HVCTARY(5),VVCTARY(5)	VECTDAT	1

COMMON/RACURV/ IDBUG,IEXTOR,KURNAM(99)	RACURV	1
LOGICAL MSGFLAG	INITIAL	23
DATA DADTCMD, DCDX, DCLX, DTDGEX, DTEMPF, DTIME, EPR, FAILGRP, FAILMOD,	INITIAL	24
& / 2.5, 0.0, 0.0, 1.0, 0.0, 0.10D0, 0.0, 'OEI', 'SEIZE', /,	INITIAL	25
& FAILST, GAMMARW, HRUNWAY, HCLEAR, ICOUNT, LUMSG, NPAGE, ROLLMU	INITIAL	26
& / 'IDLE', 0.0, 0.0, 50.0, 1, 0, 58, 0.025 /	INITIAL	27
& VKMCG	INITIAL	27
& / 0.0 /	INITIAL	28
DATA FLPARY /0.0,0.0,0.0,0.0,0.0,0.0/VFLPARY/ 0.0, 0.0, 0.0, 0.0, 0.0/	INITIAL	30
& ,HVCTARY /0.0,0.0,0.0,0.0,0.0,0.0/VVCTARY/999.,999.,999.,999.,999./	INITIAL	31
& ,XNUARY /0.0,0.0,0.0,0.0,0.0,0.0/MAXSIZEF,MAXSIZEG,MAXSIZEV/5,6,5/	INITIAL	32
IF (GROUP.EQ.'ZERO') THEN	INITIAL	33
C**** INITIALIZE CONSTANTS.	INITIAL	34
C**** ASLSQR5 = 661.48*SQRT(5.0); TWOOVR7 = 2.0/7.0	INITIAL	35
ASLSQR5 = ASL*SQRT(5.0)	INITIAL	36
TWOOVR7 = 2.0/7.0	INITIAL	37
C****	INITIAL	38
C**** INITIALIZE MVR WITH THE FIRST THREE CHARACTERS OF MANUVR. THIS	INITIAL	39
C**** ALLOWS THE PROGRAM TO DISTINGUISH IF THE INPUT FILE CONTAINS ALL	INITIAL	40
C**** ONE TYPE OF MANEUVERS (I.E. RTOS) OR A VARIETY. IF THE INPUT	INITIAL	41
C**** FILE CONTAINS A VARIETY, THE PROGRAM WILL LOOP TO LINE 15 TO READ	INITIAL	42
C**** THE NEW MANUVR INSTEAD OF LINE 25.	INITIAL	43
MVR = MANUVR(1:3)	INITIAL	44
C****	INITIAL	45
C**** RESET SOME INPUT PARAMETERS FROM ANY PREVIOUS RUN (IF ANY),	INITIAL	46
C**** BEFORE READING NAMELIST DATA FROM INPUT FILE (TAPES).	INITIAL	47
C**** PARAMETERS NOT RESET HERE RETAIN THEIR VALUE FOR THE SUBSEQUENT	INITIAL	48
C**** RUN IF NOT EXPLICITLY DECLARED IN NAMELIST DATA.	INITIAL	49
AOA3PT = ZERO	INITIAL	50
DTFAIL = ZERO	INITIAL	51
ENGGRP = 'NUL'	INITIAL	52
FLAP = -1.0	INITIAL	53
FPCTFLG = .FALSE.	INITIAL	54
GWTO = ZERO	INITIAL	55
IDBUG = 0	INITIAL	56
JDEBUG = 0	INITIAL	57
REVFLAG = .FALSE.	INITIAL	58
TIMEROL = ZERO	INITIAL	59
VKWND = ZERO	INITIAL	60
XENG = ZERO	INITIAL	61
XENGFLD = 1.0	INITIAL	62

ELSE	INITIAL	83
C**** INITIALIZE LUMSG.	INITIAL	84
IF (LUMSG.EQ.0 .OR. LUMSG.EQ.LUIN) THEN	INITIAL	85
LUMSG = LUOUT	INITIAL	86
ELSEIF (LUMSG.NE.LUOUT .AND. .NOT.(MSGFLAG)) THEN	INITIAL	87
MSGFLAG = .TRUE.	INITIAL	88
OPEN(UNIT=LUMSG,FILE='TOLAND.MSG')	INITIAL	89
ENDIF	INITIAL	90
C**** INITIALIZE PROGRAM CONTROL PARAMETERS.	INITIAL	91
DADTCMD = ABS(DADTCMD)	INITIAL	92
DTIMEJ = DTIME	INITIAL	93
KENG = 0	INITIAL	94
LINENUM = 6	INITIAL	95
FLAPLG = .TRUE.	INITIAL	96
LGRFLAG = .TRUE.	INITIAL	97
VECTLG = .TRUE.	INITIAL	98
BRKFLAG = .FALSE.	INITIAL	99
CLRHTG = .FALSE.	INITIAL	100
ERRFLAG = .FALSE.	INITIAL	101
SBKFLAG = .FALSE.	INITIAL	102
SPLFLAG = .FALSE.	INITIAL	103
TERMFLG = .FALSE.	INITIAL	104
VFFLAG = .FALSE.	INITIAL	105
C****	INITIAL	106
C**** INITIALIZE AIRCRAFT CONSTANTS.	INITIAL	107
FLAPO = FLAP	INITIAL	108
GRW = GAMMARW/RX	INITIAL	109
GWT = GWT0	INITIAL	110
VWIND = VKWIND*FPSKTS	INITIAL	111
WNGLOD = GWT0/SWING	INITIAL	112
XMU = ROLLMU	INITIAL	113
C****	INITIAL	114
C**** TERMINATE PROGRAM IF GROSS WEIGHT NOT INITIALIZED.	INITIAL	115
IF(GWT0.EQ.ZERO) CALL HALT(LUIN,LUMSG,LUOUT,	INITIAL	116
& 'GROSS WEIGHT NOT INPUT BY USER.')	INITIAL	117
C****	INITIAL	118
C**** INITIALIZE THE NUMBER OF ENGINES BEFORE AND AFTER ENGINE	INITIAL	119
C**** FAILURE.	INITIAL	120
IF (NENG.EQ. 0) NENG = 1	INITIAL	121
IF (XENG.EQ.ZERO) XENG = FLOAT(NENG)	INITIAL	122
IF (FAILST.EQ.'MIL ') THEN	INITIAL	123
XENGFLD = 1.0 - XMIL	INITIAL	124
ENDIF	INITIAL	125
XENGOUT = XENG - XENGFLD	INITIAL	126
C****	INITIAL	127
C**** INITIALIZE LOGIC CONTROL VARIABLES FOR FLAPS, GEAR AND THRUST	INITIAL	128
C**** VECTORING.	INITIAL	129
IFLAP = IGEAR = IVECT = 1	INITIAL	130
C****	INITIAL	131
C**** INITIALIZE INTEGRATION VARIABLES.	INITIAL	132
RKGRND(14) = ZERO	INITIAL	133
RKAIR (27) = ZERO	INITIAL	134

	IF (MANUVR.EQ.'TKO' .OR.MANUVR.EQ.'TOLTKO') THEN	INITIAL	115
C****	INITIALIZE FLAGS FOR THE TAKEOFF MANEUVERS.	INITIAL	116
	AOA0FLG = .TRUE.	INITIAL	117
	GEFLAG = .TRUE.	INITIAL	118
	LIFTOFF = .FALSE.	INITIAL	119
	ROTATE = .FALSE.	INITIAL	120
	RTOFLAG = .FALSE.	INITIAL	121
	IF(ENGGRP.EQ.'NUL') ENGGRP = 'AEO'	INITIAL	122
C****		INITIAL	123
C****	RESET SOME INPUT PARAMETERS FROM ANY PREVIOUS RUN (IF ANY).	INITIAL	124
C****	PARAMETERS NOT RESET HERE RETAIN THEIR VALUE FOR THE	INITIAL	125
C****	SUBSEQUENT RUN IF NOT EXPLICITLY DECLARED IN NAMELIST TKO.	INITIAL	126
	TKOTYPE = 'STATIC'	INITIAL	127
	NEQ = 2	INITIAL	128
	ALPHA = ZERO	INITIAL	129
	DIST = ZERO	INITIAL	130
	HAGL = ZERO	INITIAL	131
	PRESALT = HRUNWAY	INITIAL	132
	THETAF = AOA3PT	INITIAL	133
	TIME = ZERO	INITIAL	134
	VKFAIL = ZERO	INITIAL	135
	VKROTAT = ZERO	INITIAL	136
	VKSTART = ZERO	INITIAL	137
	XLF = 1.0	INITIAL	138
	ELSEIF(MANUVR.EQ.'RTO' .OR.MANUVR.EQ.'TOLRTO') THEN	INITIAL	139
C****	INITIALIZE FLAGS FOR THE REFUSED TAKEOFF MANEUVERS.	INITIAL	140
	AOA0FLG = .TRUE.	INITIAL	141
	GEFLAG = .TRUE.	INITIAL	142
	RTOFLAG = .TRUE.	INITIAL	143
	LIFTOFF = .FALSE.	INITIAL	144
	ROTATE = .FALSE.	INITIAL	145
	IF(ENGGRP.EQ.'NUL') ENGGRP = 'AEO'	INITIAL	146
C****		INITIAL	147
C****	RESET SOME INPUT PARAMETERS FROM ANY PREVIOUS RUN (IF ANY).	INITIAL	148
C****	PARAMETERS NOT RESET HERE RETAIN THEIR VALUE FOR THE	INITIAL	149
C****	SUBSEQUENT RUN IF NOT EXPLICITLY DECLARED IN NAMELIST TKO.	INITIAL	150
	TKOTYPE = 'STATIC'	INITIAL	151
	NEQ = 2	INITIAL	152
	ALPHA = ZERO	INITIAL	153
	DIST = ZERO	INITIAL	154
	HAGL = ZERO	INITIAL	155
	PRESALT = HRUNWAY	INITIAL	156
	THETAF = ZERO	INITIAL	157
	TIME = ZERO	INITIAL	158
	VKFAIL = ZERO	INITIAL	159
	VKROTAT = ZERO	INITIAL	160
	VKSTART = ZERO	INITIAL	161
	XLF = 1.0	INITIAL	162

	ELSEIF(MANUVR.EQ.'LND' .OR.MANUVR.EQ.TOLLND) THEN	INITIAL	163
C****	INITIALIZE FLAGS FOR THE LANDING MANEUVER.	INITIAL	164
	AOA0FLG = .FALSE.	INITIAL	165
	GEFLAG = .FALSE.	INITIAL	166
	RTOFLAG = .FALSE.	INITIAL	167
	LIFTOFF = .TRUE.	INITIAL	168
	ROTATE = .TRUE.	INITIAL	169
	IF(ENGGRP.EQ.'NUL') ENGGRP = 'AEI'	INITIAL	170
C****		INITIAL	171
C****	RESET SOME INPUT PARAMETERS FROM ANY PREVIOUS RUN (IF ANY).	INITIAL	172
C****	PARAMETERS NOT RESET HERE RETAIN THEIR VALUE FOR THE	INITIAL	173
C****	SUBSEQUENT RUN IF NOT EXPLICITLY DECLARED IN NAMELIST LND.	INITIAL	174
	NEQ = 4	INITIAL	175
	ALPHA = 99.0	INITIAL	176
	GAMMAPP = 1.0	INITIAL	177
	HFLARE = -1.0	INITIAL	178
	STEADY = .TRUE.	INITIAL	179
	VKAPP = 1.0	INITIAL	180
	WRITITR = .FALSE.	INITIAL	181
C****		INITIAL	182
C****	INITIALIZE ADDITIONAL PARAMETERS.	INITIAL	183
	ALPHMX = THTMAX	INITIAL	184
	HAGL = HCLEAR	INITIAL	185
	IF(VKFLAP.EQ.ZERO) VKFLAP = VKFLPMX	INITIAL	186
	ENDIF	INITIAL	187
	CALL SPDBRAK ('RESET',SPDBRK,SPDBRK0,ZERO)	INITIAL	188
	CALL SPOIL ('RESET',SPOILER,ZERO ,ZERO)	INITIAL	189
	ENDIF	INITIAL	190
	RETURN	INITIAL	191
	END	INITIAL	192

8Subroutine FRETRAC

SUBROUTINE FRETRAC(FLAP,VKFLAP,DFLAPDT)	FRETRAC	1
C**** THIS SUBROUTINE CONTROLS RETRACTION OF THE FLAPS BASED UPON FLAP	FRETRAC	2
C**** RETRACTION SPEED (VKFLAP) AND THE DELTA TIME FOR FLAP RETRACTION	FRETRAC	3
C**** (DFLAPDT).	FRETRAC	4
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
COMMON/FLAPDAT/ IFLAP,MAXSIZ,FLPARY(5),VFLPARY(5)	FLAPDAT	1
DOUBLE PRECISION TIMEJ	FRETRAC	8
LOGICAL INIFLAP	FRETRAC	9
IF(.NOT.INIFLAP) THEN	FRETRAC	10
C**** INITIATE FLAP RETRACTION.	FRETRAC	11
INIFLAP = .TRUE.	FRETRAC	12
LINENUM = LINENUM + 1	FRETRAC	13
IF(IFLAP .LE.0) IFLAP = 1	FRETRAC	14
IF(DFLAPDT .EQ.0.0) DFLAPDT = 1.0	FRETRAC	15
IF(IFLAP.LE.MAXSIZ) THEN	FRETRAC	16
FLAPNEW = FLPARY(IFLAP)	FRETRAC	17
ELSE	FRETRAC	18
FLAPNEW = 0.0	FRETRAC	19
ENDIF	FRETRAC	20
TIMEJ = TIME	FRETRAC	21
TIMEFLP = (FLAP - FLAPNEW)/DFLAPDT	FRETRAC	22
WRITE(LUOUT,1001) FLAPNEW,TIMEFLP	FRETRAC	23
1001 FORMAT(' FLAPS RETRACTED TO 'F4.1,' DEGREES IN '	FRETRAC	24
& F4.1,' SECONDS.)	FRETRAC	25
ENDIF	FRETRAC	26
IF(INIFLAP) THEN	FRETRAC	27
FLAP = FLAP - DFLAPDT*FLOAT(TIME - TIMEJ)	FRETRAC	28
IF(FLAP.LE.FLAPNEW) THEN	FRETRAC	29
FLAP = FLAPNEW	FRETRAC	30
IF(IFLAP.LE.MAXSIZ) THEN	FRETRAC	31
IFLAP = IFLAP + 1	FRETRAC	32
VKFLAP = VFLPARY(IFLAP)	FRETRAC	33
ENDIF	FRETRAC	34
INIFLAP = .FALSE.	FRETRAC	35
IF(FLAP.LE.0.0) FLAPFLG = .FALSE.	FRETRAC	36
ENDIF	FRETRAC	37
ENDIF	FRETRAC	38
TIMEJ = TIME	FRETRAC	39
RETURN	FRETRAC	40
END	FRETRAC	41

Subroutine GRETRAC

SUBROUTINE GRETRAC(DCDLGR,DTGEAR,HAGL)	GRETRAC	1
C**** THIS SUBROUTINE CONTROLS RETRACTION OF THE LANDING GEAR AND	GRETRAC	2
C**** RETURNS THE DELTA DRAG COEFFICIENT OF THE LANDING GEAR (DCDLGR)	GRETRAC	3
C**** BASED UPON THE DELTA TIME TO RETRACT THE GEAR (DTGEAR) AND ELAPSED	GRETRAC	4
C**** TIME FROM GEAR RETRACTION.	GRETRAC	5
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTFL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
REAL LGRARY(6)	GEARDAT	1
COMMON/GEARDAT/ IGEAR,MAXSIZG,LGRARY	GEARDAT	2
LOGICAL INIGEAR	GRETRAC	19
IF(.NOT.INIGEAR) THEN	GRETRAC	10
C**** INITIATE GEAR RETRACTION.	GRETRAC	11
INIGEAR = .TRUE.	GRETRAC	12
LINENUM = LINENUM + 1	GRETRAC	13
IF(IGEAR.LE.0) IGEAR = 1	GRETRAC	14
TGEARUP = FLOAT(TIME) + DTGEAR	GRETRAC	15
WRITE(LUOUT,1002) TIME,HAGL,TGEARUP	GRETRAC	16
1002 FORMAT(' GEAR RETRACTION (TIME = 'F6.2,' HAGL = 'F8.1,	GRETRAC	17
& ' COMPLETE AT 'F6.2,' SECONDS)')	GRETRAC	18
ENDIF	GRETRAC	19
IF(INIGEAR) THEN	GRETRAC	20
DGTIME = TGEARUP - FLOAT(TIME)	GRETRAC	21
IF(LGRARY(1).EQ.0.000) THEN	GRETRAC	22
C**** GEAR DRAG INCREMENT REDUCED LINEARLY WITH TIME IN DTGEAR	GRETRAC	23
C**** SECONDS.	GRETRAC	24
GRFACTR = DGTIME/DTGEAR	GRETRAC	25
DCDLGR = DCDLGR*GRFACTR	GRETRAC	26
IF(DCDLGR.LE.0.0000) THEN	GRETRAC	27
C**** RESET VARIABLES AND FLAGS.	GRETRAC	28
DCDLGR = 0.0000	GRETRAC	29
INIGEAR = .FALSE.	GRETRAC	30
LGRFLAG = .FALSE.	GRETRAC	31
ENDIF	GRETRAC	32

	ELSE	GRETRAC	33
C****	GEAR DRAG INCREMENT OBTAINED FROM USER INPUT ARRAY, LGRARY.	GRETRAC	34
C****	EACH ELEMENT IN THE ARRAY IS ASSIGNED TO DCDLGR FOR ONE	GRETRAC	35
C****	SECOND UNTIL DCDLGR EQUALS ZERO OR IGEAR EXCEEDS MAXSIZG.	GRETRAC	36
	IGEAR = INT(DTGEAR - DGTIME) + 1	GRETRAC	37
	IF(IGEAR.LE.MAXSIZG) THEN	GRETRAC	38
	DCDLGR = LGRARY(IGEAR)	GRETRAC	39
	ELSE	GRETRAC	40
C****	RESET VARIABLES AND FLAGS.	GRETRAC	41
	DCDLGR = 0.0000	GRETRAC	42
	INIGEAR = .FALSE.	GRETRAC	43
	LGRFLAG = .FALSE.	GRETRAC	44
	ENDIF	GRETRAC	45
	ENDIF	GRETRAC	46
	RETURN	GRETRAC	47
	END	GRETRAC	48
		GRETRAC	49

Subroutine PITCH

SUBROUTINE PITCH(MANUVR,ALPHA,DADTCMD,DTIME,DTDTGEX,LUOUT)	PITCH	1
C**** THIS SUBROUTINE MODULATES ANGLE OF ATTACK TO MATCH AN INPUT PITCH	PITCH	2
C**** ATTITUDE OR CLIMB SPEED IF LIMIT CONDITIONS ARE MET. THE RATE AT	PITCH	3
C**** WHICH ANGLE OF ATTACK IS INCREASED IS BASED ON LOAD FACTOR (XLF)	PITCH	4
C**** AND AN INPUT FIRST DERIVATIVE OF ALPHA WITH RESPECT TO TIME	PITCH	5
C**** (DADTCMD).	PITCH	6
C****	PITCH	7
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AIRBORN/ ALPHAJ,ALPHMX,DTDT,GAMMAPP,ROCFPM,THETAF,XLF,XLFJ	AIRBORN	1
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITTR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	8
CHARACTER MANUVR*7	PITCH	14
DOUBLE PRECISION DTIME	PITCH	15
IF(DTDTGEX.LE.0.0) DTDTGEX = 1.0	PITCH	16
IF (DADTCMD.LT.0.0) THEN	PITCH	17
C**** REDUCE ANGLE OF ATTACK - DADTCMD NEGATIVE.	PITCH	18
IF (MANUVR.EQ.'CLIMB') THEN	PITCH	19
IF(XLF.GE.0.85) THEN	PITCH	20
RFACTOR = 0.50*DTDTGEX	PITCH	21
ELSE	PITCH	22
RFACTOR = 0.25*DTDTGEX	PITCH	23
ENDIF	PITCH	24
ELSEIF(MANUVR.EQ.'LANDING') THEN	PITCH	25
RFACTOR = DTDTGEX	PITCH	26
ELSEIF(MANUVR.EQ.'ROLL') THEN	PITCH	27
RFACTOR = DTDTGEX*2.0	PITCH	28
ELSEIF(MANUVR.EQ.'ROTATE') THEN	PITCH	29
RFACTOR = 1.0	PITCH	30
ELSE	PITCH	31
RFACTOR = 1.0	PITCH	32
ENDIF	PITCH	33

ELSEIF(DADTCMD.GT.0.0) THEN	PITCH	34
C**** INCREASE ANGLE OF ATTACK - DADTCMD POSITIVE.	PITCH	35
IF (MANUVR.EQ.'CLIMB ') THEN	PITCH	36
IF(THETA.F.LE.THTMAX) THEN	PITCH	37
IF (XLF.GE.0.9) THEN	PITCH	38
RFACTOR = DTDGEX	PITCH	39
ELSEIF(XLF.LT.0.9 .AND. XLF.GE.0.8) THEN	PITCH	40
RFACTOR = DTDGEX*2.0	PITCH	41
ELSE	PITCH	42
RFACTOR = DTDGEX*3.0	PITCH	43
ENDIF	PITCH	44
ELSE	PITCH	45
TERMFLG = .TRUE.	PITCH	46
RFACTOR = 0.0	PITCH	47
WRITE(LUOUT,1003) THETA.F,THTMAX,XLF	PITCH	48
ENDIF	PITCH	49
1003 FORMAT(' THE PROGRAM HAS ATTEMPTED TO SIMULATE A CLIMB',	PITCH	50
& ' OR PULLUP THAT IS BEYOND THE INPUT LIMITS OF ',	PITCH	51
& ' MAXIMUM PITCH ATTITUDE.',	PITCH	52
& ' THETA, MAX THETA, LOAD FACTOR =', F10.2,	PITCH	53
& ' DEGREES',F10.2,' DEGREES',F10.2,' G')	PITCH	54
ELSEIF(MANUVR.EQ.'LANDING') THEN	PITCH	55
IF(XLF.GE.1.0) THEN	PITCH	56
RFACTOR = DTDGEX	PITCH	57
ELSE	PITCH	58
RFACTOR = DTDGEX*2.0	PITCH	59
ENDIF	PITCH	60
ELSEIF(MANUVR.EQ.'ROLL ') THEN	PITCH	61
RFACTOR = DTDGEX	PITCH	62
ELSEIF(MANUVR.EQ.'ROTATE ') THEN	PITCH	63
RFACTOR = 1.0	PITCH	64
ELSE	PITCH	65
RFACTOR = 1.0	PITCH	66
ENDIF	PITCH	67
ELSE	PITCH	68
TERMFLG = .TRUE.	PITCH	69
WRITE(LUOUT,1004)	PITCH	70
1004 FORMAT(' NO PITCH MODULATION IS POSSIBLE. DADTCMD IS SET',	PITCH	71
& ' TO ZERO.')	PITCH	72
ENDIF	PITCH	73
ALPHA = ALPHA + DADTCMD*FLOAT(DTIME)*RFACTOR	PITCH	74
GAMMA = GAMMAR*RX	PITCH	75
THETA.F = ALPHA + GAMMA	PITCH	76
IF (THETA.F.GT.THTMAX .AND. (LIFTOFF)) THEN	PITCH	77
THETA.F = THTMAX	PITCH	78
GAMMA = THTMAX - ALPHA	PITCH	79
GAMMAR = GAMMA/RX	PITCH	80
ELSEIF (THETA.F.GT.THTMAX .AND. (.NOT.LIFTOFF)) THEN	PITCH	81
THETA.F = THTMAX	PITCH	82
ALPHA = THTMAX	PITCH	83
GAMMAR = 0.0	PITCH	84
ENDIF	PITCH	85
RETURN	PITCH	86
END	PITCH	87

Subroutine SPDBRAK

SUBROUTINE SPDBRAK(ACTION,SPDBRK,SBKEND,DSBKDT)	SPDBRAK	1
C**** THIS SUBROUTINE CONTROLS THE SPEED BRAKES BASED UPON THE CHARACTER	SPDBRAK	2
C**** VARIABLE (ACTION) AND THE TIME RATE OF CHANGE OF SPEED BRAKE ANGLE	SPDBRAK	3
C**** (DSBKDT). THE VALID VALUES FOR ACTION ARE: 'RESET '	SPDBRAK	4
C**** 'DEPLOY', AND 'RETRACT'.	SPDBRAK	5
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
CHARACTER ACTION*	SPDBRAK	8
DOUBLE PRECISION TIMEJ	SPDBRAK	9
LOGICAL INISBK	SPDBRAK	10
IF (ACTION.EQ.'RESET ') THEN	SPDBRAK	11
INISBK = .FALSE.	SPDBRAK	12
SBKFLAG = .FALSE.	SPDBRAK	13
SPDBRK = SBKEND	SPDBRAK	14
RETURN	SPDBRAK	15
ELSEIF (.NOT.INISBK) THEN	SPDBRAK	16
C**** INITIATE SPEED BRAKE DEPLOYMENT OR RETRACTION.	SPDBRAK	17
INISBK = .TRUE.	SPDBRAK	18
LINENUM = LINENUM + 1	SPDBRAK	19
IF(DSBKDT.EQ.0.0) DSBKDT = 90.0	SPDBRAK	20
TIMEJ = TIME	SPDBRAK	21
TIMESBK = (SBKEND - SPDBRK)/DSBKDT	SPDBRAK	22
IF (ACTION.EQ.'DEPLOY') THEN	SPDBRAK	23
WRITE(LUOUT,1005) SBKEND,TIMESBK	SPDBRAK	24
ELSEIF(ACTION.EQ.'RETRACT') THEN	SPDBRAK	25
WRITE(LUOUT,1006) SBKEND,TIMESBK	SPDBRAK	26
ENDIF	SPDBRAK	27
1005 FORMAT(' SPEED BRAKES DEPLOYED TO',F5.1,' DEGREES IN',F5.1,	SPDBRAK	28
& ' SECONDS.')	SPDBRAK	29
1006 FORMAT(' SPEED BRAKES RETRACTED TO',F5.1,' DEGREES IN',F5.1,	SPDBRAK	30
& ' SECONDS.')	SPDBRAK	31
ENDIF	SPDBRAK	32
IF(INISBK) THEN	SPDBRAK	33
IF(ACTION.EQ.'DEPLOY') THEN	SPDBRAK	34
SPDBRK = SPDBRK + DSBKDT*FLOAT(TIME - TIMEJ)	SPDBRAK	35
IF(SPDBRK.GE.SBKEND) THEN	SPDBRAK	36
INISBK = .FALSE.	SPDBRAK	37
SBKFLAG = .TRUE.	SPDBRAK	38
SPDBRK = SBKEND	SPDBRAK	39
ENDIF	SPDBRAK	40

```

ELSE
    SBKFLAG = .FALSE.
    SPDBRK = SPDBRK - DSBKDT*FLOAT(TIME - TIMEJ)
    IF(SBKEND.LE.0.0) SBKEND = 0.0
    IF(SPDBRK.LE.SBKEND) THEN
        IF(SPDBRK.LE.0.0) INISBK = .FALSE.
        SPDBRK = SBKEND
    ENDIF
ENDIF
TIMEJ = TIME
RETURN
END

```

```

SPDBRAK 41
SPDBRAK 42
SPDBRAK 43
SPDBRAK 44
SPDBRAK 45
SPDBRAK 46
SPDBRAK 47
SPDBRAK 48
SPDBRAK 49
SPDBRAK 50
SPDBRAK 51
SPDBRAK 52
SPDBRAK 53

```

Subroutine SPOIL

SUBROUTINE SPOIL(ACTION,SPOILER,SPLREND,DSPLRDT)	SPOIL	1
C**** THIS SUBROUTINE CONTROLS THE SPOILERS BASED UPON THE CHARACTER	SPOIL	2
C**** VARIABLE (ACTION) AND THE TIME RATE OF CHANGE OF SPOILER ANGLE	SPOIL	3
C**** (DSPLRPDT). THE VALID VALUES FOR ACTION ARE: 'RESET '	SPOIL	4
C**** 'DEPLOY ', AND 'RETRACT'.	SPOIL	5
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITTR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	8
CHARACTER ACTION*7	SPOIL	8
DOUBLE PRECISION TIMEJ	SPOIL	9
LOGICAL INISPLR	SPOIL	10
IF (ACTION.EQ.'RESET ') THEN	SPOIL	11
INISPLR = .FALSE.	SPOIL	12
SPLFLAG = .FALSE.	SPOIL	13
SPOILER = SPLREND	SPOIL	14
RETURN	SPOIL	15
ELSEIF (.NOT.INISPLR) THEN	SPOIL	16
C**** INITIATE SPOILER DEPLOYMENT OR RETRACTION.	SPOIL	17
INISPLR = .TRUE.	SPOIL	18
LINENUM = LINENUM + 1	SPOIL	19
IF(DSPLRDT.EQ.0.0) DSPLRDT = 90.0	SPOIL	20
TIMEJ = TIME	SPOIL	21
TIMESPL = (SPLREND - SPOILER)/DSPLRDT	SPOIL	22
IF (ACTION.EQ.'DEPLOY ') THEN	SPOIL	23
WRITE(LUOUT,1005) SPLREND,TIMESPL	SPOIL	24
ELSEIF(ACTION.EQ.'RETRACT') THEN	SPOIL	25
WRITE(LUOUT,1006) SPLREND,TIMESPL	SPOIL	26
ENDIF	SPOIL	27
1005 FORMAT(' SPOILERS DEPLOYED TO',F5.1,' DEGREES IN',F5.1,	SPOIL	28
& ' SECONDS.')	SPOIL	29
1006 FORMAT(' SPOILERS RETRACTED TO',F5.1,' DEGREES IN',F5.1,	SPOIL	30
& ' SECONDS.')	SPOIL	31
ENDIF	SPOIL	32
IF(INISPLR) THEN	SPOIL	33
IF(ACTION.EQ.'DEPLOY ') THEN	SPOIL	34
SPOILER = SPOILER + DSPLRDT*FLOAT(TIME - TIMEJ)	SPOIL	35
IF(SPOILER.GE.SPLREND) THEN	SPOIL	36
INISPLR = .FALSE.	SPOIL	37
SPLFLAG = .TRUE.	SPOIL	38
SPOILER = SPLREND	SPOIL	39
ENDIF	SPOIL	40

ELSE	SPOIL	41
SPLFLAG = .FALSE.	SPOIL	42
SPOILER = SPOILER - DSPLRDT*FLOAT(TIME - TIMEJ)	SPOIL	43
IF(SPLREND.LE.0.0) SPLREND = 0.0	SPOIL	44
IF(SPOILER.LE.SPLREND) THEN	SPOIL	45
IF(SPOILER.LE.ZERO) INISPLR = .FALSE.	SPOIL	46
SPOILER = SPLREND	SPOIL	47
ENDIF	SPOIL	48
ENDIF	SPOIL	49
TIMEJ = TIME	SPOIL	50
RETURN	SPOIL	51
END	SPOIL	52
	SPOIL	53

Subroutine TVECTOR

SUBROUTINE TVECTOR(VTANGLE,HVECT,WVECT,DVECTDT)	TVECTOR	1
C**** THIS SUBROUTINE CONTROLS THE THRUST VECTORING BASED UPON THE	TVECTOR	2
C**** THRUST VECTORING TRIGGER ALTITUDE (HVECT), THE THRUST	TVECTOR	3
C**** VECTORING TRIGGER AIRSPEED (WVECT), AND THE DELTA TIME FOR THRUST	TVECTOR	4
C**** VECTORING (DVECTDT).	TVECTOR	5
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
COMMON/VECTDAT/ IVECT,MAXSIZV,XNUARY(5),HVCTARY(5),VVCTARY(5)	VECTDAT	1
DOUBLE PRECISION TIMEJ	TVECTOR	9
LOGICAL INIVECT	TVECTOR	10
IF(.NOT.INIVECT) THEN	TVECTOR	11
C**** INITIATE THRUST VECTOR CHANGE.	TVECTOR	12
INIVECT = .TRUE.	TVECTOR	13
LINENUM = LINENUM + 1	TVECTOR	14
IF(IVECT .LE.0) IVECT = 1	TVECTOR	15
IF(DVECTDT .EQ.0.0) DVECTDT = 1.0	TVECTOR	16
IF(IVECT.LE.MAXSIZV) THEN	TVECTOR	17
XNUNEW = XNUARY(IVECT)	TVECTOR	18
ELSE	TVECTOR	19
XNUNEW = 0.0	TVECTOR	20
ENDIF	TVECTOR	21
TIMEJ = TIME	TVECTOR	22
TIMEVCT = (XNUNEW - VTANGLE)/DVECTDT	TVECTOR	23
IF(TIMEVCT.LT.0.0) THEN	TVECTOR	24
C**** ADJUST SIGN OF TIMEVCT AND DVECTDT SO THAT TIMEVCT IS ALWAYS	TVECTOR	25
C**** POSITIVE AND DVECTDT IS POSITIVE IF THE VECTORED THRUST	TVECTOR	26
C**** ANGLE IS INCREASING AND DVECTDT IS NEGATIVE IF THE VECTOR	TVECTOR	27
C**** THRUST ANGLE IS DECREASING.	TVECTOR	28
TIMEVCT = -TIMEVCT	TVECTOR	29
DVECTDT = -DVECTDT	TVECTOR	30
ENDIF	TVECTOR	31
WRITE(LUOUT,1007) VTANGLE,XNUNEW,TIMEVCT	TVECTOR	32
1007 FORMAT(' VECTORED THRUST ANGLE CHANGED FROM ',F4.1,' TO ',F4.1,	TVECTOR	33
& ' DEGREES IN',F4.1,' SECONDS.')	TVECTOR	34
ENDIF	TVECTOR	35

IF(INIVECT) THEN	TVECTOR	36
VTANGLE = VTANGLE + DVECTDT*FLOAT(TIME - TIMEJ)	TVECTOR	37
IF(VTANGLE.LE.XNUNEW) THEN	TVECTOR	38
VTANGLE = XNUNEW	TVECTOR	39
IF(IVECT.LE.MAXSIZV) THEN	TVECTOR	40
IVECT = IVECT + 1	TVECTOR	41
HVECT = HVCTARY(IVECT)	TVECTOR	42
VVECT = VVCTARY(IVECT)	TVECTOR	43
ENDIF	TVECTOR	44
INIVECT = .FALSE.	TVECTOR	45
IF(VTANGLE.EQ.0.0) VECTFLG = .FALSE.	TVECTOR	46
ENDIF	TVECTOR	47
ENDIF	TVECTOR	48
TIMEJ = TIME	TVECTOR	49
RETURN	TVECTOR	50
END	TVECTOR	51

Subroutine TAKOFF

SUBROUTINE TAKOFF(ALPHA,GDIST)	TAKOFF	1
C**** THIS SUBROUTINE CONTROLS THE EXECUTION OF THE TAKEOFF MANEUVER	TAKOFF	2
C**** FROM BRAKE RELEASE THROUGH CLIMBOUT. SUBROUTINE TAKEOFF CALLS INTX	TAKOFF	3
C**** TO PERFORM THE NUMERICAL INTEGRATION OF THE RESULTANTS OF THE	TAKOFF	4
C**** EQUATIONS OF MOTION AND CALLS FORCEX TO OBTAIN THE FORCE	TAKOFF	5
C**** COEFFICIENTS FOR THE EQUATIONS OF MOTION.	TAKOFF	6
C****	TAKOFF	7
C**** THE CONTINUED TAKEOFF IS DONE BY ACCELERATING TO SPEED VKFAIL,	TAKOFF	8
C**** LINEARLY DECREASING WITH TIME THE NUMBER OF ENGINES FROM A VALUE	TAKOFF	9
C**** OF XENG TO A VALUE OF XENGOUT, CONTINUING THE GROUND ROLL TO	TAKOFF	10
C**** VKROTAT, ROTATE (IF VKFAIL IS LESS THAN VKROTAT), LIFTOFF AND	TAKOFF	11
C**** CLIMBOUT TO HMAX. VKFAIL MUST BE LESS THAN TAKEOFF SPEED FOR AN	TAKOFF	12
C**** ENGINE FAILURE TO BE INITIATED.	TAKOFF	13
C****	TAKOFF	14
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	15
PARAMETER (LUIN=3,LUOUT=4)	CTRL	16
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	17
& NEQ,NPAGE,TIME,TIMEROL	CTRL	18
COMMON/AIRCFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCFT	19
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCFT	20
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	21
& SPDBRK,SPOILER,VKCAS,VKTAS	AERO	22
COMMON/ENGINE/ AIT,AMACH,DTFAIL,EPR,FE,FG,FGPCT,NENG,PWRCODE,	ENGINE	23
& REVNDX,RC,THRUST,VTANGLE,WFUEL,XENG,XENGFLD,	ENGINE	24
& XENGOUT,XIDLE,XMIL,ZFN	ENGINE	25
COMMON/VECTOR/ HVECT,VVECT	VECTOR	26
COMMON/AIRBORN/ ALPHAJ,ALPHMX,DTDT,GAMMAPP,ROCFPM,THETAJ,XLF,XLFJ	AIRBORN	27
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	28
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	29
COMMON/RUNWAY/ ABARG,AOAABRK,BRAKMU,BRKFCR,GAMMARW,GRW,HAGL,	RUNWAY	30
& HCLEAR,HFLARE,HGEAR,HRUNWAY,IMU,RCR,ROLLMU,TIMEFLD,	RUNWAY	31
& TIMEBRK,TIMEFLP,TIMESBK,TIMESPL,XMU	RUNWAY	32
DOUBLEPRECISION VTAS,DIST,ACCEL,VTASJ,RKGRND	INTEG	33
COMMON/INTEG/ VTAS,DIST,ACCEL,VTASJ,RKGRND(20)	INTEG	34
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	35
& ROC,RKAIR	FPINTEG	36
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	37
& ROC,RKAIR(40)	FPINTEG	38
COMMON/ATMOS/ TEMPR,PRESS,RHO,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMPF	ATMOS	39
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	40
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	41
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	42
LOGICAL AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	43
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	44
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	45
& VFFLAG,WRITTR	FLAGS	46
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	47
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	48
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	49
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	50

