NASA CR-166,069

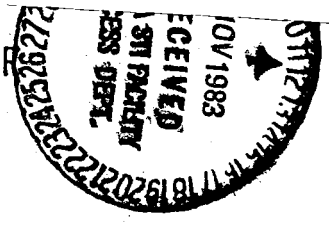
NASA CONTRACTOR REPORT 166069

(NASA-CR-166069) IMPROVEMENTS TO THE FATOLA COMPUTER PROGRAM INCLUDING ADDED ACTIVITY CONTROLLED LANDING GEAR SUBROUTINES Final Report An Early Domestic Dissemination Report (Computer Sciences Corp., Hampton,

X83-10332

Unclas





# IMPROVEMENTS TO THE FATOLA COMPUTER PROGRAM INCLUDING ADDED ACTIVELY CONTROLLED LANDING GEAR SUBROUTINES



GERALD H. MALL

COMPUTER SCIENCES CORPORATION
Hampton, VA 23666

Contract NAS1-16078
April 1983



National Aeronautics and Space Administration

**Langley Research Center**
Hampton, Virginia 23665

Table of Contents

This Page Intentionally Left Blank

# List of Figures

This Page Intentionally Left Blank

# List of Tables

# INTRODUCTION

Program TOLA (Take-Off and Landing Analysis) provides a non real time simulation of the dynamics of conventional aircraft during takeoffs and landings. The program models the performance of an aircraft during a takeoff roll or during the glide slope, flare, impact, and rollout of a landing. It includes the effects of a number of external and internal conditions such as wind shears, rough runway, engine failure, ground effect, etc. Extensive documentation of the TOLA program--its capabilities, problem formulation, and user and programmer guides--have been written (Lynch 1972; Lynch and Dueweke 1974 a & b; Young and Dueweke 1975).

TOLA has been modified to include a flexible airframe option (Dick and Benda 1975) and is identified as program FATOLA at NASA Langley. Following validation of the flexible airframe analysis and some additional modifications to the program to improve its capabilities (Carden and McGehee 1977), a provision for actively controlled landing gear has been incorporated. The active control code simulates dynamic load control during impact and rollout, and during takeoff roll on rough runways. Additionally, a program restart capability has been added as well as other program enhancements.

This report includes a brief description of the added capabilities, a detailed description of specific program changes, and includes information required for a user to exercise the new options. A complete listing of the modifications to the FATOLA program is included as an appendix.

# 1. ADDED PROGRAM CAPABILITIES

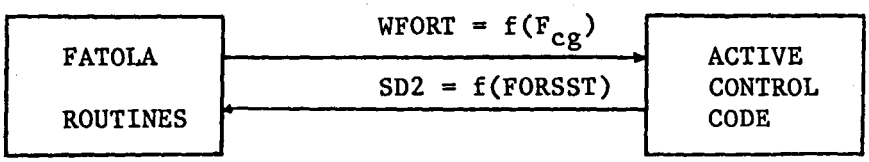
The FATOLA program has been modified to provide for actively controlled landing gears, metering pin area as a function of strut stroke, and continuously varying aerodynamic coefficients $C_{A_0}$ and $C_{N_0}$. A restart capability has been added to the program.

1.1 Active Control Landing Gear - The active gear limits the force applied at the gear-airframe interface by limiting the shock strut force (FORSST) with a closed loop series-hydraulic control. An impact limit force (WLFOR) is determined from the value of the gear-airframe interface force (WFORT) very shortly after aircraft touchdown. This determination is made when the work potential of the strut (energy dissipation potential) exceeds the aircraft energy at touchdown apportioned among the main gears. The gear-airframe interface force is compared with the impact limit force, the shock strut force then being adjusted to bring the gear-airframe interface force within specified limits about the control impact limit force.

After the initial impact energy has been dissipated and the aircraft is in the rollout phase of the landing simulation, the control limit force is reduced to a value where the shock strut force is controlled to support the aircraft weight. The transition from impact limit force (WLFOR) to rollout limit force (WLFORR) is

carried out smoothly by the use of a ramp function from impact to rollout. Active control performs similarly during a takeoff roll by using the rollout limit force value to control the interface force experienced when accelerating on a rough runway. A detailed description of the active control model is presented in a technical note by McGehee and Carden (1976).

1.2 Implementation of the Active Control Code - In FATOLA, the landing gear dynamics are modeled in subroutines LGEAR1 and LGEA3C. The active control subroutines replace the passive shock strut calculations in LGEAR1 (see Figs. 1 & 2). However, passive gear calculations can be made using the active gear subroutines. The control code is integrated into FATOLA by defining the gear-airframe interface force in terms of the total force applied to the aircraft center of gravity, which is computed in FATOLA. The active code returns a value for the shock strut force which is a term used in the FATOLA calculation of the shock strut acceleration:



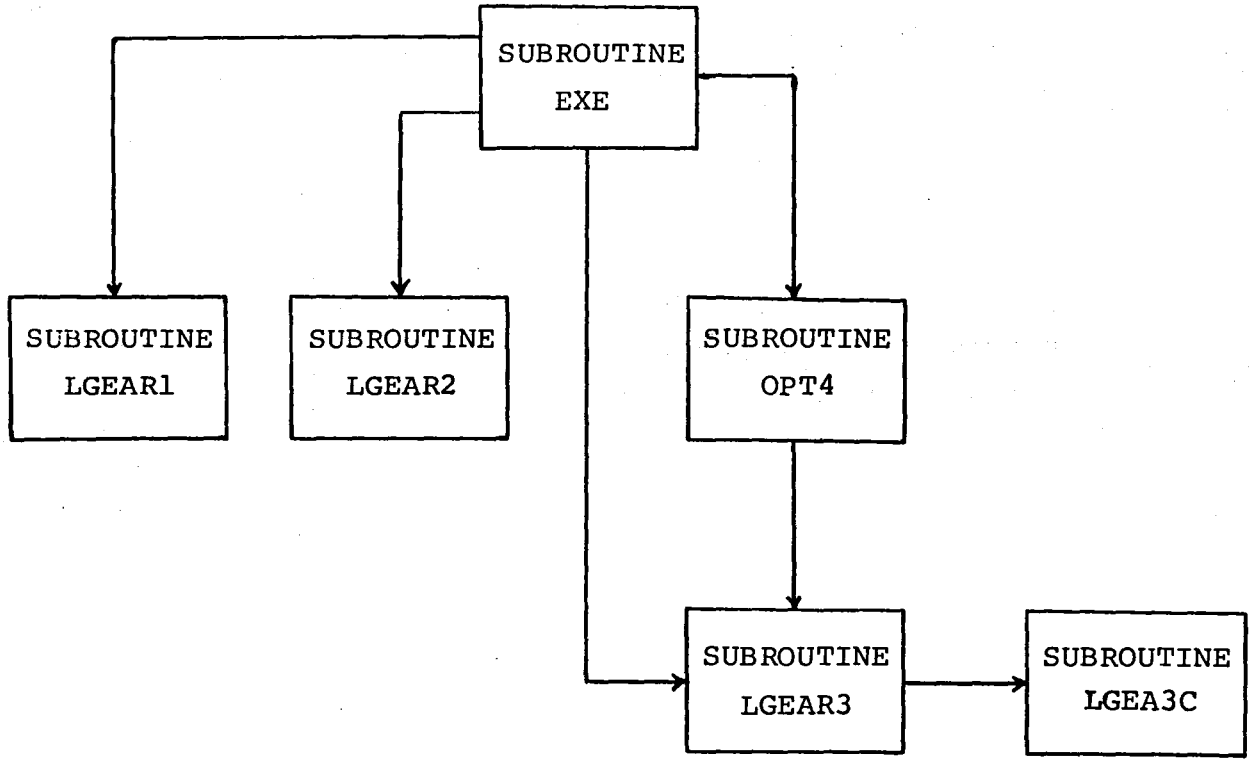



FIGURE 1: SUBROUTINE LINKAGE FOR CALCULATION OF PASSIVE LANDING GEAR DYNAMICS (ORIGINAL FATOLA ROUTINES)