CHARACTER ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
& THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/ ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
& TKOTYPE	CHARV	4
COMMON/FLAPDAT/ IFLAP,MAXSIZF,FLPARY(5),VFLPARY(5)	FLAPDAT	1
REAL LGRARY(6)	GEARDAT	1
COMMON/GEARDAT/ IGEAR,MAXSIZG,LGRARY	GEARDAT	2
COMMON/VECTDAT/ IVECT,MAXSIZV,XNUARY(5),HVCTARY(5),VVCTARY(5)	VECTDAT	1
LOGICAL INIVCLM,INISEG1,INISEG2,VCOFLAG	TAKOFF	31
REAL ATMARY(8),S(48),T(24),INTERP	TAKOFF	32
EQUIVALENCE (TEMPR,ATMARY(1)),(FPVTAS,S(1)),(VTAS,T(1))	TAKOFF	33
EXTERNAL DERIVGR,DERIVAT,INTERP	TAKOFF	34
DATA DISTMAX,HCLIMB,HCLMOUT, HGEAR,HMAX, ROLLMAX, THTCLM, THTFLY,	TAKOFF	35
& / 60760., 100., 200., 205., 1000., 120.0, 10.0, 8.0,/,	TAKOFF	36
& THTROT, THTTOL, TIMEMAX,VCLMOUT, VKEND,VKFAIL, VKFLAP VKROTAT	TAKOFF	37
& / 10.0, 0.1, 300., 0., 250., 0., 0., 0.,/	TAKOFF	38
& VKSTART	TAKOFF	39
& / 0/	TAKOFF	40
NAMelist/TKO/HGEAR,HMAX,THTCLM,THTROT,VKFAIL,VKROTAT	TAKOFF	41
NAMelist/TKO2/DISTMAX,HCLIMB,HCLMOUT,ROLLMAX,THTFLY,THTTOL,	TAKOFF	42
& TIMEMAX,TKOTYPE,VCLMOUT,VKEND,VKFLAP,VKSTART	TAKOFF	43
NAMelist/TKOARY/FLPARY,VFLPARY,HVCTARY,VVCTARY,XNUARY	TAKOFF	44
C****	TAKOFF	45
C**** INPUT DATA LOADED INTO TAKOFF THRU NAMELISTS /TKO/ AND /TKOARY/.	TAKOFF	46
READ(LUIN,TKO)	TAKOFF	47
READ(LUIN,TKO2)	TAKOFF	48
READ(LUIN,TKOARY)	TAKOFF	49
C****	TAKOFF	50
C**** TERMINATE PROGRAM IF ROTATION SPEED NOT INITIALIZED.	TAKOFF	51
IF(VKROTAT.EQ.ZERO) CALL HALT(LUIN,LUMSG,LUOUT,	TAKOFF	52
& 'VKROTAT NOT INPUT BY USER.')	TAKOFF	53
C****	TAKOFF	54
C**** INITIALIZE INTERNAL SUBROUTINE LOGIC CONTROL VARIABLES.	TAKOFF	55
INIVCLM = INISEG1 = INISEG2 = LIFTOFF = ROTATE = VCOFLAG = .FALSE.	TAKOFF	56
C****	TAKOFF	57
C**** INITIALIZE LOGIC FLAG BASED ON ENGINE FAILURE SPEED.	TAKOFF	58
IF(VKFAIL.EQ.ZERO) THEN	TAKOFF	59
FAILFLG = .FALSE.	TAKOFF	60
ELSE	TAKOFF	61
FAILFLG = .TRUE.	TAKOFF	62
ENDIF	TAKOFF	63
C****	TAKOFF	64
C**** INITIALIZE CURVE FILES.	TAKOFF	65
CALL INICURV	TAKOFF	66
C****	TAKOFF	67
C**** INITIALIZE INTERNAL SUBROUTINE PARAMETERS.	TAKOFF	68
TEMPF = TSLF + DTEMPF - 0.003566*HRUNWAY	TAKOFF	69
THTMAX = THTROT	TAKOFF	70

C****		TAKOFF	71
C****	OBTAIN ATMOSPHERIC VARIABLES.	TAKOFF	72
	CALL ATMOSPH(HRUNWAY,ATMARY)	TAKOFF	73
C****		TAKOFF	74
C****	SUM VKSTART AND VKWIND TO CALCULATE INITIAL AIRSPEED FOR ROLLING	TAKOFF	75
C****	MINIMUM INTERVAL TAKEOFFS.	TAKOFF	76
	VKTAS = VKWIND + VKSTART	TAKOFF	77
	VTAS = VKTAS*FPSKTS	TAKOFF	78
C****		TAKOFF	79
C****	CALIBRATED AIRSPEED EQUATION	TAKOFF	80
C****	ASLSQR5 = 661.48*SQRT(5.0); TWOOVR7 = 2.0/7.0	TAKOFF	81
C****	VKCAS = ASL*SQRT(5.0*(DELTA*	TAKOFF	82
C****&	((1.0 + 0.2*AMACH**2)**3.5 - 1.0) + 1.0)**(2.0/7.0) - 1.0))	TAKOFF	83
	CALL SPEED(0.0,VTAS,VWIND,AMACH,QS,VKCAS,VKEAS,VKTAS,VKTGS,VTGS)	TAKOFF	84
C****	OBTAIN INITIAL CONDITIONS OF THRUST AND FORCE COEFFICIENTS FOR	TAKOFF	85
C****	GROUND ROLL.	TAKOFF	86
	CALL FORCEX(ALPHA,CD,CL)	TAKOFF	87
C****		TAKOFF	88
C****	OBTAIN STATIC THRUST TO WEIGHT RATIO.	TAKOFF	89
	TOVERW = THRUST*(FLOAT(NENG)/XENG)/GWT	TAKOFF	90
C****		TAKOFF	91
C****	INITIALIZE THRUST ANGLE SCHEDULES FOR AIRBORNE PORTION OF	TAKOFF	92
C****	TAKEOFF.	TAKOFF	93
	HVECT = HVCTARY(1)	TAKOFF	94
	VVECT = VVCTARY(1)	TAKOFF	95
	VTANGLE = XNUARY(1)	TAKOFF	96
C****		TAKOFF	97
C****	REINITIALIZE THE VVCTARY ARRAY ELEMENTS FOR ANY CORRESPONDING	TAKOFF	98
C****	ELEMENTS IN THE HVCTARY ARRAY. THIS GIVES ALTITUDE BREAKPOINTS	TAKOFF	99
C****	PRECEDENCE OVER AIRSPEED BREAKPOINTS FOR THRUST VECTORING	TAKOFF	100
C****	MODULATION.	TAKOFF	101
	DO 10 I = 2,MAXSIZV	TAKOFF	102
	IF(HVCTARY(I).NE.ZERO) VVCTARY(I) = ZERO	TAKOFF	103
	10 CONTINUE	TAKOFF	104
C****		TAKOFF	105
C****	INITIALIZE FLAPS FOR GROUND ROLL.	TAKOFF	106
	IF (VFLPARY(1) .NE.ZERO) THEN	TAKOFF	107
	VKFLAP = VFLPARY(1)	TAKOFF	108
	ELSEIF(VKFLAP .EQ.ZERO) THEN	TAKOFF	109
	VKFLAP = VKFLPMX	TAKOFF	110
	ENDIF	TAKOFF	111

C****		TAKOFF	112
C****	ECHO BACK PROGRAM INPUTS TO OUTPUT FILE.	TAKOFF	113
	IF(VKWIND.GE.ZERO) THEN	TAKOFF	114
	WRITE(LUOUT,1001) HRUNWAY,GAMMARW,TEMPF, VKWIND,'HEADWIND'	TAKOFF	115
	ELSE	TAKOFF	116
	WRITE(LUOUT,1001) HRUNWAY,GAMMARW,TEMPF,ABS(VKWIND),'TAILWIND'	TAKOFF	117
	ENDIF	TAKOFF	118
	WRITE(LUOUT,1002) GWT,THTR0T,SWING,THTC LM,GWT*TOVERW/DELTA,THTF LY,	TAKOFF	119
&	WNGLOD,VCLMOUT,TOVERW,HGEAR,XENG,DADTCMD,DTFAIL	TAKOFF	120
	WRITE(LUOUT,1003)	TAKOFF	121
&	(XNUARY (I),I = 1,MAXSIZV),	TAKOFF	122
&	(VVCTARY (I),I = 1,MAXSIZV),(HVCTARY(I),I = 1,MAXSIZV),	TAKOFF	123
&	(FLPARY (I),I = 1,MAXSIZF),(VFLPARY(I),I = 1,MAXSIZF),	TAKOFF	124
&	HRUNWAY	TAKOFF	125
	WRITE(LUOUT,1004)	TAKOFF	126
1001	FORMAT('*** INPUTS TO TAKE OFF ***', ALTITUDE =',F7.1,	TAKOFF	127
&	ALTITUDE =',F7.1,19X,' RUNWAY SLOPE =',F4.1,' DEGREES	TAKOFF	128
&	'DEGREES',,	TAKOFF	129
&	'TEMPERATURE =',F7.1,' DEGREES F',,	TAKOFF	130
&	',F7.1,' KNOT ',A8)	TAKOFF	131
1002	FORMAT(/,' AIRCRAFT PARAMETERS',20X,	TAKOFF	132
&	'FLIGHTPATH CONTROL PARAMETERS',,	TAKOFF	133
&	/, ' GROSS RAMP WEIGHT = ',F9.0,2X,	TAKOFF	134
&	' ROTATION PITCH ANGLE = ',F6.1,	TAKOFF	135
&	/, ' WING AREA = = ',F9.0,2X,	TAKOFF	136
&	' SEGMENT I PITCH ANGLE = ',F6.1,	TAKOFF	137
&	/, ' STATIC SEA LEVEL THRUST = ',F9.0,2X,	TAKOFF	138
&	' SEGMENT II PITCH ANGLE = ',F6.1,	TAKOFF	139
&	/, ' WING LOADING = ',F9.1,2X,	TAKOFF	140
&	' CLIMB OUT AIRSPEED = ',F6.1,	TAKOFF	141
&	/, ' THRUST/WEIGHT = ',F9.3,2X,	TAKOFF	142
&	' GEAR RETRACTION ALTITUDE = ',F6.1,	TAKOFF	143
&	/, ' NUMBER OF ENGINES = ',F9.1,2X,	TAKOFF	144
&	' COMMANDED ALPHA RATE = ',F6.1,	TAKOFF	145
&	/, ',11X,	TAKOFF	146
&	' ELAPSED ENGINE FAILURE TIME = ',F6.1)	TAKOFF	147
1003	FORMAT(/,' FLAP AND VECTORED THRUST SCHEDULES',,	TAKOFF	148
&	9X,'FLAP RETRACTION SCHEDULE',,	TAKOFF	149
&	12X,'FLAP DEFLECTION',5F9.1/,	TAKOFF	150
&	12X,'SPEED ',5F9.1/,	TAKOFF	151
&	9X,'VECTORED THRUST ANGLE',12X,'ANGLE ',5F9.1/,	TAKOFF	152
&	12X,'SPEED ',5F9.1/,12X,'ALTITUDE',5F9.0/,	TAKOFF	153
&	9X,'ALL SPEEDS ARE CALIBRATED AIR SPEEDS AND ALL ',	TAKOFF	154
&	'ALTITUDES ARE ABSOLUTE ALTITUDES',,	TAKOFF	155
&	'↑ TAKEOFF (ELEVATION =',F6.0,' FEET')/,)	TAKOFF	156
1004	FORMAT(' TIME DIST. WEIGHT ALT. VCAS VTAS VTGS',	TAKOFF	157
&	' ACCEL CL CD THETA ALPHA GAMMA DTHET ',	TAKOFF	158
&	' R/C LOAD THRUST XENG',,	TAKOFF	159
&	'(SEC) (FEET) (LBS) (FEET) (KTS) (KTS) (KTS)',	TAKOFF	160
&	'(FPS2) (DEG) (DEG) (DEG) /DT ',	TAKOFF	161
&	'(FPM) FACT (LBS) OR MU',)	TAKOFF	162

C****		TAKOFF	163
C****	GROUND ROLL INTEGRATION VARIABLES.	TAKOFF	164
C****	NEQ = NUMBER OF EQUATIONS	TAKOFF	165
C****	DTIME = TIME INTERVAL, STEP SIZE (SECONDS)	TAKOFF	166
C****	VTAS = VELOCITY (FEET/SECOND) WITH RESPECT TO THE AIR	TAKOFF	167
C****	DIST = DISTANCE (FEET)	TAKOFF	168
C****	ACCEL = ACCELERATION (FEET/SECOND**2)	TAKOFF	169
	GDIST = ZERO	TAKOFF	170
	CALL INTX(NEQ,TIME,DTIME,T,DERIVGR,ALPHA)	TAKOFF	171
	IF(DTIME.GE.0.10D0) THEN	TAKOFF	172
	WRITE(LUOUT,1005) TIME,GDIST,GWT,HAGL,VKCAS,VKTAS,THETAF,ACCEL,	TAKOFF	173
&	CL,CD,ZERO,ALPHA,ZERO,ZERO,ZERO,XLF,THRUST,	TAKOFF	174
&	XENG	TAKOFF	175
	ELSE	TAKOFF	176
	WRITE(LUOUT,1006) TIME,GDIST,GWT,HAGL,VKCAS,VKTAS,THETAF,ACCEL,	TAKOFF	177
&	CL,CD,ZERO,ALPHA,ZERO,ZERO,ZERO,XLF,THRUST,	TAKOFF	178
&	XENG	TAKOFF	179
	ENDIF	TAKOFF	180
1005	FORMAT(F6.1,F8.0,F9.0,F8.1,3F7.1,F7.2,F8.4,F7.4,4F7.2,F8.1,F6.2,	TAKOFF	181
&	F9.0,F7.3)	TAKOFF	182
1006	FORMAT(F6.2,F8.0,F9.0,F8.1,3F7.1,F7.2,F8.4,F7.4,4F7.2,F8.1,F6.2,	TAKOFF	183
&	F9.0,F7.3)	TAKOFF	184
C****		TAKOFF	185
C****	GROUND ROLL INTEGRATION LOOP	TAKOFF	186
	19 DO 20 NCOUNT=1,10	TAKOFF	187
	ALPHAJ = ALPHA	TAKOFF	188
	THETAJ = THETAF	TAKOFF	189
C****		TAKOFF	190
C****	LIFTOFF CRITERION; IF LIFT IS LESS THAN WEIGHT	TAKOFF	191
	IF (QS*CY.LT.GWT*COSD(GAMMARW)) THEN	TAKOFF	192
C****	LIFTOFF CRITERION NOT MET	TAKOFF	193
C****		TAKOFF	194
C****	MAKE INTEGRATION STEP.	TAKOFF	195
	CALL INTZ(NEQ,TIME,DTIME,T,DERIVGR,ALPHA)	TAKOFF	196
	CALL INTG(DIST,RKGRND(14),DTIME,0.0,VWIND,WFUEL,	TAKOFF	197
	DDIST,GDIST,GWT)	TAKOFF	198

C****		TAKOFF	199
C****	CALCULATE DYNAMIC PRESSURE, MACH NUMBER AND VELOCITIES.	TAKOFF	200
	CALL SPEED(0.0,VTAS,VWIND,AMACH,QS,VKCAS,VKEAS,VKTAS,	TAKOFF	201
	VKTGS,VTGS)	TAKOFF	202
C****		TAKOFF	203
C****	CHECK SPEED FOR VKFAIL.	TAKOFF	204
	IF(VKCAS.GE.VKFAIL .AND. (.NOT.VFFLAG)) THEN	TAKOFF	205
C****	ENGINE FAILURE SPEED ATTAINED.	TAKOFF	206
	SPOOL = .TRUE.	TAKOFF	207
	VFFLAG = .TRUE.	TAKOFF	208
	TIMEFLD= FLOAT(TIME - DTIME)	TAKOFF	209
	IF(.NOT.RTOFLAG) THEN	TAKOFF	210
	XENGTRN = 1.0	TAKOFF	211
	IF (FAILST.EQ.'IDLE') THEN	TAKOFF	212
	XENGEND = XIDLE	TAKOFF	213
	ELSEIF (FAILST.EQ.' MIL') THEN	TAKOFF	214
	XENGEND = XMIL	TAKOFF	215
	ELSEIF (FAILST.EQ.' OFF') THEN	TAKOFF	216
	XENGEND = ZERO	TAKOFF	217
	ELSE	TAKOFF	218
	XENGEND = 1.0 -XENGFLD	TAKOFF	219
	ENDIF	TAKOFF	220
	XENGFLD = 1.0	TAKOFF	221
	ENDIF	TAKOFF	222
	IF(FAILFLG) THEN	TAKOFF	223
C****	ENGINE FAILURE OCCURS.	TAKOFF	224
C****	PARAMETER ENGGRP SWITCHED TO FAILGRP FOR	TAKOFF	225
C****	CONTINUED OR REFUSED PORTION OF TAKEOFF.	TAKOFF	226
	LINENUM = LINENUM + 1	TAKOFF	227
	ENGGRP = FAILGRP	TAKOFF	228
	CALL INICURV	TAKOFF	229
	WRITE(LUOUT,1007) TIME,GDIST,VKCAS,VKTAS	TAKOFF	230
1007	FORMAT(' ENGINE FAILURE (TIME = ',F6.2,	TAKOFF	231
&	' DIST = ',F8.1,' CAS = ',F7.1,	TAKOFF	232
&	' TAS = ',F7.1,' KNOTS)')	TAKOFF	233
	ENDIF	TAKOFF	234
	ENDIF	TAKOFF	235
C****		TAKOFF	236
C****	ENGINE FAILURE CONTROL	TAKOFF	237
	IF(ENGGRP.EQ.FAILGRP) THEN	TAKOFF	238
	IF (RTOFLAG)THEN	TAKOFF	239
C****	REFUSED TAKEOFF	TAKOFF	240
C****	THIS FLAG RETURNS EXECUTION TO THE CALLING PROGRAM	TAKOFF	241
C****	AT START OF THE ENGINE FAILURE. PROGRAM	TAKOFF	242
C****	EXECUTION WILL PROCEED WITH SUBROUTINE ROLL.	TAKOFF	243
	ICOUNT = NCOUNT + 1	TAKOFF	244
	RETURN	TAKOFF	245

	ELSEIF (FAILMOD.EQ.'SEIZE') THEN	TAKOFF	246
	SPOOL = .TRUE.	TAKOFF	247
	XENG = XENGOUT	TAKOFF	248
	ELSE	TAKOFF	249
	IF (DTFAIL.EQ.ZERO .AND. (.NOT. SPOOL)) THEN	TAKOFF	250
	CALL SPOOLDNF(TIME,XENGEND,XENGTRN,SPOOL,XENGF,	TAKOFF	251
	LUMSG)	TAKOFF	252
	XENG = XENGOUT + XENGF	TAKOFF	253
	ELSEIF(DTFAIL.NE.ZERO .AND. (.NOT. SPOOL)) THEN	TAKOFF	254
	XENG = XENG - XENGFLD*FLOAT(DTIME)/DTFAIL	TAKOFF	255
	IF(XENG.LE.XENGOUT) THEN	TAKOFF	256
	SPOOL = .TRUE.	TAKOFF	257
	ENDIF	TAKOFF	258
	ELSE	TAKOFF	259
	XENG = XENGOUT	TAKOFF	260
	ENDIF	TAKOFF	261
	ENDIF	TAKOFF	262
	ENDIF	TAKOFF	263
C****	CHECK SPEED FOR VKROTAT.	TAKOFF	264
	IF(VKCAS.GE.VKROTAT) THEN	TAKOFF	265
	IF(.NOT.ROTATE) THEN	TAKOFF	266
C****	BEGIN ROTATION	TAKOFF	267
	ROTATE = .TRUE.	TAKOFF	268
	AOA0FLG = .FALSE.	TAKOFF	269
	LINENUM = LINENUM + 1	TAKOFF	270
	WRITE(LUOUT,1008) TIME,GDIST,VKCAS,VKTGS	TAKOFF	271
1008	FORMAT(' ROTATION (TIME = ',F6.2,	TAKOFF	272
&	' DIST = ',F8.1,' CAS = ',F7.1,	TAKOFF	273
&	' VG = ',F7.1,' KNOTS)')	TAKOFF	274
	ENDIF	TAKOFF	275
C****		TAKOFF	276
C****	INCREASE ALPHA TO TAIL SCRAPE ANGLE AT RATE DADTCMD.	TAKOFF	277
	CALL PITCH('ROTATE ',ALPHA,DADTCMD,DTIME,DTDTGEX,LUOUT)	TAKOFF	278
	ENDIF	TAKOFF	279
C****		TAKOFF	280
C****	CHECK TIME FOR LIMIT OF GROUND ROLL TIME.	TAKOFF	281
	IF(FLOAT(TIME).GT.ROLLMAX) THEN	TAKOFF	282
	WRITE(LUOUT,1009)	TAKOFF	283
1009	FORMAT('/',' TIME LIMIT FOR GROUND RUN EXCEEDED.)	TAKOFF	284
	RETURN	TAKOFF	285
	ENDIF	TAKOFF	286

	IF(NCOUNT.EQ.10) THEN	TAKOFF	287
C****	CALCULATE ADDITIONAL OUTPUT PARAMETERS, WRITE OUTPUT AND	TAKOFF	288
C****	INCREMENT LINE NUMBER COUNTER LINENUM.	TAKOFF	289
C****		TAKOFF	290
C****	CALCULATE LOCAL(BASED ON THE CURRENT AND PREVIOUS TIME	TAKOFF	291
C****	POINTS) RATES OF CHANGE IN ANGLE OF ATTACK AND PITCH	TAKOFF	292
C****	ATTITUDE. THESE ARE NOT THE COMMANDED RATES OF CHANGE	TAKOFF	293
C****	IN ANGLE OF ATTACK AND PITCH ATTITUDE.	TAKOFF	294
	DADT = (ALPHA - ALPHAJ)/FLOAT(DTIME)	TAKOFF	295
	DTHDT = (THETAJ - THETAJ)/FLOAT(DTIME)	TAKOFF	296
	CALL FORCEX(ALPHA,CD,CL)	TAKOFF	297
	IF(DTIME.GE.0.10D0)THEN	TAKOFF	298
	WRITE(LUOUT,1005) TIME,GDIST,GWT,HAGL,VKCAS,VKTAS,	TAKOFF	299
&	VKTGS,ACCEL,CL,CD,THETAJ,ALPHA,ZERO,	TAKOFF	300
&	DTHDT,ZERO,XLF,THRUST,XENG	TAKOFF	301
	ELSE	TAKOFF	302
	WRITE(LUOUT,1006) TIME,GDIST,GWT,HAGL,VKCAS,VKTAS,	TAKOFF	303
&	VKTGS,ACCEL,CL,CD,THETAJ,ALPHA,ZERO,	TAKOFF	304
&	DTHDT,ZERO,XLF,THRUST,XENG	TAKOFF	305
	ENDIF	TAKOFF	306
	LINENUM = LINENUM + 1	TAKOFF	307
	IF(LINENUM.GE.NPAGE) THEN	TAKOFF	308
C****	RESET LINE NUMBER COUNTER LINENUM AND WRITE HEADER	TAKOFF	309
C****	FOR NEW PAGE.	TAKOFF	310
	LINENUM = 5	TAKOFF	311
	WRITE(LUOUT,1010)	TAKOFF	312
	WRITE(LUOUT,1004)	TAKOFF	313
1010	FORMAT('I TAKEOFF CONTINUED',/)	TAKOFF	314
	ENDIF	TAKOFF	315
C****	RESTART GROUND ROLL INTEGRATION LOOP.	TAKOFF	316
	GO TO 19	TAKOFF	317
	ENDIF	TAKOFF	318
	ELSEIF(.NOT.LIFTOFF) THEN	TAKOFF	319
C****	LIFTOFF CRITERION MET. INITIALIZE ICOUNT FOR AIRBORNE	TAKOFF	320
C****	INTEGRATION LOOP. THE VARIABLE ICOUNT, ALLOWS SUBROUTINE	TAKOFF	321
C****	TAKOFF TO WRITE OUTPUT AT DTIME*10.0 SECOND INTERVALS AS	TAKOFF	322
C****	PROGRAM EXECUTION PROCEEDS FROM THE GROUND ROLL INTEGRATION	TAKOFF	323
C****	LOOP TO THE AIRBORNE INTEGRATION LOOP OR TO THE GROUND ROLL	TAKOFF	324
C****	INTEGRATION LOOP IN SUBROUTINE ROLL FOR REFUSED TAKEOFFS.	TAKOFF	325
	ICOUNT = NCOUNT	TAKOFF	326
	LIFTOFF = .TRUE.	TAKOFF	327
	ENDIF	TAKOFF	328
20	CONTINUE	TAKOFF	329

C****		TAKOFF	330
C****	OUTPUT VARIABLES AT LTOFF.	TAKOFF	331
	LINENUM = LINENUM + 1	TAKOFF	332
	WRITE(LUOUT,1011) TIME,GDIST,VKCAS,VKTGS	TAKOFF	333
1011	FORMAT(' LTOFF (TIME = 'F8.2,' DIST = 'F8.1,	TAKOFF	334
	& ' CAS = 'F7.1,' VG = 'F7.1,' KNOTS))	TAKOFF	335
C****		TAKOFF	336
C****	AIRBORNE PORTION OF TAKEOFF	TAKOFF	337
C****		TAKOFF	338
C****	INITIALIZE THRUST ANGLE SCHEDULES FOR AIRBORNE PORTION OF TAKEOFF.	TAKOFF	339
	HVECT = HVCTARY(1)	TAKOFF	340
	WVECT = WVCTARY(1)	TAKOFF	341
C****		TAKOFF	342
C****	INITIALIZE AIRBORNE INTEGRATION VARIABLES.	TAKOFF	343
C****	FPVTAS = VELOCITY ALONG FLIGHTPATH (FEET/SECOND)	TAKOFF	344
C****	GAMMAR = FLIGHTPATH ANGLE (RADIAN)	TAKOFF	345
C****	FPDIST = DISTANCE ALONG FLIGHTPATH (FEET)	TAKOFF	346
C****	PRESALT = ALTITUDE (FEET)	TAKOFF	347
C****	FPACCEL = ACCELERATION ALONG FLIGHTPATH (FEET/SECOND**2)	TAKOFF	348
C****	DGDTR = TIME RATE OF CHANGE OF FLIGHTPATH ANGLE (RADIAN/SECOND)	TAKOFF	349
C****	VHAS = HORIZONTAL AIRSPEED (FEET/SECOND)	TAKOFF	350
C****	ROC = RATE OF CLIMB (FEET/SECOND)	TAKOFF	351
	NEQ = 4	TAKOFF	352
	FPDIST = DIST	TAKOFF	353
	FPVTAS = VTAS	TAKOFF	354
	GAMMAR = ZERO	TAKOFF	355
	RKAIR(27) = RKGRND(14)	TAKOFF	356
	CALL INTX(NEQ,TIME,DTIME,S,DERIVAT,ALPHA)	TAKOFF	357
	CALL INTG(DIST,RKGRND(14),DTIME,GAMMAR,VWIND,WFUEL,	TAKOFF	358
	DDIST,GDIST,GWT)	TAKOFF	359
C****		TAKOFF	360
C****	AIRBORNE INTEGRATION LOOP	TAKOFF	361
	IF(ICOUNT.EQ.11) ICOUNT = 1	TAKOFF	362
29	DO 30 NCOUNT=ICOUNT,10	TAKOFF	363
	ALPHAJ = ALPHA	TAKOFF	364
	THETAJ = THETAF	TAKOFF	365
	HAGL = PRESALT - HRUNWAY	TAKOFF	366
C****		TAKOFF	367
C****	MAKE INTEGRATION STEP.	TAKOFF	368
	CALL INTZ(NEQ,TIME,DTIME,S,DERIVAT,ALPHA)	TAKOFF	369
	IF(ERRFLAG) THEN	TAKOFF	370
	CALL ERROR(ROCFPM)	TAKOFF	371
	RETURN	TAKOFF	372
	ENDIF	TAKOFF	373
	CALL INTG(DIST,RKAIR(27),DTIME,GAMMAR,VWIND,WFUEL,	TAKOFF	374
	DDIST,GDIST,GWT)	TAKOFF	375
C****		TAKOFF	376
C****	OBTAIN ATMOSPHERIC VARIABLES.	TAKOFF	377
	CALL ATMOSPH(PRESALT,ATMARY)	TAKOFF	378

C****		TAKOFF	379
C****	CALCULATE DYNAMIC PRESSURE, MACH NUMBER AND VELOCITIES.	TAKOFF	380
	CALL SPEED(GAMMAR,FPVTAS,VWIND,AMACH,QS,VKAS,VKEAS,VKTAS,	TAKOFF	381
	VKTGS,VTGS)	TAKOFF	382
C****		TAKOFF	383
C****	CHECK FOR CLEARANCE HEIGHT HCLEAR.	TAKOFF	384
	IF ((.NOT.CLRHGT) .AND. HAGLLT.HCLEAR) THEN	TAKOFF	385
C****	STORE VALUES FOR HCLEAR FEET INTERPOLATION	TAKOFF	386
	FPVTASJ = FPVTAS	TAKOFF	387
	GDISTJ = GDIST	TAKOFF	388
	HAGLJ = HAGL	TAKOFF	389
	ELSEIF((.NOT.CLRHGT) .AND. HAGL.GE.HCLEAR) THEN	TAKOFF	390
C****	FIND VALUES AT HCLEAR FEET AGL	TAKOFF	391
	CLRHGT = .TRUE.	TAKOFF	392
	LINENUM = LINENUM + 1	TAKOFF	393
	GD50 = INTERP(GDIST ,GDISTJ ,HAGL,HAGLJ,HCLEAR)	TAKOFF	394
	VTAS50 = INTERP(FPVTAS,FPVTASJ ,HAGL,HAGLJ,HCLEAR)	TAKOFF	395
	CALL SPEED(GAMMAR,VTAS50,VWIND,AMACH,QS,VKAS,VKEAS,VKTAS,	TAKOFF	396
	& VKTGS,VTGS)	TAKOFF	397
	WRITE(LUOUT,1012) HCLEAR,TIME,GD50,VKAS,VKTAS	TAKOFF	398
1012	FORMAT(' ALTITUDE:',F5.0,'FEET (TIME = ',F6.2,	TAKOFF	399
	& ' DIST = ',F8.1,' CAS = ',F7.1,' TAS = ',F7.1,	TAKOFF	400
	& ' KNOTS'))	TAKOFF	401
	ENDIF	TAKOFF	402
C****		TAKOFF	403
C****	USE DEFAULT CLIMB OUT SPEED IF NO USER INPUT. SET VCOFLAG TO	TAKOFF	404
C****	.TRUE. TO PREVENT RE-INITIALIZING VCLMOUT.	TAKOFF	405
	IF(.NOT.VCOFLAG) THEN	TAKOFF	406
	IF(VCLMOUT.EQ.ZERO) VCLMOUT = VKAS	TAKOFF	407
	VCOFLAG = .TRUE.	TAKOFF	408
	ENDIF	TAKOFF	409
	IF (HAGLLT.HCLIMB) THEN	TAKOFF	410
C****	CONSTANT THETA PORTION OF TAKEOFF.	TAKOFF	411
C****	MODULATE ALPHA FOR CONSTANT PITCH ANGLE CLIMB OR TO COMPLETE	TAKOFF	412
C****	ROTATION.	TAKOFF	413
	IF(.NOT.INISEG1) THEN	TAKOFF	414
C****	INITIATE CONSTANT PITCH ANGLE CLIMB AT SEGMENT ONE CLIMB	TAKOFF	415
C****	GRADIENT. SET MAXIMUM PITCH ANGLE TO THTCLM FOR CONSTANT	TAKOFF	416
C****	THETA PORTION OF TAKEOFF.	TAKOFF	417
	INISEG1 = .TRUE.	TAKOFF	418
	THTMAX = THTCLM	TAKOFF	419
	ENDIF	TAKOFF	420
C****	CONSTANT THETA PORTION OF TAKEOFF	TAKOFF	421
C****	MODULATE ALPHA FOR CONSTANT THETA CLIMB OR TO COMPLETE	TAKOFF	422
C****	ROTATION.	TAKOFF	423
	CALL PITCH('ROTATE ',ALPHA,DADTCMD,DTIME,DTDTGEX,LUOUT)	TAKOFF	424

	ELSEIF(HAGL.GE.HCLIMB .AND. HAGL.LT.HCLMOUT) THEN	TAKOFF	425
C****	CONSTANT THETA TO CONSTANT CLIMBOUT SPEED TRANSITION	TAKOFF	426
C****	MODULATE ALPHA FOR CONSTANT SPEED CLIMB OUT WITHIN	TAKOFF	427
C****	ALTITUDES HCLIMB AND HCLMOUT.	TAKOFF	428
	IF(.NOT.INIVCLM) THEN	TAKOFF	429
C****	INITIATE CONSTANT CALIBRATED AIRSPEED CLIMB.	TAKOFF	430
	INIVCLM = .TRUE.	TAKOFF	431
	THTMAX = 89.9	TAKOFF	432
	LINENUM = LINENUM + 1	TAKOFF	433
	WRITE(LUOUT,1012) HCLIMB,TIME,GDIST,VKCAS,VKTAS	TAKOFF	434
	ENDIF	TAKOFF	435
	IF(ROCFPM.GE.ZERO) THEN	TAKOFF	436
	IF(VKCAS.LT.VCLMOUT) THEN	TAKOFF	437
C****	REDUCE ALPHA	TAKOFF	438
	CALL PITCH('CLIMB ',ALPHA,-DADTCMD,DTIME,DTDTGEX,	TAKOFF	439
&	LUOUT)	TAKOFF	440
	ELSE	TAKOFF	441
C****	INCREASE ALPHA	TAKOFF	442
	CALL PITCH('CLIMB ',ALPHA, DADTCMD,DTIME,DTDTGEX,	TAKOFF	443
&	LUOUT)	TAKOFF	444
	ENDIF	TAKOFF	445
	ELSE	TAKOFF	446
	TERMFLG = .TRUE.	TAKOFF	447
	WRITE(LUOUT,1013) ROCFPM,VCLMOUT,THTMAX	TAKOFF	448
	ENDIF	TAKOFF	449
	ELSEIF(HAGL.GE.HCLMOUT) THEN	TAKOFF	450
C****	CONSTANT THETA TO VKEND OR HMAX AT THTFLY PITCH ATTITUDE	TAKOFF	451
C****	HCLIMB AND HCLMOUT.	TAKOFF	452
	IF(.NOT.INISEG2) THEN	TAKOFF	453
C****	INITIATE CONSTANT PITCH ANGLE CLIMB AT SEGMENT TWO CLIMB	TAKOFF	454
C****	GRADIENT.	TAKOFF	455
	INISEG2 = .TRUE.	TAKOFF	456
	THTMAX = THTFLY	TAKOFF	457
	LINENUM = LINENUM + 1	TAKOFF	458
	WRITE(LUOUT,1012) HCLMOUT,TIME,GDIST,VKCAS,VKTAS	TAKOFF	459
	ENDIF	TAKOFF	460
	IF(ROCFPM.GE.ZERO) THEN	TAKOFF	461
	IF ((THETAF - THTFLY) .GE. THTTOL) THEN	TAKOFF	462
C****	REDUCE ALPHA	TAKOFF	463
	CALL PITCH('CLIMB ',ALPHA,-DADTCMD,DTIME,DTDTGEX,	TAKOFF	464
&	LUOUT)	TAKOFF	465
	ELSEIF ((THETAF - THTFLY) .LT. -THTTOL) THEN	TAKOFF	466
C****	INCREASE ALPHA	TAKOFF	467
	CALL PITCH('CLIMB ',ALPHA, DADTCMD,DTIME,DTDTGEX,	TAKOFF	468
&	LUOUT)	TAKOFF	469
	ELSE	TAKOFF	470
	THETAF = THTFLY	TAKOFF	471
	ENDIF	TAKOFF	472
	ELSE	TAKOFF	473
	TERMFLG = .TRUE.	TAKOFF	474
	WRITE(LUOUT,1013) ROCFPM,VCLMOUT,THTMAX	TAKOFF	475
	ENDIF	TAKOFF	476
	ENDIF	TAKOFF	477

1013	FORMAT(' SUBROUTINE TAKOFF IS UNABLE TO SIMULATE A CLIMB',	TAKOFF	478
&	' USING THE INPUTS OF CLIMB OUT SPEED OR PITCH',	TAKOFF	479
&	' ATTITUDE BECAUSE THE RATE OF CLIMB IS LESS THAN',	TAKOFF	480
&	' ZERO. /', RATE OF CLIMB, CLIMB OUT SPEED, AND',	TAKOFF	481
&	' MAXIMUM PITCH ATTITUDE =',	TAKOFF	482
&	F10.0, ' FEET/MINUTE',F10.0,' KNOTS',F10.0,' DEGREES.')	TAKOFF	483
C****		TAKOFF	484
C****	CHECK FOR TERMINATION CRITERIA.	TAKOFF	485
	IF(GDIST .GT.DISTMAX .OR. HAGL .GE.HMAX .OR.	TAKOFF	486
&	TIME .GT.TIMEMAX .OR. VKCAS.GE.VKEND) TERMFLG = .TRUE.	TAKOFF	487
	IF(NCOUNT.EQ.10 .OR. TERMFLG) THEN	TAKOFF	488
C****	CALCULATE ADDITIONAL OUTPUT PARAMETERS, WRITE OUTPUT AND	TAKOFF	489
C****	INCREMENT LINE NUMBER COUNTER LINENUM.	TAKOFF	490
	IF(.NOT.TERMFLG) THEN	TAKOFF	491
	DADT = (ALPHA - ALPHAJ)/FLOAT(DTIME)	TAKOFF	492
	DTHTDT = (THETAJ - THETAJ)/FLOAT(DTIME)	TAKOFF	493
	ENDIF	TAKOFF	494
	CALL FORCEX(ALPHA,CD,CL)	TAKOFF	495
	GAMMA = ATAN(ROC/VTGS)*RX	TAKOFF	496
	ROCFPM = ROC*60.0	TAKOFF	497
C****		TAKOFF	498
C****	RECALCULATE FLIGHTPATH ACCELERATION FOR OUTPUT.	TAKOFF	499
	FPA = FPACCEL + G*SIN(GAMMAR)	TAKOFF	500
	IF(DTIME.GE.0.10D0) THEN	TAKOFF	501
	WRITE(LUOUT,1005) TIME,GDIST,GWT,HAGL,VKCAS,VKTAS,VKTGS,	TAKOFF	502
&	FPA,CL,CD,THETAJ,ALPHA,GAMMA,DTHTDT,	TAKOFF	503
&	ROCFPM,XLF,THRUST,XENG	TAKOFF	504
	ELSE	TAKOFF	505
	WRITE(LUOUT,1006) TIME,GDIST,GWT,HAGL,VKCAS,VKTAS,VKTGS,	TAKOFF	506
&	FPA,CL,CD,THETAJ,ALPHA,GAMMA,DTHTDT,	TAKOFF	507
&	ROCFPM,XLF,THRUST,XENG	TAKOFF	508
	ENDIF	TAKOFF	509
	IF(TERMFLG) THEN	TAKOFF	510
	WRITE(LUOUT,1014)	TAKOFF	511
1014	FORMAT(/,' END OF TAKEOFF')	TAKOFF	512
	RETURN	TAKOFF	513
	ENDIF	TAKOFF	514
	LINENUM = LINENUM + 1	TAKOFF	515
	IF(LINENUM.GE.NPAGE) THEN	TAKOFF	516
C****	RESET LINE NUMBER COUNTER LINENUM AND WRITE HEADER FOR	TAKOFF	517
C****	NEW PAGE.	TAKOFF	518
	LINENUM = 5	TAKOFF	519
	WRITE(LUOUT,1010)	TAKOFF	520
	WRITE(LUOUT,1004)	TAKOFF	521
	ENDIF	TAKOFF	522
C****	RESET INTEGRATION LOOP COUNTER ICOUNT TO 1 AND RESTART	TAKOFF	523
C****	THE AIRBORNE INTEGRATION LOOP.	TAKOFF	524
	ICOUNT = 1	TAKOFF	525
	GO TO 29	TAKOFF	526
	ENDIF	TAKOFF	527
30	CONTINUE	TAKOFF	528
	END	TAKOFF	529

Subroutine LANDNG

SUBROUTINE LANDNG(AOA)	LANDNG	1
C**** THIS SUBROUTINE CONTROLS THE EXECUTION OF THE LANDING MANEUVER.	LANDNG	2
C**** LANDNG CALLS STEDYST TO OBTAIN THE REQUIRED VALUES OF THRUST	LANDNG	3
C**** THRUST AND ANGLE OF ATTACK FOR STEADY STATE APPROACH. LANDNG	LANDNG	4
C**** CALLS FLARE TO EXECUTE THE FLARE AND CALLS ROLLS FOR THE GROUND	LANDNG	5
C**** PORTION OF THE LANDING.	LANDNG	6
C****	LANDNG	7
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKAS,VKTAS	AERO	2
COMMON/ENGINE/ AIT,AMACH,DTFAIL,EPR,FE,FG,FGPCT,NENG,PWRCODE,	ENGINE	1
& REVNDX,RC,THRUST,VTANGLE,WFUEL,XENG,XENGFLD,	ENGINE	2
& XENGOUT,XIDLE,XMIL,ZFN	ENGINE	3
COMMON/AIRBORN/ ALPHAJ,ALPHMX,DDT,GAMMAPP,ROCFPM,THETAJ,XLF,XLFJ	AIRBORN	1
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
COMMON/RUNWAY/ ABARG,AOAABRK,BRAKMU,BRKFCR,GAMMARW,GRW,HAGL,	RUNWAY	1
& HCLAR,HFLARE,HGEAR,HRUNWAY,IMU,RCR,ROLLMU,TIMEFLD,	RUNWAY	2
& TIMEBRK,TIMEFLP,TIMESBK,TIMESPL,XMU	RUNWAY	3
DOUBLEPRECISION VTAS,DIST,ACCEL,VTASJ,RKGRND	INTEG	1
COMMON/INTEG/ VTAS,DIST,ACCEL,VTASJ,RKGRND(20)	INTEG	2
DOUBLEPRECISION FPGTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPGTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
COMMON/ATMOS/ TEMPR,PRESS,RHO,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMPF	ATMOS	1
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRS,ROTATE,	FLAGS	2
& RTFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITTR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRS,ROTATE,RTFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	8
LOGICAL FINDV	LANDNG	21
REAL ATMARY(8),S(48)	LANDNG	22
EQUIVALENCE (TEMPR,ATMARY(1)),(FPGTAS,S(1))	LANDNG	23
DATA SINKTD	LANDNG	24
& / 10.0/	LANDNG	25

NAMLIST/LND/ALPHA,GAMMAPP,HFLARE,SINKTD,VKAPP	LANDNG	26
NAMLIST/LND2/ DTDTMX,SPLFLAG,WRITTR	LANDNG	26
ALPHA = AOA	LANDNG	28
C****	LANDNG	29
C**** INPUT DATA LOADED INTO LANDNG THRU NAMLIST /LND/.	LANDNG	30
READ(LUIN,LND)	LANDNG	31
READ(LUIN,LND2)	LANDNG	32
C****	LANDNG	33
C**** TERMINATE PROGRAM IF APPROACH FLIGHT PATH ANGLE, FLARE HEIGHT OR	LANDNG	34
C**** APPROACH SPEED NOT INITIALIZED.	LANDNG	35
IF (GAMMAPP.EQ. 1.0) THEN	LANDNG	36
CALL HALT(LUIN,LUMSG,LUOUT,' GAMMAPP NOT INPUT BY USER.')	LANDNG	37
ELSEIF (GAMMAPP.GE.ZERO) THEN	LANDNG	38
CALL HALT(LUIN,LUMSG,LUOUT,' GAMMAPP MUST BE LESS THAN ZERO.')	LANDNG	39
ENDIF	LANDNG	40
IF(HFLARE.EQ.-1.0) THEN	LANDNG	41
CALL HALT(LUIN,LUMSG,LUOUT,' HFLARE NOT INPUT BY USER.')	LANDNG	42
ELSEIF(VKAPP .EQ. 1.0 .AND. ALPHA.EQ. 99.0) THEN	LANDNG	43
& CALL HALT(LUIN,LUMSG,LUOUT,	LANDNG	44
& ' VKAPP OR ALPHA NOT INPUT BY USER.')	LANDNG	45
ENDIF	LANDNG	46
IF(ALPHA.EQ.99.0) THEN	LANDNG	47
C**** FIND ANGLE OF ATTACK FOR STEADY STATE APPROACH.	LANDNG	48
FINDV = .FALSE.	LANDNG	49
ELSE	LANDNG	50
C**** FIND AIRSPEED FOR STEADY STATE APPROACH; INITIALIZE CALIBRATED	LANDNG	51
C**** AIRSPEED AT 100 KNOTS.	LANDNG	52
VKAPP = 100.0	LANDNG	53
FINDV = .TRUE.	LANDNG	54
ENDIF	LANDNG	55
C****	LANDNG	56
C**** INITIALIZE CURVE FILES.	LANDNG	57
CALL INICURV	LANDNG	58
C****	LANDNG	59
C**** OBTAIN ATMOSPHERIC VARIABLES.	LANDNG	60
IF(HCLEAR.GT.HFLARE) THEN	LANDNG	61
HAGL = HCLEAR	LANDNG	62
ELSE	LANDNG	63
HAGL = HFLARE	LANDNG	64
ENDIF	LANDNG	65
PRESALT = HRUNWAY + HAGL	LANDNG	66
CALL ATMOSPH(PRESALT,ATMARY)	LANDNG	67
C****	LANDNG	68
C**** INITIALIZE FLIGHT PATH ANGLE, TRUE AIRSPEED, MACH NUMBER AND QS.	LANDNG	69
VAPP = VKAPP*FPSKTS	LANDNG	70
FPVTAS = VAPP/SQRT(SIGMA)	LANDNG	71
GAMAPPR= GAMMAPP/RX	LANDNG	72
GAMMAR = GAMAPPR - ASIN((VWIND/FPVTAS)*SIN(GAMAPPR))	LANDNG	73
CALL SPEED(GAMMAR,FPVTAS,VWIND,AMACH,QS,VKCAS,VKEAS,VKTAS, VKTGS,	LANDNG	74
VTGS)	LANDNG	75
VKCAS = VKAPP	LANDNG	76
C****	LANDNG	77
C**** OBTAIN THRUST AND ANGLE OF ATTACK FOR STEADY STATE APPROACH.	LANDNG	78
CALL STEDYST(ALPHA,FPVTAS,GAMMAR,FINDV)	LANDNG	79

C****		LANDNG	80
C****	RECALCULATE APPROACH SPEED IF ANGLE OF ATTACK IS AN INPUT.	LANDNG	81
	IF(FINDV) THEN	LANDNG	82
	VAPP = FPVTAS*SQRT(SIGMA)	LANDNG	83
	VKAPP = VAPP/FPSKTS	LANDNG	84
	ENDIF	LANDNG	85
C****		LANDNG	86
C****	CALCULATE SINK RATE AT APPROACH IN FEET/MINUTE.	LANDNG	87
	ROC = FPVTAS*SIN(GAMMAR)	LANDNG	88
	SINKRT = -ROC*60.0	LANDNG	89
	IF(ERRFLAG) THEN	LANDNG	90
	TERMFLG = .TRUE.	LANDNG	91
	AOA = ALPHA	LANDNG	92
	WRITE(LUOUT,1001) VKAPP, AMMAPP,PRESALT,ALPHA,ALPHMX,XENG,GWT	LANDNG	93
1001	FORMAT(' *** FAILED IN THRUST AND ANGLE OF ATTACK ON APPROACH',	LANDNG	94
&	/, VKAPP,GAMMAPP,PRESALT =',3F10.2,	LANDNG	95
&	/, ALPHA,ALPHMX,XENG,GWT =',3F10.2,F10.0)	LANDNG	96
	RETURN	LANDNG	97
	ENDIF	LANDNG	98
C****		LANDNG	99
C****	DETERMINE LOAD FACTOR FOR FLARE.	LANDNG	100
	IF(HFLARE.GT.ZERO) CALL FLARENZ(ALPHA,GAMMAPP,GAMMATD,	LANDNG	101
&	HAGL,HCLEAR,HFLARE,HRUNWAY,SINKTD,	LANDNG	102
&	THRUST,VKAPP,XLFLARE,XLFMAX)	LANDNG	103
	WRITITR = .TRUE.	LANDNG	104
C****		LANDNG	105
C****	WRITE HEADER INFORMATION FOR OUTPUT.	LANDNG	106
	LINENUM = 7	LANDNG	107
	WRITE(LUOUT,1002) VKAPP ,GAMMAPP ,SINKRT ,ALPHA ,FLAP,	LANDNG	108
&	HRUNWAY,XLFMAX,	LANDNG	109
&	GAMMATD ,SINKTD*60.0 ,THRUST ,SPOILER,	LANDNG	110
&	DTEMPF ,ALPHMX,	LANDNG	111
&	WNGLOD ,GAMMARW ,DADTCMD ,SPLFLAG,	LANDNG	112
&	VKWIND ,DTDTMX	LANDNG	113
1002	FORMAT('↑ VKAPP =',F7.1, GAMMAPP =',F6.1,' SINKRT =',F6.1,	LANDNG	114
&	ALPHA =',F7.2, FLAP =',F6.1,' HRUNWAY =',F7.0,	LANDNG	115
&	XLFMAX =',F6.2, /,	LANDNG	116
&	GAMMATD =',F6.1,' SINKTD =',F6.1,	LANDNG	117
&	THRUST =',F7.0,' SPOILER =',F6.1,' DTEMPF =',F7.1,	LANDNG	118
&	ALPHMX =',F6.2, /,	LANDNG	119
&	WNGLOD =',F7.1,' GAMMARW =',F6.1,'	LANDNG	120
&	DADTCMD =',F7.2,' SPLFLAG =',L6, VKWIND =',F7.1,	LANDNG	121
&	DTDTMX =',F6.2, /)	LANDNG	122

IF(TERMFLG) THEN	LANDNG	123
C**** TERMINATE LANDING EXECUTION.	LANDNG	124
AOA = ALPHA	LANDNG	126
WRITE(LUOUT,1003)	LANDNG	125
1003 FORMAT(/,' END OF LANDING')	LANDNG	126
RETURN	LANDNG	127
ELSEIF(HFLARE.GE.HCLEAR) THEN	LANDNG	128
C**** CALCULATE VARIABLES FOR OUTPUT.	LANDNG	129
OVERFLG = .TRUE.	LANDNG	130
TIME = DBLE(ZERO)	LANDNG	131
C****	LANDNG	132
C**** WRITE HEADER RECORD FOR OUTPUT.	LANDNG	133
WRITE(LUOUT,1004)	LANDNG	134
1004 FORMAT(' TIME DIST. WEIGHT ALT. VCAS VTAS VTGS',	LANDNG	135
& ' ACCEL CL CD THETA ALPHA GAMMA DTHET '	LANDNG	136
& ' R/C LOAD THRUST XENG',	LANDNG	137
& ' (SEC) (FEET) (LBS) (FEET) (KTS) (KTS) (KTS)',	LANDNG	138
& ' (FPS2) (DEG) (DEG) (DEG) /DT '	LANDNG	139
& ' (FPM) FACT (LBS) OR MU',)	LANDNG	140
ELSE	LANDNG	141
STEADY = .TRUE.	LANDNG	142
CALL APPROCH(ALPHA,GAMMAPP,GDIST,HAGL,HCLEAR,HFLARE,HRUNWAY)	LANDNG	143
ENDIF	LANDNG	144
C****	LANDNG	145
C**** INITIALIZE APPROACH DISTANCE, APPDIST.	LANDNG	146
APPDIST = GDIST	LANDNG	147
C****	LANDNG	148
C**** INITIALIZE LOAD FACTOR FOR FLARE.	LANDNG	149
XLF = XLFLARE	LANDNG	150
XLFMAXJ = XLFMAX	LANDNG	151
XLFMAX = XLFLARE	LANDNG	152
C****	LANDNG	153
C**** CALL SUBROUTINE FLARE WITH FINAL LOAD FACTOR VALUE.	LANDNG	154
IF(HFLARE.GT.ZERO) THEN	LANDNG	155
STEADY = .FALSE.	LANDNG	156
LINENUM = LINENUM + 1	LANDNG	157
WRITE(LUOUT,1005) HFLARE,XLF	LANDNG	158
1005 FORMAT(' **** BEGIN FLARE AT 'F4.1,' FEET. ',	LANDNG	159
& ' LOAD FACTOR = 'F8.4,' G"S ****')	LANDNG	160
CALL FLARE(ALPHA,GDIST,ODIST,ROCTD,VKTGS)	LANDNG	161
ENDIF	LANDNG	162
XLFMAX = XLFMAXJ	LANDNG	163
IF(TERMFLG) THEN	LANDNG	164
AOA = ALPHA	LANDNG	165
WRITE(LUOUT,1003)	LANDNG	166
RETURN	LANDNG	167
ENDIF	LANDNG	168
CALL FORCEX(ALPHA,CD,CL)	LANDNG	169
LINENUM = LINENUM + 2	LANDNG	170
WRITE(LUOUT,1006)	LANDNG	171
1006 FORMAT(' **** TOUCHDOWN ****')	LANDNG	172
CALL SPEED, GAMMAR,FPVTAS,VWIND,AMACH,QS,VKCAS,VKEAS,VKTAS,	LANDNG	173
VKTGS,VTGS)	LANDNG	174

AIRDIST = GDIST	LANDNG	175
FLRDIST = AIRDIST - APPDIST	LANDNG	176
GAMMA = ATAN(ROC/VTGS)*RX	LANDNG	177
SINKRT = -SINKTD*60.0	LANDNG	178
THETAF = ALPHA	LANDNG	179
IF(DTIME.GE.0.10D0) THEN	LANDNG	180
WRITE(LUOUT,1007) TIME,GDIST,GWT,ZERO,VKCAS,VKTAS,VKTGS,	LANDNG	181
& FPACCEL,CL,CD,THETAF,ALPHA,GAMMA,DTDT,-SINKRT,	LANDNG	182
& XLF,THRUST,XMU	LANDNG	183
1007 FORMAT(F6.1,F8.0,F9.0,F8.1,3F7.1,F7.2,F8.4,F7.4,4F7.2,F8.1,	LANDNG	184
& F6.2,F9.0,F7.3)	LANDNG	185
ELSE	LANDNG	186
WRITE(LUOUT,1008) TIME,GDIST,GWT,ZERO,VKCAS,VKTAS,VKTGS,FPACCEL,CL,	LANDNG	187
& FPACCEL,CL,CD,THETAF,ALPHA,GAMMA,DTDT,-SINKRT,	LANDNG	188
& XLF,THRUST,XMU	LANDNG	189
1008 FORMAT(F6.2,F8.0,F9.0,F8.1,3F7.1,F7.2,F8.4,F7.4,4F7.2,F8.1,	LANDNG	190
& F6.2,F9.0,F7.3)	LANDNG	191
ENDIF	LANDNG	192
C****	LANDNG	193
C**** INITIALIZE GROUND ROLL INTEGRATION VARIABLES WITH PREVIOUS	LANDNG	194
C**** AIRBORNE INTEGRATION VARIABLES FOR SUBROUTINE ROLL	LANDNG	195
TIMEFLD = FLOAT(TIME)	LANDNG	196
VTAS = FPVTAS	LANDNG	197
CALL ROLL(ALPHA,GDIST)	LANDNG	198
AOA = ALPHA	LANDNG	199
WRITE(LUOUT,1009) INT(HCLEAR),GDIST-ODIST,AIRDIST,GDIST-AIRDIST,	LANDNG	200
& APPDIST,FLRDIST,ABARG	LANDNG	201
1009 FORMAT(/' DISTANCE OVER A',I4,' FOOT OBSTACLE =',F7.1,' FEET; ',	LANDNG	202
& ' AIR DISTANCE =',F7.1,' FEET; GROUND ROLL DISTANCE =',	LANDNG	203
& F7.1,' FEET; ',	LANDNG	204
& ' PRE-FLARE DISTANCE =',F7.1,' FEET; ',	LANDNG	205
& ' FLARE DISTANCE =',F7.1,' FEET; AVERAGE DECELERATION =',	LANDNG	206
& F6.3,' G',/,)	LANDNG	207
RETURN	LANDNG	208
END	LANDNG	209

Subroutine STEDYST

SUBROUTINE STEDYST(ALPHA,FPVTAS,GAMMAR,FINDV)	STEDYST	1
C**** THIS SUBROUTINE CALCULATES THE REQUIRED VALUES OF THRUST AND	STEDYST	2
C**** ANGLE OF ATTACK OR THRUST AND AIRSPEED FOR ZERO ACCELERATION ALONG	STEDYST	3
C**** AND NORMAL TO THE FLIGHT PATH, DV/DT AND DG/DT, RESPECTIVELY. THE	STEDYST	4
C**** OUTER LOOP VARIES THRUST, WHILE THE TWO INNER LOOPS VARY	STEDYST	5
C**** ALPHA FROM ALPHM TO ALPHMX OR FPVTAS FROM VTASMN TO VTASMX. FOR	STEDYST	6
C**** A FIXED VALUE OF THRUST (OUTER LOOP), FUNCTIONS DVDT (ACCELERATION	STEDYST	7
C**** ALONG FLIGHT PATH), AND DGD (ACCELERATION NORMAL TO FLIGHT PATH),	STEDYST	8
C**** ARE CALLED WITH THE VALUES OF ALPHA (INNER LOOP). THESE TWO	STEDYST	9
C**** FUNCTION SUBROUTINES CALL FORCEX. SUBROUTINES ITRLND AND ZEROX	STEDYST	10
C**** ARE BOTH ZERO FINDERS.	STEDYST	11
C****	STEDYST	12
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKCAS,VKTAS	AERO	2
COMMON/ENGINE/ AIT,AMACH,DTFAIL,EPR,FE,FG,FGPCT,NENG,PWRCODE,	ENGINE	1
& REVNDX,RC,THRUST,VTANGLE,WFUEL,XENG,XENGFLD,	ENGINE	2
& XENGOUT,XIDLE,XMIL,ZFN	ENGINE	3
COMMON/ATMOS/ TEMPR,PRESS,RHO,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMPF	ATMOS	1
PARAMETER (ASL=661.48,FPSKTS=1.687806,G=32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRS,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRS,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
LOGICAL FINDV	STEDYST	20
REAL INTERP	STEDYST	21
EXTERNAL DGD,VDVT	STEDYST	22
DATA ALPHA0,ALPHA4,DALPHA,DVTAS, EPS, FACTOR, FNSTEP,MAXITER	STEDYST	23
& / 0.0, 4.0, 1.0, 1.0, 0.0001, 0.95, 1.05, 31/	STEDYST	24
ERRFLAG = .FALSE.	STEDYST	25
C****	STEDYST	26
C**** INITIALIZE NET THRUST TOLERANCE TO THE GREATER OF 0.1% OF GROSS	STEDYST	27
C**** WEIGHT OR 5 POUNDS.	STEDYST	28
IF(GWT.GT.5000.) THEN	STEDYST	29
TOLRNC = 0.001*GWT	STEDYST	30
ELSE	STEDYST	31
TOLRNC = 5.0	STEDYST	32
ENDIF	STEDYST	33

C****		STEDYST	34
C****	INITIALIZE UPPER AND LOWER SEARCH BOUNDARIES FOR THRUST TO	STEDYST	35
C****	THE QUOTIENT OF GROSS WEIGHT OVER THE ENGINE MULTIPLICATIVE FACTOR	STEDYST	36
C****	AND 1% OF THE GROSS WEIGHT, RESPECTIVELY.	STEDYST	37
	BOUNDU = GWT/XENG	STEDYST	38
	BOUNDL = GWT/100.0	STEDYST	39
C****		STEDYST	40
C****	MAKE FIRST GUESS AT ALPHA OR VKAPP.	STEDYST	41
	IF(.NOT.FINDV) THEN	STEDYST	42
C****	FIND ANGLE OF ATTACK.	STEDYST	43
	CALL FORCEX(ALPHA4,CD4,CL4)	STEDYST	44
	CALL FORCEX(ALPHA0,CD0,CL0)	STEDYST	45
	ALPHA = INTERP(ALPHA4,ALPHA0,CL4,CL0,GWT/QS)	STEDYST	46
	CALL FORCEX(ALPHA,CD,CL)	STEDYST	47
	ALPHMX = ALPHA + 2.5	STEDYST	48
	ALPHMN = -ALPHMX	STEDYST	49
	ELSE	STEDYST	50
C****	FIND AIRSPEED.	STEDYST	51
C****		STEDYST	52
C****	FIND APPROXIMATE CL AND CD FOR MACH NUMBER AT 100 KNOTS	STEDYST	53
C****	AIRSPEED.	STEDYST	54
	CALL FORCEX(ALPHA,CD,CL)	STEDYST	55
C****		STEDYST	56
C****	CALCULATE FLIGHT PATH TRUE AIRSPEED BASED ON THE APPROXIMATED	STEDYST	57
C****	CL AND ASSUMPTION OF LIFT EQUAL TO WEIGHT AND ALPHA EQUAL TO	STEDYST	58
C****	ZERO.	STEDYST	59
	FPVTAS = SQRT(2.0*GWT*COS(GAMMAR)/(CL*RHO*SWING))	STEDYST	60
	CALL SPEED(GAMMAR,FPVTAS,0.0,AMACH,QS,VKAS,VKEAS,VKTAS,	STEDYST	61
	VKTGS,VTGS)	STEDYST	62
	CALL FORCEX(ALPHA,CD,CL)	STEDYST	63
	VTASMX = FPVTAS + 10.0	STEDYST	64
	VTASMN = FPVTAS - 10.0	STEDYST	65
	ENDIF	STEDYST	66
C****		STEDYST	67
C****	MAKE FIRST GUESS AT THRUST.	STEDYST	68
	TFIRST = 1.06*(CD*QS + GWT*SIN(GAMMAR))	STEDYST	69
	IF(JDEBUG.EQ.6666 .OR. JDEBUG.EQ.9999)	STEDYST	70
	& WRITE(LUOUT,9001) TFIRST,ALPHA,VKAS,CD,CL,QS	STEDYST	71
9001	FORMAT('**** INITIAL CONDITIONS ****',/,	STEDYST	72
	& 'THRUST, ALPHA, VKAPP, CD, CL, QS=',/,	STEDYST	73
	& F7.0, F7.2, F9.1, 2F8.4, F10.1,/,	STEDYST	74
19	LABLNO = 19	STEDYST	75
	ERROR = 99.	STEDYST	76
	JFLAG = 0	STEDYST	77
	THRUST = TFIRST	STEDYST	78
	IF(FINDV) THEN	STEDYST	79
	FG = THRUST/COSD(ALPHA+AIT)	STEDYST	80
	IF(JDEBUG.EQ.6666 .OR. JDEBUG.EQ.9999)	STEDYST	81
	& WRITE(LUOUT,9002) ICOUNT,LABLNO,THRUST,FPVTAS,DVDTHI,	STEDYST	82
	& JFLAG,ERROR,ERRORJ	STEDYST	83

	DO 10 ICOUNT=1,MAXITER	STEDYST	84
	IF(JFLAG.LE.1) ERRORJ = ERROR	STEDYST	85
C****		STEDYST	86
C****	FIND VALUE OF THRUST SUCH THAT DV/DT IS NEGATIVE AT FPGTAS	STEDYST	87
C****	EQUAL TO VTASMX, WITH A REDUCTION IN FPGTAS RESULTING IN	STEDYST	88
C****	LESS DRAG DV/DT WILL CHANGE SIGN(I.E. A BOUNDED INTERVAL IN	STEDYST	89
C****	WHICH DV/DT EQUAL TO ZERO WILL BE FOUND).	STEDYST	90
	VTASHI = VTASMX	STEDYST	91
	CALL SPEED(GAMMAR,VTASHI,0.0,AMACH,QS,VKCAS,VKEAS,VKTAS,	STEDYST	92
	VKTGS,VTGS)	STEDYST	93
	DVDTHI = DVD(ALPHA,VTASHI,GAMMAR,GWT)	STEDYST	94
	IF (DVDTHI.GE.0.0) THEN	STEDYST	95
	TFIRST = TFIRST/FNSTEP	STEDYST	96
	IF(TFIRST.GE.BOUNDL) GO TO 19	STEDYST	97
	GO TO 199	STEDYST	98
	ENDIF	STEDYST	99
C****		STEDYST	100
C****	SEARCH FOR DV/DT EQUAL TO ZERO FROM VTASMX TO VTASMN.	STEDYST	101
29	LABLNO = 29	STEDYST	102
	VTASLO = VTASHI - DVTAS	STEDYST	103
	FG = THRUST/COSD(ALPHA+AIT)	STEDYST	104
	IF(JDEBUG.EQ.6666 .OR. JDEBUG.EQ.9999)	STEDYST	105
&	WRITE(LUOUT,9002) ICOUNT,LABLNO,THRUST,VTASLO,DVDTHI,	STEDYST	106
&	JFLAG,ERROR,ERRORJ	STEDYST	107
	IF(VTASLO.LT.VTASMN) THEN	STEDYST	108
	ERRFLAG = .TRUE.	STEDYST	109
	WRITE(LUOUT,1001)THRUST,VTASLO,ERROR,ERRORJ,ICOUNT	STEDYST	110
1001	FORMAT(' *** FAILED IN STEDYST. NO SOLUTION FOR DVDTHI,	STEDYST	111
&	/, THRUST,FPGTAS,ERROR,ERRORJ,ICOUNT =',	STEDYST	112
&	2F9.2,2F10.4,I3)	STEDYST	113
	RETURN	STEDYST	114
	ENDIF	STEDYST	115
	CALL SPEED(GAMMAR,VTASLO,0.0,AMACH,QS,VKCAS,VKEAS,VKTAS,	STEDYST	116
	VKTGS,VTGS)	STEDYST	117
	DVDTLO = DVD(ALPHA,VTASLO,GAMMAR,GWT)	STEDYST	118
C****		STEDYST	119
C****	CHECK TO SEE IF DV/DT EQUAL TO ZERO IS BOUNDED.	STEDYST	120
	IF (DVDTLO*DVDTHI.GT.0.0) THEN	STEDYST	121
	VTASHI = VTASLO	STEDYST	122
	DVDTHI = DVDTLO	STEDYST	123
	GO TO 29	STEDYST	124
	ENDIF	STEDYST	125

C****		STEDYST	126
C****	ONCE FPVTAS INTERVAL FOR DV/DT EQUAL TO ZERO IS BOUNDED,	STEDYST	127
C****	CALL FUNCTION ZEROX TO FIND FPVTAS1, THE VALUE OF FPVTAS FOR	STEDYST	128
C****	DV/DT EQUALS ZERO.	STEDYST	129
	FPVTAS1 = ZEROX(VTASLO,VTASHI,DVDT,ALPHA,GAMMAR,GWT,EPS,	STEDYST	130
&	FINDV)	STEDYST	131
C****		STEDYST	132
C****	SEARCH FOR DG/DT SAME AS ABOVE SEARCH FOR DV/DT EQUALS ZERO.	STEDYST	133
C****	IF DG/DT EQUALS ZERO AT FPVTAS = VTASMX, PROGRAM FAILS.	STEDYST	134
C****	FPVTAS2 IS REQUIRED VALUE OF FPVTAS FOR DG/DT EQUAL TO ZERO.	STEDYST	135
	VTASHI = VTASMX	STEDYST	136
	CALL SPEED(GAMMAR,VTASHI,0.0,AMACH,QS,VKCAS,VKEAS,VKTAS,	STEDYST	137
	VKTGS,VTGS)	STEDYST	138
	DGDTHI = DGD(ALPHA,VTASHI,GAMMAR,GWT)	STEDYST	139
	IF(DGDTHI.LT.0.0) GO TO 199	STEDYST	140
39	LABLNO = 39	STEDYST	141
	VTASLO = VTASHI - DVTAS	STEDYST	142
	FG = THRUST/COSD(ALPHA+AIT)	STEDYST	143
	IF(JDEBUG.EQ.6666 .OR. JDEBUG.EQ.9999)	STEDYST	144
&	WRITE(LUOUT,9003) ICOUNT,LABLNO,THRUST,VTASLO,DGDTHI,	STEDYST	145
&	JFLAG,ERROR,ERRORJ	STEDYST	146
	IF(VTASLO.LT.VTASMN) THEN	STEDYST	147
	TFIRST = TFIRST/FNSTEP	STEDYST	148
	IF(TFIRST.GE.BOUNDL) GO TO 19	STEDYST	149
	GO TO 199	STEDYST	150
	ENDIF	STEDYST	151
	CALL SPEED(GAMMAR,VTASLO,0.0,AMACH,QS,VKCAS,VKEAS,VKTAS,	STEDYST	152
	VKTGS,VTGS)	STEDYST	153
	DGDTLO = DGD(ALPHA,VTASLO,GAMMAR,GWT)	STEDYST	154
	IF(DGDTLO.DGDTHI.GT.0.0) THEN	STEDYST	155
	VTASHI = VTASLO	STEDYST	156
	DGDTHI = DGDTLO	STEDYST	157
	GO TO 39	STEDYST	158
	ENDIF	STEDYST	159
	FPVTAS2 = ZEROX(VTASLO,VTASHI,DGDT,ALPHA,GAMMAR,GWT,EPS,	STEDYST	160
&	FINDV)	STEDYST	161

C***		STEDYST	162
C***	FIND ERROR BETWEEN FPVTAS1 AND FPVTAS2.	STEDYST	163
	ERROR = (FPVTAS1 - FPVTAS2)/VTASMX	STEDYST	164
	IF ((ABS(ERROR).LT.0.001	STEDYST	165
&	(ABS(ERROR).LT.0.005 .AND. ICOUNT.GT.25).OR.	STEDYST	166
&	(JFLAG.EQ.3	STEDYST	167
	FPVTAS = FPVTAS1	STEDYST	168
	CALL SPEED(GAMMAR,FPVTAS,0.0,AMACH,QS,VKCAS,VKEAS,VKTAS,	STEDYST	169
	VKTGS,VTGS)	STEDYST	170
	RETURN	STEDYST	171
	ELSE	STEDYST	172
C***	MAKE NEW GUESS AT THRUST AND REPEAT DV/DT AND DG/DT LOOPS	STEDYST	173
C***	UNTIL FPVTAS1 EQUALS FPVTAS2 (WITHIN TOLERANCES).	STEDYST	174
	CALL ITRLND(ERROR,ERRORJ,THRUST,FACTOR,TOLRNCE,JFLAG)	STEDYST	175
	ENDIF	STEDYST	176
10	CONTINUE	STEDYST	177
199	WRITE(LUOUT,1003) TFIRST,DGDTHI,ERROR,ERRORJ,FPVTAS1,FPVTAS2,	STEDYST	178
&	ICOUNT	STEDYST	179
1003	FORMAT(' *** FAILED IN SUBROUTINE STEDYST.',	STEDYST	180
&	' TFIRST, DGDTHI, ERROR, ERRORJ, FPVTAS1,',	STEDYST	181
&	' FPVTAS2,ICOUNT = ',F8.0,5F10.4,I7)	STEDYST	182

ELSE	STEDYST	183
FG = THRUST/COSD(ALPHHI+AIT)	STEDYST	184
IF(JDEBUG.EQ.6666 .OR. JDEBUG.EQ.9999)	STEDYST	185
& WRITE(LUOUT,9004) ICOUNT,LABLNO,THRUST,ALPHA,DVDTHI,	STEDYST	186
& JFLAG,ERROR,ERRORJ	STEDYST	187
DO 20 ICOUNT=1,MAXITER	STEDYST	188
IF(JFLAG.LE.1) ERRORJ = ERROR	STEDYST	189
C****	STEDYST	190
C****	STEDYST	191
C****	STEDYST	192
C****	STEDYST	193
C****	STEDYST	194
FIND VALUE OF THRUST SUCH THAT DV/DT IS NEGATIVE AT ALPHA	STEDYST	195
EQUAL TO ALPHMX, WITH A REDUCTION IN ALPHA RESULTING IN LESS	STEDYST	196
LESS DRAG DV/DT WILL CHANGE SIGN(I.E. A BOUNDED INTERVAL IN	STEDYST	197
WHICH DV/DT EQUAL TO ZERO WILL BE FOUND).	STEDYST	198
ALPHHI = ALPHMX	STEDYST	199
DVDTHI = DVDTHI,FPVTAS,GAMMAR,GWT)	STEDYST	200
IF (DVDTHI.GE.0.0) THEN	STEDYST	201
TFIRST = TFIRST/FNSTEP	STEDYST	202
IF(TFIRST.GE.BOUNDL) GO TO 19	STEDYST	203
GO TO 299	STEDYST	204
ENDIF	STEDYST	205
C****	STEDYST	206
C****	STEDYST	207
SEARCH FOR DV/DT EQUAL TO ZERO FROM ALPHMX TO -ALPHMX.	STEDYST	208
LABLNO = 49	STEDYST	209
ALPHLO = ALPHHI - DALPHA	STEDYST	210
FG = THRUST/COSD(ALPHLO+AIT)	STEDYST	211
IF(JDEBUG.EQ.6666 .OR. JDEBUG.EQ.9999)	STEDYST	212
& WRITE(LUOUT,9004) ICOUNT,LABLNO,THRUST,ALPHLO,DVDTHI,	STEDYST	213
& JFLAG,ERROR,ERRORJ	STEDYST	214
IF(ALPHLO.LT.ALPHMN) THEN	STEDYST	215
ERRFLAG = .TRUE.	STEDYST	216
WRITE(LUOUT,1002)THRUST,ALPHLO,ERROR,ERRORJ,ICOUNT	STEDYST	217
1002 FORMAT(' *** FAILED IN STEDYST. NO SOLUTION FOR DVDTHI,	STEDYST	218
& /, THRUST,ALPHA,ERROR,ERRORJ,ICOUNT =,	STEDYST	219
& 2F9.2,2F10.4,I3)	STEDYST	220
RETURN	STEDYST	221
ENDIF	STEDYST	222
DVDTLO = DVDTHI,FPVTAS,GAMMAR,GWT)	STEDYST	223
C****	STEDYST	224
C****	STEDYST	225
CHECK TO SEE IF DV/DT EQUAL TO ZERO IS BOUNDED.	STEDYST	226
IF (DVDTLO*DVDTHI.GT.0.0) THEN	STEDYST	227
ALPHHI = ALPHLO	STEDYST	228
DVDTHI = DVDTLO	STEDYST	229
GO TO 49	STEDYST	230
ENDIF	STEDYST	231
C****		
C****		
C****		
C****		
ONCE ALPHA INTERVAL FOR DV/DT EQUAL TO ZERO IS BOUNDED,		
CALL FUNCTION ZEROX TO FIND ALPHA1, THE VALUE OF ALPHA FOR		
DV/DT EQUALS ZERO.		
ALPHA1 = ZEROX(ALPHLO,ALPHHI,DVDTHI,FPVTAS,GAMMAR,GWT,EPS,		
& FINDV)		

C****		STEDYST	232
C****	SEARCH FOR DG/DT SAME AS ABOVE SEARCH FOR DV/DT EQUALS ZERO.	STEDYST	233
C****	IF DG/DT EQUALS ZERO AT ALPHA = ALPHMX, PROGRAM FAILS.	STEDYST	234
C****	ALPHA2 IS REQUIRED VALUE OF ALPHA FOR DG/DT EQUAL TO ZERO.	STEDYST	235
	ALPHHI = ALPHMX	STEDYST	236
	DGDTHI = DGD(T(ALPHHI,FPVTAS,GAMMAR,GWT)	STEDYST	237
	IF(DGDTHI.LT.0.0) GO TO 299	STEDYST	238
59	LABLNO = 59	STEDYST	239
	ALPHLO = ALPHHI - DALPHA	STEDYST	240
	FG = THRUST/COSD(ALPHLO+AIT)	STEDYST	241
	IF(JDEBUG.EQ.6666 .OR. JDEBUG.EQ.9999)	STEDYST	242
&	WRITE(LUOUT,9005) ICOUNT,LABLNO,THRUST,ALPHLO,DGDTHI,	STEDYST	243
&	JFLAG,ERROR,ERRORJ	STEDYST	244
	IF(ALPHLO.LT.ALPHMN) THEN	STEDYST	245
	TFIRST = TFIRST/FNSTEP	STEDYST	246
	IF(TFIRST.GE.BOUNDL) GO TO 19	STEDYST	247
	GO TO 299	STEDYST	248
	ENDIF	STEDYST	249
	DGDTLO = DGD(T(ALPHLO,FPVTAS,GAMMAR,GWT)	STEDYST	250
	IF(DGDTLO*DGDTHI.GT.0.0) THEN	STEDYST	251
	ALPHHI = ALPHLO	STEDYST	252
	DGDTHI = DGDTLO	STEDYST	253
	GO TO 59	STEDYST	254
	ENDIF	STEDYST	255
	ALPHA2 = ZEROX(ALPHLO,ALPHHI,DGDT,FPVTAS,GAMMAR,GWT,EPS,	STEDYST	256
&	FINDV)	STEDYST	257
C****		STEDYST	258
C****	FIND ERROR BETWEEN ALPHA1 AND ALPHA2.	STEDYST	259
	ERROR = (ALPHA1 - ALPHA2)/ALPHMX	STEDYST	260
	IF ((ABS(ERROR).LT.0.005	STEDYST	261
&	(ABS(ERROR).LT.0.010 .AND. ICOUNT.GT.25).OR.	STEDYST	262
&	(JFLAG.EQ.3	STEDYST	263
	)) THEN	STEDYST	264
	ALPHA = ALPHA1	STEDYST	265
	RETURN	STEDYST	266
ELSE		STEDYST	267
C****	MAKE NEW GUESS AT THRUST AND REPEAT DV/DT AND DG/DT LOOPS	STEDYST	268
C****	UNTIL ALPHA1 EQUALS ALPHA2 (WITHIN TOLERANCES).	STEDYST	269
	CALL ITRLND(ERROR,ERRORJ,THRUST,FACTOR,TOLRNCE,JFLAG)	STEDYST	270
	ENDIF	STEDYST	271
20	CONTINUE	STEDYST	272
239	WRITE(LUOUT,1004) TFIRST,DGDTHI,ERROR,ERRORJ,ALPHA1,ALPHA2,	STEDYST	273
&	ICOUNT	STEDYST	274
1004	FORMAT(' *** FAILED IN SUBROUTINE STEDYST.',	STEDYST	275
&	' TFIRST, DGDTHI, ERROR, ERRORJ, ALPHA1,',	STEDYST	276
&	' ALPHA2,ICOUNT = ',F8.0,5F10.4,I7)	STEDYST	277
ENDIF			

```

      ERRFLAG = .TRUE.
9002 FORMAT( ICOUNT, LABLNO, THRUST, FPVTAS, DVDT, JFLAG,
&          ' ERROR, ERRORJ = ',
&          2I8, F8.0, F8.1, F8.4, I7, 2F8.3)
9003 FORMAT( ICOUNT, LABLNO, THRUST, FPVTAS, DGDT, JFLAG,
&          ' ERROR, ERRORJ = ',
&          2I8, F8.0, F8.1, F8.4, I7, 2F8.3)
9004 FORMAT( ICOUNT, LABLNO, THRUST, ALPHA, DVDT, JFLAG,
&          ' ERROR, ERRORJ = ',
&          2I8, F8.0, F8.1, F8.4, I7, 2F8.3)
9005 FORMAT( ICOUNT, LABLNO, THRUST, ALPHA, DGDT, JFLAG,
&          ' ERROR, ERRORJ = ',
&          2I8, F8.0, F8.1, F8.4, I7, 2F8.3)
      RETURN
      END

```

```

STEDYST 278
STEDYST 279
STEDYST 280
STEDYST 281
STEDYST 282
STEDYST 283
STEDYST 284
STEDYST 285
STEDYST 286
STEDYST 287
STEDYST 288
STEDYST 289
STEDYST 290
STEDYST 291
STEDYST 292