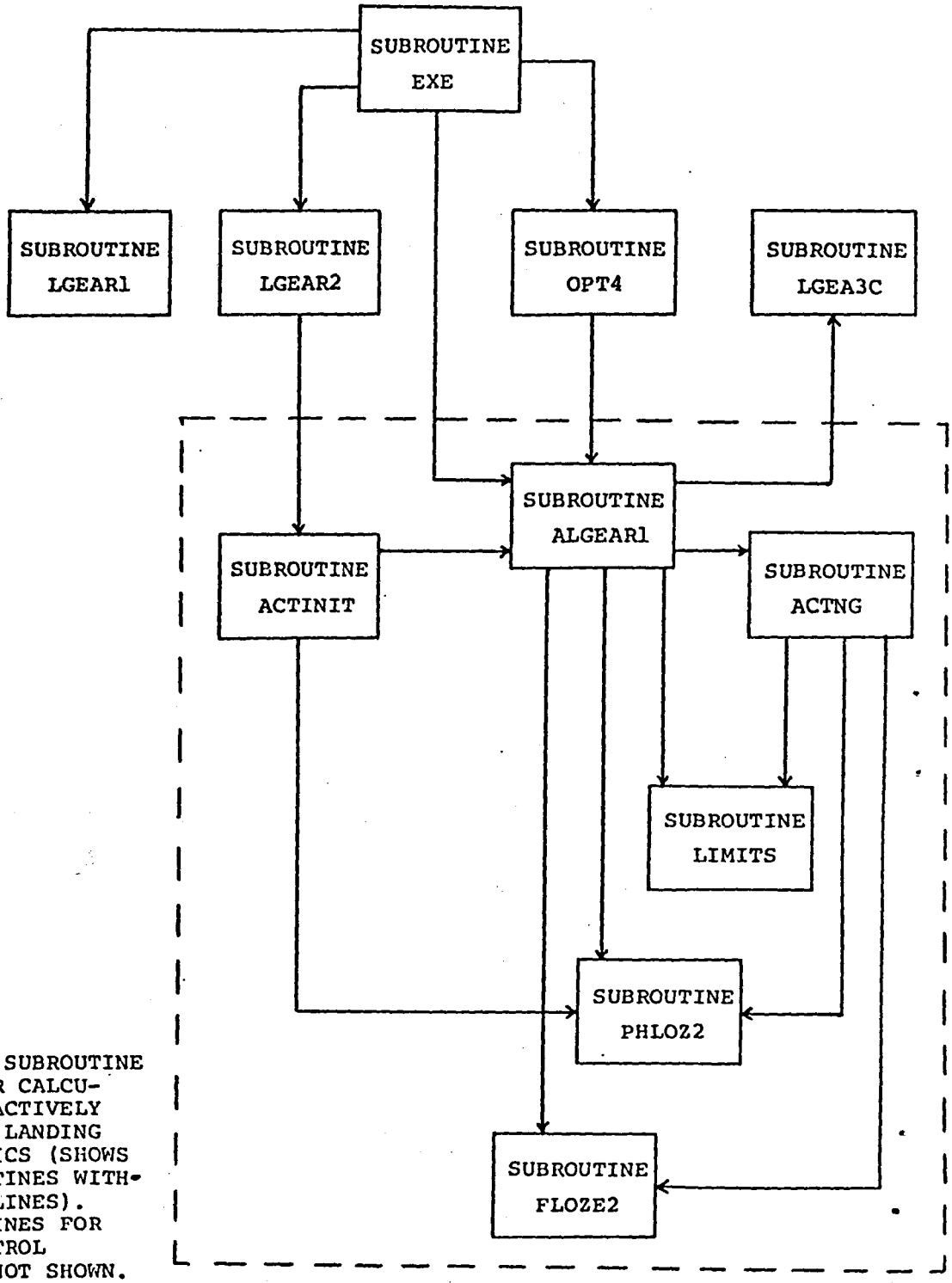



FIGURE 2: SUBROUTINE LINKAGE FOR CALCULATION OF ACTIVELY CONTROLLED LANDING GEAR DYNAMICS (SHOWS NEW SUBROUTINES WITHIN DASHED LINES). INPUT ROUTINES FOR ACTIVE CONTROL VARIABLES NOT SHOWN.

The active control feature has been incorporated in FATOLA with eight new subroutines:

DIRACT--block data input variables

ACTIN--reads active control input variables

ACTINIT--initializes the active control variables

ALGEAR1--performs the main active control calculations

ACTNG--performs the active nose gear calculations

PHLOZ2--calculates initial servovalve hydraulic variables

FLOZE2--calculates servovalve hydraulic variables

LIMITS--sets limits on servovalve power spool motion and displacements

The active control variables have all been blocked into labeled common, including the additional input variables. This common block is named ACTIVE and has been added to FATOLA subroutines EXE, OPT1, LGEA3C, and SDFLGP, as well as the active subroutines ACTINIT, ACTIN, and ALGEAR1. See Tables 1 and 2 for lists of the added input variables and other key program variables.

The active control mode is enabled by setting the landing gear type indicator switch (INDLG) to -3 in the input data set. Tests are performed on this indicator whenever gear calculations are needed, leading to calls to the active subroutines rather than to the LGEAR3 entry point. To perform passive gear calculations, using the active gear subroutines, the switch IMODE(I) must be set to zero

for each gear. Set IMODE(I) to one for each gear for active gear calculations. These tests take place in subroutines EXE and OPT1. Subroutine SDFLGP has been extensively modified to output values of selected active control variables to a listing and to a plot data file.

TABLE 1
INPUT VARIABLES

| VARIABLE NAME | TYPE | UNITS | DESCRIPTION |
|---|---|---|---|
| AMUH | | LBF-SEC/FT$^2$ | Dynamic viscosity of hydraulic fluid |
| APINT | ARRAY | FT$^2$ | Area of metering pin as function of strut stroke |
| AREA1 | ARRAY | FT$^2$ | Area of strut hydraulic chamber (piston) |
| AREA2 | ARRAY | FT$^2$ | Area of strut pneumatic chamber (cylinder) |
| AREA3 | ARRAY | FT$^2$ | Area of chamber between piston and cylinder |
| AREMO | ARRAY | FT$^2$ | Area of strut main orifice |
| AREO3 | ARRAY | FT$^2$ | Orifice area of chamber between piston and cylinder |
| BETA | | LBF/IN$^2$ | Bulk modulus of hydraulic fluid |
| BLMU | ARRAY | -- | Coefficient of friction for strut lower bearing |

| | | | |
|---|---|---|---|
| BUMU | ARRAY | -- | Coefficient of friction for strut upper bearing |
| CDMOC | ARRAY | -- | Strut main orifice compression discharge coefficient |
| CDMOE | ARRAY | -- | Strut main orifice extension discharge coefficient |
| CDSV | ARRAY | -- | Servovalve orifice discharge coefficient |
| CD3 | ARRAY | -- | Discharge coefficient for chamber between piston and cylinder |
| CFFOR | ARRAY | LBF | Coulomb friction force between piston and cylinder bearings |
| DIOTA | | DEGREES | Angle between body x-axis and wing main chord |
| DSV | ARRAY | INCHES | Diameter of servovalve spool |
| EPSILO | ARRAY | LBF | Tolerance about impact control limit force |
| EPSROL | ARRAY | LBF | Tolerance about rollout control limit force |
| EPSSLP | | LBF/SEC | Time rate of change of tolerance about transition control limit force |
| ETASV | | 1/SEC | Damping coefficient in servovalve transfer function |
| FWORK | ARRAY | -- | Coefficient for selecting value of impact limit force |

| | | | | |
|---|---|---|---|---|
| GAMAH | ARRAY | $LBF/FT^3$ | | Specific weight of hydraulic fluid |
| GNR | | — | | Constant — set to 0.0 |
| IMODE | INTEGER ARRAY | — | 0=Off<br>1=On | Switch to select active control for each gear |
| IRST | | — | 0=None<br>1=Write<br>2=Read<br>3=Read/ Write | Restart indicator |
| KAPT | INTEGER ARRAY | — | 0=No Pin<br>1=Constant Area<br>2=Variable Area | Indicator for metering pin |
| LTAB10 | REAL ARRAY | — | | $C_{N_0}$ limiting value |
| LTAB80 | REAL ARRAY | — | | $C_{A_0}$ limiting value |
| OMRUN | | DEGREES | | Runway slope |
| PATM | | $LBF/FT^2$ | | Atmospheric pressure |
| PERCNT | ARRAY | — | | Constant |
| PGAHAC | ARRAY | $LBF/FT^2$ | | Pressure in high pressure control reservoir |
| PGALAC | ARRAY | $LBF/FT^2$ | | Pressure in low pressure control accumulator |
| PGA1I | ARRAY | $LBF/FT^2$ | | Strut hydraulic charging pressure |
| PGA2I | ARRAY | $LBF/FT^2$ | | Strut pneumatic charging pressure |
| PGA3I | ARRAY | $LBF/FT^2$ | | Hydraulic pressure in chamber between strut piston and cylinder |

| | | | |
|---|---|---|---|
| PINM | ARRAY | $FT^2$ | Main gear metering pin area table |
| PINN | ARRAY | $FT^2$ | Nose gear metering pin area table |
| QPUMPS | ARRAY | $FT^3/SEC$ | Maximum hydraulic pump flow rate |
| RCLSV | ARRAY | INCHES | Radial clearance between spool and sleeve of servovalve |
| RHOH | | $SLUGS/FT^3$ | Mass density of hydraulic fluid |
| RTAB10 | ARRAY | | $C_{N_0}$ rate of change |
| RTAB80 | ARRAY | | $C_{A_0}$ rate of change |
| STROM | ARRAY | FT | Main gear stroke table for pin area |
| STRON | ARRAY | FT | Nose gear stroke table for pin area |
| TAUF | | SEC | Time constant in strut position feedback loop |
| TC1 | | SEC | Time constant of electronic compensation network |
| TC2 | | SEC | Same definition as TC1 |
| TC3 | | SEC | " " " " |
| TC4 | | SEC | " " " " |
| VOLACI | ARRAY | $FT^3$ | Total volume of high pressure accumulator |
| VOLANI | ARRAY | $FT^3$ | Initial volume of charging nitrogen of high pressure accumulator |

| | | | |
|---|---|---|---|
| VOL1I | ARRAY | $FT^3$ | Initial volume of hydraulic fluid in shock strut piston |
| VOL2I | ARRAY | $FT^3$ | Pneumatic volume of charged strut |
| VOL3I | ARRAY | $FT^3$ | Volume between strut piston and cylinder |
| WC | | $SEC^{-1}$ | Corner frequency in active control servovalve transfer function |
| WC1 | | $SEC^{-1}$ | Natural frequency in electronic compensation network |
| WLFOR | ARRAY | LBF | Control limit force for impact |
| WLFORR | | LBF | Control limit force for rollout |
| WSV | | $SEC^{-1}$ | Natural frequency in active control servovalve transfer function |
| WSV1 | | IN | Window width of servovalve orifice for high pressure |
| WSV3 | | IN | Window width of servovalve orifice for low pressure |
| XBIAS | ARRAY | IN | Servovalve spool displacement for controlling strut charging pressure |
| XDDMAX | ARRAY | $IN/SEC^2$ | Maximum positive acceleration of servovalve spool |
| XDDMIN | ARRAY | $IN/SEC^2$ | Maximum negative acceleration of servovalve spool |

| | | | | |
|---|---|---|---|---|
| XKA | ARRAY | AMPS/VOLT | | Amplifier gain in active control loop |
| XKF | ARRAY | VOLT/IN | | Position feedback gain in strut position control loop |
| XKSV | ARRAY | IN/AMP | | Position gain of servovalve in active control loop |
| XLPSV1 | ARRAY | IN | | Overlap or underlap between spool and sleeve at null for flow Q1 |
| XLPSV3 | ARRAY | IN | | Overlap or underlap between spool and sleeve at null for flow Q3 |
| XSCOM | ARRAY | IN | | Commanded (static) position of shock strut |
| XSTHR | | FT | Elec cont. function not used in program | Threshold strut stroke for determining takeoff or landing mode |
| XSVDMN | ARRAY | IN/SEC | | Maximum negative velocity of servovalve spool. |
| XSVDMX | ARRAY | IN/SEC | | Maximum positive velocity of servovalve spool |
| XSVMAX | ARRAY | IN | | Maximum positive displacement of servovalve spool |
| XSVMIN | ARRAY | IN | | Maximum negative displacement of servovalve spool |
| ZETAC1 | | | | Damping coefficient in electronic compensation network |
| ZETAC2 | | | | Damping coefficient in electronic compensation network |

TABLE 2
OTHER VARIABLES

| VARIABLE NAME | TYPE | UNITS | INITIAL VALUE | DESCRIPTION |
|---|---|---|---|---|
| ACON | | $SEC^3$ | | $1./W_{sv}**2*W_c$ |
| AIC | ARRAY | | 0.0 | Switch for regulating flow between strut piston and cylinder |
| AP2TO | ARRAY | $LBF/FT^2$ | | Absolute value of pneumatic charging pressure in strut |
| BCON | | $SEC^2$ | | $1./W_{sv}**2+2.*\eta_{sv}/(W_{sv}*W_c)$ |
| BLFORT | ARRAY | LBF | | Normal force between strut piston and cylinder at lower bearing |
| BUFORT | ARRAY | LBF | | Normal force between strut piston and cylinder at upper bearing |
| CCON | | SEC | | $2.*\eta_{sv}/W_{sv}+1./W_c$ |
| CMASNG | | SLUGS | | Effective mass at nose gear root |
| COEFO | ARRAY | $FT^2/LBF-SEC^2$ | | Strut main orifice flow coefficient modified for $d\gamma_h/d\rho$ |
| COEF1 | ARRAY | $IN^2/SEC\sqrt{LBF}$ | | Coefficient of flow from high-pressure accumulator to strut piston |
| COEF3 | ARRAY | $IN^2/SEC\sqrt{LBF}$ | | Coefficient of flow from strut piston to low-pressure reservoir |
| COPA | | -- | | $\cos(\phi+\alpha)$ |

| | | | | |
|---|---|---|---|---|
| CSV1 | ARRAY | $FT^3/SEC\sqrt{LBF}$ | | Servovalve orifice coefficient from accumulator to strut piston |
| CSV3 | ARRAY | $FT^3/SEC\sqrt{LBF}$ | | Servovalve orifice coefficient from strut piston to reservoir |
| DCON | | | 1.0 | |
| DELTX | ARRAY | INCHES | 0.0 | Difference between strut stroke and command stroke |
| DELTX1 | ARRAY | VOLTS | | Product of DELTX and position feedback gain in piston control loop |
| DF | ARRAY | LBF | 0.0 | Variation of gear-airframe interface force about control force |
| DLTX1D | ARRAY | VOLTS/SEC | 1.0 | Time rate of change of DELTX1 |
| DMD1 | ARRAY | | | Restart data block |
| DMTANH | ARRAY | — | | Function of hyberbolic tangent |
| DP1 | ARRAY | $LBF\text{-}SEC/FT^2$ | 0.0 | Time rate of change of hydraulic pressure in strut piston |
| DSTOP | | FT | .004 | Stopping distance for strut extension stop |
| DWFORT | | LBF | | Earth axes force at aircraft center of gravity |
| ENCG | | FT-LBF | | Kinetic energy at aircraft center of gravity |
| ENUP | ARRAY | FT-LBF | 0.0 | Kinetic energy of aircraft at landing gear root |
| FFORT | ARRAY | LBF | 0.0 | Strut axial friction force |

| | | | | |
|---|---|---|---|---|
| FONHST | ARRAY | LBF | 0.0 | Ground force normal to strut axis |
| FORCHT | ARRAY | LBF | 0.0 | Charging force in fully extended strut |
| FORSST | ARRAY | LBF | 0.0 | Strut force along strut axis |
| FSTOP | ARRAY | LBF | 0.0 | Strut extension stop force |
| FSTOPK | | LBF/FT | 0.0 | Strut extension stop spring rate |
| HMM | ARRAY | -- | 0.0 | Active control operation indicator 0=off, 1=on |
| IA1 | ARRAY | -- | 0 | Indicator for defining the return of notch-network compensation current to zero when gear is fully-extended during rebound; =0, non-zero current; =1, zero current |
| IA2 | ARRAY | -- | 0 | Same as IA1 |
| IA3 | ARRAY | -- | 0 | Same as IA1 |
| IA4 | ARRAY | -- | 0 | Same as IA1 |
| IA6 | ARRAY | -- | 0 | Indicator for defining the return of first lead-lag network compensation current to zero when gear is fully-extended during rebound: =0, non-zero current; =1, zero current |
| IA7 | ARRAY | -- | 0 | Same as IA6 |

| | | | | |
|---|---|---|---|---|
| IA9 | ARRAY | — | 0 | Indicator for defining the return of second lead-lag network compensation current to zero when gear is fully-extended during rebound:<br>=0, non-zero current;<br>=1, zero current |
| IA10 | ARRAY | — | 0 | Same as IA9 |
| ICOSV | INTEGER ARRAY | — | 0 | Servovalve control indicator for return to null position |
| ICU | ARRAY | — | 0 | Indicator for defining when the fluid volume in the strut piston has returned to the fully-extended strut value during rebound:<br>=0, has not returned;<br>=1, returned |
| IFR | INTEGER ARRAY | — | 0 | Indicator for selection of tanh function |
| IFSTOP | INTEGER ARRAY | — | 0 | Indicator for strut extension stopping force |
| IGE | ARRAY | — | 0 | Indicator for defining fully-extended gear during rebound:<br>=0, gear compressed;<br>=1, gear fully-extended |
| IGO | ARRAY | — | 0 | Indicator for defining accelerated return of servo-controller parameters to fully-extended gear values:<br>=0, inactive;<br>=1, active |
| IIXSVH | INTEGER ARRAY | — | 0 | Indicator for servovalve returning to null from high pressure |