```

Subroutine FLARENZ

SUBROUTINE FLARENZ(ALPHA,GAMMAPP,GAMMATD,HAGL,HCLEAR,HFLARE,	FLARENZ	1
& HRUNWAY,SINKTD,THRUST,VKAPP,XLFLARE,XLFMAX)	FLARENZ	2
C**** THIS SUBROUTINE DETERMINES THE CORRECT TARGET LOAD FACTOR FOR	FLARENZ	3
C**** THE LANDING FLARE.	FLARENZ	4
C****	FLARENZ	5
C**** SUBROUTINE ITRLND IS A ZERO-FINDING ROUTINE, WHICH VARIES THE	FLARENZ	6
C**** INDEPENDENT VARIABLE BASED ON THE SIZE AND SIGN OF THE ERROR.	FLARENZ	7
C**** THE LOOP CONTAINING ITRLND IS EXITED SUCCESSFULLY WHEN JFLAG = 3,	FLARENZ	8
C**** OR THE MAGNITUDE OF THE ERROR IS LESS THAN SOME ACCEPTABLE VALUE.	FLARENZ	9
C**** VARIABLE JCOUNT STORES THE NUMBER OF ITERATIONS WHICH IS LIMITED	FLARENZ	10
C**** TO 25.	FLARENZ	11
C****	FLARENZ	12
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
CHARACTER ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
& THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/ ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
& TKOTYPE	CHARV	4
REAL INTERP	FLARENZ	17
DATA EPSROC,FACTOR,MAXITER,TOLRNCE	FLARENZ	18
& / 0.5, 1.03, 25, 0.00001/	FLARENZ	19
C****	FLARENZ	20
C**** CALCULATE INITIAL LOAD FACTOR GUESS FOR FLARE.	FLARENZ	21
VAPP = VKAPP*FPSKTS	FLARENZ	22
GAMAPPR = GAMMAPP/RX	FLARENZ	23
GAMATDR = ASIN(-SINKTD/VAPP)	FLARENZ	24
XLFLARE = (VAPP**2*(COS(GAMATDR) - COS(GAMAPPR))/ & (G*HFLARE)) + 1.0	FLARENZ	25
GAMMATD = GAMATDR*RX	FLARENZ	26
IF(XLFLARE.GT.XLFMAX) CALL HALT(LUIN,LUMSG,LUOUT,	FLARENZ	27
& ' FLARE LOAD FACTOR EXCEEDS MAXIMUM LIMIT.')	FLARENZ	28
	FLARENZ	29

```

C****
C**** INITIALIZE PARAMETERS FOR FLARE ITERATIONS.
      AOAAPP = ALPHA
      BOUNDU = XLFMAX
      ERROR  = 1.0
      FNAPP  = THRUST
      FPVTASJ = FPVTAS
      JFLAG  = 0
      IF(.NOT.WRITITR) THEN
        IF(JDEBUG.EQ.6666 .OR. JDEBUG.EQ.9999) THEN
          WRITE(LUOUT,1001)
        ELSE
          WRITE(LUOUT,1002)
        ENDIF
      ENDIF
      DO 10 JCOUNT=1,MAXITER+1
        IF (JFLAG.NE.4) THEN
          IF (JFLAG.LE.1) ERRORJ = ERROR
          ALPHA  = AOAAPP
          DTIME  = DTIMEJ
          FDIST  = ZERO
          FPVTAS = FPVTASJ
          ROCTDJ = ROCTD
          THRUST = FNAPP
          TIME   = DBLE(ZERO)
          XLF    = XLFLARE
          XLFMAX = XLFLARE
          LINENUM = 6
          IF(WRITITR) WRITE(LUOUT,1003) XLF,SINKTD,FACTOR,ERROR,JFLAG,
                                     JCOUNT
          IF(HCLEAR.LE.HFLARE)THEN
            IF(WRITITR) WRITE(LUOUT,1004)
          ELSE
            STEADY = .TRUE.
            CALL APPROCH(ALPHA,GAMMAPP,FDIST,HAGL,HCLEAR,HFLARE,
                        & HRUNWAY)
          ENDIF
          STEADY = .FALSE.
          CALL FLARE(ALPHA,FDIST,ODIST,ROCTD,VKTGS)
          IF(TERMFLG) RETURN
          WRITE(LUOUT,1005) -ROCTD*60.0,JFLAG,JCOUNT
          DELTROC = -ROCTD - SINKTD
          IF(SINKTD.NE.ZERO) THEN
            ERROR = DELTROC/SINKTD
          ELSE
            ERROR = DELTROC
          ENDIF
        ENDIF
      ENDIF

```

```

FLARENZ 30
FLARENZ 31
FLARENZ 32
FLARENZ 33
FLARENZ 34
FLARENZ 35
FLARENZ 36
FLARENZ 37
FLARENZ 38
FLARENZ 39
FLARENZ 40
FLARENZ 41
FLARENZ 42
FLARENZ 43
FLARENZ 44
FLARENZ 45
FLARENZ 46
FLARENZ 47
FLARENZ 48
FLARENZ 49
FLARENZ 50
FLARENZ 51
FLARENZ 52
FLARENZ 53
FLARENZ 54
FLARENZ 55
FLARENZ 56
FLARENZ 57
FLARENZ 58
FLARENZ 59
FLARENZ 60
FLARENZ 61
FLARENZ 62
FLARENZ 63
FLARENZ 64
FLARENZ 65
FLARENZ 66
FLARENZ 67
FLARENZ 68
FLARENZ 69
FLARENZ 70
FLARENZ 71
FLARENZ 72
FLARENZ 73
FLARENZ 74
FLARENZ 75
FLARENZ 76

```

IF (JFLAG.LT.3 .AND. ABS(DELTROC).GE.EPSROC/60.0) THEN	FLARENZ	77
XLFJ = XLF	FLARENZ	78
C****	FLARENZ	79
C****	FLARENZ	80
GUESS NEW LOAD FACTOR BASED ON ERROR--CALL ITRLND.	FLARENZ	81
CALL ITRLND(ERROR,ERRORJ,XLFLARE,FACTOR,TOLRNCE,JFLAG)	FLARENZ	82
IF(JCOUNT.GT.25 .OR.	FLARENZ	83
& XLFLARE.GT.BOUNDU .OR. XLFLARE.LE.1.0) THEN	FLARENZ	84
C****	FLARENZ	85
RESET PARAMETERS TO PRE-FLARE SETTINGS.	FLARENZ	86
ALPHA = AOAAPP	FLARENZ	87
DTIME = DTIMEJ	FLARENZ	88
FPVTAS = FPVTASJ	FLARENZ	89
THRUST = FNAPP	FLARENZ	90
XLF = XLFLARE	FLARENZ	91
XLFMAX = BOUNDU	FLARENZ	92
WRITE(LUOUT,1006)XLFLARE,ROCTD,SINKTD,ERROR,ERRORJ,	FLARENZ	93
& JCOUNT,JFLAG	FLARENZ	94
RETURN	FLARENZ	95
ENDIF	FLARENZ	96
ELSE	FLARENZ	97
JFLAG = 4	FLARENZ	98
ENDIF	FLARENZ	99
ENDIF	FLARENZ	100
10 CONTINUE	FLARENZ	101
1001 FORMAT('↑ ** LANDING **')	FLARENZ	102
1002 FORMAT(' ** LANDING **')	FLARENZ	103
1003 FORMAT('↑ XLFLARE, SINKTD, FACTOR, ERROR, JFLAG, JCOUNT=',	FLARENZ	104
& F11.4, 2F8.2, F8.4, 2I8 ,)	FLARENZ	105
1004 FORMAT(' TIME DIST. WEIGHT ALT. VCAS VTAS VTGS',	FLARENZ	106
& ' ACCEL CL CD THETA ALPHA GAMMA DTHET ',	FLARENZ	107
& ' R/C LOAD THRUST XENG',	FLARENZ	108
& ' (SEC) (FEET) (LBS) (FEET) (KTS) (KTS) (KTS)',	FLARENZ	109
& ' (FPS2) (DEG) (DEG) (DEG) /DT ',	FLARENZ	110
& ' (FPM) FACT (LBS) OR MU',)	FLARENZ	111
1005 FORMAT(' *** SINK RATE AT TOUCHDOWN =',F7.1,' FEET/MINUTE',	FLARENZ	112
& ' CONVERGENCE FLAG =',I3,	FLARENZ	113
& ' FLARENZ ITERATION NUMBER =',I3,' ***')	FLARENZ	114
1006 FORMAT(' *** FAILED IN FLARENZ',	FLARENZ	115
& ' XLFLARE,ROCTD,SINKTD =',F8.4,2F9.2/,	FLARENZ	116
& ' ERROR,ERRORJ,JCOUNT,JFLAG =',2F7.4,2I3)	FLARENZ	117
C****	FLARENZ	118
C**** TOUCHDOWN RATE WITHIN TOLERANCE; RESET PARAMETERS TO PRE-FLARE	FLARENZ	119
C**** SETTINGS.	FLARENZ	120
ALPHA = AOAAPP	FLARENZ	121
DTIME = DTIMEJ	FLARENZ	122
FPVTAS = FPVTASJ	FLARENZ	123
THRUST = FNAPP	FLARENZ	124
XLF = XLFLARE	FLARENZ	125
XLFMAX = BOUNDU	FLARENZ	126
IF(JCOUNT.GE.1) THEN	FLARENZ	127
XLFLARE = (INTERP(XLF,XLFJ,ROCTD,ROCTDJ,-SINKTD) + 2.0*XLF)/3.0	FLARENZ	128
ENDIF	FLARENZ	129
RETURN		
END		

Subroutine APPROCH

SUBROUTINE APPROCH(ALPHA,GAMMAPP,GDIST,HAGL,HCLEAR,HFLARE,HRUNWAY)	APPROCH	1
C**** THIS SUBROUTINE CCNROLS THE EXECUTION OF THE LANDING APPROACH	APPROCH	2
C**** WHEN THE OBSTACLE CLEARANCE HEIGHT, HCLEAR, IS ABOVE THE FLARE	APPROCH	3
C**** HEIGHT, HFLARE. USING THE VALUES FROM SUBROUTINE STEDYST AS	APPROCH	4
C**** CONSTANTS, APPROCH DOES NOT USE RUNGE-KUTTA NUMERICAL INTEGRATION.	APPROCH	5
C****	APPROCH	6
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,OS,	AERO	1
& SPDBRK,SPOILER,VKCAS,VKTAS	AERC	2
COMMON/ENGINE/ AIT,AMACH,DTFAIL,EPR,FE,FG,FGPCT,NENG,PWRCODE,	ENGINE	1
& REVNDX,RC,THRUST,VTANGLE,WFUEL,XENG,XENGFLD	ENGINE	2
& XENGOUT,XIDLE,XMIL,ZFN	ENGINE	3
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
COMMON/ATMOS/ TEMPR,PRESS,RHO,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMPF	ATMOS	1
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITTR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	8
REAL ATMARY(8),S(48),INTERP	APPROCH	16
EQUIVALENCE (TEMPR,ATMARY(1)),(FPVTAS,S(1))	APPROCH	17
DATA APPDTIM0.5/	APPROCH	18
C****	APPROCH	19
C**** WRITE HEADER RECORD FOR OUTPUT.	APPROCH	20
IF(WRITTR) WRITE(LUOUT,1001)	APPROCH	21
1001 FORMAT(' TIME DIST. WEIGHT ALT. VCAS VTAS VTGS',	APPROCH	22
& ' ACCEL CL CD THETA ALPHA GAMMA DTHET ',	APPROCH	23
& ' R/C LOAD THRUST 'ENG',	APPROCH	24
& ' (SEC) (FEET) (L) (KTS) (KTS) (KTS)',	APPROCH	25
& ' (FPS2) (DEG) (DEG) (DEG) /DT ',	APPROCH	26
& ' (FPM) FACT (LBS) OR MU',)	APPROCH	27
GAMAPPR = GAMMAPP/RX	APPROCH	28
GAMMAR = GAMAPPR - ASIN((VWIND/FPVTAS)*SIN(GAMAPPR))	APPROCH	29
VAPP = VKAPP*FPSKTS	APPROCH	30

C****		APPROCH	31
C****	CALCULATE SINK RATE AT APPROACH IN FEET/MINUTE.	APPROCH	32
	ROC = FPVTAS*SIN(GAMMAR)	APPROCH	33
	SINKRT = -ROC*60.0	APPROCH	34
C****		APPROCH	35
C****	CALCULATE VARIABLES FOR OUTPUT ON APPROACH.	APPROCH	36
	DELTAH = -ROC*APPDTIM	APPROCH	37
	DGDIST = -DELTAH/TAN(GAMAPPR) - VWIND*APPDTIM	APPROCH	38
	DTDT = FPACCEL = ZERO	APPROCH	39
	GDIST = -DGDIST	APPROCH	40
	HAGL = HCLEAR + DELTAH	APPROCH	41
	OVERFLG = .FALSE.	APPROCH	42
	TIME = DBLE(-APPDTIM)	APPROCH	43
	CALL SPEED(GAMMAR,FPVTAS,VWIND,AMACH,QS,VKCAS,VKEAS,VKTAS, VKTGS,	APPROCH	44
	VTGS)	APPROCH	45
	VKCAS = VKAPP	APPROCH	46
	XLF = 1.0	APPROCH	47
19	HAGL = HAGL - DELTAH	APPROCH	48
	FPVTASJ = FPVTAS	APPROCH	49
	IF(HAGL.GT.HFLARE) THEN	APPROCH	50
C****	OBTAIN ATMOSPHERIC VARIABLES.	APPROCH	51
	PRESALT = HRUNWAY + HAGL	APPROCH	52
	CALL ATMOSPH(PRESALT,ATMARY)	APPROCH	53
	CALL FORCEX(ALPHA,CD,CL)	APPROCH	54
	FPVTAS = VAPP/SC*SIGMA	APPROCH	55
	FPACCEL = (FPVTAS - FPVTASJ)/APPDTIM	APPROCH	56
	GDIST = GDIST + DGDIST	APPROCH	57
	HAGLJ = HAGL	APPROCH	58
	TIME = TIME + DBLE(APPDTIM)	APPROCH	59
	XLF = 1.0	APPROCH	60
	THETA F = ALPHA + GAMMAR*RX	APPROCH	61
	CALL SPEED(GAMMAR,FPVTAS,VWIND,AMACH,QS,VKCAS,VKEAS,VKTAS, VKTGS,	APPROCH	62
	VTGS)	APPROCH	63
	VKCAS = VKAPP	APPROCH	64
	LINENUM = LINENUM + 1	APPROCH	65
	IF(WRITITR) THEN	APPROCH	66
	IF(APPDTIM.GE.0.10) THEN	APPROCH	67
	WRITE(LUOUT,1002) TIME,GDIST,GWT,HAGL,VKCAS,VKTAS,VKTGS,	APPROCH	68
&	FPACCEL,CL,CD,THETA F,ALPHA,GAMMAPP,	APPROCH	69
&	DTDT,-SINKRT,XLF,THRUST,XENG	APPROCH	70
	ELSE	APPROCH	71
	WRITE(LUOUT,1003) TIME,GDIST,GWT,HAGL,VKCAS,VKTAS,VKTGS,	APPROCH	72
&	FPACCEL,CL,CD,THETA F,ALPHA,GAMMAPP,	APPROCH	73
&	DTDT,-SINKRT,XLF,THRUST,XENG	APPROCH	74
	ENDIF	APPROCH	75
	ENDIF	APPROCH	76
1002	FORMAT(F6.1,F8.0,F9.0,F8.1,3F7.1,F7.2,F8.4,F7.4,4F7.2,F8.1,	APPROCH	77
&	F6.2,F9.0,F7.3)	APPROCH	78
1003	FORMAT(F6.2,F8.0,F9.0,F8.1,3F7.1,F7.2,F8.4,F7.4,4F7.2,F8.1,	APPROCH	79
&	F6.2,F9.0,F7.3)	APPROCH	80
	GO TO 19	APPROCH	81
	ENDIF	APPROCH	82

C****	APPROCH	83
C**** CALCULATE TIME PLUS DTIMEJ FOR TIME WITHOUT ALTITUDE	APPROCH	84
C**** EXTRAPOLATION. THEN, AFTER GROUND DISTANCE AND TIME ARE	APPROCH	85
C**** EXTRAPOLATED FOR THE EXACT FLARE HEIGHT, HFLARE, CALCULATE	APPROCH	86
C**** NEW DELTA TIME (WHICH WILL ALWAYS BE LESS THAN THE ORIGINAL	APPROCH	87
C**** DELTA TIME, DTIMEJ) SO THAT THE ELAPSED TIME WILL BE AN EVEN	APPROCH	88
C**** MULTIPLE OF THE ORIGINAL DELTA TIME, DTIMEJ. LATER IN	APPROCH	89
C**** SUBROUTINE FLARE, DTIME WILL BE RESET TO THE ORIGINAL DELTA	APPROCH	90
C**** TIME, DTIMEJ	APPROCH	91
TTAP PDT = FLOAT(TIME) + APPDTIM	APPROCH	92
TTD TIME = FLOAT(TIME + DTIME)	APPROCH	93
C****	APPROCH	94
C**** EXTRAPOLATE VALUES FOR DISTANCE AND TIME TO CORRESPOND TO	APPROCH	95
C**** ALTITUDE HFLARE.	APPROCH	96
GDIST = INTERP(GDIST+DGDIST ,GDIST ,HAGL,HAGLJ,HFLARE)	APPROCH	97
TIME = INTERP(FLOAT(TIME+APPDTIM) ,FLOAT(TIME),HAGL,HAGLJ,HFLARE)	APPROCH	98
IF(APPDTIM.GT.FLOAT(DTIME)) THEN	APPROCH	99
IF(TTDTIME-FLOAT(TIME).LE.ZERO) THEN	APPROCH	100
DTIME = DBLE(TTAP PDT) - TIME	APPROCH	101
ELSE	APPROCH	102
DTIME = DBLE(TTDTIME) - TIME	APPROCH	103
ENDIF	APPROCH	104
IMINUS = INT(APPDTIM/FLOAT(DTIMEJ)) + 1	APPROCH	105
DO 10 I=1,IMINUS	APPROCH	106
IF(DTIME.GT.DTIMEJ) DTIME = DTIME - DTIMEJ	APPROCH	107
10 CONTINUE	APPROCH	108
ELSE	APPROCH	109
IF(TTAP PDT-FLOAT(TIME).LE.ZERO) THEN	APPROCH	110
DTIME = DBLE(TTDTIME) - TIME	APPROCH	111
ELSE	APPROCH	112
DTIME = DBLE(TTAP PDT) - TIME	APPROCH	113
ENDIF	APPROCH	114
IMINUS = INT(FLOAT(DTIMEJ)/APPDTIM) + 1	APPROCH	115
DO 20 I=1,IMINUS	APPROCH	116
IF(FLOAT(DTIME).GT.APPDTIM) DTIME = DTIME - DBLE(APPDTIM)	APPROCH	117
20 CONTINUE	APPROCH	118
ENDIF	APPROCH	119
RETURN	APPROCH	120
END	APPROCH	121

Subroutine FLARE

SUBROUTINE FLARE(ALPHA,GDIST,ODIST,ROCTD,VKTGS)	FLARE	1
C**** THIS SUBROUTINE CALCULATES THE FLARE MANEUVER BY RUNGE-KUTTA	FLARE	2
C**** NUMERICAL INTEGRATION. THE INTEGRATION STEP SIZE IS DTIME	FLARE	3
C**** SECONDS. THE INPUT VARIABLE, HFLARE, IS THE HEIGHT AT WHICH THE	FLARE	4
C**** FLARE IS STARTED.	FLARE	5
C****	FLARE	6
C**** THE VARIABLE DTDGEX IS PROVIDED TO THE USER AS A MEANS TO ACCOUNT	FLARE	7
C**** FOR A LOSS IN THE PITCH RATE CAPABILITY DUE TO GROUND EFFECT.	FLARE	8
C**** DTDGEX MAY BE A FUNCTION OF WING HEIGHT/WING SURFACE AREA, TOTAL	FLARE	19
C**** LIFT COEFFICIENT, C _L ; CIRCULATION LIFT COEFFICIENT; ETC. DTDGEX	FLARE	10
C**** WOULD EQUAL 1.0 OUT OF GROUND EFFECT AND LESS THAN 1.0 IN GROUND	FLARE	11
C**** EFFECT. DTDGEX SHOULD BE CALCULATED IN SUBROUTINE 'FXAERO',	FLARE	12
C**** PASSED THROUGH THE CALLING STATEMENT TO SUBROUTINE FORCEX. THE	FLARE	13
C**** USER MAY IGNORE DTDGEX WITHOUT AFFECTING PROGRAM EXECUTION.	FLARE	14
C****	FLARE	15
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DDTDMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AERC/ CX,CY,DADTCMD,DCDX,DCLX,DDTDGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKCAS,VKTAS	AERO	2
COMMON/ENGINE/ AIT,AMACH,DTFAIL,EPR,FE,FG,FGPCT,NENG,PWRCODE,	ENGINE	1
& REVNDX,RC,THRUST,VTANGLE,WFUEL,XENG,XENGFLD,	ENGINE	2
& XENGOUT,XIDLE,XMIL,ZFN	ENGINE	3
COMMON/AIRBORN/ ALPHAJ,ALPHMX,DDTD,GAMMAPP,ROCFPM,THETAJ,XLF,XLFJ	AIRBORN	1
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
COMMON/RUNWAY/ ABARG,AOAABRK,BRAKMU,BRKFCR,GAMMARW,GRW,HAGL,	RUNWAY	1
& HCLEAR,HFLARE,HGEAR,HRUNWAY,IMU,RCR,ROLLMU,TIMEFLD,	RUNWAY	2
& TIMEBRK,TIMEFLP,TIMESBK,TIMESPL,XMU	RUNWAY	3
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
COMMON/ATMOS/ TEMPR,PRESS,RHO,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMPF	ATMOS	1
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRS,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITTR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRS,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	8

LOGICAL INIRKI,INIRKJ	FLARE	27
REAL ATMARY(8),S(48),INTERP	FLARE	28
EQUIVALENCE (TEMPR,ATMARY(1)),(FPVTAS,S(1))	FLARE	29
EXTERNAL DERIVAL,INTERP	FLARE	30
C****	FLARE	31
C**** AIRBORNE INTEGRATION VARIABLES	FLARE	32
C**** NEQ = NUMBER OF EQUATIONS	FLARE	33
C**** DTIME = TIME INTERVAL, STEP SIZE (SECONDS)	FLARE	34
C**** FPVTAS = VELOCITY ALONG FLIGHT PATH (FEET/SECOND)	FLARE	35
C**** GAMMAR = FLIGHT PATH ANGLE (RADIAN)	FLARE	36
C**** FPDIST = DISTANCE ALONG FLIGHT PATH (FEET)	FLARE	37
C**** PRESALT = ALTITUDE (FEET)	FLARE	38
C**** FPACCEL = ACCELERATION ALONG FLIGHT PATH (FEET/SECOND**2)	FLARE	39
C**** DGDTR = TIME RATE OF CHANGE OF FLIGHT PATH ANGLE	FLARE	40
C**** (RADIAN/SECOND)	FLARE	41
C**** VHAS = HORIZONTAL SPEED (FEET/SECOND)	FLARE	42
C**** ROC = RATE OF CLIMB (FEET/SECOND)	FLARE	43
C****	FLARE	44
C**** INITIALIZE VARIABLES FOR FLARE INTEGRATION LOOP.	FLARE	45
GAMAPPR = GAMMAP/RX	FLARE	46
GAMMAR = GAMAPPR - ASIN((VWIND/FPVTAS)*SIN(GAMAPPR))	FLARE	47
HAGL = HFLARE	FLARE	48
ICOUNT = 1	FLARE	49
INIRKI = .FALSE.	FLARE	50
ODIST = ZERO	FLARE	51
C****	FLARE	52
C**** FLARE INTEGRATION LOOP	FLARE	53
19 DO 10 NCOUNT=ICOUNT,10	FLARE	54
PRESALT = HAGL + HRUNWAY	FLARE	55
ALPHAJ = ALPHA	FLARE	56
C****	FLARE	57
C**** OBTAIN ATMOSPHERIC VARIABLES.	FLARE	58
CALL ATMOSPH(PRESALT,ATMARY)	FLARE	59
CALL FORCEX(ALPHA,CD,CL)	FLARE	60
CALL SPEED(GAMMAR,FPVTAS,VWIND,AMACH,QS,VKCAS,VKEAS,VKTAS,VKTGS,	FLARE	61
VTGS)	FLARE	62

	IF(.NOT.INIRKI) THEN	FLARE	63
C****	MAKE INITIAL INTEGRATION STEP.	FLARE	64
	CALL INTX(NEQ,TIME,DTIME,S,DERIVAL,ALPHA)	FLARE	65
	ROC = FPVTAS*SIN(GAMMAR)	FLARE	66
	SINKRT = -ROC*60.0	FLARE	67
	THETAF = ALPHA + GAMMAR*RX	FLARE	68
	GAMMA = ATAN(ROC/VTGS)*RX	FLARE	69
	DTDT = ZERO	FLARE	70
	IF(DTIMEJ.EQ.DTIME) THEN	FLARE	71
C****	NO SHORT TIME INCREMENT IN THIS CALL TO FLARE.	FLARE	72
	INIRKI = .TRUE.	FLARE	73
	INIRKJ = .FALSE.	FLARE	74
	ELSE	FLARE	75
C****	A SHORT TIME INCREMENT IN THIS CALL TO FLARE IS NEEDED	FLARE	76
C****	TO BRING THE ELAPSED TIME TO AN EVEN MULTIPLE OF DELTA	FLARE	77
C****	TIME FOR THIS INTEGRATION STEP ONLY.	FLARE	78
	INIRKI = .TRUE.	FLARE	79
	INIRKJ = .TRUE.	FLARE	80
	ENDIF	FLARE	81
	ELSE	FLARE	82
C****	INCREASE ANGLE OF ATTACK AT A RATE OF DADTCMD*DTDTGEX	FLARE	83
C****	DEGREES PER SECOND.	FLARE	84
	CALL PITCH('LANDING',ALPHA,DADTCMD,DTIME,DTDTGEX,LUOUT)	FLARE	85
C****		FLARE	86
C****	MAKE INTEGRATION STEP.	FLARE	87
	CALL INTZ(NEQ,TIME,DTIME,S,DERIVAL,ALPHA)	FLARE	88
	IF(ERRFLAG) THEN	FLARE	89
C****	TERMINATE LANDING EXECUTION.	FLARE	90
	TERMFLG = .TRUE.	FLARE	91
	RETURN	FLARE	92
	ELSEIF(INIRKJ) THEN	FLARE	93
C****	A SHORT TIME INCREMENT IN THIS CALL TO FLARE WAS NEEDED	FLARE	94
C****	TO BRING THE ELAPSED TIME TO AN EVEN MULTIPLE OF DELTA	FLARE	95
C****	TIME FOR THIS INTEGRATION STEP ONLY; RESET DTIME TO THE	FLARE	96
C****	ORIGINAL VALUE, DTIMEJ. SET DTIMEJ TO ZERO SO THE	FLARE	97
C****	FLAGS INIRKI AND INIRKJ WILL BE RESET TO PROPER VALUES.	FLARE	98
	INIRKI = .FALSE.	FLARE	99
	ICOUNT = ICOUNT - 1	FLARE	100
	DTIME = DTIMEJ	FLARE	101
	ENDIF	FLARE	102
	GAMMA = ATAN(ROC/VTGS)*RX	FLARE	103
	GDISTJ = GDIST	FLARE	104
	CALL INTG(DIST,FIKAIR(27),DTIME,GAMMAR,VWIND,0.0,	FLARE	105
	DDIST,GDIST,GWT)	FLARE	106

HAGLJ = HAGL	FLARE	107
HAGL = PRESALT - HRUNWAY	FLARE	108
SINKRT = -ROC*60.0	FLARE	109
TIMEJ = FLOAT(TIME - DTIME)	FLARE	110
C****	FLARE	111
C****	FLARE	112
OBTAIN SINK RATE AT TOUCHDOWN.	FLARE	113
IF (HAGL.LE.0.10) THEN	FLARE	114
STORE CURRENT TIME IN TTDTIME VARIABLE. THEN, AFTER	FLARE	115
GROUND DISTANCE AND TIME ARE INTERPOLATED FOR THE	FLARE	116
TOUCHDOWN POINT, CALCULATE NEW DELTA TIME (WHICH	FLARE	117
WILL ALWAYS BE LESS THAN THE ORIGINAL DELTA TIME,	FLARE	118
DTIMEJ) SO THAT THE ELAPSED TIME WILL BE AN EVEN	FLARE	119
MULTIPLE OF THE ORIGINAL DELTA TIME, DTIMEJ. LATER IN	FLARE	120
SUBROUTINE ROLL, DTIME WILL BE RESET TO THE ORIGINAL	FLARE	121
DELTA TIME, DTIMEJ.	FLARE	122
NPTS = INT(TIME/DTIME)	FLARE	123
ICOUNT = NPTS - INT(10.0*FLOAT(NPTS/10)) - 1	FLARE	124
TTDTIME = FLOAT(TIME)	FLARE	125
GDIST = INTERP(GDIST ,GDISTJ ,HAGL,HAGLJ,ZERO)	FLARE	126
ROCTD = INTERP(ROC ,ROCJ ,HAGL,HAGLJ,ZERO)	FLARE	127
TIME = INTERP(FLOAT(TIME) ,TIMEJ ,HAGL,HAGLJ,ZERO)	FLARE	128
DTIME = DBLE(TTDTIME) - TIME	FLARE	129
RETURN	FLARE	130
ELSEIF (ROC.GE.ZERO) THEN	FLARE	131
NPTS = INT(TIME/DTIME)	FLARE	132
ICOUNT = NPTS - INT(10.0*FLOAT(NPTS/10)) - 1	FLARE	133
ROCTD = ZERO	FLARE	134
RETURN	FLARE	135
ENDIF	FLARE	136
ROCJ = ROC	FLARE	137
ENDIF		

	IF(HAGLLT.ZERO) HAGL = 0.10	FLARE	138
	IF(OVERFLG .AND. HAGLLT.HCLEAR) THEN	FLARE	139
C****	INTERPOLATE TO FIND DISTANCE AT OBSTACLE CLEARANCE HEIGHT,	FLARE	140
C****	HCLEAR, IF THE FLARE HEIGHT, HFLARE, IS HIGHER THAN THE	FLARE	141
C****	OBSTACLE CLEARANCE HEIGHT, HCLEAR.	FLARE	142
	OVERFLG = .FALSE.	FLARE	143
	ODIST = INTERP(GDIST,GDISTJ,HAGL,HAGLJ,HCLEAR)	FLARE	144
	ENDIF	FLARE	145
	IF(WRITTR .AND. (.NOT. INIRKJ .OR. INIRKI)) THEN	FLARE	146
	IF(DTIME.GE.0.10D0) THEN	FLARE	147
	WRITE(LUOUT,1002) TIME,GDIST,GWT,HAGL,VKCAS,VKTAS,VKTGS,	FLARE	148
&	FPACCEL,CL,CD,THETA,ALPHA,GAMMA,	FLARE	149
&	DTDT,-SINKRT,XLF,THRUST,XENG	FLARE	150
	ELSE	FLARE	151
	WRITE(LUOUT,1003) TIME,GDIST,GWT,HAGL,VKCAS,VKTAS,VKTGS,	FLARE	152
&	FPACCEL,CL,CD,THETA,ALPHA,GAMMA,	FLARE	153
&	DTDT,-SINKRT,XLF,THRUST,XENG	FLARE	154
	ENDIF	FLARE	155
	LINENUM = LINENUM + 1	FLARE	156
	IF(LINENUM.GE.NPAGE) THEN	FLARE	157
C****	RESET LINE NUMBER COUNTER LINENUM AND WRITE HEADER FOR	FLARE	158
C****	NEW PAGE.	FLARE	159
	LINENUM = 3	FLARE	160
	WRITE(LUOUT,1001)	FLARE	161
	ENDIF	FLARE	162
	ENDIF	FLARE	163
1001	FORMAT('↑ TIME DIST. WEIGHT ALT. VCAS VTAS VTGS',	FLARE	164
&	' ACCEL CL CD THETA ALPHA GAMMA DTHET ',	FLARE	165
&	' R/C LOAD THRUST XENG'./,	FLARE	166
&	' (SEC) (FEET) (LBS) (FEET) (KTS) (KTS) (KTS)',	FLARE	167
&	' (FPS2) (DEG) (DEG) (DEG) /DT ',	FLARE	168
&	' (FPM) FACT (LBS) OR MU'./)	FLARE	169
1002	FORMAT(F6.1,F8.0,F9.0,F8.1,3F7.1,F7.2,F8.4,F7.4,4F7.2,F8.1,	FLARE	170
&	F6.2,F9.0,F7.3)	FLARE	171
1003	FORMAT(F6.2,F8.0,F9.0,F8.1,3F7.1,F7.2,F8.4,F7.4,4F7.2,F8.1,	FLARE	172
&	F6.2,F9.0,F7.3)	FLARE	173
10	CONTINUE	FLARE	174
	GO TO 19	FLARE	175
	END	FLARE	176

Subroutine ROLL

SUBROUTINE ROLL(ALPHA,GDIST)	ROLL	1
C**** THIS SUBROUTINE CONTROLS THE EXECUTION OF THE GROUND ROLL OF THE	ROLL	2
C**** LANDING MANEUVER OR THE DECELERATION PORTION OF A REFUSED TAKEOFF	ROLL	3
C**** MANEUVER. SUBROUTINE ROLL CALLS INTX TO PERFORM THE NUMERICAL	ROLL	4
C**** INTEGRATION OF THE RESULTANTS OF THE EQUATIONS OF MOTION AND CALLS	ROLL	5
C**** FORCEX TO OBTAIN THE FORCE COEFFICIENTS FOR THE EQUATIONS	ROLL	6
C**** OF MOTION.	ROLL	7
C****	ROLL	8
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKAS,VKTAS	AERO	2
COMMON/ENGINE/ AIT,AMACH,DTFAIL,EPR,FE,FG,FGPCT,NENG,PWRCODE,	ENGINE	1
& REVNDX,RC,THRUST,VTANGLE,WFUEL,XENG,XENGFLD,	ENGINE	2
& XENGOUT,XIDLE,XMIL,ZFN	ENGINE	3
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
COMMON/RUNWAY/ ABARG,AOAABRK,BRAKMU,BRKFCR,GAMMARW,GRW,HAGL,	RUNWAY	1
& HCLEAR,HFLARE,HGEAR,HRUNWAY,IMU,RCR,ROLLMU,TIMEFLD,	RUNWAY	2
& TIMEBRK,TIMEFLP,TIMESBK,TIMESPL,XMU	RUNWAY	3
DOUBLEPRECISION VTAS,DIST,ACCEL,VTASJ,RKGRND	INTEG	1
COMMON/INTEG/ VTAS,DIST,ACCEL,VTASJ,RKGRND(20)	INTEG	2
COMMON/ATMOS/ TEMPR,PRESS,RHO,ALF,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMP,F	ATMOS	1
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITTR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	8
CHARACTER ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
& THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/ ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
& TKOTYPE	CHARV	4

LOGICAL IDLE,IDLFLAG,INIRKI,INIRKJ,SPOOLF,THROTTL	ROLL	20
REAL ATMARY(8),T(24)	ROLL	21
EQUIVALENCE (TEMPR,ATMARY(1)),(VTAS,T(1))	ROLL	22
EXTERNAL DERIVGR	ROLL	23
DATA AOAABRK, BRAKMU,IMU, RCR, TIMEBRK,TIMEFLP,TIMEIDL, TIMEREV	ROLL	24
& / 0.0, 0.250, 0, 0.0, 3.0, 999., 3.0, 0.0/,	ROLL	25
& TIMESBK, TIMESPL, VKABRK	ROLL	26
& / 0.0, 0.0, 0.0/	ROLL	27
NAMELIST/ROL/BRAKMU,IMU,RCR	ROLL	28
NAMELIST/ROL2/AOAABRK,BRKFCR,TIMEBRK,TIMEFLP,TIMEIDL,TIMEREV,TIMESPL,	ROLL	29
& VKABRK,VKBRAKE	ROLL	30
C****	ROLL	31
C**** RESET SOME INPUT PARAMETERS FROM ANY PREVIOUS RUN (IF ANY).	ROLL	32
C**** PARAMETERS NOT RESET HERE RETAIN THEIR VALUE FOR THE SUBSEQUENT	ROLL	33
C**** RUN IF NOT EXPLICITLY DECLARED IN NAMELIST ROL. AND ROL2.	ROLL	34
BRKFCTR = 1.0	ROLL	35
VKBRAKE = 999.	ROLL	36
C****	ROLL	37
C**** INPUT DATA LOADED INTO ROLL THRU NAMELIST /ROL/ AND /ROL2/.	ROLL	38
READ(LUIN,ROL)	ROLL	39
READ(LUIN,ROL2)	ROLL	40
C****	ROLL	41
C**** INITIALIZE CURVE FILES.	ROLL	42
CALL INICURV	ROLL	43
C****	ROLL	44
C**** INITIALIZE LOAD FACTOR TO 1.0.	ROLL	45
XLF = 1.0	ROLL	46
IF(.NOT.RTOFLAG) THEN	ROLL	47
C**** LANDING	ROLL	48
C****	ROLL	49
C**** INITIALIZE RUNGE-KUTTA INTEGRATION FLAG.	ROLL	50
INIRKI = .FALSE.	ROLL	51
ICOUNT = ICOUNT - 1	ROLL	52
C****	ROLL	53
C**** INITIALIZE ENGINE THROTTLING FLAGS AND VARIABLES.	ROLL	54
IDLFLAG = .FALSE.	ROLL	55
REVRSE = .FALSE.	ROLL	56
IDLE = .TRUE.	ROLL	57
THROTTL = .TRUE.	ROLL	58
XENGEND = FLOAT(NENG)*XIDLE	ROLL	59
XENGTRN = XENG	ROLL	60

ELSE	ROLL	61
C**** REFUSED TAKEOFF	ROLL	62
C****	ROLL	63
C**** INITIALIZE RUNGE-KUTTA INTEGRATION FLAG.	ROLL	64
INIRKI = .TRUE	ROLL	65
C****	ROLL	66
C**** INITIALIZE ENGINE THROTTLING FLAGS AND VARIABLES.	ROLL	67
IF (NENG.GT.1) THEN	ROLL	68
IDLE = .FALSE.	ROLL	69
ELSE	ROLL	70
IDLE = .TRUE.	ROLL	71
ENDIF	ROLL	72
IDLFLAG = .TRUE.	ROLL	73
REVRSE = .FALSE.	ROLL	74
SPOOLF = .FALSE.	ROLL	75
THROTTL = .TRUE.	ROLL	76
C****	ROLL	77
C**** FOR THE FAILED ENGINE(S)	ROLL	78
FENGTRN = XENGFLD	ROLL	79
IF (FAILST.EQ.'IDLE') THEN	ROLL	80
FENGEND = XIDLE	ROLL	81
ELSEIF(FAILST.EQ.' MIL') THEN	ROLL	82
FENGEND = XMIL	ROLL	83
ELSEIF(FAILST.EQ.' OFF') THEN	ROLL	84
FENGEND = ZERO	ROLL	85
ELSE	ROLL	86
FENGEND = 1.0 -XENGFLD	ROLL	87
ENDIF	ROLL	88
IF (FAILMOD.EQ.'SEIZE') THEN	ROLL	89
SPOOLF = .TRUE.	ROLL	90
XENG = XENGOUT	ROLL	91
XENGF = FENGEND	ROLL	93
ELSE	ROLL	94
XENGF = 1.0	ROLL	95
ENDIF	ROLL	96
C****	ROLL	97
C**** FOR THE REMAINING ENGINE(S)	ROLL	98
RENGTRN = FLOAT(INT(XENGOUT))	ROLL	99
RENGEND = FLOAT(NENG - INT(XENGFLD))*XIDLE	ROLL	100
XENGR = FLOAT(INT(XENGOUT))	ROLL	101
XENGTRN = XENGOUT	ROLL	102
XENG = XENGF + XENGR	ROLL	103
ENDIF	ROLL	104
C****	ROLL	105
C**** INITIALIZE AERODYNAMIC VARIABLES.	ROLL	106
CALL ATMOSPH(HRUNWAY,ATMARY)	ROLL	107
CALL FORCEX(ALPHA,CD,CL)	ROLL	108
C****	ROLL	109
C**** RUNGE-KUTTA INTEGRATION OF GROUND ROLL	ROLL	110
NEQ = 2	ROLL	111
HAGL = ZERO	ROLL	112

C****		ROLL	113
C****	GROUND ROLL INTEGRATION LOOP	ROLL	114
	IF(ICOUNT.EQ.11) ICOUNT = 1	ROLL	115
	19 DO 10 NCOUNT=ICOUNT,10	ROLL	116
	IF(.NOT.INIRKI) THEN	ROLL	117
C****	INITIALIZE INTEGRATION VARIABLES.	ROLL	118
	CALL INTX(NEQ,TIME,DTIME,T,DERIVGR,ALPHA)	ROLL	119
	CALL INTG(DIST,DISTJ,DTIME,0.0,VWIND,WFUEL,	ROLL	120
	DDIST,GDIST,GWT)	ROLL	121
	IF(DTIMEJ.EQ.0.0D0) THEN	ROLL	122
C****	NO SHORT TIME INCREMENT IN THIS CALL TO ROLL	ROLL	123
	INIRKI = .TRUE.	ROLL	124
	INIRKJ= .FALSE.	ROLL	125
	ELSE	ROLL	126
C****	A SHORT TIME INCREMENT IN THIS CALL TO ROLL IS NEEDED	ROLL	127
C****	TO BRING THE ELAPSED TIME TO AN EVEN MULTIPLE OF DELTA	ROLL	128
C****	TIME FOR THIS INTEGRATION STEP ONLY.	ROLL	129
	INIRKI = .TRUE.	ROLL	130
	INIRKJ= .TRUE.	ROLL	131
	ENDIF	ROLL	132