| | | | | |
|---|---|---|---|---|
| IIXSVL | INTEGER ARRAY | -- | 0 | Indicator for servovalve returning to null from low pressure |
| ITRIP | ARRAY | -- | 0 | Indicator for defining the return of gear and servo-valve parameters to fully-extended gear values: =0, inactive; =1, active |
| INDEACT | INTEGER ARRAY | -- | 0 | Indicator for control regime 0=off, 1=impact, 2=rollout |
| INITSW | INTEGER | -- | 1 | Switch indicating first pass through program 1=first pass |
| IOPCO | INTEGER ARRAY | -- | 1 | Indicator for control operation during return to null position |
| IPASS | INTEGER ARRAY | -- | 0 | " |
| IPSTOP | INTEGER | -- | 0 | " |
| IQCU | ARRAY | -- | 0 | Indicator for defining when the cumulative fluid volume supplied by the control system has returned to zero when the strut is fully-extended during rebound: =0, non-zero; =1, zero |
| ISET | INTEGER ARRAY | -- | 1 | See IOPCO |
| ISTROK | INTEGER ARRAY | -- | 0 | Indicator for strut stroke initialization |

| | | | | |
|---|---|---|---|---|
| IXS | ARRAY | -- | 0 | Indicator defining the condition of servovalve spool displacement equal to bias value for the fully-extended strut during rebound:<br>=0, unequal<br>=1, equal |
| IXSVH | INTEGER ARRAY | -- | 0 | Indicator for control operation during return to null position |
| IXSVL | INTEGER ARRAY | -- | 0 | " |
| NAC | INTEGER ARRAY | -- | 0 | Fluid flow indicator<br>0=off;<br>1=from strut;<br>2=to strut |
| NITER | INTEGER | -- | | Number of iterations required to converge to solution |
| PGA1T | ARRAY | $LBF/FT^2$ | | Gauge pressure of hydraulic fluid in strut piston |
| PGA2T | ARRAY | $LBF/FT^2$ | | Gauge pressure of nitrogen in strut cylinder |
| PGA3T | ARRAY | $LBF/FT^2$ | | Gauge pressure of hydraulic fluid in volume 3 |
| PGA1T1 | ARRAY | $LBF/FT^2$ | | See PGA1T |
| PR | ARRAY | $LBF/IN^2$ | | Servovalve return pressure |
| PS | ARRAY | $LBF/IN^2$ | | Servovalve supply pressure |
| P1 | ARRAY | $LBF/IN^2$ | | Strut piston pressure |
| QC | ARRAY | $IN^3/SEC$ | 0.0 | Flow rate to the load |
| QO | ARRAY | $FT^3/SEC$ | 0.0 | Flow rate through strut main orifice |

| | | | | |
|---|---|---|---|---|
| QS1 | ARRAY | $IN^3/SEC$ | | Flow rate from accumulator to strut piston |
| QS3 | ARRAY | $IN^3/SEC$ | | Flow rate from strut piston to reservoir |
| QSV | ARRAY | $FT^3/SEC$ | | Fluid flow rate through servovalve, positive from supply to strut |
| QSVCU | ARRAY | $FT^3$ | 0.0 | Cumulative volume of fluid added to or removed from strut |
| QSVN | ARRAY | $FT^3/SEC$ | 0.0 | Volume expansion or compression rate of nitrogen in high pressure accumulator |
| QSV1 | ARRAY | | | See QS1 |
| QSV3 | ARRAY | | | See QS3 |
| QTOLER | | $IN^3/SEC$ | .0001 | Tolerance allowed in calculating flow rates |
| REDSLP | ARRAY | LBF/SEC | 1.E5 | Slope for reducing control limit force from impact value to rollout value |
| RESA | ARRAY | | 0.0 | Control activation switch |
| SA | ARRAY | | 0.0 | " |
| SBFOT | | | 0.0 | Strut elastic bending force |
| SIPA | | | | sin ($\phi+\alpha$) |
| UNSPRNG | | SLUGS | | Mass of strut piston and gear components attached |
| VCUM | ARRAY | $FT^3$ | 0.0 | Cumulative volume of fluid flowed from piston to cylinder |
| VELDEC | ARRAY | FT/SEC | 0.0 | Velocity for initiating transition from impact limit force to rollout value |

| | | | | |
|---|---|---|---|---|
| VMASS | ARRAY | SLUGS | | Vehicle mass |
| VOLAHT | ARRAY | $FT^3$ | | Volume of hydraulic fluid in high pressure accumulator |
| VOLANT | ARRAY | $FT^3$ | | Volume of nitrogen in high pressure accumulator |
| VOL1T | ARRAY | $FT^3$ | | Volume of hydraulic fluid in strut piston |
| VOL2T | ARRAY | $FT^3$ | | Pneumatic volume in strut cylinder |
| VOL3T | ARRAY | $FT^3$ | | Volume between piston and cylinder |
| WFORT | ARRAY | LBF | 0.0 | Gear-airframe interface force |
| XMA | ARRAY | AMPERES | | Input signal to electronic compensation networks |
| XMA1 ... 4 | ARRAY | " | 0.0 | modification of input signal to the notch and lead-lag networks |
| XMA5 | ARRAY | " | | " |
| XMA6,7 | ARRAY | " | 0.0 | " |
| XMA8 | ARRAY | AMPERES | | " |
| XMA9,10 | ARRAY | AMPERES | | " |
| XMA11 | ARRAY | AMPERES | | Output signal from electronic compensation networks to servovalve |
| XMU | | CENTIPOISE | | Viscosity of hydraulic fluid |
| XSTOT | ARRAY | FT | | Instantaneous stroke required to dissipate remaining kinetic energy at simultaneous force level |

| | | | | |
|---|---|---|---|---|
| XSV | ARRAY | INCHES | | Servovalve spool displacement |
| XSVDD | ARRAY | $IN/SEC^2$ | 0.0 | Acceleration of servovalve spool |
| XSVDDD | ARRAY | $IN/SEC^3$ | 0.0 | Jerk on servovalve spool |
| XSVDOT | ARRAY | IN/SEC | 0.0 | Velocity of servovalve spool |
| XVALVE | ARRAY | IN | | Analytically controlled servovalve spool displacement |
| ZDANT | | FT/SEC | | Velocity of nose gear root |
| ZSSC | ARRAY | FT | | Allowable strut stroke for activating control |

1.3 Stroke Dependent Metering Pin Area - Following implementation of the active control landing gear, an additional capability was required to model the F4 aircraft landing gears. The metering pin area was allowed to have either a constant value (the original design) or to be functionally dependent on the strut stroke. No new subroutines were required and modifications were limited to the active input routines, ACTINIT, and ALGEAR. Input variables are presented and defined in Table 1.

1.4 Variable Aerodynamic Coefficients - The use of staging to introduce changes in the aerodynamic coefficients $C_{A_0}$ and $C_{N_0}$ was a cumbersome process and led to discontinuities in program results. New program variables were introduced to allow the specification of a rate of change for each of these coefficients providing a much smoother transition. This feature required two new subroutines:

AERO4--initializes and updates coefficients

AEROIN--reads rates and limiting values

the new variables have been stored in a new common block, AEROCO. Input variables are included in Table 1.

## 1.5 Restart Capability

Recent attempts to model the F4 gear have shown that at the time of touchdown extremely large changes occur in several variables resulting in a small integration step size. The final result is a large amount of computer time. The problem was compounded by having to rerun the entire simulation whenever a program failure was experienced.

The introduction of a restart capability eliminated the problem of total reruns. Whenever any data is staged into the program, a disk file is generated which contains all the information required to restart the program from that point in time. Thus when program failure occurs, it is only necessary to rerun from the most recent stage. This technique has been found etremely useful in the analysis of the effects of various runway conditions. Rather than running the entire landing simulation for each runway to be considered, it is now only necessary to begin the simulation at that point in time where the runway bump is encountered.

No new subroutines were required, but many existing routines were modified by relocating their local variables into labeled common blocks. With all program variables in common blocks declared in the main program, their storage locations are contiguous and a single read or write transfers the entire block of restart data.

New program variables required for the restart option are included in Tables 1 and 2.

## 2. CHRONOLOGICAL HISTORY OF PROGRAM MODIFICATIONS

Numerous program changes have been made to the FATOLA program since the last formal documentation (Reference 3) in 1975. A total of 52 UPDATE corrections sets have been applied since that time. Numerous modifications were performed by Structures and Dynamics Division (SDD) personnel and were reported in References (1) and (2). The remaining coding changes were performed by Computer Sciences Corporation (CSC) personnel. For the sake of completeness, all SDD and CSC modifications are included in this history.

2.1 Structures and Dynamics Division Modifications - Beginning in November 1975 and ending in November 1978, several improvements were made to the FATOLA program to enhance its simulation capabilities. The net effect of the three correction sets C82477, C11778, and SDDMODS is reported in References (1) and (2) with the following exceptions.