	ELSE	ROLL	133
	IF(.NOT.(AOA0FLG)) THEN	ROLL	134
C****	FOR AIRCRAFT NOT AT A THREE POINT ATTITUDE, SET THE	ROLL	135
C****	TARGET ANGLE OF ATTACK TO EITHER AOAABRK OR AOA3PT	ROLL	136
C****	DEGREES.	ROLL	137
	ALPHAJ = ALPHA	ROLL	138
	IF(AOAABRK.EQ.ZERO .OR. VKABRK.EQ.ZERO) THEN	ROLL	139
	ALPHTGT = AOA3PT	ROLL	140
	ELSE	ROLL	141
	IF(VKCAS.GT.VKABRK) THEN	ROLL	142
	ALPHTGT = AOAABRK	ROLL	143
	ELSE	ROLL	144
	ALPHTGT = AOA3PT	ROLL	145
	ENDIF	ROLL	146
	ENDIF	ROLL	147
	DALPHA = ABS(ALPHTGT - ALPHA)	ROLL	148
	IF(DALPHA.GT.DADTCMD*FLOAT(DTIME)) THEN	ROLL	149
C****	CHANGE ANGLE OF ATTACK UNTIL EQUAL TO ALPHTGT.	ROLL	150
	IF (ALPHA.LT.ALPHTGT) THEN	ROLL	151
	CALL PITCH('ROLL ',ALPHA,DADTCMD,DTIME,DTDTGEX,	ROLL	152
&	LUOUT)	ROLL	153
	ELSE IF (ALPHA.GE.ALPHTGT) THEN	ROLL	154
	CALL PITCH('ROLL ',ALPHA,-DADTCMD,DTIME,DTDTGEX,	ROLL	155
&	LUOUT)	ROLL	156
	ENDIF	ROLL	157
	DADT = (ALPHA - ALPHAJ)/FLOAT(DTIME)	ROLL	158
	ELSE	ROLL	159
C****	IF THE ANGLE OF ATTACK IS WITHIN ONE PITCH INCREMENT	ROLL	160
C****	OF THE TARGET ANGLE OF ATTACK, SET ALPHA EQUAL TO	ROLL	161
C****	ALPHTGT.	ROLL	162
	ALPHA = ALPHTGT	ROLL	163
	DADT = ZERO	ROLL	164
	DADTCMD = ZERO	ROLL	165
	IF(ALPHTGT.EQ.ZERO) AOA0FLG = .TRUE.	ROLL	166
	ENDIF	ROLL	167
	IF(QS*CY.GE.GV**COSD(GAMMARW)) THEN	ROLL	168
C****	LIFTOFF CRITERION MET.	ROLL	169
	LIFTOFF = .TRUE.	ROLL	170
	LINENUM = LINENUM + 2	ROLL	171
	XMU = ZERO	ROLL	172
	WRITE(LUOUT,1001) ALPHA,VKCAS,ROLLMU	ROLL	173
1001	FORMAT(' *** RUN-TIME ERROR. LIFT GREATER THAN'	ROLL	174
&	' WEIGHT DURING AEROBRAKING.',	ROLL	175
&	' ALPHA = ',F6.2,' CAS = ',F7.1,' KNOTS ',	ROLL	176
&	' COEFFICIENT OF FRICTION = ',F6.3)	ROLL	177
	ELSE	ROLL	178
	LIFTOFF = .FALSE.	ROLL	179
	XMU = ROLLMU	ROLL	180
	ENDIF	ROLL	181
	ENDIF	ROLL	182

	TIMEROL = FLOAT(TIME) - TIMEFLD	ROLL	183
	IF(.NOT.(BRKFLAG) .AND.	ROLL	184
&	(VKCAS.LT.VKBRAKE .AND. TIMEROL.GE.TIMEBRK)) THEN	ROLL	185
C****	BRAKE APPLICATION	ROLL	186
	BRKFLAG = .TRUE.	ROLL	187
	BRKENGY = 0.5*(GWT/G)*VTAS**2	ROLL	188
	XMU = BRAKMU	ROLL	189
	LINENUM = LINENUM + 1	ROLL	190
	WRITE(LUOUT,1002) TIME,GDIST,BRAKMU,VKCAS,VKTAS	ROLL	191
1002	FORMAT(' BRAKE APPLICATION (TIME = ',F6.2,	ROLL	192
&	' DIST = ',F8.1,' BRAKING COEFFICIENT = ',	ROLL	193
&	F6.3,' CAS = ',F7.1,' KNOTS TAS = ',	ROLL	194
&	F7.1,' KNOTS)')	ROLL	195
	ENDIF	ROLL	196
	IF(RTOFLAG) THEN	ROLL	197
C****	FAILED ENGINE TRANSIENT	ROLL	198
	IF (DTFAIL.EQ.ZERO .AND. (.NOT.SPOOLF)) THEN	ROLL	199
	CALL SPOOLDNF(TIME,FENGEND,FENGTRN,SPOOLF,XENGF,LUMSG)	ROLL	200
C****		ROLL	201
C****	SUM ENGINE MULTIPLICATIVE FACTORS.	ROLL	202
	XENG = XENGF + XENGR	ROLL	203
	ELSEIF (DTFAIL.NE.ZERO .AND. (.NOT.SPOOLF)) THEN	ROLL	204
	XENGF = FENGTRN - (FENGTRN - FENGEND)*TIMEROL/DTFAIL	ROLL	205
C****		ROLL	206
C****	SUM ENGINE MULTIPLICATIVE FACTORS.	ROLL	207
	XENG = XENGF + XENGR	ROLL	208
	IF(XENGF.LE.FENGEND) THEN	ROLL	209
	SPOOLF = .TRUE.	ROLL	210
	XENGF = FENGEND	ROLL	211
	ENDIF	ROLL	212
	ENDIF	ROLL	213
	ELSE	ROLL	214
	XENGF = ZERO	ROLL	215
	ENDIF	ROLL	216
	IF((NCOUNT.EQ.10.OR.STEADY) .AND.	ROLL	217
&	(JDEBUG.EQ.6666.OR.JDEBUG.EQ.9999)) THEN	ROLL	218
	LINENUM = LINENUM + 2	ROLL	219
	WRITE(LUMSG,6666) THROTTL,IDLFLAG,SPOOLF,IDLE,	ROLL	220
&	XENGF,XENGR,XENG	ROLL	221
6666	FORMAT(' THROTTL IDLFLAG SPOOLF IDLE XENGF,XENGR,XENG =',	ROLL	222
&	/, 2L8, L7, L5, 3F10.3)	ROLL	223
	ENDIF	ROLL	224

	IF(THROTTL) THEN	ROLL	225
	IF ((IDLFLAG) .AND. TIMEROL .GE. TIMEIDL) THEN	ROLL	226
C****	RETARD THROTTLE TO IDLE.	ROLL	227
	IF(XENGFLD.NE.1.0 .AND. (.NOT. SPOOLF)) THEN	ROLL	228
C****	FOR PART POWER ENGINE FAILURES MOVE THE REMAINING	ROLL	229
C****	XENGF COMPONENT TO XENGR AND RESET XENGF TO ZERO.	ROLL	230
	SPOOLF = .TRUE.	ROLL	231
	XENGR = XENGF + XENGR	ROLL	232
	XENGF = ZERO	ROLL	233
	IF(DTFAIL.NE.0.0) THEN	ROLL	234
	RENGTRN = XENGOUT	ROLL	235
	ELSE	ROLL	236
	RENGTRN = (RENGEND*TIMEROL - XENGR*DTFAIL)/	ROLL	237
&	(TIMEROL - DTFAIL)	ROLL	238
	ENDIF	ROLL	239
	ENDIF	ROLL	240
	IF (DTFAIL.EQ.ZERO .AND. (.NOT. IDLE)) THEN	ROLL	241
	CALL SPOOLDNR(TIME,RENGEND,RENGTRN,IDLE,XENGR)	ROLL	242
	LUMSG)	ROLL	243
&		ROLL	244
C****	SUM ENGINE MULTIPLICATIVE FACTORS.	ROLL	245
C****	XENG = XENGF + XENGR	ROLL	246
	ELSEIF (DTFAIL.NE.ZERO .AND. (.NOT.IDLE)) THEN	ROLL	247
	XENGR = RENGRN - (RENGTRN - RENGRN)*	ROLL	248
&	TIMEROL/DTFAIL	ROLL	249
	IF(XENGR.LE.RENGEND) THEN	ROLL	250
	IDLE = .TRUE.	ROLL	251
	XENGR = RENGRN	ROLL	252
	ENDIF	ROLL	253
C****	SUM ENGINE MULTIPLICATIVE FACTORS.	ROLL	254
C****	XENG = XENGF + XENGR	ROLL	255
	ELSE	ROLL	256
	IDLFLAG = .FALSE.	ROLL	257
	ENGGRP = 'AEI'	ROLL	258
	IF ((.NOT.REVFLAG) .AND. (.NOT.RTOFLAG)) THEN	ROLL	259
	XENG = FLOAT(NENG)	ROLL	260
	ELSEIF(.NOT.REVFLAG) THEN	ROLL	261
	IF(FAILST.EQ.'IDLE') THEN	ROLL	262
	XENG = FLOAT(NENG)	ROLL	263
	ELSE	ROLL	264
	XENG = FLOAT(NENG - INT(XENGFLD))	ROLL	265
	ENDIF	ROLL	266
	ENDIF	ROLL	267
	ENDIF	ROLL	268
	ENDIF	ROLL	269

	ELSEIF((REVFLAG) .AND. TIMEROL .GE. TIMEREV) THEN	ROLL	270
C****	ADVANCE THROTTLE FOR REVERSE THRUST.	ROLL	271
	IF(.NOT. REVRSE) THEN	ROLL	272
	IF(.NOT. RTOFLAG) ENGGRP = 'AER'	ROLL	273
	CALL SPOOLUP(REVNDX, TIME, 0.0, XENGTRN, REVHSE, XENGR)	ROLL	274
C****		ROLL	275
C****	SUM ENGINE MULTIPLICATIVE FACTORS.	ROLL	276
	XENG = XENGF + XENGR	ROLL	277
	ELSE	ROLL	278
	REVFLAG = .FALSE.	ROLL	279
	ENDIF	ROLL	280
	ELSEIF((.NOT. IDLFLAG) .AND. (.NOT. REVFLAG)) THEN	ROLL	281
	THROTTL = .FALSE.	ROLL	282
	ENDIF	ROLL	283
	ENDIF	ROLL	284
C****		ROLL	285
C****	OBTAIN FORCE COEFFICIENTS, CX AND CY.	ROLL	286
	CALL FORCECX(ALPHA, CD, CL)	ROLL	287
C****		ROLL	288
C****	MAKE INTEGRATION STEP.	ROLL	289
	CALL INTZ(NEQ, TIME, DTIME, T, DERIVGR, ALPHA)	ROLL	290
	CALL INTG(DIST, RKGRND(14), DTIME, 0.0, VWIND, WFUEL,	ROLL	291
	DDIST, GDIST, GWT)	ROLL	292
	CALL SPEED(0.0, VTAS, VWIND, AMACH, QS, VKCAS, VKEAS, VKTAS, VKTGS,	ROLL	293
	VTGS)	ROLL	294
	ACLSUM = ACLSUM + ACCEL	ROLL	295
	BRKENGY = BRKENGY	ROLL	296
	+ (THRUST - CD*QS - GWT*ROLLMU*COS(GRW))*DDIST	ROLL	297
&	IF(INIRKJ) THEN	ROLL	298
C****	A SHORT TIME INCREMENT IN THIS CALL TO ROLL WAS NEEDED	ROLL	299
C****	TO BRING THE ELAPSED TIME TO AN EVEN MULTIPLE OF DELTA	ROLL	300
C****	TIME FOR THIS INTEGRATION STEP ONLY; RESET DTIME TO THE	ROLL	301
C****	ORIGINAL VALUE, DTIMEJ. SET DTIMEJ TO ZERO SO THE	ROLL	302
C****	FLAGS INIRKI AND INIRKJ WILL BE RESET TO PROPER VALUES.	ROLL	303
	INIRKI = .FALSE.	ROLL	304
	DTIME = DTIMEJ	ROLL	305
	DTIMEJ = DBLE(ZERO)	ROLL	306
	ENDIF	ROLL	307

	IF (TIMEROL.GT.90.0) THEN	ROLL	308
C****	LIMIT ON GROUND ROLL OF SIXTY SECONDS.	ROLL	309
	WRITE(LUOUT,1003) BRAKMU,BRKFCR,IMU	ROLL	310
1003	FORMAT(' *** FAILED IN SUBROUTINE ROLL.',	ROLL	311
&	' BRAKMU =',F6.3,' BRKFCR =',F6.3,' IMU =',	ROLL	312
&	I3)	ROLL	313
	RETURN	ROLL	314
	ELSEIF(VKTGS.LT. 1.0) THEN	ROLL	315
C****	WITH GROUND SPEED LESS THAN 1.0 FPS, ASSUME GROUND ROLL	ROLL	316
C****	ENDS AT THE CURRENT DISTANCE BUT AN ADDITIONAL 0.2	ROLL	317
C****	SECONDS. WRITE FINAL OUTPUT.	ROLL	318
	TIME = TIME + 0.2D0	ROLL	319
	IF(DTIME.GE.0.10D0) THEN	ROLL	320
	WRITE(LUOUT,1006) TIME,GDIST,GWT,ZERO,ZERO,ZERO,ZERO,	ROLL	321
&	ACCEL,CL,CD,ZERO,ALPHA,ZERO,ZERO,	ROLL	322
&	ZERO,XLF,THRUST,XMU	ROLL	323
	ELSE	ROLL	324
	WRITE(LUOUT,1007) TIME,GDIST,GWT,ZERO,ZERO,ZERO,ZERO,	ROLL	325
&	ACCEL,CL,CD,ZERO,ALPHA,ZERO,ZERO,	ROLL	326
&	ZERO,XLF,THRUST,XMU	ROLL	327
	ENDIF	ROLL	328
C****		ROLL	329
C****	CALCULATE AVERAGE DECELERATIONS ABAR AND ABARG FOR	ROLL	330
C****	RUNGE-KUTTA INTEGRATION OF GROUND ROLL.	ROLL	331
	ABAR = -ACLSUM/FLOAT(TIME/DTIME)	ROLL	332
	ABARG = ABAR/G	ROLL	333
	WRITE(LUOUT,1004) BRAKMU,BRKFCR,IMU,BRKENGY	ROLL	334
1004	FORMAT(' BRAKMU =',F6.3,' BRKFCR =',F6.3,' IMU =',	ROLL	335
&	I3,' BRAKE ENERGY =',1PE12.5E2,' FOOT-POUNDS')	ROLL	336
	RETURN	ROLL	337
	ELSEIF(NCOUNT.EQ.10) THEN	ROLL	338
C****	CALCULATE ADDITIONAL OUTPUT PARAMETERS.	ROLL	339
	THETA F = ALPHA	ROLL	340
	IF(LINENUM.GE.NPAGE) THEN	ROLL	341
C****	RESET LINE NUMBER COUNTER LINENUM AND WRITE HEADER	ROLL	342
C****	FOR NEW PAGE.	ROLL	343
	LINENUM = 4	ROLL	344
	WRITE(LUOUT,1005)	ROLL	345
	ENDIF	ROLL	346
	ENDIF	ROLL	347
	ENDIF	ROLL	348
10	CONTINUE	ROLL	349

C****	ROLL	350
C**** WRITE OUTPUT AND INCREMENT LINE NUMBER COUNTER LINENUM.	ROLL	351
IF(DTIME.GE.0.10D0) THEN	ROLL	352
WRITE(LUOUT,1006) TIME,GDIST,GWT,ZERO,VKCAS,VKTAS,VKTGS,ACCEL,	ROLL	353
& CL,CD,THETA,ALPHA,ZERO,DADT,ZERO,XLF,THRUST,	ROLL	354
& XMU	ROLL	355
ELSE	ROLL	356
WRITE(LUOUT,1007) TIME,GDIST,GWT,ZERO,VKCAS,VKTAS,VKTGS,ACCEL,	ROLL	357
& CL,CD,THETA,ALPHA,ZERO,DADT,ZERO,XLF,THRUST,	ROLL	358
& XMU	ROLL	359
ENDIF	ROLL	360
LINENUM = LINENUM + 1	ROLL	361
C**** RESET INTEGRATION LOOP COUNTER ICOUNT TO 1 AND RESTART	ROLL	362
C**** THE GROUND ROLL INTEGRATION LOOP.	ROLL	363
ICOUNT = 1	ROLL	364
GO TO 19	ROLL	365
1005 FORMAT('↑ GROUND ROLL CONTINUED',/,	ROLL	366
& ' TIME DIST. WEIGHT ALT. VCAS VTAS VTGS',	ROLL	367
& ' ACCEL CL CD THETA ALPHA GAMMA DTHET ',	ROLL	368
& ' R/C LOAD THRUST XENG',/,	ROLL	369
& ' (SEC) (FEET) (LBS) (FEET) (KTS) (KTS) (KTS)',	ROLL	370
& ' (FPS2) (DEG) (DEG) (DEG) /DT ',	ROLL	371
& ' (FPM) FACT (LBS) OR MU',/)	ROLL	372
1006 FORMAT(F6.1,F8.0,F9.0,F8.1,3F7.1,F7.2,F8.4,F7.4,4F7.2,F8.1,F6.2,	ROLL	373
& F9.0,F7.3)	ROLL	374
1007 FORMAT(F6.2,F8.0,F9.0,F8.1,3F7.1,F7.2,F8.4,F7.4,4F7.2,F8.1,F6.2,	ROLL	375
& F9.0,F7.3)	ROLL	376
END	ROLL	377

Subroutine INTX

```

SUBROUTINE INTX(NEQ,TIME,DTIME,T,DERIV,ALPHA)
C**** THIS SUBROUTINE PERFORMS NUMERICAL INTEGRATION USING A FOURTH
C**** ORDER RUNGE-KUTTA SCHEME WITH THE ADAMS-BASHFORTH-MOULTON
C**** PREDICTOR-CORRECTOR METHOD.
C****
C**** REFERENCE: MORRIS, JOHN LL.
C****      COMPUTATIONAL METHODS IN ELEMENTARY NUMERICAL ANALYSIS
C****      COPYRIGHT 1983 BY JOHN WILEY AND SONS LTD.
C****      LIBRARY OF CONGRESS CATALOG QA297.M647 519.4
C****      ISBN 0 471 10419 1
C****      PP 388-97
C****
      DOUBLE PRECISION DTIME,DTIME24,HK,HK2,HK4,RKUTTA(4),T(1),TIME
      INTEGER N(11)
      DO 10 I=2,11
10        N(I) = NEQ*1
      NCOUNT = N(6)
      DTIME24 = DTIME/24.0D0
      CALL DERIV(ALPHA)
      DO 20 I=1,N(2)
20        T(I+N(4)) = T(I)
      RETURN
      ENTRY INTZ(NEQ,TIME,DTIME,T,DERIV,ALPHA)
      IF (NCOUNT.LE.N(10)) THEN
        RKUTTA(1) = RKUTTA(2) = TIME + DTIME*0.5D0
        RKUTTA(3) = RKUTTA(4) = TIME + DTIME
        DO 40 J=1,4
          TIME = RKUTTA(J)
          DO 30 I=1,NEQ
            HK2 = DTIME*T(I+NEQ)
            HK = 0.5D0*HK2
            HK4 = 2.0D0*HK2
            IF (J.EQ.1) THEN
              T(I+N(2)) = T(I)
              T(I+N(3)) = HK2
              T(I) = T(I+N(2)) + HK
            ELSEIF (J.EQ.2) THEN
              T(I) = T(I+N(2)) + HK
              T(I+N(3)) = T(I+N(3)) + HK4
            ELSEIF (J.EQ.3) THEN
              T(I) = T(I+N(2)) + HK2
              T(I+N(3)) = T(I+N(3)) + HK4
            ELSEIF (J.EQ.4) THEN
              T(I) = T(I+N(2)) + (T(I+N(3)) + HK2)/6.0D0
            ENDIF
          30      CONTINUE
          CALL DERIV(ALPHA)
        40      CONTINUE
        DO 50 I=1,N(2)
          T(I+NCOUNT) = T(I)
          NCOUNT = NCOUNT + N(2)
        50
      
```

INTX	1
INTX	2
INTX	3
INTX	4
INTX	5
INTX	6
INTX	7
INTX	8
INTX	9
INTX	10
INTX	11
INTX	12
INTX	13
INTX	14
INTX	15
INTX	16
INTX	17
INTX	18
INTX	19
INTX	20
INTX	21
INTX	22
INTZ	1
INTZ	2
INTZ	3
INTZ	4
INTZ	5
INTZ	6
INTZ	7
INTZ	8
INTZ	9
INTZ	10
INTZ	11
INTZ	12
INTZ	13
INTZ	14
INTZ	15
INTZ	16
INTZ	17
INTZ	18
INTZ	19
INTZ	20
INTZ	21
INTZ	22
INTZ	23
INTZ	24
INTZ	25
INTZ	26
INTZ	27
INTZ	28
INTZ	29

ELSE	INTZ	30
DO 60 I=1,NEQ	INTZ	31
T(I+N(3)) = 19.0D0*T(I+N(11)) - 5.0D0*T(I+N(9)) + T(I+N(7))	INTZ	32
60 T(I) = T(I+N(10)) + DTIME24*(55.0D0*T(I+N(11))	INTZ	33
& - 59.0D0*T(I+N(9)) + 37.0D0*T(I+N(7))	INTZ	34
& - 9.0D0*T(I+N(5)))	INTZ	35
TIME = TIME + DTIME	INTZ	36
CALL DERIV(ALPHA)	INTZ	37
DO 70 I=1,NEQ	INTZ	38
T(I+N(2)) = T(I)	INTZ	39
70 T(I) = T(I+N(10))	INTZ	40
& + DTIME24*(9.0D0*T(I+NEQ) + T(I+N(3)))	INTZ	41
CALL DERIV(ALPHA)	INTZ	42
DO 80 I=1,N(6)	INTZ	43
80 T(I+N(4)) = T(I+N(6))	INTZ	44
DO 90 I=1,N(2)	INTZ	45
90 T(I+N(10)) = T(I)	INTZ	46
ENDIF	INTZ	47
RETURN	INTZ	48
END	INTZ	49

Subroutine INTG

```
      SUBROUTINE INTG(DIST,DISTJ,DTIME,GAMMAR,VWIND,WFUEL,DDIST,GDIST,GWT)
C**** THIS SUBROUTINE CALCULATES INCREMENTAL GROUND DISTANCE, DDIST;
C**** TOTAL GROUND DISTANCE, GDIST; AND GROSS WEIGHT, GWT, USING FINITE
C**** DIFFERENCE INTEGRATION EQUATIONS.
C****
      DOUBLE PRECISION DTIME
      DDIST = (DIST - DISTJ)*COS(GAMMAR) - VWIND*FLOAT(DTIME)
      GDIST = GDIST + DDIST
      GWT = GWT - (WFUEL/3600.)*FLOAT(DTIME)
      RETURN
      END
```

INTG	1
INTG	2
INTG	3
INTG	4
INTG	5
INTG	6
INTG	7
INTG	8
INTG	9
INTG	10
INTG	11

Subroutine DERIVGR

SUBROUTINE DERIVGR(ALPHA)	DERIVGR	1
C**** THIS SUBROUTINE DERIVGR CALCULATES THE ACCELERATION, ACCEL, FOR	DERIVGR	2
C**** THE GROUND ROLL OF A TAKEOFF, LANDING OR REFUSED TAKEOFF.	DERIVGR	3
C****	DERIVGR	4
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKCAS,VKTAS	AERO	2
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
COMMON/RUNWAY/ ABARG,AOAABRK,BRAKMU,BRKFCR,GAMMARW,GRW,HAGL,	RUNWAY	1
& HCLEAR,HFLARE,HGEAR,HRUNWAY,IMU,RCR,ROLLMU,TIMEFLD,	RUNWAY	2
& TIMEBRK,TIMEFLP,TIMESBK,TIMESPL,XMU	RUNWAY	3
DOUBLEPRECISION VTAS,DIST,ACCEL,VTASJ,RKGRND	INTEG	1
COMMON/INTEG/ VTAS,DIST,ACCEL,VTASJ,RKGRND(20)	INTEG	2
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRS,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITTR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRS,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	8
REAL LODMAIN,LODNOSE	DERIVGR	13
DATA ITER/9/	DERIVGR	14
C****	DERIVGR	15
C**** OBTAIN FORCE COEFFICIENTS CX AND CY.	DERIVGR	16
CALL FORCEX(ALPHA,CD,CL)	DERIVGR	17
IF(BRKFLAG .AND. IMU.NE.0) THEN	DERIVGR	18
TIMEB = TIMEFLD + TIMEBRK	DERIVGR	19
IF(FLOAT(TIME).GE.TIMEB) THEN	DERIVGR	20
VKTGS = (VTAS - VWIND)/FPSKTS	DERIVGR	21
C****	DERIVGR	22
C**** CALCULATE WEIGHT MINUS LIFT.	DERIVGR	23
WTML = GWT*COS(GRW) - CY*QS	DERIVGR	24
CALL GENMU(VKTGS,WTML,XMAIN,XNOSE,YCG)	DERIVGR	25

DO 10 I=1,ITER	DERIVGR	26
LODMAIN = ((GWT*(ACCEL/G + SIN(GRW)) + CX*QS)*YCG	DERIVGR	27
& + WTML*XNOSE)/(XMAIN + XNOSE)	DERIVGR	28
LODNOSE = WTML - LODMAIN	DERIVGR	29
FEX = -(GWT*SIN(GRW) + CX*QS	DERIVGR	30
& + BRAKMU*LODMAIN - ROLLMU*LODNOSE)	DERIVGR	31
ACCEL = FEX*(G/GWT)	DERIVGR	32
IF(ABS(ACCEL-ACCELJ).GE.0.01) THEN	DERIVGR	33
ACCELJ = ACCEL	DERIVGR	34
ELSE	DERIVGR	35
VTASJ = VTAS	DERIVGR	36
RETURN	DERIVGR	37
ENDIF	DERIVGR	48
10 CONTINUE	DERIVGR	49
WRITE(LUOUT,1001) ACCEL,ACCELJ,LODMAIN,LODNOSE	DERIVGR	40
1001 FORMAT(' *** UNABLE TO CONVERGE ON SOLUTION FOR EQUATION',	DERIVGR	41
& ' OF MOTION WITH NOSE AND MAIN GEAR FORCES',	DERIVGR	42
& ' RESOLVED. ACCEL,ACCELJ,LODMAIN,LODNOSE =',	DERIVGR	43
& 2F10.2,2F10.0,' ***')	DERIVGR	44
RETURN	DERIVGR	45
ENDIF	DERIVGR	46
ENDIF	DERIVGR	47
C****	DERIVGR	48
C**** CALCULATE GROUND ROLL ACCELERATION (FT/SEC**2), ACCEL.	DERIVGR	49
FEX = -(GWT*(SIN(GRW) + XMU*COS(GRW)) + QS*(CY*XMU-CX))	DERIVGR	50
ACCEL = FEX*(G/GWT)	DERIVGR	51
VTASJ = VTAS	DERIVGR	52
RETURN	DERIVGR	53
END	DERIVGR	54

Subroutine DERIVAT

SUBROUTINE DERIVAT(ALPHA)	DERIVAT	1
C**** THIS SUBROUTINE CALCULATES THE TIME DERIVATIVES FOR THE AIRBORNE	DERIVAT	2
C**** PORTION OF THE TAKEOFF AND MANAGES THE FLIGHTPATH CONTROL	DERIVAT	3
C****	DERIVAT	4
COMMON/AIRCRAFT/ AOA0PT,AP,B,CGPCT,CL,CLPM,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKCAS,VKTAS	AERO	2
COMMON/AIRBORNE/ ALPHA,ALPHMX,DTDT,GAMMAPP,ROCFPM,THETA,FL,XLF,XLFJ	AIRBORNE	1
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRS,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITTR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRS,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	8
DATA DALPHA/0.05/	DERIVAT	13
ALPHMX = THTMAX	DERIVAT	14
ITER = INT((ALPHMX + 2.0)/DALPHA)	DERIVAT	15
DO 20 I=1,ITER	DERIVAT	16
C****	DERIVAT	17
C**** OBTAIN FORCE COEFFICIENTS CX AND CY.	DERIVAT	18
CALL FORCEX(ALPHA,CD,CL)	DERIVAT	19
C****	DERIVAT	20
C**** CALCULATE LOAD FACTOR.	DERIVAT	21
XLF = (QS*CY)/GWT + (1.0 - COS(GAMMAR))	DERIVAT	22
C****	DERIVAT	23
C**** CHECK FUSELAGE ANGLE. IF THE FUSELAGE ANGLE IS GREATER THAN	DERIVAT	24
C**** THE MAXIMUM FUSELAGE ANGLE, REDUCE ANGLE OF ATTACK.	DERIVAT	25
THETA = ALPHA + GAMMAR*RX	DERIVAT	26

IF(THETA.F.GT.THTMAX) THEN	DERIVAT	27
ALPHA = THTMAX - GAMMAR*RX	DERIVAT	28
ELSE	DERIVAT	29
C**** CHECK FLIGHTPATH ACCELERATION. IF THE ACCELERATION IS	DERIVAT	30
C**** LESS THAN ZERO, REDUCE ANGLE OF ATTACK.	DERIVAT	31
FPACCEL = (G/GWT)*(-CX*QS - GWT*SIN(GAMMAR))	DERIVAT	32
IF(FPACCEL.LT.ZERO) THEN	DERIVAT	33
ALPHA = ALPHA - DALPHA	DERIVAT	34
IF(CL.LT.0.0000) THEN	DERIVAT	35
WRITE(LUOUT,1001) FPACCEL,THTMAX	DERIVAT	36
1001 FORMAT(' *** UNABLE TO MAINTAIN FLIGHTPATH',	DERIVAT	37
& ' ACCELERATION GREATER THAN OR EQUAL TO',	DERIVAT	38
& ' ZERO. FPACCEL,THTMAX =',F10.3,F10.1,' ****)	DERIVAT	39
ERRFLAG = .TRUE.	DERIVAT	40
RETURN	DERIVAT	41
ENDIF	DERIVAT	42
ELSE	DERIVAT	43
C**** CALCULATE DGAMMA/DT, HORIZONTAL SPEED AND RATE OF	DERIVAT	44
C**** CLIMB, DGDTR, VHAS AND ROC.	DERIVAT	45
DGDTR = (G/(GWT*FPVTAS))*(CY*QS - GWT*COS(GAMMAR))	DERIVAT	46
VHAS = FPVTAS*COS(GAMMAR)	DERIVAT	47
ROC = FPVTAS*SIN(GAMMAR)	DERIVAT	48
RETURN	DERIVAT	49
ENDIF	DERIVAT	50
ENDIF	DERIVAT	51
20 CONTINUE	DERIVAT	52
END	DERIVAT	53

Subroutine DERIVAL

SUBROUTINE DERIVAL(ALPHA)	DERIVAL	1
C**** THIS SUBROUTINE CALCULATES THE TIME DERIVATIVES FOR THE AIRBORNE	DERIVAL	2
C**** PORTION OF THE LANDING AND MANAGES THE FLIGHTPATH CONTROL	DERIVAL	3
C****	DERIVAL	4
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKCAS,VKTAS	AERO	2
COMMON/AIRBORN/ ALPHAJ,ALPHMX,DTDT,GAMMAPP,ROCFPM,THETAJ,XLF,XLFJ	AIRBORN	1
DOUBLEPRECISION FSVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FSVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITTR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITTR	FLAGS	8
DATA DALPHA/0.05/	DERIVAL	12
ITER = INT(2.0*ALPHMX/DALPHA)	DERIVAL	13
DO 30 I=1,ITER	DERIVAL	14
C****	DERIVAL	15
C**** LIMIT ALPHA TO ALPHMX	DERIVAL	16
IF(ALPHA.GT.ALPHMX) ALPHA = ALPHMX	DERIVAL	17
C****	DERIVAL	18
C**** OBTAIN FORCE COEFFICIENTS CX AND CY.	DERIVAL	19
CALL FORCEX(ALPHA,CD,CL)	DERIVAL	20
C****	DERIVAL	21
C**** CHECK LOAD FACTOR. IF LOAD FACTOR IS GREATER THAN THE	DERIVAL	22
C**** MAXIMUM LOAD FACTOR, REDUCE ANGLE OF ATTACK.	DERIVAL	23
XLF = (QS*CY)/GWT + (1.0 - COS(GAMMAR))	DERIVAL	24
IF(XLF.GT.XLFMAX) THEN	DERIVAL	25
ALPHA = ALPHA - DALPHA	DERIVAL	26
IF(ALPHA.LT.-ALPHMX) THEN	DERIVAL	27
ERRFLAG = .TRUE.	DERIVAL	28
WRITE(LUOUT,1002)	DERIVAL	29
1002 FORMAT(' *** ERROR IN DERIVAL. ALPHA IS LESS THAN',	DERIVAL	30
& ' -ALPHMX AND XLF IS GREATER THAN XLFMAX. ****)	DERIVAL	31
RETURN	DERIVAL	32
ENDIF	DERIVAL	33

	ELSE	DERIVAL	34
C****	CALCULATE TIME RATE OF CHANGE OF FLIGHTPATH ANGLE DGDTR AND	DERIVAL	35
C****	THE PITCH RATE DTDTR. IF DTHETA/DT IS GREATER THAN THE	DERIVAL	36
C****	MAXIMUM, DTDTRMX, REDUCE ANGLE OF ATTACK.	DERIVAL	37
	DGDTR = (G/(GWT*FPVTAS))*(CY*QS - GWT*COS(GAMMAR))	DERIVAL	38
	DADT = (ALPHA - ALPHAJ)/FLOAT(DTIME)	DERIVAL	39
	DTDTR = DGDTR*RX + DADT	DERIVAL	40
	IF(DTDTR.GT.DTDTRMX) THEN	DERIVAL	41
	ALPHA = ALPHA - DADT	DERIVAL	42
	IF(ALPHA LT.-ALPHMX) THEN	DERIVAL	43
	ERRFLAG = .TRUE.	DERIVAL	44
	WRITE(LUOUT,1003)	DERIVAL	45
1003	FORMAT(' *** ERROR IN DERIVAL. ALPHA IS LESS THAN',	DERIVAL	46
&	' -ALPHMX AND DTDTR IS GREATER THAN DTDTRMX.',	DERIVAL	47
&	' ****')	DERIVAL	48
	RETURN	DERIVAL	49
	ENDIF	DERIVAL	50
	ELSE	DERIVAL	51
C****	CALCULATE ACCELERATION ALONG FLIGHTPATH AND VELOCITY	DERIVAL	52
C****	COMPONENTS.	DERIVAL	53
	FPACCEL = (G/GWT)*(-CX*QS - GWT*SIN(GAMMAR))	DERIVAL	54
	VHAS = FPVTAS*COS(GAMMAR)	DERIVAL	55
	ROC = FPVTAS*SIN(GAMMAR)	DERIVAL	56
	RETURN	DERIVAL	57
	ENDIF	DERIVAL	58
	ENDIF	DERIVAL	59
30	CONTINUE	DERIVAL	60
	END	DERIVAL	61

Subroutine ERROR

SUBROUTINE ERROR(LUOUT,ROCFPM)	ERROR	1
C**** THIS SUBROUTINE IS CALLED DURING A TAKEOFF SIMULATION WHEN THE	ERROR	2
C**** FLAG ERRFLAG IS SET TO .TRUE. PROGRAM EXECUTION IS TERMINATED FOR	ERROR	3
C**** THE PRESENT SET OF NAMELIST INPUTS AND WILL CONTINUE IF ADDITIONAL	ERROR	4
C**** NAMELIST INPUTS ARE TO BE PROCESSED.	ERROR	5
C****	ERROR	6
IF(ROCFPM.LE.0.0) THEN	ERROR	7
WRITE(LUOUT,1001)	ERROR	8
ELSE	ERROR	9
WRITE(LUOUT,1002)	ERROR	10
ENDIF	ERROR	11
1001 FORMAT(/,' CANNOT CLIMB AT USER INPUT CONDITIONS.',	ERROR	12
& /,' *** ABNORMAL TERMINATION OF TAKOFF ***)	ERROR	13
1002 FORMAT(/,' *** ABNORMAL TERMINATION OF TAKOFF ***)	ERROR	14
RETURN	ERROR	15
END	ERROR	16

Subroutine HALT

SUBROUTINE HALT(LUIN,LUMSG,LUOUT,TERMMSG)	HALT	1
C**** THIS SUBROUTINE TERMINATES PROGRAM EXECUTION, CLOSES THE INPUT AND	HALT	2
C**** OUTPUT FILES, AND WRITES A TERMINATION MESSAGE, TERMMSG.	HALT	3
C****	HALT	4
CHARACTER*(*) TERMMSG	HALT	5
WRITE(LUOUT,1001) TERMMSG	HALT	6
1001 FORMAT(/,A)	HALT	7
CLOSE (UNIT=LUIN)	HALT	8
CLOSE (UNIT=LUMSG)	HALT	9
CLOSE (UNIT=LUOUT)	HALT	10
STOP ' TO LAND NORMAL TERMINATION'	HALT	11
END	HALT	12

Subroutine ATMOSPH

```

SUBROUTINE ATMOSPH(PRESALT,ARRAY)
C**** THIS SUBROUTINE CALCULATES PRESSURE, TEMPERATURE, DENSITY, SPEED
C**** OF SOUND, KINEMATIC VISCOSITY, AND PRESSURE, TEMPERATURE AND
C**** DENSITY RATIOS. THE INPUTS ARE PRESSURE ALTITUDE, PRESALT, AND
C**** THE TEMPERATURE INCREMENT FROM STANDARD DAY, DTEMPF.
C****
C**** ARRAY(1) = TEMPERATURE (DEG R)  ARRAY(6) = PRESSURE RATIO
C**** ARRAY(2) = PRESSURE (PSF)  ARRAY(7) = DENSITY RATIO
C**** ARRAY(3) = DENSITY (SLUG/FT3)  ARRAY(8) = TEMPERATURE RATIO
C****
C**** ARRAY(4) = SPEED OF SOUND (FEET/SECOND)
C**** ARRAY(5) = KINEMATIC VISCOSITY (FT**2/SEC)
C****
COMMON/ATMOS/TEMPR,PRESS,RHO,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMPF
REAL ARRAY(8)
DATA TSLR,PSLPSF, RHOSL,ASLFPS,ASLKTS, CONST1
& / 518.67, 2116.22, 2.3769E-03, 1116.47, 661.48,0.270558E-06/
IF(PRESALT.LT.36089.0) THEN
    FACTOR = 1.0 - 6.8750E-06*PRESALT
    ARRAY(6) = DELTA = FACTOR**5.2559
    ARRAY(8) = THETA = FACTOR + DTEMPF/518.67
ELSE
    ARRAY(6) = DELTA = 0.22336*EXP((36089.0-PRESALT)/20786.0)
    ARRAY(8) = THETA = 0.75187 + DTEMPF/518.67
ENDIF
ARRAY(7) = SIGMA = DELTA/THETA
ARRAY(1) = TEMPR = THETA*TSLR
ARRAY(2) = PRESS = DELTA*PSLPSF
ARRAY(3) = RHO = SIGMA*RHOSL
ARRAY(4) = AFPS = SQRT(THETA)*ASLFPS
ARRAY(5) = VISCOSK = CONST1*TEMPR**1.5/
& ((PRESS/144.0)*(1.0 + 198.72/TEMPR))
RETURN
END

```

ATMOSPH	1
ATMOSPH	2
ATMOSPH	3
ATMOSPH	4
ATMOSPH	5
ATMOSPH	6
ATMOSPH	7
ATMOSPH	8
ATMOSPH	9
ATMOSPH	10
ATMOSPH	11
ATMOSPH	12
ATMOSPH	13
ATMOS	1
ATMOSPH	15
ATMOSPH	16
ATMOSPH	17
ATMOSPH	18
ATMOSPH	29
ATMOSPH	26
ATMOSPH	21
ATMOSPH	22
ATMOSPH	23
ATMOSPH	24
ATMOSPH	25
ATMOSPH	26
ATMOSPH	27
ATMOSPH	28
ATMOSPH	29
ATMOSPH	30
ATMOSPH	31
ATMOSPH	32
ATMOSPH	33
ATMOSPH	34

Subroutine SPEED

```

SUBROUTINE SPEED(GAMMAR,VTASX,VWIND,AMACH,QS,VKCAS,VKEAS,VKTAS,
                VKTGS,VTGS)
C**** THIS SUBROUTINE IS CALCULATES DYNAMIC PRESSURE, MACH NUMBER, AND
C**** VELOCITIES.
C****
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,
&                LOADING,SWING,THTMAX,WNGLOD,XLFMAX
COMMON/ATMOS/TEMPR,PRESS,RHO,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMPF
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,
&          RX=57.29577951308,TSLF=59.0,ZERO=0.0)
COMMON/CONST/ ASLSQR5,TWOVR7
VKTAS = VTASX/FPSKTS
VTGS = VTASX*COS(GAMMAR) - VWIND
VKTGS = VTGS/FPSKTS
IF (VTASX.GT.ZERO) THEN
    AMACH = VTASX/AFPS
    QS = 0.5*RHO*VTASX**2*SWING
    VKEAS = VKTAS*SQRT(SIGMA)
    VKCAS = ASLSQR5*SQRT((DELTA*
&                ((1.0 + 0.2*AMACH**2)**3.5 - 1.0) + 1.0)**(TWOVR7)
&                - 1.0)
ELSE
C**** IF TRUE AIRSPEED IS NEGATIVE, RESET THESE PARAMETERS TO
C**** ZERO; THIS CAN OCCUR IN A SIMULATION WITH TAILWIND.
    AMACH = VKEAS = VKCAS = ZERO
    QS = 0.1
ENDIF
RETURN
END

```

SPEED	1
SPEED	2
SPEED	3
SPEED	4
SPEED	5
AIRCRAFT	1
AIRCRAFT	2
ATMOS	1
CONST	1
CONST	2
CONST	3
SPEED	9
SPEED	10
SPEED	11
SPEED	12
SPEED	13
SPEED	14
SPEED	15
SPEED	16
SPEED	17
SPEED	18
SPEED	19
SPEED	20
SPEED	21
SPEED	22
SPEED	23
SPEED	24
SPEED	25
SPEED	26

Subroutine ITRLND

SUBROUTINE ITRLND(ERROR,ERRORJ,DRIVER,FACTOR,TOLRNCE,JFLAG)	ITRLND	1
C**** THIS SUBROUTINE IS A ZERO FINDING ROUTINE WHICH VARIES THE	ITRLND	2
C**** INDEPENDENT VARIABLE BASED ON THE SIGN AND SIZE OF THE ERROR. THE	ITRLND	3
C**** LOOP CONTAINING THE CALL TO ITRLND IS EXITED SUCCESSFULLY WHEN	ITRLND	4
C**** JFLAG = 3 OR WHEN THE MAGNITUDE OF THE ERROR IS LESS THAN THE	ITRLND	5
C**** TOLERANCE VALUE, TOLRNCE. ERROR AND ERRORJ ARE THE PRESENT AND	ITRLND	6
C**** PREVIOUS ERRORS, RESPECTIVELY. JFLAG IS A VARIABLE WHICH	ITRLND	7
C**** INDICATES THE STATUS OF THE SEARCH. JFLAG IS SET TO 0 BEFORE	ITRLND	8
C**** ITRLND IS CALLED. AFTER THE FIRST CALL TO ITRLND, JFLAG IS SET TO	ITRLND	9
C**** 1. JFLAG IS SET TO 2 ONCE THE ZERO IS ISOLATED BETWEEN THE UPPER	ITRLND	10
C**** AND LOWER BOUNDARIES, BOUNDU AND BOUNDL, RESPECTIVELY. JFLAG IS	ITRLND	11
C**** SET TO 3 AND THE ZERO IS CONSIDERED FOUND WHEN THE ABSOLUTE VALUE	ITRLND	12
C**** OF THE DIFFERENCE BETWEEN BOUNDU AND BOUNDL IS LESS THAN TOLRNCE,	ITRLND	13
C**** THE TOLERANCE PASSED TO SUBROUTINE ITRLND.	ITRLND	14
C****	ITRLND	15
C**** DURING THE FIRST CALL TO ITRLND DRIVERJ, THE PREVIOUS DRIVER, IS	ITRLND	16
C**** SET TO DRIVER AND DRIVER IS THEN MULTIPLIED BY FACTOR. ON	ITRLND	17
C**** SUBSEQUENT CALLS TO ITRLND THESE TWO DRIVER VALUES ARE SET TO	ITRLND	18
C**** EITHER UPPER AND LOWER BOUND OR A MULTIPLICATION OR DIVISION BY	ITRLND	19
C**** FACTOR IS APPLIED TO DRIVER. ONCE THE LOCATION OF THE ZERO IS	ITRLND	20
C**** DETERMINED TO BE ON ONE SIDE OF THE DRIVER OR THE OTHER, DRIVER	ITRLND	21
C**** IS SET TO EITHER THE UPPER OR LOWER BOUND AS THE SEARCH IS	ITRLND	22
C**** NARROWED.	ITRLND	23
C****	ITRLND	24
LOGICAL FFLAG,MFLAG,PFLAG	ITRLND	25
IF(JFLAG.EQ.0) THEN	ITRLND	26
FFLAG = MFLAG = PFLAG = .FALSE.	ITRLND	27
JFLAG = 1	ITRLND	28
ENDIF	ITRLND	29
IF (.NOT.PFLAG) THEN	ITRLND	30
PFLAG = .TRUE.	ITRLND	31
DRIVERJ = DRIVER	ITRLND	32
DRIVER = DRIVER*FACTOR	ITRLND	33
RETURN	ITRLND	34
ELSEIF (.NOT.FFLAG) THEN	ITRLND	35
IF (ERROR*ERRORJ.LT.0.0) THEN	ITRLND	36
IF(DRIVERJ.LE.DRIVER) THEN	ITRLND	37
BOUNDL = DRIVERJ	ITRLND	38
BOUNDU = DRIVER	ITRLND	39
ELSE	ITRLND	40
BOUNDU = DRIVERJ	ITRLND	41
BOUNDL = DRIVER	ITRLND	42
ERRORJ = ERROR	ITRLND	43
ENDIF	ITRLND	44
JFLAG = 2	ITRLND	45
FFLAG = .TRUE.	ITRLND	46
DRIVER = (BOUNDU + BOUNDL)/2.0	ITRLND	47

```

      ELSEIF(.NOT.MFLAG .AND.
&      ((ERROR.LT.0.0 .AND. ERROR.GE.ERRORJ) .OR.
&      (ERROR.GE.0.0 .AND. ERROR.LE.ERRORJ))) THEN
      DRIVERJ = DRIVER
      DRIVER = DRIVER*FACTOR
      ELSE
      MFLAG = .TRUE.
      BOUNDU = DRIVER
      DRIVERJ = DRIVER
      DRIVER = BOUNDU/FACTOR
      ENDIF
      RETURN
ENDIF
IF(ERROR*ERRORJ.GT.0.0) THEN
      BOUNDL = DRIVER
      ERRORJ = ERROR
ELSE
      BOUNDU = DRIVER
ENDIF
DRIVER = (BOUNDU + BOUNDL)/2.0
IF(ABS(BOUNDU-BOUNDL) .LT. TOLRNCE) JFLAG = 3
RETURN
END

```

```

ITRLND 48
ITRLND 49
ITRLND 50
ITRLND 51
ITRLND 52
ITRLND 53
ITRLND 54
ITRLND 55
ITRLND 56
ITRLND 57
ITRLND 58
ITRLND 59
ITRLND 60
ITRLND 61
ITRLND 62
ITRLND 63
ITRLND 64
ITRLND 65
ITRLND 66
ITRLND 67
ITRLND 68
ITRLND 69
ITRLND 70

```

Program Functions

Function DGDT

FUNCTION DGDT(ALPHA,FPVTAS,GAMMAR,GWT)	DGDT	1
C**** THIS FUNCTION CALCULATES DG/DT AS A FUNCTION OF ALPHA. IT	DGDT	2
C**** REQUIRES VARIABLES CY, FPVTAS, GAMMAPP, GWT, AND QS.	DGDT	3
C****	DGDT	4
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKAS,VKTAS	AERO	2
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOOVR7	CONST	3
CALL FORCEX(ALPHA,CD,CL)	DGDT	8
DGDT = (G/(GWT*FPVTAS))*(CY*QS - GWT*COS(GAMMAR))	DGDT	9
RETURN	DGDT	10
END	DGDT	11

Function DVDT

FUNCTION DVDT(ALPHA,FPVTAS,GAMMAR,GWT)	DVDT	1
C**** THIS FUNCTION CALCULATES DV/DT AS A FUNCTION OF ALPHA. IT	DVDT	2
C**** REQUIRES VARIABLES CX, FPVTAS, GAMMAPP, GWT, AND QS.	DVDT	3
C****	DVDT	4
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKAS,VKTAS	AERO	2
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOOVR7	CONST	3
CALL FORCEX(ALPHA,CD,CL)	DVDT	8
DVDH = DVTDH(FPVTAS,' VC')	DVDT	9
DVDT = G/GWT*(-CX*QS - GWT*SIN(GAMMAR)) - DVDH*FPVTAS*SIN(GAMMAR)	DVDT	10
RETURN	DVDT	11
END	DVDT	12

Function DVTDH

FUNCTION DVTDH (VTAS,CONSTANT)	DVTDH	1
C**** THIS FUNCTION CALCULATES THE CLIMB SPEED DERIVATIVE, DV/DH FOR	DVTDH	2
C**** CONSTANT CALIBRATED AIRSPEED, EQUIVALENT AIRSPEED, OR MACH NUMBER.	DVTDH	3
C****	DVTDH	4
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
COMMON/ATMOS/ TEMPR,PRESS,RHO,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMPF	ATMOS	1
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOOVR7	CONST	3
COMMON/RACURV/ IDBUG,IEXTOR,KURNAM(99)	RACURV	1
CHARACTER CONSTANT*4	DVTDH	10
IF(IDBUG.GE.1) WRITE(LUMSG,9001)	DVTDH	11
9001 FORMAT (' ENTERED DVTDH')	DVTDH	12
CALL SPEED(GAMMAR,VTAS,VWIND,AMACH,QS,VKCAS,VKEAS,VKTAS,	DVTDH	13
VKTGS,VTGS)	DVTDH	14
IF (CONSTANT.EQ.' VC') THEN	DVTDH	15
C**** CONSTANT CALIBRATED AIRSPEED	DVTDH	16
DERIVA = DADH(PRESALT)	DVTDH	17
DERIVD = DDELTDH(PRESALT)	DVTDH	18
IF(IDBUG.GE.1) WRITE(LUMSG,1001) DERIVA,DERIVD,TIME	DVTDH	19
1001 FORMAT(' ***** FROM FUNCTION DVTDH, DADH,DDELTDH =',2E16.8,	DVTDH	20
& ' AT TIME ',E14.8, ' *****')	DVTDH	21
IF (VKCAS.LE.ASL .AND. AMACH.LE.1.0) THEN	DVTDH	22
C**** SUBSONIC EQUATION	DVTDH	23
CONSTX = 1.0 + 0.2*AMACH**2	DVTDH	24
DVTDH = AMACH*DERIVA - (5.0*AFPS*((CONSTX)**3.5 - 1.0)*	DVTDH	25
& DERIVD)/(7.0*AMACH*(CONSTX)**2.5)	DVTDH	26
ELSEIF (VKCAS.GT.ASL .AND. AMACH.GT.1.0) THEN	DVTDH	27
C**** SUPERSONIC EQUATION	DVTDH	28
CONSTY = 7.0*AMACH**2 - 1.0	DVTDH	29
DVTDH = AMACH*DERIVA - (AFPS*DERIVD*(166.921*AMACH**7 -	DVTDH	30
& (CONSTY)**2.5)*(CONSTY))/	DVTDH	31
& (1168.45*AMACH**6*(2.0*AMACH**2 - 1.0))	DVTDH	32
ELSE	DVTDH	33
WRITE(LUMSG,1002)	DVTDH	34
1002 FORMAT(' ILLEGAL VALUES FOR VKCAS AND AMACH PASSED TO',	DVTDH	35
& ' FUNCTION DVTDH. EITHER VKCAS IS GREATER THAN',	DVTDH	36
& ' ASL AND AMACH IS LESS THAN 1.0 OR VKCAS IS LESS',	DVTDH	37
& ' THAN OR EQUAL TO ASL AND AMACH IS GREATER THAN',	DVTDH	38
& ' 1.0; EXECUTION RETURNS TO CALLING SUBROUTINE.')	DVTDH	39
ENDIF	DVTDH	40

	ELSEIF(CONSTANT.EQ.' VE') THEN	DVTDH	41
C****	CONSTANT EQUIVALENT AIRSPEED	DVTDH	42
	DXDH = DSIGDH(PRESALT)	DVTDH	43
	IF(IDBUG.GE.1) WRITE(LUMSG,1003) DXDH,TIME	DVTDH	44
1003	FORMAT(' ***** FROM FUNCTION DVTDH, DSIGDH =',E16.8,	DVTDH	45
&	' AT TIME ',E14.8, ' *****')	DVTDH	46
	DVTDH = -(VKEAS*FPSKTS)*DXDH/(2.0*SIGMA**1.5)	DVTDH	47
	ELSEIF(CONSTANT.EQ.'MACH') THEN	DVTDH	48
C****	CONSTANT MACH NUMBER	DVTDH	49
	DXDH = DADH(PRESALT)	DVTDH	50
	IF(IDBUG.GE.1) WRITE(LUMSG,1004) DXDH,TIME	DVTDH	51
1004	FORMAT(' ***** FROM FUNCTION DVTDH, DADH =',E16.8,	DVTDH	52
&	' AT TIME ',E14.8, ' *****')	DVTDH	53
	DVTDH = AMACH*DXDH	DVTDH	54
	ELSE	DVTDH	55
	WRITE(LUMSG,1005)	DVTDH	56
1005	FORMAT(' ILLEGAL OPTION CODE PASSED TO FUNCTION DVTDH;',	DVTDH	57
&	' EXECUTION RETURNS TO CALLING SUBROUTINE.')	DVTDH	58
	ENDIF	DVTDH	59
	RETURN	DVTDH	60
	END	DVTDH	61

Function DADH

```

FUNCTION DADH(HC)
C**** THIS FUNCTION CALCULATES THE SPEED OF SOUND DERIVATIVE WITH
C**** RESPECT TO PRESSURE ALTITUDE.
C****
DOUBLEPRECISION DTIME,DTIMEJ,TIME
PARAMETER (LUIN=3,LUOUT=4)
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,
& NEQ,NPAGE,TIME,TIMEROL
IF (HC.GE. -1000.00 .AND. HC.LE. 36089.24) THEN
DADH = -0.0851520/SQRT(288.15 - .0019812*HC)
ELSEIF (HC.GT. 36089.24 .AND. HC.LE. 65616.80) THEN
DADH = 0.0
ELSEIF (HC.GT. 65616.80 .AND. HC.LE. 104986.88) THEN
DADH = 0.0100234/SQRT(216.65 + .0003048*(HC - 65616.8))
ELSEIF (HC.GT. 104986.88 .AND. HC.LE. 154199.48) THEN
DADH = 0.0280841/SQRT(228.65 + .0008534*(HC - 104986.88))
ELSE
WRITE(LUMSG,1001)
1001 FORMAT(' ILLEGAL PRESSURE ALTITUDE PASSED TO FUNCTION DADH;',
& ' EXECUTION RETURNED TO CALLING SUBROUTINE')
ENDIF
RETURN
END

```

DADH	1
DADH	2
DADH	3
DADH	4
CTRL	1
CTRL	2
CTRL	3
CTRL	4
DADH	6
DADH	7
DADH	8
DADH	9
DADH	10
DADH	11
DADH	12
DADH	13
DADH	14
DADH	15
DADH	16
DADH	17
DADH	18
DADH	19
DADH	20

Function DDELTDH

FUNCTION DDELTDH(HC)	DDELTDH	1
C**** THIS FUNCTION CALCULATES THE PRESSURE RATIO DERIVATIVE WITH	DDELTDH	2
C**** RESPECT TO PRESSURE ALTITUDE.	DDELTDH	3
C****	DDELTDH	4
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
IF (HC.GE. -1000.00 .AND. HC.LE. 36089.24) THEN	DDELTDH	6
DDELTDH = -0.361374E-04/(1.0 - HC*6.875586E-06)	DDELTDH	7
ELSEIF (HC.GT. 36089.24 .AND. HC.LE. 65616.80) THEN	DDELTDH	8
DDELTDH = -0.480637E-04	DDELTDH	9
ELSEIF (HC.GT. 65616.80 .AND. HC.LE. 104986.88) THEN	DDELTDH	10
DDELTDH = -0.480637E-04/(1.0 + (HC - 65616.80)*1.40688E-06)	DDELTDH	11
ELSEIF (HC.GT. 104986.88 .AND. HC.LE. 154199.48) THEN	DDELTDH	12
DDELTDH = -0.455412E-04/(1.0 + (HC - 104986.88)*3.73252E-06)	DDELTDH	13
ELSE	DDELTDH	14
WRITE(LUMSG,1001)	DDELTDH	15
1001 FORMAT(' ILLEGAL PRESSURE ALTITUDE PASSED TO FUNCTION',	DDELTDH	16
& ' DDELTDH; EXECUTION RETURNED TO CALLING SUBROUTINE')	DDELTDH	17
ENDIF	DDELTDH	18
RETURN	DDELTDH	19
END	DDELTDH	20

Function DSIGDH

```

FUNCTION DSIGDH(HC)
C**** THIS SUBROUTINE CALCULATES THE DENSITY RATIO DERIVATIVE WITH
C**** RESPECT TO PRESSURE ALTITUDE.
C****
DOUBLEPRECISION DTIME,DTIMEJ,TIME
PARAMETER (LUIN=3,LUOUT=4)
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,
& NEQ,NPAGE,TIME,TIMEROL
IF (HC.GE. -1000.00 .AND. HC.LE. 36089.24) THEN
    DSIGDH = -2.926178E-05*(1.0 - HC*6.87558E-06)**3.2559
ELSEIF (HC.GT. 36089.24 .AND. HC.LE. 65616.80) THEN
    DSIGDH = EXP (-4.80637E-05*(HC - 36089.24))*(-1.427853E-05)
ELSEIF (HC.GT. 65616.80 .AND. HC.LE. 104986.88) THEN
    DSIGDH = -3.555171E-06*
& (1.0 + 1.40688E-06*(HC - 65616.80))**(-36.1634)
ELSEIF (HC.GT. 104986.88 .AND. HC.LE. 154199.48) THEN
    DSIGDH = -5.319445E-07*
& (1.0 + 3.73252E-06*(HC - 104986.88))**(-14.2012)
ELSE
    WRITE(LUMSG,1001)
1001    FORMAT(' ILLEGAL PRESSURE ALTITUDE PASSED TO FUNCTION',
&          ' DSIGDH; EXECUTION RETURNED TO CALLING SUBROUTINE')
    ENDIF
    RETURN
END

```

DSIGDH	1
DSIGDH	2
DSIGDH	3
DSIGDH	4
CTRL	1
CTRL	2
CTRL	3
CTRL	4
DSIGDH	6
DSIGDH	7
DSIGDH	8
DSIGDH	9
DSIGDH	10
DSIGDH	11
DSIGDH	12
DSIGDH	13
DSIGDH	14
DSIGDH	15
DSIGDH	16
DSIGDH	17
DSIGDH	18
DSIGDH	19
DSIGDH	20
DSIGDH	21
DSIGDH	22

Function INTERP

REAL FUNCTION INTERP(YNOW,YPAST,XNOW,XPAST,XANS)	INTERP	1
C**** THIS FUNCTION RETURNS THE INTERPOLATED VALUE BETWEEN TWO ORDINATE	INTERP	2
C**** INPUTS BASED ON THE ADDITIONAL INPUTS OF THREE ABSCISSAS.	INTERP	3
C****	INTERP	4
XDIFF = XNOW - XPAST	INTERP	5
C****	INTERP	6
C**** CHECK DIFFERENCE BETWEEN X-COORDINATES TO PREVENT ARITHMETIC	INTERP	7
C**** DIVISION BY ZERO.	INTERP	8
IF(XDIFF.EQ.0.0) XDIFF = 1.0	INTERP	9
INTERP = YNOW - ((YNOW - YPAST)*(XNOW - XANS)/XDIFF)	INTERP	10
RETURN	INTERP	11
END	INTERP	12

Function ZEROX

FUNCTION ZEROX(A,B,FUNCTN,VAR2,VAR3,VAR4,TOLRNCE,FINDV)	ZEROX	1
C**** THIS FUNCTION FINDS THE ZERO OF THE FUNCTION FUNCTN(X) ON AN	ZEROX	2
C**** INTERVAL X = A TO X = B. THE PROGRAM ASSUMES THAT F(AX) AND	ZEROX	3
C**** F(BX) ARE OF OPPOSITE SIGN, IMPLYING THAT FUNCTN(X) IS ZERO	ZEROX	4
C**** SOMEWHERE IN THAT INTERVAL. THE ARGUMENT, "FUNCTN", MUST BE	ZEROX	5
C**** DECLARED EXTERNAL IN THE MAIN CALLING PROGRAM AND HAVE FOUR	ZEROX	6
C**** CALLING ARGUMENTS, A OR B, VAR2, VAR3, AND VAR4.	ZEROX	7
C****	ZEROX	8
LOGICAL FINDV	ZEROX	9
REAL A,B,FUNCTN,TOLRNCE	ZEROX	10
DATA EPSILON/2.22E-16/	ZEROX	11
AX = CX = A	ZEROX	12
BX = B	ZEROX	13
IF(FINDV) THEN	ZEROX	14
FAX = FCX = FUNCTN(VAR2,AX,VAR3,VAR4)	ZEROX	15
FBX = FUNCTN(VAR2,BX,VAR3,VAR4)	ZEROX	16
ELSE	ZEROX	17
FAX = FCX = FUNCTN(AX,VAR2,VAR3,VAR4)	ZEROX	18
FBX = FUNCTN(BX,VAR2,VAR3,VAR4)	ZEROX	19
ENDIF	ZEROX	20
D = E = BX - AX	ZEROX	21
19 IF(ABS(FCX).LT.ABS(FBX)) THEN	ZEROX	22
AX = BX	ZEROX	23
BX = CX	ZEROX	24
CX = AX	ZEROX	25
FAX = FBX	ZEROX	26
FBX = FCX	ZEROX	27
FCX = FAX	ZEROX	28
ENDIF	ZEROX	29
TOLX = 2.0*EPSILON*ABS(BX) + TOLRNCE	ZEROX	30
DIFCXBX = CX - BX	ZEROX	31
HALFDIF = 0.5*DIFCXBX	ZEROX	32
IF ((ABS(HALFDIF).LE.TOLX) .OR. (FBX.EQ.0.0)) THEN	ZEROX	33
ZEROX = BX	ZEROX	34
RETURN	ZEROX	35
ELSEIF((ABS(E).LT.TOLX) .OR. (ABS(FAX).LE.ABS(FBX))) THEN	ZEROX	36
D = E = HALFDIF	ZEROX	37
ELSE	ZEROX	38
S = FBX/FAX	ZEROX	39
IF(AX.EQ.CX) THEN	ZEROX	40
P = DIFCXBX*S	ZEROX	41
Q = 1.0 - S	ZEROX	42
ELSE	ZEROX	43
Q = FAX/FCX	ZEROX	44
R = FBX/FCX	ZEROX	45
P = S*(DIFCXBX*Q*(Q - R) - (BX - AX)*(R - 1.0))	ZEROX	46
Q = (Q - 1.0)*(R - 1.0)*(S - 1.0)	ZEROX	47
ENDIF	ZEROX	48

IF(P.LE.0.0) THEN	ZEROX	49
P = -P	ZEROX	50
ELSE	ZEROX	51
Q = -Q	ZEROX	52
ENDIF	ZEROX	53
S = E	ZEROX	54
E = D	ZEROX	55
IF(((2.0*P) .GE.(3.0*HALFDIF*Q - ABS(TOLX*Q))).OR.	ZEROX	56
& (P .GE. ABS(0.5*S*Q))) THEN	ZEROX	57
D = E = HALFDIF	ZEROX	58
ELSE	ZEROX	59
D = P/Q	ZEROX	60
ENDIF	ZEROX	61
ENDIF	ZEROX	62
AX = BX	ZEROX	63
FAX = FBX	ZEROX	64
IF (ABS(D).GT.TOLX) THEN	ZEROX	65
BX = BX + D	ZEROX	66
ELSEIF (CX.GT.BX) THEN	ZEROX	67
BX = BX + TOLX	ZEROX	68
ELSE	ZEROX	69
BX = BX - TOLX	ZEROX	70
ENDIF	ZEROX	71
IF(FINDV) THEN	ZEROX	72
FBX = FUNCTN(VAR2,BX,VAR3,VAR4)	ZEROX	73
ELSE	ZEROX	74
FBX = FUNCTN(BX,VAR2,VAR3,VAR4)	ZEROX	75
ENDIF	ZEROX	76
IF(FBX*FCX/ABS(FCX).GT.0.0) THEN	ZEROX	77
CX = AX	ZEROX	78
FCX = FAX	ZEROX	79
D = E = BX - AX	ZEROX	80
ENDIF	ZEROX	81
GO TO 19	ZEROX	82
END	ZEROX	83

User Provided Subroutine Examples

Subroutine INICURV

SUBROUTINE INICURV	INICURV	1
C**** THIS SUBROUTINE INITIALIZES THE NAMES, TABLES AND VALUE NAMES OF	INICURV	2
C**** THE RANDOM ACCESS CURVE FILES AND OPENS THE APPROPRIATE CURVES AS	INICURV	3
C**** NEEDED.	INICURV	4
C****	INICURV	5
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLG,REV7SE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLG,	FLAGS	6
& REV7SE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
CHARACTER ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
& THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/ ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
& TKOTYPE	CHARV	4
COMMON/CINDEX/MSTRNDX(3),NDXLEN,LUCURV	CINDEX	1
COMMON/RACURV/ IDBUG,IEXTOR,KURNAM(99)	RACURV	1
COMMON/VALUES/VALIN(15),VFUEL(4),VIDLE(4),VLIFT(2),VAERO(15)	VALUES	2
COMMON/TABLES/TFUEL (172),TIDLE (119),TAERO07(72),TAERO08(88),	TABLES	1
& TAERO09(79),TAERO10(62),TAERO11(91),TAERO12(76),	TABLES	2
& TAERO13(64),TAERO14(73),TAERO15(74),TAERO16(66),	TABLES	3
& TAERO17(66),TAERO18(77),TAERO19(73),TAERO20(76),	TABLES	4
& TAERO21(76),TAERO22(81),TAERO23(74)	TABLES	5
LOGICAL CRVFLAG	INICURV	12
C INTEGER	INICURV	13
C REAL	INICURV	14
DATA IDBUG, IEXTOR,LUCURV,NDXLEN,NDXTYPE	INICURV	15
& / 0, 0, 2, 3, 0/	INICURV	16
C****	INICURV	17
C**** OPEN RANDOM CURVE FILE WITH EXTENDED CORE STORAGE.	INICURV	18
IF(.NOT.CRVFLAG) THEN	INICURV	19
C CALL OPENMS (LUCURV,MSTRNDX,NDXLEN,NDXTYPE)	INICURV	20
ENDIF	INICURV	21
C****	INICURV	22
C**** INITIALIZE CURVE ARRAYS.	INICURV	23
IF(.NOT.CRVFLAG) THEN	INICURV	24
CRVFLAG = .TRUE.	INICURV	25
NDXTYPE = 0	INICURV	26
C CALL CURVSET(KURNAM,TABLE,NDIM(X),NAMEIN,NUNITI,NIPIN,	INICURV	27
C & NAMEOUT,NUNITO,NDPOUT(1),IDBUG,LUMSG)	INICURV	28
ENDIF	INICURV	29
RETURN	INICURV	30
END	INICURV	31

Subroutine FORCEX

SUBROUTINE FORCEX(ALPHA,CD,CL)	FORCEX	1
C**** THIS SUBROUTINE PROVIDES THE TOTAL FORCE COEFFICIENTS ALONG	FORCEX	2
C**** AND NORMAL TO THE FLIGHTPATH BY CALLING USER PROVIDED SUBROUTINE	FORCEX	3
C**** 'FXXAERO' TO DETERMINE LIFT AND DRAG COEFFICIENTS AND USER	FORCEX	4
C**** PROVIDED SUBROUTINES 'FXXENG', SPOOLUP, SPOOLDNF, AND SPOOLDNR TO	FORCEX	5
C**** PROVIDE THRUST AND FUEL FLOW.	FORCEX	6
C****	FORCEX	7
C**** SUBROUTINE FORCEX SHOULD REQUIRE ONLY THE FOLLOWING MODIFICATIONS:	FORCEX	8
C****	FORCEX	9
C**** 1) INITIALIZATION OF THE DATA STATEMENT PROVIDING VALUES FOR	FORCEX	10
C**** AIT, AR, B, CGPCT, CLALPH, CONFIG, DTDTMX, FLAP, FLPPCT, FLPPCT,	FORCEX	11
C**** HZ, LOADING, NENG, PWRCODE, RC, SWING, THRCRV, THTMAX,	FORCEX	12
C**** XIDLE, XLFMAX, AND XMIL.	FORCEX	13
C****	FORCEX	14
C**** 2) CHANGING THE CALLING STATEMENT FOR SUBROUTINES 'FXXAERO'	FORCEX	15
C**** AND 'FXXENG' TO THE USER PROVIDED NAMES WITH THE	FORCEX	16
C**** APPROPRIATE CALLING ARGUMENTS..	FORCEX	17
C****	FORCEX	18
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKCAS,VKTAS	AERO	2
COMMON/ENGINE/ AIT,AMACH,DTFAIL,EPR,FE,FG,FGPCT,NENG,PWRCODE,	ENGINE	1
& REVNDX,RC,THRUST,VTANGLE,WFUEL,XENG,XENGFLD,	ENGINE	2
& XENGOUT,XIDLE,XMIL,ZFN	ENGINE	3
COMMON/AIRBORNE/ ALPHA,ALPHMX,DTDT,GAMMAPP,ROCFPM,THETA,FLAP,XLF,XLFJ	AIRBORNE	1
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
CHARACTER ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
& THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/ ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
& TKOTYPE	CHARV	4
DATA AIT, AR, B, CGPCT, CLALPH, CONFIG, DTDTMX, FLAP, FLPPCT	FORCEX	27
& / 0.0, 7.000, 75.0, 25.0, 0.120, 1.0, 7.0, -1.0, 50.0 /,	FORCEX	28
& HZ, LOADING, NENG, PWRCODE, RC, SWING, THRCRV, THTMAX	FORCEX	29
& / 5.50, 1, 4, 4444., 0.0, 800., 'RC', 15.0 /,	FORCEX	30
& XIDLE, XLFMAX, XMIL	FORCEX	31
& / 0.06, 2.0, 0.50 /	FORCEX	32

CALL FXXAERO(ALPHA,AMACH,CL,CD,FLAP,SPOILER,VKCAS,XENGOUT)	FORCEX	33
IF(.NOT. STEADY) THEN	FORCEX	34
CALL FXXENG(AIT,AMACH,NENG,PWRCODE,QS,VKCAS,XENG,XIDLE,	FORCEX	35
FE,FG,THRUST,WFUEL,EPR)	FORCEX	36
C****	FORCEX	37
C**** ADJUST THRUST BY FGPCT FACTOR	FORCEX	38
FACTOR = 1.00 + FGPCT/100.	FORCEX	39
FG = FG*FACTOR	FORCEX	40
THRUST = THRUST*FACTOR	FORCEX	41
WFUEL = WFUEL*FACTOR	FORCEX	42
ENDIF	FORCEX	43
C****	FORCEX	44
C**** CALCULATE NET THRUST COEFFICIENT.	FORCEX	45
CTX = (FG*COSD (ALPHA+AIT) - FE)/QS	FORCEX	46
CTY = (FG*SIND (ALPHA+AIT))/QS	FORCEX	47
C****	FORCEX	48
C**** ADD USER INCREMENTS TO CL AND CD.	FORCEX	49
CL = CL + DCLX	FORCEX	50
CD = CD + DCDX	FORCEX	51
C****	FORCEX	52
C**** CALCULATE FORCE COEFFICIENTS CX AND CY.	FORCEX	53
CX = CD - CTX	FORCEX	54
CY = CL + CTY	FORCEX	55
RETURN	FORCEX	56
END	FORCEX	57

Subroutine FXXAERO

SUBROUTINE FXXAERO(Alpha,AMACH,CL,CD,FLAP,SPOILER,VKCAS,XENGOUT)	FXXAERO	1
C**** THIS SUBROUTINE CALCULATES OR PERFORMS A TABLE LOOKUP TO	FXXAERO	2
C**** DETERMINE THE LIFT AND DRAG COEFFICIENTS FOR THE AIRCRAFT. THIS	FXXAERO	3
C**** SUBROUTINE AND ITS NAME ARE PROVIDED BY THE USER.	FXXAERO	4
C****	FXXAERO	5
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/AIRCFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCFT	2
COMMON/VECTOR/ HVECT,VVECT	VECTOR	1
COMMON/AIRSPED/ VKABRK,VKAPP,VKBRAKE,VKEND,VKFAIL,VKFLAP,VKFLPMX,	AIRSPED	1
& VKMCG,VKROTAT,VKSTART,VKWIND,VWIND	AIRSPED	2
COMMON/RUNWAY/ ABARG,AOAABRK,BRAKMU,BRKFCR,GAMMARW,GRW,HAGL,	RUNWAY	1
& HCLEAR,HFLARE,HGEAR,HRUNWAY,IMU,RCR,ROLLMU,TIMEFLD,	RUNWAY	2
& TIMEBRK,TIMEFLP,TIMESBK,TIMESPL,XMU	RUNWAY	3
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
COMMON/ATMOS/ TEMPR,PRESS,RHO,AFPS,VISCOSK,DELTA,SIGMA,THETA,DTEMPF	ATMOS	1
LOGICAL AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRS,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHGT,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRS,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
CHARACTER ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
& THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/ ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
& TKOTYPE	CHARV	4
COMMON/RACURV/ IDBUG,IEXTOR,KURNAM(99)	RACURV	1
COMMON/VALUES/VALIN(15),VFUEL(4),VIDLE(4),VLIFT(2),VAERO(15)	VALUES	2
COMMON/TABLES/TFUEL (172),TIDLE (119),TAERO07(72),TAERO08(88),	TABLES	1
& TAERO09(79),TAERO10(62),TAERO11(91),TAERO12(76),	TABLES	2
& TAERO13(64),TAERO14(73),TAERO15(74),TAERO16(66),	TABLES	3
& TAERO17(66),TAERO18(77),TAERO19(73),TAERO20(76),	TABLES	4
& TAERO21(76),TAERO22(81),TAERO23(74)	TABLES	5
DATA DCDLFL,DCDCG,DCDWML,DCDEOT,DFLAPDT,DSPLRDT,DTGEAR,VKFLPMX	FXXAERO	17
& / 0.0000, 0.0000, 0.0000, 0.0000, 3.0, 9.0, 6.0, 225./	FXXAERO	18
C****	FXXAERO	19
C**** EMPIRICAL FORMULA FOR LANDING GEAR DRAG.	FXXAERO	20
C**** IF(DCDLGR.EQ.0.0) DCDLGR = (0.0032/SWING)*GWT**0.80	FXXAERO	21

C****		FXXAERO	22
C****	INITIALIZE VALIN ARRAY FOR CURVE FILE LOOKUPS.	FXXAERO	23
	VALIN(IA) = PRESALT	FXXAERO	24
	VALIN(IB) = AMACH	FXXAERO	25
	VALIN(IC) = FLTNDX	FXXAERO	26
	VALIN(ID) = FLAP	FXXAERO	27
	VALIN(IE) = VKCAS	FXXAERO	28
	VALIN(IF) = CGPCT	FXXAERO	29
	VALIN(IG) = ALPHA	FXXAERO	30
C****		FXXAERO	31
C****	DETERMINE LIFT COEFFICIENT, CL, USING CURVE LIFT.	FXXAERO	32
C	CALL CURVEL(KURNAM(XX),TABLE,TABLE,VALIN,VLIFT,	FXXAERO	33
C &	IDBUG,IEXTOR,LUMSG)	FXXAERO	34
	CLSLOPE = VLIFT(1)	FXXAERO	35
	CLINTCP = VLIFT(2)	FXXAERO	36
	VALIN(9) = CL = CLSLOPE*ALPHA + CLINTCP	FXXAERO	37
	IF((NCOUNT.EQ.100R.STEADY) .AND.	FXXAERO	38
	& (JDEBUG.EQ.7001.OR.JDEBUG.EQ.7777.OR.JDEBUG.EQ.9999)) THEN	FXXAERO	39
	WRITE(LUMSG,7001) VLIFT,VALIN(9),VALIN(7),VALIN(8)	FXXAERO	40
7001	FORMAT(' CLSLOPE,CLINTCP,CL,FLTNDX,FLAP =',3F10.6,2F8.1)	FXXAERO	41
	ENDIF	FXXAERO	42
C****		FXXAERO	43
C****	FLAP RETRACTION	FXXAERO	44
	IF((FLAPFLG .AND. (VKCAS.GE.VKFLAP .OR. VKCAS.GE.VKFLPMX)) .OR.	FXXAERO	45
	& (FLAPFLG .AND. TIMEROL.GE.(TDELAY+TFLP)))	FXXAERO	46
	& CALL FRETRAC(FLAP,VKFLAP,DFLAPDT)	FXXAERO	47
C****		FXXAERO	48
C****	SPOILER DEPLOYMENT	FXXAERO	49
	IF(.NOT.(SPLFLAG) .AND. TIMEROL.GE.(TDELAY+TSPL)) THEN	FXXAERO	50
	SPLREND = 90.0	FXXAERO	51
	CALL SPOIL('DEPLOY',SPOILER,SPLREND,DSPLRDT)	FXXAERO	52
	ENDIF	FXXAERO	53
	IF(LGRFLAG) THEN	FXXAERO	54
C****	DETERMINE DELTA DRAG COEFFICIENT FOR LANDING GEAR, DCDLGR,	FXXAERO	55
C****	USING CURVE LGRDRAG.	FXXAERO	56
C	CALL CURVEL(KURNAM(XX),TABLE,TABLE,VALIN,VAERO(Y),	FXXAERO	57
C &	IDBUG,IEXTOR,LUMSG)	FXXAERO	58
	DCDLGR = VAERO(Y)	FXXAERO	59
	IF(HAGL.GE.HGEAR) CALL GRETRAC(DCDLGR,DTGEAR,HAGL)	FXXAERO	60
	ELSE	FXXAERO	61
	DCDLGR = 0.0000	FXXAERO	62
	ENDIF	FXXAERO	63

C****		FXXAERO	64
C****	SUM INCREMENTS FROM CURVE FILES	FXXAERO	65
	CL = CLALPH*ALPHA	FXXAERO	66
	CL = CL + DCLSPL + DCLX	FXXAERO	67
	CD = CD + DCDSPL + DCDFLP + DCDREN + DCDLGR + DCDGE*(GEFACTR)	FXXAERO	68
	& + DCDG + DCDWML + DCDEOT + DCDX	FXXAERO	69
	IF ((NCOUNT.EQ.10.OR.STEADY) .AND.	FXXAERO	70
	& (JDEBUG.EQ.7002.OR.JDEBUG.EQ.7777.OR.JDEBUG.EQ.9999)) THEN	FXXAERO	71
	WRITE(LUMSG,7002) (VAERO(I),I=1,10)	FXXAERO	72
7002	FORMAT(' VAERO(1),SPLDEF,DCDSPL,DCDREN,DCDLGR,DCDGE,DCDCG,',	FXXAERO	73
	& ' DCDWML,DCDEOT,DCLSPL = ',11F10.6)	FXXAERO	74
	ENDIF	FXXAERO	75
	RETURN	FXXAERO	76
	END	FXXAERO	77