In program TOLA, the call to FTNBIN was commented out when the Integrated Computer Operating System (ICOPS) was replaced with the Network Operating System (NOS).

Dummy space was not added to the end of the DIRCOM common block in subroutine TFFS9 as reported in Reference (2). This addition is not necessary for proper program execution.

A call to subroutine FLEX5 was inserted in subroutine OPT1 immediately before the call to FLEX6. The reason for this addition is not apparent since it is a do-nothing entry point.

The function ATAN2 was declared EXTERNAL immediately following the SUBROUTINE LGEA3C statement. Again, there is no apparent reason for this addition.

In an attempt to prevent over-extension or compression of the struts, coding changes were made to two subroutines. In subroutine LGDET, the FORMAT statements labeled 49 and 53 were replaced by the following statements

```
49 FORMAT(58X,4H-ES(,I1,19H) EXCEEDED IN LGDET)
53 FORMAT(58X,4H ES(,I1,20H) EXCEEDED IN LGDET /
```

The following lines of code were inserted before statement number 50,

```
Y(J)=-.5*ES(I)
Y(JJ)=-1.0E-10
P(JJ)=-1.0E-10
```

and the statements between and including statement numbers 51 and 21 were replaced with the following.

```
51 IF(SD2(1,I).LT.0.)GO TO 30
   IF(SD1(1,I).LT.0.)SD1(1,I)=0.
   GO TO 55
30 SD2(1,I)=0.
   SD1(1,I)=0.
```

In subroutine LGEAR1, the formats were similarly replaced as follows.

```
49 FORMAT(58X,4H-ES(,I1,20H) EXCEEDED IN LGEAR1/
53 FORMAT(58X,4H ES(,I1,20H) EXCEEDED IN LGEAR1/
```

the following code was inserted before statement number 50

```
S(1,I)=-0.5*ES(I)
SD1(1,I)=-1.0E-10
SD2(1,I)=-1.0E-10
```

and the two lines beginning with statement number 51 were replaced with the following.

```
51 IF(SD2(1,I).LT.0.)GO TO 30
   IF(SD1(1,I).LT.0.)SD1(1,I)=0.
   GO TO 55
30 SD2(1,I)=0.
```

Additionally, the following code was inserted immediately before statement number 20.

```
C     RE-CHECK SHOCK STRUT FORCE FOR RE-CONDITIONED FULLY EXTENDED STATE
      IF(SD1(1,I).EQ.0.0.AND.FT(I).LE.ABS(SF(I)))SF(I)=-FT(I)
```

2.2 Computer Sciences Corporation Modifications - In early 1978, CSC was requested to support a major modification to the FATOLA program, by introducing the capability of an active control landing gear. A mathematical model of an active control landing gear

(ACOLAG) had been developed and programmed (Reference 7), however this program did not include airframe elastic effects. The goal of the program modifications was to incorporate the series-hydraulic active control logic and equations for all landing gear oleo-pneumatic shock struts into the FATOLA computer program.

The correction idents ACOBLK, CSCMODS, ACTINIT, ALGEAR, PHLOZ2, FLOZE2, and LIMITS contained the code required for a first cut at implementing the active control. A comdeck, ACOBLK, was inserted following the deck TOLA to establish a common block containing the active gear variables. The original contents of this common block are shown below.

```
C*****************************  ACTIVE VARIABLES  ****************************
      COMMON /ACTIVE/ AMUH   ,ACON       ,BCON       ,CCON       ,DCON
     1,APINT(5)   ,AP2TO(5)  ,AREA1(5)   ,AREA2(5)   ,AREA3(5)
     2,AREMO(5)   ,ARE03(5)  ,BLFORT(5)  ,BLMU(5)    ,BUFORT(5)  ,BUMU(5)
     3,BETA       ,CDMOC(5)  ,CDMOE(5)   ,CDSV(5)    ,CFFOR(5)   ,CD3(5)
     4,COEF       ,COEFO(5)  ,COEF3(5)   ,COPA
     5,CSV1(5)    ,CSV3(5)   ,DELT       ,DELTX(5)   ,DELTX1(5)  ,DLTX1D(5)
     6,DF(5)      ,DP1(5)    ,DSV(5)     ,DSTOP      ,DIOTA      ,DMTANH(5)
     7,ENUP(5)    ,EPSILO(5) ,EPSROL(5)  ,EPSSLP     ,ETASV      ,FSTOPK
     8,FFORT(5)   ,FOAHST(5) ,FONHST(5)  ,FORCHT(5)  ,FORSST(5)  ,FSTOP(5)
     9,GAMAH(5)   ,GAMAN     ,GNR        ,HMM(5)     ,ICOSV(5)   ,IDEACT
     *,IFRI(5)    ,IIXSVH(5) ,IIXSVL(5)  ,IOPCO(5)   ,IPASS(5)   ,ISET(5)
     1,IXSVH(5)   ,IXSVL(5)  ,IFSTOP(5)  ,ISTROK(5)  ,KAPT(5)
     2,NAC(5)     ,NITER     ,OMRUN      ,PATM
     3,PERCNT(5)  ,PGAHAC(5) ,PGALAC(5)  ,PGA1I(5)   ,PGA1T(5)
     4,PGA2I(5)   ,PGA2T(5)  ,PGA3I(5)   ,PGA3T(5)
     5,PR(5)      ,PS(5)     ,P1(5)      ,QC(5)      ,QO(5)      ,QPUMPS(5)
     6,QSV(5)     ,QSVCU(5)  ,QSVN(5)    ,QSV1(5)    ,QSV3(5)    ,QS1(5)
     7,QS3(5)     ,QTOLER    ,RCLSV(5)   ,REDSLP(5)  ,RHOH       ,SBFOT
     8,SIPA       ,SA(5)     ,RESA(5)    ,TAUF       ,TC1        ,TC2
      COMMON /ACTIVE/ TC3    ,TC4        ,VCUM(5)    ,VELDEC     ,VOL1I(5)
     1,VOL1T(5)   ,VOL2I(5)  ,VOL2T(5)   ,VOL3I(5)   ,VOL3T(5)   ,WC
     2,WC1        ,WFORT(5)  ,WLFOR(5)   ,WLFORR     ,XBIAS(5)   ,XVALVE(5)
     3,XKA(5)     ,XKF(5)    ,XKSV(5)    ,XLPSV1(5)  ,XLPSV3(5)  ,XSTHR
     4,XMA(5)     ,XMA1(5)   ,XMA2(5)    ,XMA3(5)    ,XMA4(5)    ,XMA5(5)
     5,XMA6(5)    ,XMA7(5)   ,XMA8(5)    ,XMA9(5)    ,XMA10(5)   ,XMA11(5)
     6,XMU        ,XSCOM(5)  ,XSTOT(5)   ,XSV(5)     ,XSVDOT(5)  ,XSVDD(5)
```

```
     7,XSVDDD(5) ,XSVDMN(5) ,XSVDMX(5) ,XSVMAX(5) ,XSVMIN(5) ,XDDMAX(5)
     8,XDDMIN(5) ,VOLACI(5) ,VOLAHT(5) ,VOLANI(5) ,VOLANT(5)
     9,WSV       ,WSV1      ,WSV3      ,ZETAC1    ,ZETAC2    ,ZSSC
C*****************************************************************
```