Subroutine GEFFECT

SUBROUTINE GEFFECT(HAGL,SRATIO)	GEFFECT	1
C SUBROUTINE GEFFECT(ALPHA,CL,HAGL,DCLGE,DCDGE) PARKS	GEFFECT	2
C**** THIS SUBROUTINE SUPPLIES THE GROUND EFFECT INCREMENTS TO THE	GEFFECT	3
C**** AERODYNAMIC COEFFICIENTS. THIS GENERIC SUBROUTINE PROVIDES	GEFFECT	4
C**** PREDICTED GROUND EFFECT EQUATIONS WHICH CAN BE MODIFIED BY THE	GEFFECT	5
C**** USER FOR A PARTICULAR AIRCRAFT.	GEFFECT	6
C****	GEFFECT	7
COMMON/AIRCRAFT/ AOA3PT,AR,B,CGPCT,CLALPH,CONFIG,DTDTMX,FLT,GWT,HZ,	AIRCRAFT	1
& LOADING,SWING,THTMAX,WNGLOD,XLFMAX	AIRCRAFT	2
COMMON/AERO/ CX,CY,DADTCMD,DCDX,DCLX,DTDTGEX,FLAP,FLPPCT,QS,	AERO	1
& SPDBRK,SPOILER,VKCAS,VKTAS	AERO	2
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOOVR7	CONST	3
CHARACTER ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
& THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/ ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
& TKOTYPE	CHARV	4
C**** B = WINGSPAN HZ = HEIGHT WING ABOVE TIRES	GEFFECT	12
C**** AR = ASPECT RATIO CLALPH = DCL/DALPHA (DIMENSIONLESS)	GEFFECT	13
C**** HAGL = ABSOLUTE ALTITUDE OF AIRCRAFT TIRES ABOVE RUNWAY	GEFFECT	14
PARAMETER (Fi=3.141592653589793)	GEFFECT	15
C**** HZB = HZ/B	GEFFECT	16
C**** HB = (HZ + HAGL)/B	GEFFECT	17
C****	GEFFECT	18
C**** SIGMA INTERPOLATION FORMULA: NACA TN D-970	GEFFECT	19
C**** "EFFECT OF GROUND PROXIMITY ON THE AERODYNAMIC" -- ETC	GEFFECT	20
C**** GIVEN BOTH IN-GROUND EFFECT AND OUT OF GROUND EFFECT AERODYNAMICS,	GEFFECT	21
C**** THIS FORMULA PROVIDES A RATIO BETWEEN THEM AS A FUNCTION OF	GEFFECT	22
C**** ALTITUDE (AGL). THE COMMENTED EQUATIONS BELOW SHOW HOW THIS RATIO	GEFFECT	23
C**** CAN BE USED IN THE CALLING SUBROUTINE.	GEFFECT	24
C**** CDOUT = CD OUT OF GROUND EFFECT; CD0 = CD AT HAGL = 0.0	GEFFECT	25
C**** CLOUT = CL OUT OF GROUND EFFECT; CL0 = CL AT HAGL = 0.0	GEFFECT	26
SIGMAZ = (1.0 - 1.32*HZB)/(1.05 + 7.4*HZB)	GEFFECT	27
SIGMA = (1.0 - 1.32*HB)/(1.05 + 7.4*HB)	GEFFECT	28
SRATIO = AMAX1(SIGMA/SIGMAZ,0.0)	GEFFECT	29
C**** CL = CLOGE + SRATIO*(CL0 - CLOGE)	GEFFECT	30
C**** CD = CDOGE + SRATIO*(CD0 - CDOGE)	GEFFECT	31
C**** DCLGE = SRATIO*(CL0 - CLOGE)	GEFFECT	32
C**** DCDGE = SRATIO*(CD0 - CDOGE)	GEFFECT	33
C****	GEFFECT	34
C**** PREDICTED GROUND EFFECT BY DR E K PARKS	GEFFECT	35
C**** WITHOUT IN-GROUND EFFECT AERODYNAMICS, THIS EQUATION PROVIDES AN	GEFFECT	36
C**** ESTIMATE OF THE GROUND EFFECT INCREMENTS TO THE OUT OF GROUND	GEFFECT	37
C**** EFFECT AERODYNAMICS.	GEFFECT	38
C X = 25.94*HB**2 + 1.0	GEFFECT	39
C DCDGE = -8.0*CL**2/(AR*PI**3*X**2)	GEFFECT	40
C DCLGE = 4.0*CL*CLALPH/(AR*PI**3*HB)*(X - SQRT(X)*CL))	GEFFECT	41
RETURN	GEFFECT	42
END	GEFFECT	43

Subroutine FXXENG

SUBROUTINE FXXENG(AIT,ALPHA,AMACH,NENG,PWRCODE,QS,VKCAS,XENG,	FXXENG	1
& XIDLE,FE,FG,FN,WF,EPR)	FXXENG	2
C**** THIS SUBROUTINE CALCULATE OR PERFORMS TABLE LOOKUPS TO DETERMINE	FXXENG	3
C**** THE NET THRUST AND FUEL FLOW FOR THE AIRCRAFT. THIS SUBROUTINE	FXXENG	4
C**** AND ITS NAME ARE PROVIDED BY THE USER.	FXXENG	5
C****	FXXENG	6
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/VECTOR/ HVECT,VVECT	VECTOR	1
COMMON/RUNWAY/ ABARG,AOABRK,BRAKMU,BRKFCR,GAMMARW,GRW,HAGL,	RUNWAY	1
& HCLEAR,HFLARE,HGEAR,HRUNWAY,IMU,RCR,ROLLMU,TIMEFLD,	RUNWAY	2
& TIMEBRK,TIMEFLP,TIMESBK,TIMESPL,XMU	RUNWAY	3
DOUBLEPRECISION FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	1
& ROC,RKAIR	FPINTEG	2
COMMON/FPINTEG/ FPVTAS,GAMMAR,FPDIST,PRESALT,FPACCEL,DGDTR,VHAS,	FPINTEG	3
& ROC,RKAIR(40)	FPINTEG	4
COMMON/ATMOS/ TEMPR,PRESS,RHO,AFPS,VISCOSE,DELTA,SIGMA,THETA,DTEMPF	ATMOS	1
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
LOGICAL AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,FPCTFLG,	FLAGS	1
& GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,REVRSE,ROTATE,	FLAGS	2
& RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,TERMFLG,VECTFLG,	FLAGS	3
& VFFLAG,WRITITR	FLAGS	4
COMMON/FLAGS/ AOA0FLG,BRKFLAG,CLRHTG,ERRFLAG,FAILFLG,FLAPFLG,	FLAGS	5
& FPCTFLG,GEFLAG,LGRFLAG,LIFTOFF,OVERFLG,REVFLAG,	FLAGS	6
& REVRSE,ROTATE,RTOFLAG,SBKFLAG,SPLFLAG,SPOOL,STEADY,	FLAGS	7
& TERMFLG,VECTFLG,VFFLAG,WRITITR	FLAGS	8
CHARACTER ENGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
& THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/ ENGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
& TKOTYPE	CHARV	4
COMMON/RACURV/ IDBUG,IEXTOR,KURNAM(99)	RACURV	1
COMMON/VALUES/VALIN(15),VFUEL(4),VIDLE(4),VLIFT(2),VAERO(15)	VALUES	2
COMMON/TABLES/TFUEL (172),TIDLE (119),TAERO07(72),TAERO08(88),	TABLES	1
& TAERO09(79),TAERO10(82),TAERO11(91),TAERO12(76),	TABLES	2
& TAERO13(64),TAERO14(73),TAERO15(74),TAERO16(66),	TABLES	3
& TAERO17(66),TAERO18(77),TAERO19(73),TAERO20(76),	TABLES	4
& TAERO21(76),TAERO22(81),TAERO23(74)	TABLES	5
LOGICAL SPOOL	FXXENG	18
REAL ATMARY(8)	FXXENG	19
EQUIVALENCE (TEMPR,ATMARY(1))	FXXENG	20
DATA DVECTDT/10.0/	FXXENG	21
FE = 0.	FXXENG	22
FG = 20000.	FXXENG	23
CALL ATMOSPH(PRESALT,ATMARY)	FXXENG	24

```

      IF ((NCOUNT.EQ.10.OR.STEADY) .AND.
&      (JDEBUG.EQ.8001.OR.JDEBUG.EQ.8888.OR.JDEBUG.EQ.9999)) THEN
        WRITE(LUMSG,8001) (VALIN(I),I=1,6),SIGMA
8001    FORMAT(' TAMB,PRESALT,VTT,DTEMPF,AMACH,PWSET,SIGMA = ',/
&      F10.1,2F10.0,F10.1,F10.3,F10.1,F10.3)
      ENDIF
      VALIN(1) = AMACH
      VALIN(2) = EPR
      VALIN(3) = PWRCODE
C    CALL CURVEL(KURNAM(XX),TABLE,VALIN,VFUEL,
C      IDBUG,IEXTOR,LUMSG)
C****
C**** VECTORED THRUST ANGLE REDUCTION
      IF (VECTFLG .AND. (HAGL.GE.HVECT .AND. VKCAS.GE.VVECT))
&      CALL TVECTOR(VTANGLE,HVECT,VVECT,DVECTDT)
      IF ((NCOUNT.EQ.10.OR.STEADY) .AND.
&      (JDEBUG.EQ.8002.OR.JDEBUG.EQ.8888.CR.JDEBUG.EQ.9999)) THEN
        WRITE(LUMSG,8002) VFUEL
8002    FORMAT(' FG,FE,WF,EPR = ',3F10.0,F10.3)
      ENDIF
      CDRAM = FE/QS
      FN = FG*COSD(ALPHA + AIT) - FE
      WF = 4000.
C****
C**** CALL SPOOLUP FOR ROLLING TAKEOFFS
      IF (TKOTYPE.EQ.'ROLLING' .AND. (.NOT.SPOOL)) THEN
        CALL SPOOLUP(0.0,TIME,FLOAT(NENG)*XIDLE,XENG,SPOOL,XENG)
      ELSEIF(TKOTYPE.EQ.'ROLLING' .AND. ( SPOOL)) THEN
        SPOOL = .FALSE.
        TKOTYPE = 'SPOOLED'
      ENDIF
      RETURN
END

```

```

FXXENG 25
FXXENG 26
FXXENG 27
FXXENG 28
FXXENG 29
FXXENG 30
FXXENG 31
FXXENG 32
FXXENG 33
FXXENG 34
FXXENG 35
FXXENG 36
FXXENG 37
FXXENG 38
FXXENG 39
FXXENG 40
FXXENG 41
FXXENG 42
FXXENG 43
FXXENG 44
FXXENG 45
FXXENG 46
FXXENG 47
FXXENG 48
FXXENG 49
FXXENG 50
FXXENG 51
FXXENG 52
FXXENG 53
FXXENG 54
FXXENG 55
FXXENG 56
FXXENG 57

```

Subroutine SPOOLUP

SUBROUTINE SPOOLUP(ENGNDX,TIME,XENG0,XENGTRN,SPOOL,XENG)	SPOOLUP	1
C**** THIS SUBROUTINE DETERMINES THE RATIO OF THRUST TO TAKEOFF RATED	SPOOLUP	2
C**** THRUST DURING A ROLLING TAKEOFF USING SUBROUTINE TABINT AND DATA	SPOOLUP	3
C**** ARRAY SPOOLA . THIS SUBROUTINE IS JUST AN EXAMPLE AND DOES NOT	SPOOLUP	4
C**** APPLY TO ALL AIRCRAFT.	SPOOLUP	5
C****	SPOOLUP	6
DOUBLE PRECISION TIME	SPOOLUP	7
LOGICAL NDXSET,SPOOL	SPOOLUP	8
REAL SPOOLA(26)	SPOOLUP	9
DATA SPOOLA/ 0.0, 2.0, 4.0, 6.0, 8.0,10.0,20.0,30.0,0.0,1.0,	SPOOLUP	10
& 0.37,0.37,0.37,0.37,0.37,0.37,1.00,1.00,	SPOOLUP	11
& 0.37,0.37,0.37,0.37,0.37,0.37,1.00,1.00/	SPOOLUP	12
IF(.NOT.(NDXSET)) THEN	SPOOLUP	13
C**** INITIALIZE PREVIOUS SPOOL FACTOR, SPOOLXJ, START TIME, TIMEZ,	SPOOLUP	14
C**** AND ENGINE LOOKUP INDEX, ENGNDX.	SPOOLUP	15
NDXSET = .TRUE.	SPOOLUP	16
SPOOL = .FALSE.	SPOOLUP	17
SPOOLXJ = 99.0	SPOOLUP	18
TIMEZ = FLOAT(TIME)	SPOOLUP	19
ENDIF	SPOOLUP	20
TIMEDIF = FLOAT(TIME) - TIMEZ	SPOOLUP	21
IF (TIMEDIF.GE.0.0 .AND. TIMEDIF.LT.30.0) THEN	SPOOLUP	22
SPOOLXJ = SPOOLX	SPOOLUP	23
CALL TABINT(TIMEDIF,SPOOLX,ENGNDX,8,2,SPOOLA(1),INDEX)	SPOOLUP	24
XENG = XENG0 + XENGTRN*SPOOLX	SPOOLUP	25
ELSEIF(TIMEDIF.GT.0.0 .AND. SPOOLXJ.EQ.SPOOLX) THEN	SPOOLUP	26
C**** THROTTLE TRANSIENT COMPLETE. RESET NDXSET FLAG AND SET STATUS	SPOOLUP	27
C**** FLAG TO .TRUE.	SPOOLUP	28
NDXSET = .FALSE.	SPOOLUP	29
SPOOL = .TRUE.	SPOOLUP	30
ENDIF	SPOOLUP	31
RETURN	SPOOLUP	32
END	SPOOLUP	33

Subroutine SPOOLDNF

```

SUBROUTINE SPOOLDNF(TIME,XENGEND,XENGTRN,SPOOL,XENG,LUMSG)
C**** THIS SUBROUTINE DETERMINES THE TOTAL NUMBER OF ENGINES REMAINING
C**** DURING A THROTTLE CHOP OR FUEL CUT USING SUBROUTINE TABINT AND
C**** DATA ARRAY SPOOLA . THIS SUBROUTINE IS JUST AN EXAMPLE AND DOES
C**** NOT APPLY TO ALL AIRCRAFT.
C****
      CHARACTER ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,
&              THRCRV*3,TKOTYPE*7
      COMMON/CHARV/  ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,
&                  TKOTYPE
      DOUBLE PRECISION TIME
      LOGICAL NDXSET,SPOOL
      REAL SPOOLA(56)
      DATA SPOOLA/ 0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8,
&                  2.0, 2.5, 3.0, 4.0, 5.0, 6.0, 7.0,10.0,0.00,1.00,
&                  1.00,0.85,0.64,0.44,0.35,0.27,0.23,0.20,0.18,0.17,
&                  0.16,0.14,0.11,0.10,0.08,0.07,0.06,0.05,
&                  1.00,0.85,0.64,0.44,0.31,0.21,0.15,0.12,0.10,0.08,
&                  0.07,0.06,0.04,0.03,0.02,0.01,0.00,0.00/
      IF(.NOT.(NDXSET)) THEN
C****   INITIALIZE PREVIOUS SPOOL FACTOR, SPOOLXJ, START TIME, TIMEZ,
C****   AND ENGINE LOOKUP INDEX, ENGNDX.
      NDXSET = .TRUE.
      SPOOL = .FALSE.
      IF      (FAILST.EQ.'IDLE') THEN
        ENGNDX = 0.0
      ELSEIF (FAILST.EQ.'OFF ') THEN
        ENGNDX = 1.0
      ELSE
        WRITE(LUMSG,1001) FAILST
1001      FORMAT(' INVALID FAILST PASSED TO SPOOLDNF, FAILST = ',A5)
      ENDIF
      SPOOLXJ = 99.0
      TIMEZ   = FLOAT(TIME)
      ENDIF
      TIMEDIFF = FLOAT(TIME) - TIMEZ
      IF(SPOOLXJ.NE.SPOOLX .AND. XENG.GT.XENGEND) THEN
        SPOOLXJ = SPOOLX
        CALL TABINT(TIMEDIFF,SPOOLX,ENGNDX,18,2,SPOOLA(1),INDEX)
        XENG = XENGTRN*SPOOLX
      ELSE
C****   THROTTLE TRANSIENT COMPLETE. RESET NDXSET FLAG AND SET STATUS
C****   FLAG TO .TRUE.
      NDXSET = .FALSE.
      SPOOL = .TRUE.
      XENG = XENGEND
      ENDIF
      RETURN
      END

```

SPOOLDNF	1
SPOOLDNF	2
SPOOLDNF	3
SPOOLDNF	4
SPOOLDNF	5
SPOOLDNF	6
CHARV	1
CHARV	2
CHARV	3
CHARV	4
SPOOLDNF	8
SPOOLDNF	9
SPOOLDNF	10
SPOOLDNF	11
SPOOLDNF	12
SPOOLDNF	13
SPOOLDNF	14
SPOOLDNF	15
SPOOLDNF	16
SPOOLDNF	17
SPOOLDNF	18
SPOOLDNF	19
SPOOLDNF	20
SPOOLDNF	21
SPOOLDNF	22
SPOOLDNF	23
SPOOLDNF	24
SPOOLDNF	25
SPOOLDNF	26
SPOOLDNF	27
SPOOLDNF	28
SPOOLDNF	29
SPOOLDNF	30
SPOOLDNF	31
SPOOLDNF	32
SPOOLDNF	33
SPOOLDNF	34
SPOOLDNF	35
SPOOLDNF	36
SPOOLDNF	37
SPOOLDNF	38
SPOOLDNF	39
SPOOLDNF	40
SPOOLDNF	41
SPOOLDNF	42
SPOOLDNF	43
SPOOLDNF	44
SPOOLDNF	45
SPOOLDNF	46

Subroutine SPOOLDNR

SUBROUTINE SPOOLDNR(TIME,XENGEND,XENGTRN,SPOOL,XENG,LUMSG)	SPOOLDNR	1
C**** THIS SUBROUTINE DETERMINES THE TOTAL NUMBER OF ENGINES REMAINING	SPOOLDNR	2
C**** DURING A THROTTLE CHOP OR FUEL CUT USING SUBROUTINE TABINT AND	SPOOLDNR	3
C**** DATA ARRAY SPOOLA . THIS SUBROUTINE IS JUST AN EXAMPLE AND DOES	SPOOLDNR	4
C**** NOT APPLY TO ALL AIRCRAFT.	SPOOLDNR	5
C****	SPOOLDNR	6
CHARACTER ENGGRP*3,FAILGRP*3,FAILMOD*5,FAILST*4,MVR*3,MANUVR*6,	CHARV	1
& THRCRV*3,TKOTYPE*7	CHARV	2
COMMON/CHARV/ ENGGRP,FAILGRP,FAILMOD,FAILST,MVR,MANUVR,THRCRV,	CHARV	3
& TKOTYPE	CHARV	4
DOUBLE PRECISION TIME	SPOOLDNR	8
LOGICAL NDXSET,SPOOL	SPOOLDNR	9
REAL SPOOLA(56)	SPOOLDNR	10
DATA SPOOLA/ 0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8,	SPOOLDNR	11
& 2.0, 2.5, 3.0, 4.0, 5.0, 6.0, 7.0,10.0,0.00,1.00,	SPOOLDNR	12
& 1.00,0.85,0.64,0.44,0.35,0.27,0.23,0.20,0.18,0.17,	SPOOLDNR	13
& 0.16,0.14,0.11,0.10,0.08,0.07,0.06,0.06,	SPOOLDNR	14
& 1.00,0.85,0.64,0.44,0.31,0.21,0.15,0.12,0.10,0.08,	SPOOLDNR	15
& 0.07,0.06,0.04,0.03,0.02,0.01,0.00,0.00/	SPOOLDNR	16
IF(.NOT.(NDXSET)) THEN	SPOOLDNR	17
C**** INITIALIZE PREVIOUS SPOOL FACTOR, SPOOLXJ, START TIME, TIMEZ,	SPOOLDNR	18
C**** AND ENGINE LOOKUP INDEX, ENGNDX.	SPOOLDNR	19
NDXSET = .TRUE.	SPOOLDNR	20
SPOOL = .FALSE.	SPOOLDNR	21
IF (FAILST.EQ.'IDLE') THEN	SPOOLDNR	22
ENGNDX = 0.0	SPOOLDNR	23
ELSEIF (FAILST.EQ.'OFF ') THEN	SPOOLDNR	24
ENGNDX = 1.0	SPOOLDNR	25
ELSE	SPOOLDNR	26
WRITE(LUMSG,1001) FAILST	SPOOLDNR	27
1001 FORMAT(' INVALID FAILST PASSED TO SPOOLDNF, FAILST = ',A5)	SPOOLDNR	28
ENDIF	SPOOLDNR	29
SPOOLXJ = 99.0	SPOOLDNR	30
TIMEZ = FLOAT(TIME)	SPOOLDNR	31
ENDIF	SPOOLDNR	32
TIMEDIF = FLOAT('TIME') - TIMEZ	SPOOLDNR	33
IF(SPOOLXJ.NE.SPOOLX .AND. XENG.GT.XENGEND) THEN	SPOOLDNR	34
SPOOLXJ = SPOOLX	SPOOLDNR	35
CALL TABINT(TIMEDIF,SPOOLX,ENGNDX,18,2,SPOOLA(1),INDEX)	SPOOLDNR	36
XENG = XENGTRN*SPOOLX	SPOOLDNR	37
ELSE	SPOOLDNR	38
C**** THROTTLE TRANSIENT COMPLETE. RESET NDXSET FLAG AND SET STATUS	SPOOLDNR	39
C**** FLAG TO .TRUE.	SPOOLDNR	40
NDXSET = .FALSE.	SPOOLDNR	41
SPOOL = .TRUE.	SPOOLDNR	42
XENG = XENGEND	SPOOLDNR	43
ENDIF	SPOOLDNR	44
RETURN	SPOOLDNR	45
END	SPOOLDNR	46

Subroutine GENMU

SUBROUTINE GENMU(VKTGS,WTML,XMAIN,XNOSE,YCG)	GENMU	1
C**** THIS SUBROUTINE PROVIDES THE BRAKING COEFFICIENT OF FRICTION.	GENMU	2
C****	GENMU	3
DOUBLEPRECISION DTIME,DTIMEJ,TIME	CTRL	1
PARAMETER (LUIN=3,LUOUT=4)	CTRL	2
COMMON/CTRL/ DTIME,DTIMEJ,ICOUNT,JDEBUG,KENG,LINENUM,LUMSG,NCOUNT,	CTRL	3
& NEQ,NPAGE,TIME,TIMEROL	CTRL	4
COMMON/RUNWAY/ ABARG,AOAABRK,BRAKMU,BRKFCR,GAMMARW,GRW,HAGL,	RUNWAY	1
& HCLEAR,HFLARE,HGEAR,HRUNWAY,IMU,RCR,ROLLMU,TIMEFLD,	RUNWAY	2
& TIMEBRK,TIMEFLP,TIMESBK,TIMESPL,XMU	RUNWAY	3
PARAMETER (ASL=661.48,FPSKTS=1.687806,G= 32.174,	CONST	1
& RX=57.29577951308,TSLF=59.0,ZERO=0.0)	CONST	2
COMMON/CONST/ ASLSQR5,TWOVR7	CONST	3
COMMON/RACURV/ IDBUG,IEXTOR,KURNAM(99)	RACURV	1
COMMON/VALUES/VALIN(15),VFUEL(4),VIDLE(4),VLIFT(2),VAERO(15)	VALUES	2
COMMON/TABLES/TFUEL (172),TIDLE (119),TAERO07(72),TAERO08(88),	TABLES	1
& TAERO09(79),TAERO10(62),TAERO11(91),TAERO12(76),	TABLES	2
& TAERO13(64),TAERO14(73),TAERO15(74),TAERO16(66),	TABLES	3
& TAERO17(66),TAERO18(77),TAERO19(73),TAERO20(76),	TABLES	4
& TAERO21(76),TAERO22(81),TAERO23(74)	TABLES	5
C****	GENMU	10
C**** F-XX CONSTANTS	GENMU	11
C**** XMAIN = DISTANCE MAIN GEAR TO CG LODMAIN = MAIN GEAR LOAD	GENMU	12
C**** XNOSE = DISTANCE NOSE GEAR TO CG LODNOSE= NOSE GEAR LOAD	GENMU	12
C**** YCG = HEIGHT OF CG ABOVE GROUND WTML = WEIGHT - LIFT	GENMU	13
XMAIN = 9.25	GENMU	14
XNOSE = 54.92	GENMU	15
YCG = 14.00	GENMU	16
C****	GENMU	17
C**** IMU = 0 USE CONSTANT MU MODEL; IMU = 1 USE MAXMU CURVE	GENMU	18
C**** IMU = 2 USE GENERIC DRY RUNWAY; IMU = 3 USE GENERIC WET RUNWAY	GENMU	19
C**** IF IMU = 0, GENMU IS NOT CALLED BY DERIVGR.	GENMU	20
IF (IMU.EQ.1) THEN	GENMU	21
C**** RANDOM ACCESS CURVE MAXMU	GENMU	22
C VALIN(13) = WTML*VKTGS**2/1.0E+09	GENMU	23
C VALIN(14) = RCR	GENMU	24
C CALL CURVEL(KURNAM(19),TAERO19,TAERO19,VALIN,VAERO(12),	GENMU	25
C & IDBUG,IEXTOR,LUMSG)	GENMU	26
C BRAKMU = VAERO(12)	GENMU	27
C IF (NCOUNT.EQ.10 .AND.	GENMU	28
C & (JDEBUG.EQ.7003.OR.JDEBUG.EQ.7777.OR.JDEBUG.EQ.9999) THEN	GENMU	29
C WRITE(LUMSG,7003) VAERO(12)	GENMU	30
C7003 FORMAT(' BRAKMU = ',F10.3)	GENMU	31
C ENDDIF	GENMU	32
ELSEIF(IMU.EQ.2) THEN	GENMU	33
C**** DRY RUNWAY	GENMU	34
VKTGSX = AMAX1((130.0 - VKTGS),ZERO)	GENMU	35
BRAKMU = (0.490 - VKTGSX*0.00127)	GENMU	36

ELSEIF(IMU.EQ.3) THEN	GENMU	37
C**** WET RUNWAY	GENMU	38
IF(VKTGS .LT. 50.0) BRAKMU = (0.3630 - 0.0006737*VKTGS)	GENMU	39
IF(VKTGS .GE. 50.0) BRAKMU = (0.3063 - 0.0022300*VKTGS)	GENMU	40
ENDIF	GENMU	41
BRAKMU = BRAKMU*BRIKCTR	GENMU	42
XMU = BRAKMU	GENMU	43
RETURN	GENMU	44
END	GENMU	45





APPENDIX C: MICROSOFT FORTRAN COMPILER INFORMATION

This section describes the internal software structure used to generate the TOLAND libraries and executables with the Microsoft™ FORTRAN compiler (version 5.1 or later). The location of the subroutines within each library is also detailed.

Executables (.EXE files) are generated from the user provided source code and the following libraries: CURVE.LIB, PANDFQ.LIB, and TOLAND.LIB. The libraries (PANDFQ and CURVE) satisfy external references for UFTAS subroutines such as TABINT, HORP, STASK, and other random access curve file subroutines. The TOLAND.LIB library contains the following object libraries: LIBMAIN.OBJ, LIBTKO.OBJ, LIBLND.OBJ, and LIBGRND.OBJ. These object libraries are generated from their source library counterparts: LIBMAIN.FOR, LIBTKO.FOR, LIBLND.FOR, LIBGRND.FOR, and LIBGENU.FOR, respectively. LIBGENU.OBJ is not included in TOLAND.LIB. Because many of the generic subroutines will be replaced by user specific subroutines, any generic subroutines within the executable file must be explicitly added during the build process. Individual generic subroutine object files are provided for this purpose or a user may generate their own custom general user object library (LIBGENU.OBJ). This object library (along with TOLAND.LIB and user TOLAND source file) would comprise the program list for the executable file build. Table C-1 indicates which subroutines are contained within each library.

TABLE C-1
SUBROUTINE LOCATIONS

LIBMAIN	LIBTKO	LIBLND	LIBGRND	LIBGENU
INITIAL	TAKOFF	LANDNG	ROLL	INICURV
INTX	DERIVAT	STEDYST	DERIVGR	FORCEX
INTG	ERROR	FLARENZ		FXXAERO
HALT		APPROCH		GEFFECT
ATMOSPH		FLARE		FXXENG
SPEED		DERIVAL		SPOOLUP
INTERP		ITRLND		SPOOLDNF
		DGDT		SPOOLDNR
		DVDT		GENMU
		DVTDH		FRETRAC
		DADH		GRETRAC
		DDELTDH		PITCH
		DSIGDH		SPDBRAK
		ZEROX		SPOIL
				TVECTOR

All common blocks for the TOLAND program are contained in separate files with the .CMN extension.

External references satisfied with PANDFQ.LIB, a general purpose flight dynamics library, are found within four sublibraries, LIBAERO, LIBARRAY, LIBTRIG, and LIBUTIL. These sublibraries contain aerodynamic, array processing, trigonometric, and miscellaneous subroutines, respectively.

LIBAERO contains UFTAS routines HORP, MACH, QORV, SRHOS, STASK, and VCMACH. LIBARRAY contains UFTAS routines DIVC, IDENT, INVERS, and NORM. LIBTRIG contains routines COSD, SIND, and TAND; they accept arguments in degrees instead of radians. LIBUTIL contains UFTAS routines LBSORT, RDBFLE, TABINT and WPSDIFF. WPSDIFF is a new name for DUZ2 which performed wild-pointing, smoothing and differentiation of time history data.

Program compilation, input file editing, program execution, output file viewing and output file printing are accomplished with dCOM Directory Commander™ macros. The macro file is named TOLAND.MAC and should be stored in the dCOM subdirectory. The output file printing macro adds a filename, date/time tag to the print spooler and also formats the output to landscape mode.



APPENDIX D: NOMENCLATURE

This section provides an index of program variables. Variables are included here if they are namelist inputs, program outputs, stored within a common block, or passed as a calling argument to a subroutine.

FORTRAN Name	Symbol	Description	Dimension	Usage	
				COMMON	SUBROUTINE NAME LIST
A	a	Dummy variable passed to subroutine ZEROX as angle of attack argument to function FUNCTN	----		ZEROX
ABARG	A	Average deceleration	feet/second ²	RUNWAY	
ACCEL	a _t	Acceleration along ground path	feet/second ²	INTEG	
ACTION		Action for spoiler or speed brake deflection (e.g. DEPLOY, RETRACT)	----		SPOIL, SPDBRAK
AFPS	a	Current speed of sound	feet/second	ATMOS	
AIRDIST	S _{TPD}	Ground distance at touchdown	feet		
AIT	i _t	Thrust incidence angle	degrees	ENGINE	
ALPHA	α	Current angle of attack	degrees		APPROCH, DERIVAL, LND DERIVAT, DERIVGR, D'GDT, DVDVT, FLARE, FLARENZ, FORCEX, INITIAL, INTX, LANDNG, PITCH, ROLL, STEDYST, TAKOFF
ALPHAJ	α _j	Previous angle of attack from last integration	degrees	AIRBORN	
ALPHAX	α	Dummy variable passed to functions DGDT and DVDVT	degrees		DGDT, DVDVT
ALPHMX	α _{max}	Maximum angle of attack limit	degrees	AIRBORN	
AMACH	M	Mach number	----	ENGINE	SPEED

FORTRAN Name	Symbol	Description	Dimension	Usage		NAME LIST
				COMMON	SUBROUTINE	
AOAABRK	α_{arobrk}	Angle of attack for aerobraking	degrees	RUNWAY		ROL2
AOA0FLG	β_{aoa0}	Zero angle of attack flag	----	FLAGS		
AOA3PT	$\alpha_{3\text{-point}}$	Angle of attack for a 3-point attitude	----	AIRCRAFT		DATA2
APPDIST	S_{app}	Approach ground distance	feet			
AR	b^2/S_{wing}	Aspect ratio of the wing	----	AIRCRAFT		
ARRAY		Dummy argument for subroutine ATMOSPH	----		ATMOSPH	
ASL	a_4	Speed of sound at sea level	knots	CONST		
ASLSQRS	$a_4/\sqrt{5}$	Product of the Speed of sound at sea level and $\sqrt{5}$	knots	CONST		
B	b	span of the wing	feet	AIRCRAFT		
B	B	Dummy variable passed to subroutine ZEROX as angle of attack argument to function FUNCTN	----		ZEROX	
BRAKMU	μ_{brake}	Braking coefficient of friction	----	RUNWAY		ROL
BRKFCTR		Braking Factor (applied to BRAKMU)	----	RUNWAY		ROL2
BRKFLAG	β_{brake}	Brake flag	----	FLAGS		
CD	C_D	Drag coefficient	----		FORCEX	
CGPCT	c_g	Longitudinal center of gravity	percent mean aerodynamic chord	AIRCRAFT		DATA
CL	C_L	Lift coefficient	----		FORCEX, GEFFECT	
CLALPH	C_{L_α}	Lif: curve slope	radians ⁻¹	AIRCRAFT		
CLRHGT	β_{clrhgt}	Clearance height flag	----	FLAGS		
CONFIG		Aircraft Configuration Variable (user definable)	----	AIRCRAFT		DATA2
CONSTANT		Character value containing either 'VC', 'VE' or 'MACH' for determining the climb speed derivative with respect to pressure altitude, dV/dH	----		DVTDH	
CX	C_x	Force coefficient along flightpath	----	AERO		
CY	C_y	Force coefficient normal to flightpath	----	AERO		
DADT	$d\alpha/dt$	Time rate of change of angle of attack	degree/second			

FORTRAN Name	Symbol	Description	Dimension	Usage		NAME LIST
				COMMON	SUBROUTINE	
DADTCMD	$(da/dt)_{cmd}$	Commanded time rate of change of angle of attack	degree/second	AERO	PITCH	DATA
DCDGE	δC_{D_p}	Ground effect delta drag coefficient	----		GEFFECT	
DCDLGR	$\delta C_{D_{gear}}$	Landing gear delta drag coefficient	----		GRETRAC	
DCDX	δC_D	Delta drag coefficient	----	AERO	GEFFECT	DATA2
DCLGE	δC_{L_p}	Ground effect delta lift coefficient	----			
DCLX	δC_L	Delta lift coefficient	----	AERO		DATA2
DDIST	S_i	Incremental ground distance (distance per integration step)	feet		INTG	
DELTA	δ	Current pressure ratio	----	ATMOS	INTX	
DERIV		Subroutine name variable passed to Runge-Kutta numerical integration subroutine INTX	----			
DFLAPDT	$\Delta \delta / dt$	Rate of change of flap deflection	degree/second		FRETRAC	
DGDTR	a_n	Acceleration normal to flightpath	radian/second ²	FPINTEG		
DIST	S_a	Current air distance (ground distance uncorrected for wind)	feet	INTEG	INTG	
DISTJ	S_a	Previous air distance (ground distance uncorrected for wind)	feet	INTEG	INTG	TKO2
DISTMAX	$S_{a_{max}}$	Ground distance limit for takeoff simulation	feet			
DRIVER		Boundary limit variable on zero finding subroutine	----		ITRLND	
DSBKDT	$\Delta \delta_{sk}/dt$	Rate of change of speed brake deflection	degree/second		SPDBRAK	
DSPLRDT	$\Delta \delta_{sp}/dt$	Rate of change of spoiler deflection	degree/second		SPOIL	
DTDT	$d\theta/dt$	Current pitch rate	degree/second	AIRBORN		
DTDTGEX		Pitch rate capability loss factor in ground effect	----	AERO	PITCH	LND2
DTDTMX	$(d\theta/dt)_{max}$	Maximum pitch rate during landing simulation	degree/second	AIRCRAFT		DATA
DTEMPF	δT_{td}	Delta temperature from standard day	degrees Fahrenheit	ATMOS		DATA2
DTFAIL	δt_{fail}	Time for the failed engine to lose thrust	seconds	ENGINE		
DTGEAR	δt_{gear}	Time for gear retraction	seconds		GRETRAC	

FORTRAN Name	Symbol	Description	Dimension	Usage	
				COMMON	SUBROUTINE NAME LIST
DTIME	Δt	Runge-Kutta numerical integration step size	seconds	CTRL	INTX, PITCH DATA2
DTIMEJ	Δt_j	Runge-Kutta numerical integration step size	seconds	CTRL	
DVECTDT	$\Delta \delta / dt$	Rate of change of thrust vector angle	degree/second		TVECTOR
ENGGRP	$\phi_{\text{engin.}}$	Operating Engine Group	----	CHARV	DATA2
ENGNDX		Engine Spoolup index (user definable)	----		
EPR	EPR	Engine pressure ratio	----	ENGINE	DATA2
ERRFLAG	P_{error}	Error flag	----	FLAGS	
ERROR	ϵ	Present error for zero finding routine	----		ITRLND
ERRORJ	ϵ_j	Previous error for zero finding routine	----		ITRLND
FACTOR		Multiplicative jump factor on zero finding subroutine	----		ITRLND
FAILFLG	P_{engfail}	Engine failure flag	----	FLAGS	
FAILGRP	ϕ_{fail}	Fail Engine Group	----	CHARV	DATA2
FAILMOD		Engine Failure Mode (SEIZE or SPOOL)	----	CHARV	DATA2
FAILST		Engine Failure State (IDLE or OFF)	----	CHARV	DATA2
FE	F_E	Ram Drag	pound-force	ENGINE	
FG	F_G	Gross Thrust	pound-force	ENGINE	
FGPCT	$\%F_G$	Gross Thrust Percentage Increment	percent F_G		DATA2
FINDV	P_{findv}	Steady state condition flag	----		STEDYST, ZEROX
FLAP	δ_f	Flap deflection	degrees	AERO	FRETRAC DATA
FLAPFLG	P_{flap}	Flap flag	----	FLAGS	
FLAP0	δ_{f0}	Initial flap deflection	degrees		INITIAL
FLPARY	ϕ_{flap}	Flap deflection array	degrees		TKOARY
FLPPCT	$\% \text{Flap}$	Flap percentage setting	percent max flap	FLAPDAT	DATA2
FLRDIST	S_{flare}	Flare ground distance	feet	AERO	
FLT	$\#_{\text{flight}}$	Flight number	----	AIRCRAFT	DATA2
FLTNDX		Flight index (user definable)	----		DATA2
FPACCEL	a_t	Acceleration along flightpath	feet/second ²	FPINTEG	

FORTRAN Name	Symb.	Description	Dimension	Usage	
				COMMON	SUBROUTINE NAME LIST
FPCTFLG	P_{flap}	Flap percentage flag	----	FLAGS	
FPDIST	S_{fp}	Flightpath air distance	feet	FPINTEG	
FPSKTS		Conversion factor from knots to feet per second	feet/second/knot	CONST	
FPVTAS	V_t	Flightpath true airspeed	feet/second	FPINTEG	DGDT, DVDT, STEDYST ZEROX
FUNCTN		Function DGDT or DVDT passed to Function ZEROX zero finding routine	----		
G	g	Acceleration due to gravity	feet/second ²	CONST	
GAMMA	γ	Flightpath angle (inertial reference frame)	degrees		
GAMMAPP	γ_{approach}	Approach flightpath angle (inertial reference frame)	degrees	AIRBORN	APPROCH, FLARENZ LND
GAMMAR	γ	Flightpath angle	radians	FPINTEG	DGDT, DVDT, INTG, SPEED, STEDYST
GAMMARW	γ_{runway}	Runway slope	degrees	RUNWAY	DATA
GAMMATD	$\gamma_{\text{touchdown}}$	Touchdown flightpath angle (inertial reference frame)	degrees		FLARENZ
GDIST	S_g	Ground distance	feet		APPROCH, FLARE, INTG, ROLL
GEFLAG	P_{geffect}	Ground effect flag	----	FLAGS	
GROUP		Group of variables to be initialized	----		INITIAL
CRW	γ_{runway}	Runway slope	radians	RUNWAY	
GWT	W_{gross}	Gross weight	pound-force	AIRCRAFT	DGDT, DVDT, INTG
GWT0	W_{gross0}	Initial gross weight	pound-force		INITIAL
HAGL	H_{gl}	Current altitude above liftoff point for takeoffs	feet	RUNWAY	APPROCH, FLARENZ, GEFFECT, GRETRAC
HAGL	H_{gl}	Current altitude above runway for landings	feet	RUNWAY	APPROCH, FLARENZ, GEFFECT, GRETRAC

FORTRAN Name	Symbol	Description	Dimension	Usage	
				COMMON	SUBROUTINE NAME LIST
HC	H _c	Pressure altitude	feet		DADH, DDELTDH, DSIGDH
HCLEAR	H _c _{clear}	Obstacle clearance height	feet	RUNWAY	APPROCH, FLARENZ DATA
HCLIMB	H _c _{limb}	Height simulation switches to constant airspeed climb	feet		TKO2
HCLMOUT	H _c _{limbout}	Height simulation switches to constant theta climb	feet		TKO2
HFLARE	H _{flare}	Flare initiation height	feet	RUNWAY	APPROCH, FLARENZ LND
HGEAR	H _{gear}	Landing Gear Retraction Height:	feet	RUNWAY	TKO
HMAX	H _{max}	Termination Height Limit	feet	RUNWAY	TKO
HRUNWAY	H _{runway}	Runway pressure altitude	feet	RUNWAY	APPROCH, FLARENZ DATA
HVCTARY	## _{hvect}	Vectored thrust altitude array	feet	VECTDAT	TKOARY
HVECT	H _{vect}	Current vectored thrust altitude array element	feet		TVECTOR
HZ	H _z	Height of wing above the bottom of the tires	feet	AIRCRAFT	
ICOUNT	i _c _{count}	Integration loop initializing variable	----	CTRL	
IDBUG		UFTAS debug code	----	RACURV	DATA2
IEXTOR		Extrapolation override code	----	RACURV	
IFLAP	i _{flap}	Current element of arrays FLPARY and VFLPARY	----	FLAPDAT	
IGEAR	i _{gear}	Current element of array LGRARY	----	GEARDAT	
IMU		Braking coefficient selector	----	RUNWAY	ROL
IVECT	i _{vect}	Current element of arrays XNUARY, HVCTARY, and VVCTARY	----	VECTDAT	
JDEBUG		TOLAND Debug Code	----	CTRL	DATA2
JFLAG	P _j	Subroutine ITRLND search status variable	----		ITRLND
KENG		Subroutine 'FXXENG' output switch	----	CTRL	
KURNAM	## _{trname}	Random access curve file name array	----	RACURV	
LGRARY	## _{gear}	Landing gear drag array	----	GEARDAT	
LGRFLAG	P _j _{gear}	Landing gear flag	----	FLAGS	
LIFTOFF	P _j _{liftoff}	Liftoff flag	----	FLAGS	

FORTRAN Name	Symbol	Description	Dimension	Usage	
				COMMON	SUBROUTINE NAME LIST
LINENUM		Current line number of current page of output	----	CTRL	
LOADING		Aircraft External Stores Loading Variable (user definable)	----	AIRCRAFT	DATA2
LUCURV	$\square_{curve}$	Logical unit for the random access curve file	----	CINDEX	
LUTN	$\square_{in}$	Logical unit for program input	----	CTRL	HALT
LUMSG	$\square_{msg}$	Logical unit for message output	----	CTRL	HALT
LUOUT	$\square_{out}$	Logical unit for program output	----	CTRL	ERROR, HALT
MANUVR		Maneuver input to main program	----	CHARV	PITCH
MANUVR		Maneuver passed to subroutine PITCH	----		PITCH
MAXSIZF		Maximum size of arrays FLPARY and VFLPARY	----	FLAPDAT	
MAXSIZG		Maximum size of array LGRARY	----	GEARDAT	
MAXSIZV		Maximum size of arrays XNUARY, HVCTARY, and VVCTARY	----	VECTDAT	
MSTRNDX	$\#_{master}$	Master index array for random access curve file lookups	----	CINDEX	
MVR		First three characters of MANUVR	----	CHARV	
NCOUNT	n_{count}	Integration loop counter		CTRL	
NDXLIN		Length of master index array, MSTRNDX	----	CINDEX	
NENG	n_{eng}	Total number of engines	----	ENGINE	
NEQ	n_{eq}	Number of equations of motion to be integrated	----	CTRL	INTX
NPAGE	n_{page}	Number of lines of output per page	----	CTRL	
ODIST	S_{obs}	Pre-Flare ground distance	feet		FLARE
OVERFLG	P_{over}	Flare height over obstacle clearance height flag	----	FLAGS	
PLA	$\angle_{pla}$	Average power lever angle	degrees		FXXENG
PLAARY	$\#_{pla}$	Power lever angle array	degrees		FXXENG
PRESALT	h_p	Pressure altitude	feet	FPINTEG	ATMOSPH

FORTRAN Name	Symbol	Description	Dimension	Usage	
				COMMON	SUBROUTINE NAME LIST
PRESS	p_a	Current ambient pressure	pound-force/feet ²	ATMOS	DATA2
PWRCODE		Aircraft power code	----	FXENG	
QS	q_s	Product of dynamic pressure and reference wing area			
RC		Engine thrust rating code	pound-force	AERO	SPEED
RCR		Runway condition reading	----	ENGINE	DATA2
REVFLAG	P_{rev}	Reverse thrust flag	----	RUNWAY	ROL
REVNDX		Reverse engine spoolup index (user definable)	----	FLAGS	DATA2
REVRSE	P_{revise}	Reverse thrust active flag	----	ENGINE	DATA2
RHO	ρ	Current density	----	FLAGS	DATA2
RKAIR	$\#R_{k,air}$	Runge-Kutta flightpath array	slugs/feet ³	ATMOS	
RKGRND	$\#R_{k,grnd}$	Runge-Kutta ground roll array	----	FPINTEG	
ROC	R/C	Current rate of climb	feet/second	INTEG	
ROCFPM	R/C	Current rate of climb	feet/minute	FPINT'7	
ROCTD	R/C _{TD}	Rate of climb at touchdown	feet/second	AIRBORN	ERROR
ROLLMAX	$(t_{roll})_{max}$	Takeoff Ground Roll Time Limit	seconds	FLARE	
ROLLMU	μ_{roll}	Rolling coefficient of friction	----		TKO2
ROTATE	P_{rotate}	Rotation flag	----	RUNWAY	DATA2
RTOFLAG	P_{rtt}	Refused takeoff flag	----	FLAGS	
RX		Conversion factor from radians to degrees	degree/radian	CONST	
SBKEND	δ_{sbk_end}	Final spoiler deflection	degrees	SPDBRAK	
SBKFLAG	$P_{speedbrake}$	Speed Brake flag	----	FLAGS	
SIGMA	σ	Current density ratio	----	ATMOS	
SINKTD		Sink rate at touchdown	feet/second	FLARENZ	LND
SPDBRK	δ_{sbk}	Speed Brake deflection	degrees	AERO	SPDBRAK
SPDBRK0	δ_{sbk0}	Initial Speed Brake deflection	degrees	INITIAL, SPDBRAK	DATA2
SPLFLAG	$P_{spoiler}$	Spoiler flag	----	FLAGS	LND2

FORTRAN Name	Symbol	Description	Dimension	Usage	
				COMMON	SUBROUTINE NAME LIST
SPLREND	$\delta_{\text{spl_end}}$	Final spoiler deflection	degrees		SPOIL
SPOILER	δ_{spl}	Spoiler deflection	degrees	AERO	SPOIL
SPOOL	δ_{spool}	Engine spooling status flag	----	FLAGS	SPOOLDNF, SPOOLDNR, SPOOLUP
STEADY	δ_{steady}	Steady state flag	----	FLAGS	
SWING	S_{wing}	Reference wing area	feet ²	AIRCRAFT	
T	TEMP	Runge-Kutta numerical integration array passed to subroutine INTX T is either the values of common block INTEG or FPINTEG.	----		INTX
TEMPR	T	Current temperature	degrees Rankine	ATMOS	
TERMFLG	δ_{term}	Program terminate flag	----	FLAGS	
TERMMMSG	δ_{term}	Termination message	----	HALT	
THETA	θ	Current temperature ratio	----	ATMOS	
THETAF	θ	Current pitch attitude	degrees	AIRBORN	
THRCRV		Thrust curve type (user defined)	----	CHARV	DATA2
THRUST	F_N	Net Thrust	pound-force	ENGINE	FLARENZ
THTCCLM	θ_{climb}	Pitch attitude tracked between liftoff and H_{climb}	degrees		TKO
THTFLY	θ_{fly}	Pitch attitude tracked above H_{climbout}	degrees		TKO2
THTMAX	θ_{max}	Current maximum pitch attitude limit	degrees	AIRCRAFT	
THTROT	θ_{rotate}	Pitch attitude tracked between rotation and liftoff	degrees		TKO
THTTOL	$\theta_{\text{tolerance}}$	Pitch attitude tolerance	degrees		TKO2
TIME	t	Elapsed time	seconds	CTRL	INTX, SPOOLDNF, SPOOLDNR, SPOOLUP
TIMEBRK	t_{brake}	Time delay between touchdown and brake application for a landing or the time delay between engine failure and brake application for a refused takeoff.	seconds	RUNWAY	ROL2

FORTRAN Name	Symbol	Description	Dimension	Usage		NAME LIST
				COMMON	SUBROUTINE	
TIMEFLD	t_{failed}	Elapsed time from engine failure	seconds	RUNWAY		ROL2
TIMEFLP	t_{flap}	Time delay between touchdown and flap retraction for a landing or the time delay between engine failure and flap retraction for a refused takeoff.	seconds	RUNWAY		ROL2
TIMEIDL	t_{idle}	Time delay between engine failure and engine spool down initiation for a refused takeoff.	seconds	RUNWAY		ROL2
TIMEMAX	t_{max}	Time limit for takeoff simulation	seconds			TKO2
TIMEREV	t_{reverse}	Time delay between touchdown and engine spool up initiation of thrust reversers for a landing or the time delay between engine failure and spool up initiation of thrust reversers for a refused takeoff.	seconds	RUNWAY		ROL2
TIMEROL	t_{roll}	Elapsed time of the ground roll	seconds	CTRL		
TIMESBK	$t_{\text{speedbrake}}$	Time delay between touchdown and speedbrake deployment for a landing or the time delay between engine failure and speedbrake deployment for a refused takeoff.	seconds	RUNWAY		ROL2
TIMESPL	t_{spoiler}	Time delay between touchdown and spoiler deployment for a landing or the time delay between engine failure and spoiler deployment for a refused takeoff.	seconds	RUNWAY		ROL2
TKOTYPE		Takeoff type (either STATIC or ROLLING)				TKO2
TOLRNCE	ϵ_i	Maximum tolerance constant for zero finding routines	----	CHARV		
TSLF	T_{sl}	Standard temperature at sea level	degrees Fahrenheit	CONST		
TWOOVR	2/7	Ratio of two over seven	----	CONST		
					ITRLND, ZEROX	

FORTRAN Name	Symbol	Description	Dimension	Usage		NAME LIST
				COMMON	SUBROUTINE	
VAR2	var ₂	Dummy variable passed to subroutine ZEROX as an argument to FUNCTN	----	ZEROX		
VAR3	var ₃	Dummy variable passed to subroutine ZEROX as an argument to FUNCTN	----	ZEROX		
VAR4	var ₄	Dummy variable passed to subroutine ZEROX as an argument to FUNCTN	----	ZEROX		
VCLMOUT	V _{climout}	Airspeed tracked between H _{climb} and H _{climout}				TKO2
VECTFLG	P _{vect}	Thrust vectoring flag	----	FLAGS		
VFFLAG	P _{off}	Engine failure velocity flag	----	FLAGS		
VFLPARY	# _{flap}	Flap deflection airspeed array	knots calibrated airspeed	FLAPDAT		TKOARY
VISCOSK	v	Current kinematic viscosity	feet ² /second	ATMOS		
VHAS	V _h	Current horizontal airspeed	feet/second	FPINTEG		
VKAPRK	V _{scr brk}	Aerobreaking termination airspeed	knots calibrated airspeed	AIRSPED		ROL2
VKAPP	V _{approach}	Approach airspeed for landing simulation			FLARENZ	LND
VKBRAKE	V _{brake}	Wheel braking airspeed	knots calibrated airspeed	AIRSPED		ROL2
VKCAS	V _c	Current calibrated airspeed	knots	AERO	SPEED	
VKEAS	V _e	Current equivalent airspeed	knots		SPEED	
VKEND	V _{end}	Simulation end airspeed	knots calibrated airspeed	AIRSPED		TKO2
VKFAIL	V _{fail}	Engine failure airspeed	knots calibrated airspeed	AIRSPED		TKO
VKFLAP	V _{flap}	Flap retraction airspeed	knots calibrated airspeed	AIRSPED	FRETRAC	TKO2
VKFLPMX	(V _{flap}) _{max}	Maximum flap retraction airspeed	knots calibrated airspeed	AIRSPED		
VKMCG	V _{mcg}	Minimum control airspeed on the ground				DATA2
VKROTAT	V _{rotate}	Rotation airspeed	knots calibrated airspeed	AIRSPED		TKO
VKSTART	V _{start}	Simulation start groundspeed	knots calibrated airspeed	AIRSPED		TKO2
VKTAS	V _t	Current true airspeed	knots	AERO	SPEED	

FORTRAN Name	Symbol	Description	Dimension	Usage	
				COMMON	SUBROUTINE NAME LIST
VKTGS	V_g	Current groundspeed	knots		FLARE, GENMU, SPEED
VKTGTD	$(V_g)_{TD}$	Groundspeed at touchdown	knots		
VKWIND	V_{wind}	Wind speed	knots		
VTANGLE	ν	Vectored thrust angle	degrees	AIRSPED	DATA
VTAS	V_t	Current true airspeed	feet/second	ENGINE	TVECTOR
VTASJ	V_{tj}	Previous true airspeed	feet/second	INTEG	DVTDH
VTASX	V_{tx}	Dummy argument true airspeed passed to subroutine SPEED	feet/second	INTEG	SPEED
VTGS	V_{tg}	Current true groundspeed	feet/second		
VVCTARY	$\vec{V}_{vect}$	Vectored thrust airspeed array	knots calibrated airspeed	VECTDAT	TKOARY
VVECT	V_{vect}	Current vectored thrust airspeed array element	knots calibrated airspeed		TVECTOR
VWIND	V_{wind}	Wind speed	feet/second	AIRSPED	INTG, SPEED
WFUEL	W_f	Fuel flow	pound-meter/hour	ENGINE	
WNGLOD	W/S_{wing}	Wing loading at brake release	pound-force/feet ²	AIRCRFT	LND2
WRITTR	P_{writer}	Write iterations flag for subroutine FLARE	----	FLAGS	
WTML	W_{lift}	Gross weight minus lift	pound-force		GENMU
XANS	x_{interp}	X-axis coordinate for interpolation to calculate INTERP function value	----		INTERP
XENG	X_{eng}	Current engine multiplicative factor	----		
XENGEND	X_{engend}	Engine multiplicative factor at the end of a throttle transient	----	ENGINE	DATA2 SPOOLDNF,SPOOLDNR
XENGFLD	X_{failed}	Failed engines multiplicative factor	----		
XENGOUT	X_{engout}	Engine Multiplicative Factor after engine failure	----	ENGINE	DATA2
XENGTRN	$X_{transient}$	Engine Multiplicative Factor of transient engines	----		SPOOLDNF, SPOOLDNR, SPOOLUP

FORTRAN Name	Symbol	Description	Dimension	Usage	
				COMMON	SUBROUTINE NAME LIST
XENG0	X_{eng0}	Engine multiplicative factor at the beginning of a throttle transient	----		SPOOLUP
XIDLE	X_{idle}	Idle Engine Multiplicative Factor for a single engine	----	ENGINE	
XLF	n_x	Current normal load factor	gravity	AIRBORN	
XLFI	n_y	Previous normal load factor from last integration	gravity	AIRBORN	
XLFLARE	$(n_x)_{max}$	Flare normal load factor limit	gravity	FLARENZ	
XLFLMAX	$(n_x)_{max}$	Maximum normal load factor limit	gravity	FLARENZ	
XMAIN	x_{main}	Distance from main gear to cg	feet	GENMU	
XMIL	X_{mil}	Military Thrust Engine Multiplicative Factor for a single engine	----	ENGINE	
XMU	μ	Current coefficient of friction	----	RUNWAY	
XNOSE	x_{nose}	Distance from nose gear to cg	feet		GENMU
XNOW	x	Current x-axis variable for interpolation boundary	----		INTERP
XNUARY	θ_{θ}	Vectored thrust angle array	degrees	VECTDAT	
XPAST	x_j	Previous x-axis variable for interpolation boundary	----		INTERP
YCG	y_{cg}	Height of cg above ground	feet		GENMU
YNOW	y	Current y-axis variable for interpolation boundary	----		INTERP
YPAST	y_j	Previous y-axis variable for interpolation boundary	----		INTERP
ZERO		Constant = 0.0	----	CONST	
ZFN	$\rightarrow$	User definable variable passed through common ENGINE	----	ENGINE	
	$\uparrow$	Page Eject Symbol in FORTRAN Source Listing*	----		

* The page eject symbol is an unprintable ASCII character of value 014 decimal. It appears as the symbol for the planet Venus in the actual source code within several format statements. This symbol is substituted so actual page ejections will not occur when the source code is printed.

This page intentionally left blank.



APPENDIX E: INDEX

—A—

ABARG, 39, D-1
 ACCEL, 17, 28, 41, D-1
 acceleration.
 due to gravity. See G.
 flightpath. See FPACCEL.
 ground roll. See ACCEL.
 AERO, 22, 23, 24, 26, 28, 29, 30, 34, D-2, D-3, D-4, D-7,
 D-10, D-11
 AFPS, 42, D-1
 AIRBORN, 25, 26, 27, 28, 31, 32, 42, 2, 4, 7
 AIRCRAFT, 25, 26, 27, 29, 31, 32, 33, 37, 1, 2, 4, 5, 6
 AIRSPED, 22, 23, 24, 28, 30, 38, D-10, D-11
 airspeed.
 aerobraking termination speed. See VKABRK.
 approach speed. See VKAPP.
 braking airspeed. See VKBRAKE.
 calibrated airspeed. See VKCAS.
 constant climbout. See VCLMOUT.
 engine failure airspeed. See VKFAIL.
 equivalent airspeed. See VKEAS.
 flap limit airspeed. See VKFLPMX.
 flap retraction airspeed. See VKFLAP.
 horizontal. See VHAS.
 rotation. See VKROTAT.
 starting. See VKSTART.
 true airspeed. See VKTAS, VTAS.
 flightpath true airspeed. See FPVTAS.
 simulation end (maximum). See VKEND.
 AIT, 22, 35, D-1
 ALPHA, 3, 4, 14, 17, 22, 23, 24, 26, 28, 29, 30, 44, D-1
 ALPHAJ, 37, D-1
 altitude (AGL). See HAGL.
 climbout. See HCLIMB, HCLMOUT.
 flare. See HFLARE.
 gear retraction. HGEAR.
 maximum. See HMAX.
 obstacle clearance. See HCLEAR.
 runway. See HRUNWAY.
 AMACH, 22, 23, 29, 35, D-1
 Angle of Attack.
 aircraft. See ALPHA.
 aerobraking. See AOAABRK.
 three-point attitude. See AOA3PT.
 derivative with respect to time. See DADT.
 AOA0FLG, 33, 44, D-2
 AOA3PT, 4, 7, 22, 33, D-2
 AOAABRK, 4, 5, 15, 16, 38, 39, D-2
 APPROCH, 22, 23, 27, 31, 35, C-1, D-1, D-5
 AR, 22, 33, D-1
 Aspect Ratio. See AR.
 ASL, 43, D-1
 ASLSQR5, 43, D-1
 ATMOS, 23, 24, 26, 28, 29, 30, 42, D-1, D-3, D-7, D-8,
 D-9, D-10
 ATMOSPH, 29, C-1, D-2, D-7

atmosphere.
 density. See RHO.
 pressure. See PRESS.
 temperature (R). See TEMPR.
 average deceleration during ground roll. See
 ABARG.

—B—

B. See wingspan.
 Bibliography, 53
 braking coefficient selector switch. See IMU.
 BRAKMU, 5, 15, 22, 39, 40, D-2
 BRKFCTR, 15, 18, 39, D-2
 BRKFLAG, 44, D-2

—C—

CD, 2, 8, 17, 22, 34, 49, A-1, A-3, D-2, D-3
 center of gravity (as a function of MAC). See CGPCT.
 CGPCT, 7, 22, 33, D-2
 CHARV, 22, 23, 24, 28, 46, D-3, D-4, D-7, D-9, D-10
 CINDEX, 22, 51, D-6, D-7
 CL, 2, 8, 17, 22, 23, 28, 34, A-1, D-2, D-3
 CLALPH, 22, 33, D-2
 climbout altitudes. See HCLIMB, HCLMOUT.
 CLRHGT, 44
 .CMN file extension, C-1
 coefficient of friction.
 aircraft. See XMU.
 braking. See BRAKMU.
 rolling. See ROLLMU.
 CONFIG, 8, 22, 33, D-2
 CONST, 23, 24, 26, 28, 29, 30, 43, D-2, D-5, D-8, D-10,
 D-12
 COSD, C-1
 CTRL, 22, 23, 24, 26, 28, 29, 30, 31, D-3, D-6, D-7, D-9
 curve file names array. See KURNAM.
 CURVELIB, C-1
 CX, 34, A-1, D-2
 CY, 34, A-1, D-2

—D—

DADH, 30, C-1, D-5
 DADT, 17, D-2
 DADTCMD, 3, 4, 5, 8, 27, 39, 53, 54, 2
 DATA, 7, 33, 34, 38, 39, 40, 42, 46, 47, D-2, D-3, D-4, D-5
 DATA2, 7, 33, 34, 35, 38, 39, 46, 47, 51, D-2, D-3, D-4,
 D-6, D-7, D-8, D-9, D-11, D-12
 DCDGE, 23, D-3
 DCDLGR, 24, 49, D-3
 DCDX, 8, 17, 18, 34, D-3
 DCLGE, 23, D-3
 DCLX, 8, 17, 18, 34, D-3
 dCOM macros, C-2

drag coefficient. See CD.
 ground effect increment. See DCDGE.
 landing gear increment. See DCDLGR.
 user increment. See DCDX.
 DDELTDH, 30, C-1, D-5
 DELTA, 42, D-3
 delta temperature from standard day. See DTEMPF.
 delta time for integration. See DTIME
 delta time for engine failure. See DTFAIL.
 density ratio. See SIGMA.
 DERIVAL, 20, 21, 28, 41, 44, A-1, C-1, D-1
 DERIVAT, 20, 21, 29, 41, 44, A-1, C-1, D-1
 DERIVGR, 20, 21, 29, 41, 44, A-1, C-1, D-1
 DFLAPDT, 13, 24, 48, D-3
 DGDTR, 26, 30, C-1, D-1, D-4, D-5
 DGDTR, 41, D-3
 DIST, 28, 41, D-3
 distance.
 air distance. See DIST, FPDIST.
 ground distance. See GDIST.
 DISTMAX, 3, 11, D-3
 DIVC, C-1
 DSIGDH, 30, C-1, D-5
 DTD, 17, 37, D-3
 DTDGEX, 24, 28, 34, D-3
 DTDTHX, 14, 22, 33, D-3
 DTEMPF, 7, 17, 18, 29, 42, D-3
 DTFAIL, 5, 6, 8, 35, 44, D-3
 DTGEAR, 24, 49, D-3
 DTIME, 3, 8, 17, 24, 28, 31, 32, D-3
 DTIMEJ, 31, D-3
 DVDT, 26, 30, C-1, D-1, D-4, D-5
 DVECTDT, 13, 24, 50, D-3
 DVTDH, 30, C-1, D-2, D-11
 DUZ2. See WPDIFF.

—E—

ENGGRP, 5, 6, 8, 17, 44, 46, D-3
 ENGINE, 22, 23, 24, 26, 28, 29, 35, D-1, D-3, D-4, D-7,
 D-9, D-11, D-12
 engine failure mode. See FAILMOD.
 engine failure speed. See VKFAIL.
 engine failure state. See FAILST.
 engine pressure ratio. See EPR.
 ENGNDX, 23, D-3
 EPR, 8, 9, 23, 35, 47, D-3
 ERRFLAG, 29, 44, D-4
 ERROR, 21, 29, 33, C-1, D-4, D-6, D-8

—F—

FAILFLG, 44, D-4
 FAILGRP, 5, 6, 8, 44, 46, D-4
 FAILMOD, 5, 6, 8, 46, D-4
 FAILST, 5, 6, 8, 10, 17, 35, 44, D-4
 failure mode. See FAILMOD.
 failure state. See FAILST

FE, 22, 23, 35, A-1, D-4
 FG, 22, 23, 35, A-1, D-4
 FGPCT, 9, 17, 35, D-4
 FINDV, 24, 26, 30, D-4
 FLAGS, 22, 23, 24, 26, 28, 29, 44, D-2, D-4, D-5, D-6,
 D-7, D-10, D-11
 FLAP, 3, 7, 9, 13, 17, 18, 21, 22, 24, 34, 44, 46, 47, D-3,
 D-4
 flap deflection percentage. See FLPPCT.
 flap limit airspeed. See VKFLPMX.
 flap retraction. See FRETRAC.
 FLAP0, 17, 18, 23, D-4
 FLAPDAT, 23, 24, 48, D-4, D-6, D-7, D-10
 FLAPFLG, 44, D-4
 FLARE, 4, 14, 17, 20, 24, 27, 28, 31, 39, A-2, C-1, D-1,
 D-5, D-7, D-8, D-11
 flare initiation height. See HFLARE.
 FLARENZ, 14, 20, 21, 24, 27, 45, A-3, C-1, D-1, D-5, D-
 D-6, D-8, D-9, D-9, D-11, D-12
 FLPARY, 13, 48, D-4, D-6, D-7
 FLPPCT, 7, 9, 22, 34, 44, D-4
 FLT, 9, 17, 18, 33, D-4
 FLTNDX, 9, D-4
 FORCEX, 7, 8, 9, 10, 13, 14, 20, 22, 24, 26, 28, 33, 34, 35,
 36, 47, 48, 49, 50, A-1, C-1, D-1, D-2
 flightpath
 acceleration. See ACCEL, FPACCEL.
 angle.
 in degrees. See GAMMA.
 in radians. See GAMMAR.
 derivative with respect to time. See DGDTR.
 distance. See FPDIST.
 FPACCEL, 3, 17, 41, D-4
 FPCTFLG, 44, D-4
 FPDIST, 41, D-4
 FPINTEG, 23, 24, 28, 29, 30, 41, D-3, D-4, D-5, D-7, D-8
 FPSKTS, 43, D-4
 FPVTAS, 26, 30, 41, D-4
 FRETRAC, 21, 22, C-1, D-4
 fuel flow. See WFUEL.
 FXXAERO, 12, 13, 20, 22, 24, 28, 34, 38, 48, 49, C-1
 FXKENG, 13, 20, 22, 23, 31, 35, 45, 50, C-1, D-6, D-7

—G—

G, 43, D-4
 GAMMA, 17, D-5
 GAMMAPP, 4, 14, 27, 28, 37, D-5
 GAMMAR, 26, 28, 29, 30, 41, D-5
 GAMMARW, 7, 39, D-5
 GDIST, 17, 28, D-5
 GEARDAT, 23, 24, 49, D-6, D-7
 GEFFECT, 22, 23, 33, C-1, D-2, D-3, D-5
 GEFLAG, 44, D-5
 GENMU, 15, 22, 23, 39, 40, C-1, D-11, D-12
 glideslope. See GAMMAPP.
 GRETRAC, 21, 24, 49, C-1, D-3, D-5
 gross thrust. See FG.
 user adjustment. See FGPCT.
 gross weight. See GWT, GWT0.

ground effect. See GEFFECT.
groundspeed. See VTGS. VKTGS.
 minimum control. See VKMCG
GRW, 39, D-5
GWT, 17, 28, 30, 33, 47, 54, D-5
GWT0, 7, 17, 23, 33, 46, D-5

—H—

HAGL, 17, 23, 24, 28, 39, 44, D-5
HALT, 29, C-1, D-5
headwind component. See VKWIND.
HCLEAR, 3, 4, 7, 17, 18, 20, 21, 24, 28, 39, 44, D-5
HCLIMB, 3, 11, 12, D-5, D-9, D-10
HCLMOUT, 3, 11, 12, D-5, D-9, D-10
HFLARE, 1, 4, 14, 17, 18, 20, 21, 24, 27, 28, 39, 44, 46,
 A-2, A-3, D-5
HGEAR, 11, 39, 49, D-6
HORP, C-1
HMAX, 3, 11, 47, D-6
HRUNWAY, 8, 28, 39, 41, D-6
HVCTARY, 13, 50, D-6, D-7
HZ, 22, 33, D-6

—I—

ICOUNT, 31, D-6
IDBUG, 9, 51, D-6
IDENT, C-1
IFLAP, 48, D-6
IGEAR, 49, D-6
IMU, 15, 39, 46, 47, D-6
INICURV, 2, 22, 51, C-1
INITIAL, 23, C-1, D-1, D-4, D-8
INTEG, 23, 24, 28, 41, C-1, D-1, D-3, D-8, D-11
integration step size. See DTIME.
INTERP, 30, C-1, D-12, D-13
INTG, 28, C-1, D-3, D-5, D-11
INTX, 20, 24, 28, C-1, D-1, D-3, D-7, D-8, D-9
INVERS, C-1
ITRLND, 26, 27, 29, C-1, D-3, D-4, D-6, D-10
IVECT, 50, D-6

—J—

JDEBUG, 9, 31, D-6

—K—

KENG, 31, D-6
KURNAM, 51, D-6

—L—

landing gear retraction. See GRETRAC.
 altitude. See HGEAR.
landing gear drag. See DCDLGR.
LANDNG, 19, 20, 24, 37, C-1, D-1
LBLSORT, C-1

LGRARY, 49, D-6, D-7
LGRFLAG, 24, 44, D-6
LIBARRAY, C-1
LIBAERO, C-1
LIBGRND, C-1
LIBGENU, C-1
LIBLND, C-1
LIBMAIN, C-1
LIBTKO, C-1
LIBTRIG, C-1
LIBUTIL, C-1
LIFTOFF, 3, 44, D-6
lift coefficient. See CL.
 ground effect increment. See DCLGE.
 user increment. See DCLX.
LINENUM, 31, D-6
LND, 7, 14, 37, 38, 39, 46, D-1, D-5, D-8, D-11
LND2, 7, 14, 33, 45, 46, D-3, D-8, D-12
LOADING, 9, 22, 33, D-6
logical units for input and output.
 curve file. See LUCURV.
 input. See LUIN.
 messages. See LUMSG.
 output. See LUOUT.
LUCURV, 51, D-6
LUIN, 29, 31, D-6
LUMSG, 9, 23, 29, 31, D-6
LUOUT, 9, 24, 29, 31, D-7

—M—

MACH (subroutine), C-1
Mach number. See AMACH.
manuever. See MANUVR, MVR.
MANUVR, 24, 46, 47, D-7
maximum airspeed. See VKEND.
maximum altitude. See HMAX.
maximum distance. See DISTMAX.
MAXSIZE, 48, D-7
MAXSIZG, 49, D-7
MAXSIZV, 50, D-7
minimum control groundspeed. See VKMCG.
minimum interval takeoffs, 1, 12
MVR, 47, D-7

—N—

namelists
 DATA, 7
 DATA2, 7
 LND, 7
 LND2, 7
 ROL, 7
 ROL2, 7
 TKO, 7
 TKO2, 7
 TKOARY, 7
NASA TOLAND, 1
NCOUNT, 31, D-7

NENG, 22, 35, D-7
NEQ, 28, 32, D-7
net thrust. See FN.
NORM, C-1
NPAGE, 31, 32, D-7

—O—

obstacle clearance height. See HCLEAR.
OVERFLG, 44, D-7

—P—

PANDFQ.LIB, C-1
PITCH, 2, 24, C-1, D-1, D-2, D-3, D-7
pitch attitude. See THETAF.
 maximum. See THTMAX.
 rotation. See THTROT.
 tolerance. See THTTOL
PRESALT, 29, 41, D-7
PRESS, 42, D-7
pressure altitude. See PRESALT.
pressure ratio. See DELTA.
propulsive drag. See FE.
PWRCODE, 5, 6, 9, 22, 23, 35, D-7

—Q—

QORV, C-1
QS, 23, 29, 34, D-7

—R—

RACURV, 22, 23, 30, 51, D-6
rate of climb. See ROC, ROCFPM.
rating code. See RC.
RC, 9, 22, 47, D-7
RCR, 15, 39, 46, 47, D-7
RDBFLE, C-1
REVFLAG, 9, 44, D-7
REVNDX, 35, D-7
REVRSE, 44, D-8
RHO, 42, D-8
RKAIR, 41, D-8
RKGRND, 41, D-8
ROC, 17, 41, D-8
ROCFPM, 29, 37, D-8
ROL, 7, 15, 39, 40, 46, 47, D-2, D-6, D-7
ROL2, 7, 15, 38, 39, 40, 46, 47, D-2, D-9, D-10, D-11
ROLL, 20, 24, 28, 31, 44, 52, C-1, D-1, D-5, D-8
ROLLMAX, 3, 11, D-8
ROLLMU, 5, 9, 39, D-8
ROTATE, 44, D-8
rotation airspeed. See VKROTAT.
RTOFLAG, 44, D-8
RUNWAY, 23, 24, 28, 39, D-1, D-2, D-5, D-6, D-7, D-8,
 D-10, D-12
runway condition reading. See RCR.
runway pressure altitude. See HRUNWAY.

runway slope. See GAMMARW.
RX, 43, D-8

—S—

SBKFLAG, 45, D-8
SIGMA, 42, D-8
SINKTD, 4, 14, 27, 28, 46, D-8
SIND, C-1
SPDBRAK, 24, C-1, D-1, D-3, D-8
SPDBRK, 24, 34, D-8
SPDBRK0, 9, 23, D-8
SPEED, 29, C-1, D-1, D-5, D-7, D-11
speed of sound.
 sea level. See ASL.
 units in feet/sec. See AFPS.
speedbrakes. See SPDBRAK., SPDBRK.
SPLFLAG, 14, 45, D-8
SPOIL, 21, 24, C-1, D-1, D-3, D-8
SPOILER, 1, 4, 5, 14, 16, 21, 22, 24, 34, 40, 45, D-1, D-3,
 D-8, D-10
spoilers. See SPOIL, SPOILER.
SPOOL, 5, 6, 8, 15, 23, 35, 38, 45, 46, D-4, D-8, D-9, D-
 10
SPOOLDNF, 5, 6, 21, 22, 23, C-1, D-8, D-9, D-12
SPOOLDNR, 5, 6, 21, 22, 23, C-1, D-8, D-9, D-12
SPOOLUP, 9, 21, 22, 23, 35, C-1, D-4, D-7, D-8, D-9, D-12
SRHOS, C-1
STASK, C-1
STEADY, 45
STEDYST, 4, 20, 24, 25, 26, C-1, D-1, D-4, D-5
SWING, 22, 33, 34, A-1, A-3, D-8

—T—

TABINT, 23, C-1
TABLES, 22, 23, 51
TAKOFF, 19, 20, 24, 33, 40, 44, 48, 49, 50, C-1, D-1
temperature ratio. See THETA.
TAND, C-1
TEMPR, 42, D-9
TERMFLG, 45, D-9
TERMMSG, 29, D-9
THETA, 42, D-9
THETAF, 3, 17, 37, D-9
THRCRV, 9, 22, 47, D-
THRUST, 8, 17, 23, 26, 28, 35, D-9
 gross thrust. See FG.
 net thrust. See FN.
 propulsive drag. See FE.
thrust incidence angle. See AIT.
thrust reversing. See REVFLAG, REVNDX, REVRSE,
 and TIMEREV.
thrust vectoring, 2, 7, 21, 45, 50, D-10. See also
 TVECTOR.
THTCLM, 3, 11, 47, D-9
THTFLY, 3, 11, D-9
THTMAX, 3, 22, 33, 37, D-9
THTROT, 3, 11, 47, D-9

THITOL, 3, 11, D-9
 TIME, 17, 23, 28, 32, D-9
 TIMEBRK, 4, 5, 15, 39, 40, 44, D-9
 TIMEFLD, 40, D-9
 TIMEFLP, 4, 5, 15, 40, D-9
 TIMEIDL, 5, 15, D-9
 TIMEMAX, 3, 11, D-9
 TIMEREV, 4, 5, 15, D-10
 TIMEROL, 32, 44, D-10
 TIMESBK, 4, 5, 15, 16, 40, D-10
 TIMESPL, 4, 5, 16, 40, D-10
 TKO, 7, 11, 38, 39, 46, 47, D-6, D-9, D-11
 TKO2, 7, 11, 38, 47, D-3, D-5, D-8, D-9, D-11
 TKOARY, 7, 13, 47, 48, 50, D-4, D-6, D-10, D-11
 TKOTYPE, 12, 47, D-10
 TOLANDLIB, C-1
 touchdown.
 altitude. See HRUNWAY.
 sink rate. See SINKTD.
 TSLF, 43, D-10
 TVECTOR, 21, 24, C-1, D-3, D-6, D-11
 TWOOVR7, 43, D-10

—V—

VALUES, 22, 23, 51
 VCLMOUT, 3, 11, 12, D-10
 VCMACH, C-1
 VECTDAT, 23, 24, 50, D-6, D-7, D-11, D-12
 VECTFLG, 45, D-10
 VECTOR, 23
 VFFLAG, 44, 45, D-10
 VFLPARY, 13, 48, D-6, D-7, D-10
 VHAS, 41, D-10
 viscosity (kinematic). See VISCOSK.
 VISCOSK, 42, D-11
 VKABRK, 4, 5, 15, 16, 38, 39, D-11
 VKAPP, 4, 14, 27, 38, 46, D-11
 VKBRAKE, 4, 5, 16, 38, 44, 46, 47, D-11
 VKCAS, 17, 22, 23, 29, 34, 44, D-11
 VKEAS, 29, D-11
 VKEND, 3, 12, 38, D-11
 VKFAIL, 5, 6, 9, 11, 38, 40, 44, 45, 47, D-11
 VKFLAP, 12, 13, 24, 38, 48, D-11
 VKFLPMX, 12, 13, 38, 48, D-11
 VKMCG, 5, 6, 9, 17, 38, D-11

VKROTAT, 3, 11, 38, 47, D-11
 VKSTART, 12, 38, 47, D-11
 VKTAS, 17, 29, 34, D-11
 VKTGS, 17, 23, 28, 29, D-11
 VKWIND, 8, 38, D-11
 VTANGLE, 13, 24, 35, D-11
 VTAS, 30, 41, D-11
 VTASJ, 41, D-11
 VVCTARY, 13, 50, D-6, D-7, D-11
 VWIND, 28, 29, 38, D-11

—W—

WFUEL, 23, 28, 35, D-11
 wind speed. See VKWIND.
 wingspan, 28, 33
 WNGLOD, 33, D-12
 WPSDIFF, C-1
 WRITTR, 14, 45, D-12

—X—

XENG, 5, 6, 10, 17, 23, 26, 35, 44, D-12
 XENGFLD, 8, 10, 35, D-12
 XENGOUT, 22, 35, 44, D-12
 XIDLE, 8, 22, 36, D-12
 XLF, 17, 24, 37, D-12
 XLFJ, 37, D-12
 XLFMAX, 22, 27, 33, D-12
 XMAIN, 23, D-12
 XMIL, 8, 36, D-12
 XMU, 5, 17, 40, D-12
 XNOSE, 23, D-12
 XNUARY, 13, 50, D-6, D-7, D-12

—Y—

YCG, 23, D-12

—Z—

ZERO, 43, D-13
 ZEROX, 26, 30, C-1, D-1, D-2, D-4, D-5, D-10
 ZFN, 36, D-13