New decks ACTINIT, for initializing the active gear variables, and ALGEAR, PHLOZ2, FLOZE2, and LIMITS, containing the active gear logic, were added after the LGEA3C deck. These new decks are provided below.

```
      SUBROUTINE ACTINIT
C
C******************** FATOLA VARIABLES ********************************
       COMMON/DIRCOM/DM1(115),ALPHD,DM1A(20),AMASS,DM2(147),DCL1,DCM1,
     CDCN1,DCL2,DCM2,DCN2,DCL3,DCM3,DCN3,DM3(99),FXB7P,
     CDUM4(3),FYB7P(4),FZB7P,DM5(17),GXB7F,DM6(8),GZB7F,
     CDM7(218),INDSTE(48),PHIPD,INDSTE1(23),PSIPD,INDSTE2(156),THTPD,
     CINDSTE3(5),TIME,DM8(287),PI77R(2),PI77R1(2),DM9(4),
     CQI77R(2),QI77R1(2),DM10(4),RI77R(2),RI77R1(2),DM11(48),
     CXG77F(2),XG77F1(12),YG77F(2),YG77F1(12),
     CZG77F(2),ZG77F1(2),DUM13(52),
     CNSTRUT,MASS(5),RX(5),RY(5),RZ(5),THETAD(5),ERDEG,RGR,
     CNTIRES(5),RZERO(5),W(5),DELTAM(5),MOMENT(5),
     CRF(5),VZ  ,IFD,PZERO(5),VZERO(5),A(5),P20(5),V20(5),
     CA2(5),IL,S2T(5),ES2(5),C2L(5),MASS2(5),MUS(5),
     CCC(5),CE(5),C2C(5),C2E(5),NVGPT,NPP,MB(5),RLT,NDELTA,
     CES(5),SB(5),SD21(2),SD22(2),SD23(2),SD24(2),SD25(2)
       COMMON/DIRCOM/
     CSD11(2),SD12(2),SD13(2),SD14(2),SD15(2),
     CS1(2),SS2(2),S3(2),S4(2),S5(2),
     CS2D21(2),S2D22(2),S2D23(2),S2D24(2),S2D25(2),
     CS2D11(2),S2D12(2),S2D13(2),S2D14(2),S2D15(2),
     CS21(2),S22(2),S23(2),S24(2),S25(2),
     COMTD11(2),OMTD12(2),OMTD13(2),OMTD14(2),OMTD15(2),
     COMT1(2),OMT2(2),OMT3(2),OMT4(2),OMT5(2),
     CAI(5),BI(5),DELTA1,DELTA2,DELTA3,DELTA4,DELTA5,
     CDDELT1,DDELT2,DDELT3,DDELT4,DDELT5,ISTAGE,
     CPRTMIN,IPLT,ISDF,ISTPL1,ISTPL2,ISTPL3,ISTPL4,ISTPL5,
     CDM14(22),IB(5),DM15(127),INDLG,DM16(107),CASK(44),INDFLX,
     *NMODE,DM18(40),SXMOD(100),SYMOD(100),SZMOD(100),DM19(1686),
     *GQD2(20),DDM20(20)
     *,DDM21(20),SLEN1(5),SLEN2(5),DUM15(13),INDNWS,DDM22(10),ETADES,
     *DDM23(5),AH(5),PH(5),DDM24(30)
       COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
C
*CALL ACOBLK
```

```
      DIMENSION CD3(5), SD1(2,5)
      EQUIVALENCE(SD11(1),SD1(1,1)), (DM5(16),GREFF)
C
C        DATA STATEMENTS TO SET VALUES OF INPUT VARIABLES
C
      DATA AMUH/0.00018/, BETA/14400000./
      DATA BLMU,BUMU/10*0.15/
      DATA AREA1,AREA2,AREA3/5*0.21139,5*0.32167,5*0.00698/
      DATA AREMO,AREO3,APINT/5*0.0066,5*0.19635,5*0.0/
      DATA CDMOC,CDMOE,CD3/5*0.9,5*0.14197,5*0.6/, CFFOR/5*50.0/
      DATA CDSV,DELT,DIOTA/5*0.62,0.0001,0.0/
      DATA EPSILO,EPSSLP,EPSROL,ETASV/5*200.0,0.0,5*2000.0,0.436/
      DATA GAMAN,GAMAH,GNR/1.06,5*52.36,0.0/
      DATA ICOSV,IFSTOP/10*0/
      DATA IDEACT,II,KAPT,OMRUN,PERCNT/7*0,-0.07575,5*1.66115/
      DATA PATM,PGAHAC,PGALAC/2116.8,5*432000.0,5*0.0/
      DATA PGA1I,PGA2I,PGA3I/15*40320.0/
      DATA TAUF,TC1,TC2,TC3,TC4/0.1,0.281,0.141,0.001,0.0001/
      DATA VOL1I,VOL2I,VOL3I/5*0.36379,5*0.47164,5*0.0/
      DATA RHOH,WLFOR,WLFORR/1.626,5*500.0,0.0/
      DATA DSV,RCLSV,WSV1,WSV3/5*1.1875,5*0.0000475,2*3.480/
      DATA WC,WC1,WSV,XKSV,XSTHR/1263.0,251.3,655.5,5*0.00250,0.020833/
      DATA XKA,XKF,XLPSV1,XLPSV3/5*0.04,5*60.0,5*0.0,5*0.0/
      DATA XBIAS,XSCOM,XSVDMN,XSVDMX/5*-0.00014,5*1.206,5*-30.1,5*30.1/
      DATA XDDMAX,XDDMIN/5*1.0E20,5*-1.0E20/
      DATA XSVMAX,XSVMIN,ZETAC1,ZETAC2/5*0.100,5*-0.100,5.1,0.1/
      DATA VOLACI,VOLANI,QPUMPS/5*1.33333,5*0.26667,5*0.02005/
C*****************************************************************************
      IF (TIME .GT. DELT) RETURN
C
      ACON=1./(WSV*WSV*WC)
      BCON=1./(WSV*WSV)+2*ETASV/(WSV*WC)
      CCON=2.*ETASV/WSV+1./WC
      DCON=1.
      XMU=AMUH/.0000209
      QTOLER=.0001
      SBFOT=0.0
      VELDEC=0.
      OMRUN=OMRUN*0.01745329
      II = 0
C
      DO 100 I=1,NSTRUT
      REDSLP(I) = 100000.0
      ISTROK(I)=0
      NAC(I)=0
      VOLANT(I) = VOLANI(I)
      VOL1T(I)=VOL1I(I)
      VOL2T(I)=VOL2I(I)
      VOL3T(I)=VOL3I(I)
      QSVCU(I)=0.0
      DF(I)=0.
```

```
      DELTX(I)=0.
      DELTX1(I)=DELTX(I)*XKF(I)
      DLTX1D(I)=0.
      XMA(I)=(DF(I)+DELTX1(I))*XKA(I)
      XMA5(I)=XMA(I)
      XMA8(I)=XMA5(I)
      XMA11(I)=XMA8(I)
      XMA1(I)=0.
      XMA2(I)=0.
      XMA3(I)=0.
      XMA4(I)=0.
      XMA6(I)=0.
      XMA7(I)=0.
      XMA9(I)=0.
      XMA10(I)=0.
      XSVDD(I)=0.
      XSVDDD(I)=0.
      XSV(I)=XKSV(I)*XMA11(I)+XBIAS(I)
      XSVDOT(I)=0.0
      DP1(I)=0.
      COEF3(I)=CD3(I)*SQRT(2.*GREFF/GAMAH(I))
C
C       NOTE: SUBROUTINE 'PHLOZ2' COMPUTES INITIAL PRESSURES AND FLOWS IN
C       UNITS OF INCHES.
      COEF=CDSV(I)*SQRT(2.*GREFF/GAMAH(I))*144.
      CSV1(I)=COEF*WSV1
      CSV3(I)=COEF*WSV3
      PS(I)=PGAHAC(I)/144.
      PR(I)=PGALAC(I)/144.
      QC(I)=0.
      CALL PHLOZ2(PS(I),PR(I),XSV(I),QC(I),XLPSV1(I),XLPSV3(I),RCLSV,DSV
     & ,CSV1(I),CSV3(I),XMU,QTOLER,NITER,P1(I),QS1(I),QS3(I))
      PGA1I(I)=P1(I)*144.
      PGA2I(I)=PGA1I(I)
      PGA3I(I)=PGA2I(I)
      QSV1(I)=QS1(I)/1728.
      QSV3(I)=QS3(I)/1728.
C
      PGA1T(I)=PGA1I(I)
      PGA2T(I)=PGA2I(I)
      PGA3T(I)=PGA3I(I)
      QD(I)=0.
      QSV(I)=QSV1(I)-QSV3(I)
      VCUM(I)=0.
C
      AP2TO(I)=PGA2I(I)+PATM
      FFORT(I) = 0.0
      FORSST(I)=0.0
      WFORT(I)=0.0
      FONHST(I)=0.0
      FORCHT(I)=0.0
      XVALVE(I)=XSV(I)
  100 CONTINUE
```

```
C
      CALL SETUP
      RETURN
      END
*IDENT    ALGEAR
*INSERT     ACTINIT.145
*DECK ALGEAR
      SUBROUTINE ALGEAR1
C
********************** FATOLA VARIABLES ***********************************
      COMMON/DIRCOM/DM1(115),ALPHD,DM1A(20),AMASS,DM2(147),DCL1,DCM1,
     CDCN1,DCL2,DCM2,DCN2,DCL3,DCM3,DCN3,DM3(99),FXB7P,
     CDUM4(3),FYB7P(4),FZB7P,DM5(17),GXB7F,DM6(8),GZB7F,
     CDM7(218),INDSTE(48),PHIPD,INDSTE1(23),PSIPD,INDSTE2(156),THTPD,
     CINDSTE3(5),TIME,DM8(287),PI77R(2),PI77R1(2),DM9(4),
     COI77R(2),OI77R1(2),DM10(4),RI77R(2),RI77R1(2),DM11(48),
     CXG77F(2),XG77F1(12),YG77F(2),YG77F1(12),
     CZG77F(2),ZG77F1(2),DUM13(52),
     CNSTRUT,MASS(5),RX(5),RY(5),RZ(5),THETAD(5),ERDEG,RGR,
     CNTIRES(5),RZERO(5),W(5),DELTAM(5),MOMENT(5),
     CRF(5),VZ  ,IFD,PZERO(5),VZERO(5),A(5),P20(5),V20(5),
     CA2(5),IL,S2T(5),ES2(5),C2L(5),MASS2(5),MUS(5),
     CCC(5),CE(5),C2C(5),C2E(5),NVGPT,NPP,MB(5),RLT,NDELTA,
     CFS(5),SB(5),SD21(2),SD22(2),SD23(2),SD24(2),SD25(2)
      COMMON/DIRCOM/
     CSD11(2),SD12(2),SD13(2),SD14(2),SD15(2),
     CS1(2),SS2(2),S3(2),S4(2),S5(2),
     CS2D21(2),S2D22(2),S2D23(2),S2D24(2),S2D25(2),
     CS2D11(2),S2D12(2),S2D13(2),S2D14(2),S2D15(2),
     CS21(2),S22(2),S23(2),S24(2),S25(2),
     COMTD11(2),OMTD12(2),OMTD13(2),OMTD14(2),OMTD15(2),
     COMT1(2),OMT2(2),OMT3(2),OMT4(2),OMT5(2),
     CAI(5),BI(5),DELTA1,DELTA2,DELTA3,DELTA4,DELTA5,
     CDDELT1,DDELT2,DDELT3,DDELT4,DDELT5,ISTAGE,
     CPRTMIN,IPLT,ISDF,ISTPL1,ISTPL2,ISTPL3,ISTPL4,ISTPL5,
     CDM14(22),IB(5),DM15(127),INDLG,DM16(107),CASK(44),INDFLX,
     *NMODE,DM18(40),SXMOD(100),SYMOD(100),SZMOD(100),DM19(1686),
     *GOD2(20),DDM20(20)
     *,DDM21(20),SLEN1(5),SLEN2(5),DUM15(13),INDNWS,DDM22(10),ETADES,
     *DDM23(5),AH(5),PH(5),DDM24(30)
      REAL MASS,MOMENT,   MASS2,MUS,NTIRES,MB
      DIMENSION DLGAUT(14),DLGDE(47),DLG(7),DLGE(299)
      COMMON/LGAUTS/ARG11,ARG13,ARG31,ARG33,AMA(5),VAXLE(5)
      COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
      COMMON/LG/FXM,FYM,FZM,LM,MM,NM,EPSLO2
      COMMON/LGE/A11(5),A13(5),A31(5),A33(5),RRCGX(5),
     CRL(3,3),RI(3,3,5),RAX(5),RAY(5),RAZ(5),TMP(3),ZZERO(5),
     CXR(5),YR(5),EPSLON(5),PA(5),FDELTA(5),
     CFTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
     CVTX(5),VTY(5),VTZ(5),GZ(5),      VGPT(5),FTRX(5),FTRY(5),
     CDX(5),DY(5),DZ(5),FT(5),FDX(5),FDY(5),FF(5),AA(5),C2(5),
     CSR(5),SF(5),PSKD(5),MUVP(5),MTRX(5),MTRY(5),
```

```
      CMTRZ(5),MA(5),RG11,RG13,RG31,RG33,IPRT,
      CMTX,MTY,MTZ,SFTRX,SFTRY,SFTRZ,FTRA,
      CFTRB,FTRC,SMTRX,SMTRY,SMTRZ
       COMMON/FLXOP/GFORC2(100),GFORC3(100),GFORC4(100),BH1(308)
       EQUIVALENCE (DLGAUT(1),ARG11),(DLGDE(1),LA(1)),
      *(DLG(1),FXM),(DLGE(1),A11(1))
       COMMON/TABSRC/DUMM1(103),LOC(7)
       COMMON/HTCOM/HT,HT1,HT2
C
*CALL ACOBLK
C
       REAL MUVP,MTRX,MTRY,MTRZ,MA,
      CMTX,MTY,MTZ,LM,MM,NM
       DIMENSION DELTA(5),DDELTA(5),P(5),
      C       SD2(2,5),SD1(2,5),S(2,5),S2D2(2,5),S2D1(2,5),S2(2,5),
      COMETD1(2,5),OMET(2,5)
       EQUIVALENCE (P(1),PRES(1))
       EQUIVALENCE (DELTA(1),DELTA1),(DDELTA(1),DDELT1)
       EQUIVALENCE
      C(SD21(1),SD2(1,1)),(SD11(1),SD1(1,1)),(S1(1),S(1,1)),
      C(S2D21(1),S2D2(1,1)),(S2D11(1),S2D1(1,1)),(S21(1),S2(1,1)),
      C(OMTD11(1),OMETD1(1,1)),(OMT1(1),OMET(1,1))
C
C**************ACTIVE CODE*****************************************
       EQUIVALENCE (DM5(16),GREFF)
       DATA IFRI/5*0/,FSTOPK/0.0/,FSTOP/5*0.0/,DSTOP/0.004/
       DATA SA,RESA,HMM,IXSVL,IXSVH,IIXSVL,IIXSVH,IPASS
      1     /15*0.,25*0/
       DATA ENUP/5*0.0/
       DATA ISET,IOPCO,QSVN/10*1,5*0.0/
       Q1(T1,T2)=SIGN(1.,(T1-T2))*SQRT(ABS(T1-T2))
C*******************************************************************
C
       DATA RADDEG,DEGRAD/57.2957795,.01745329/
C
C      MAIN COMPUTATIONAL AREA
C      RL MATRIX ELEMENTS
       RL(1,1)=DCL1*RG11+DCL3*RG13
       RL(1,2)=DCL2
       RL(1,3)=DCL1*RG31+DCL3*RG33
       RL(2,1)=DCM1*RG11+DCM3*RG13
       RL(2,2)=DCM2
       RL(2,3)=DCM1*RG31+DCM3*RG33
       RL(3,1)=DCN1*RG11+DCN3*RG13
       RL(3,2)=DCN2
       RL(3,3)=DCN1*RG31+DCN3*RG33
       CALL LGEA3C
C
C******************** ACTIVE CODE **********************************
C    START ACTIVE GEAR CALCULATIONS
       COPA = COS((THTPD+DIOTA)*DEGRAD)
       SIPA = SIN((THTPD+DIOTA)*DEGRAD)
       IF(TIME .GT. DELT)GO TO 18
```

```
   15  WRITE(6,1013)
 1013  FORMAT (1H020H ACTIVE CONTROL GEAR)
   18  IF(IDEACT.EQ.1) GO TO 56
   19  IF(IDEACT.EQ.2) GO TO 80
       IF(HMM(I) .EQ. 0.) GO TO 90
       IF(OMRUN .GT. 0.0)GO TO 25
       IF(ZG77F1(1) .GE. VELDEC) GO TO 90
       GO TO 40
   25  IF(ZG77F1(1) .GE. VELDEC+XG77F1(1)*TAN(OMPUN))GO TO 90
   40  WRITE(6,1014)TIME
 1014  FORMAT (1H036H REDUCE CONTROL LIMIT FORCE AT TIME=,E16.8)
       IDEACT=1
   56  WLFOR(I)=WLFOR(I)-REDSLP(I)*DELT
       EPSILO(I)=EPSILO(I)+EPSSLP*DELT
       IF(WLFOR(I) .GT. WLFORR) GO TO 90
   60  WRITE(6,1015)TIME
 1015  FORMAT (1H027H CONTROL AT WLFORR AT TIME=,E16.8)
       IDEACT=2
   80  WLFOR(I)=WLFORR
       EPSILO(I)=EPSROL(I)
   90  CONTINUE
C*******************************************************************************
       DO 65 I=1,NSTRUT
       WFORT(I) = (SQRT(FXB7P*FXB7P+FYB7P*FYB7P+FZB7P*FZB7P))/(NSTRUT-1)
      &      +(FORSST(I)*COPA)
C************************* ACTIVE CODE *********************************
       IF(KAPT(I).NE.0)GO TO 210
       APINT(I)=0.0
  210  CONTINUE
       IF(PGA1T(I) .LE. -1600.0) PGA1T(I) =  -1600.0
       VOL1T(I)=VOL1I(I)-(AREA1(I)-APINT(I))*S(1,I)
       VOL3T(I)=VOL3I(I)+AREA3(I)*S(1,I)
       VOL2T(I)=VOL2I(I)-(AREA2(I)-AREA1(I)+APINT(I))*S(1,I)+(VOL3T(I)
      X -VOL3I(I))-VCUM(I)
       PGA2T(I)=AP2TO(I)*((VOL2I(I)/VOL2T(I))**GAMAN)-PATM
       IF(SD1(1,I) .EQ. 0.0)GO TO 104
       PGA3T(I)=((COEF3(I)*AREO3(I))**2*PGA2T(I)-SD1(1,I)/ABS(SD1(1,I))
      X *(SD1(1,I)*AREA3(I))**2)
      &/((COEF3(I)*AREO3(I))**2)
       GO TO 105
  104  PGA3T(I)=PGA2T(I)
  105  IF(PGA1T(I) .GE. PGA2T(I))GO TO 106
       GO TO 107
  106  GAMAH(I)=RHOH*GREFF*(1.0+(PGA1T(I)*3.04E-08)-
      * (PGA1T(I)**2*2.72E-15))
       GO TO 108
  107  GAMAH(I)=RHOH*GREFF*(1.0+(PGA2T(I)*3.04E-08)-
      * (PGA2T(I)**2*2.72E-15))
  108  IF(PGA1T(I) .GE. PGA2T(I))COEFO(I)=
      *  CDMOC(I)*SQRT(ABS(2.*GREFF/GAMAH(I)))
       IF(PGA2T(I) .GT. PGA1T(I))COEFO(I)=
      *  CDMOE(I)*SQRT(ABS(2.*GREFF/GAMAH(I)))
```

```
      IF(SD1(1,I).LE.0.0) GO TO 109
      QQ(I)=COEFQ(I)*(AREMQ(I)-APINT(I))*Q1(PGA1T(I),PGA2T(I))
  109 IF(PGA2T(I) .LE. -1600.0)PGA2T(I)=-1600.0
      IF(PGA3T(I) .LE. -1600.0)PGA3T(I)=-1600.0
  100 IF(DELTA(I) .LE. 0.0 .AND. TIME .GT. DELT)GO TO 101
      GO TO 110
  101 FFORT(I)=0.0
      GO TO 140
  110  CONTINUE
C     COMPUTE STRUT AXIAL BINDING FRICTION FORCE
      BLFORT(I)=FONHST(I)*((SLEN2(I)-S(1,I))/(SLEN1(I)+S(1,I))+1.0)
      BUFORT(I)=FONHST(I)*(SLEN2(I)-S(1,I))/(SLEN1(I)+S(1,I))
      FFORT(I)=BUMU(I)*ABS(BUFORT(I))+BLMU(I)*ABS(BLFORT(I))
  140 CONTINUE
C     COMPUTE SHOCK STRUT CHARGING FORCE
      IF(S(1,I) .GT. 0.0)GO TO 142
  141 FORCHT(I)=PGA1T(I)*AREA1(I)+PGA2T(I)*(AREA2(I)-AREA1(I))-PGA3T(I)
     X *AREA3(I)+FFORT(I)+CFFOR(I)
C     COMPUTE NORMAL AND AXIAL HUB TO SHOCK STRUT FORCES AT HUB
  142 FONHST(I)=SQRT(FDX(I)**2+FDY(I)**2)-MASS(I)*GREFF*SIPA+SBFOT
      IF(ABS(FT(I)) .LE. FORCHT(I) .AND. S(1,I) .EQ. 0.0)GO TO 150
      GO TO 801
  150 CONTINUE
      FORSST(I)=FT(I)
      SD1(1,I)=0.0
      IF(I.EQ.1) GO TO 450
      ISTROK(I)=1
C****** 289 CHANGED TO 295 FOR DEBUGGING PURPOSES:
      GO TO 295
C     COMPRESSION VELOCITY OF SHOCK STRUT IS POSITIVE
  801 IF(SD1(1,I) .LE. 0.8 .AND. IFRI(I) .EQ. 0)GO TO 2
      GO TO 3
    2 DMTANH(I)=1.0
      GO TO 284
    3 DMTANH(I)=ABS(TANH(2.0*SD1(1,I)))
      IFRI(I)=1
  284 IF(S(1,I) .LE. DSTOP .AND. SD1(1,I) .LT. 0.0)GO TO 900
      GO TO 902
  900 IF(S(1,I) .LE. 0.0001)GO TO 903
      GO TO 904
  903 SD2(1,I)=0.0
      SD1(1,I)=0.0
      S(1,I)=0.0
      DP1(I)=0.0
      FFORT(I)=0.0
      VCUM(I)=0.0
      QQ(I)=0.0
      II=0
C     CALL VIRK4(II,N,NT,CI,SPEC,CIMAX,IERR,VAR,CUVAR,DER,ELE1,ELE2,
C    1           ELT,ERRVAL,DERSUB,CHSUB,ITEXT)
      GO TO 902
  904 CONTINUE
```

```
      IF(IFSTOP(I) .NE. 0)GO TO 906
  905 DSTOP=S(1,I)
      FSTOPK=2.0*MASS(I)*SD1(1,I)**2/DSTOP**2
  906 IF(S(1,I) .LE. DSTOP/2.0)GO TO 908
  907 FSTOP(I)=-FSTOPK*(DSTOP-S(1,I))
      GO TO 909
  908 FSTOP(I)=-FSTOPK*S(1,I)
  909 IFSTOP(I)=1
      IFRI(I)=0
      GO TO 901
  902 FSTOP(I)=0.0
  901 IF(ABS(FT(I)) .LE. FORCHT(I) .AND. S(1,I) .EQ. 0.0)GO TO 500
      IF(SD1(1,I) .LT. 0.0)GO TO 470
      GO TO 471
  470 FFORT(I)=-FFORT(I)
      CFFOR(I)=-CFFOR(I)
  471 FORSST(I)=-((PGA1T(I)-PGA2T(I))*(AREA1(I)-APINT(I))
     & +PGA2T(I)*AREA2(I)
     X -PGA3T(I)*AREA3(I)+(FFORT(I)
     1       +CFFOR(I))*DMTANH(I)+FSTOP(I))
  500 IF(INDFLX.GE. 1)GO TO 295
      IF(I.EQ.1) GO TO 450
      ISTROK(I)=1
C***** ****** BRANCH FOR DEBUGGING PURPOSES:
      GO TO 295
  289 IF(S(1,I) .LE. 0.0)290,295
  290 IF(IOPCO(I) .EQ. 1)GO TO 295
      IF((PGA1I(I)-1000.0).LT.PGA1T(I).AND.
     &  PGA1T(I).LT.(PGA1I(I)+1000.0))299,298
  299 IF(XVALVE(I) .NE. 0.0)GO TO 311
      IF(IPASS(I) .EQ. 1)GO TO 296
      XVALVE(I)=XKSV(I)*XMA11(I)+XBIAS(I)
      IPASS(I)=1
      GO TO 294
  298 IF(ICOSV(I) .EQ. 1)GO TO 291
      IOPCO(I)=0
      IF(XSV(I) .LT. 0.002 .AND. XSV(I) .GT. -0.002)291,295
  291 IF(SD2(1,I) .LE. 0.0 .AND. ICOSV(I) .EQ. 1)GO TO 311
      IF(IOPCO(I) .EQ. 1)GO TO 295
      IF(PGA1T(I) .GT. PGA1I(I))292,293
  292 IF(IXSVL(I) .EQ. 1)GO TO 294
      XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
      IF(XVALVE(I) .LE. -0.1)300,294
  300 XVALVE(I)=-0.1
      IXSVL(I)=1
      GO TO 294
  293 IF(IXSVH(I) .EQ. 1)GO TO 294
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
      IF(XVALVE(I) .GE. 0.1)302,294
  302 XVALVE(I)=0.1
      IXSVH(I)=1
  294 CONTINUE
```

```
      II=0
C     CALL VIRK4(II,N,NT,CI,SPEC,CIMAX,IERR,VAR,CUVAR,DER,ELE1,ELE2,
C    1           ELT,ERRVAL,DERSUB,CHSUB,ITEXT)
      DLTX1D(I)=0.0
      ICOSV(I)=1
  296 IF(WFORT(I) .GT. 0.0 .AND. S(1,I) .LE. 0.0)GO TO 410
  311 IF(NAC(I) .EQ. 1)GO TO 307
      IF(IIXSVH(I) .EQ. 1)GO TO 305
      XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
      IF(XVALVE(I) .LE. 0.0)305,400
  305 XVALVE(I)=0.0
      IIXSVH(I)=1
      GO TO 400
  307 IF(IIXSVL(I) .EQ. 1)GO TO 308
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
      IF(XVALVE(I) .GE. 0.0)308,400
  308 XVALVE(I)=0.0
      IIXSVL(I)=1
  400 II=0
C     CALL VIRK4(II,N,NT,CI,SPEC,CIMAX,IERR,VAR,CUVAR,DER,ELE1,ELE2,
C    1           ELT,ERRVAL,DERSUB,CHSUB,ITEXT)
  410 IF(XVALVE(I) .NE. 0.0)GO TO 295
      ICOSV(I)=0
      DELTX1(I)=DELTX(I)*XKF(I)
      XMA(I)=(DF(I)+DELTX1(I))*XKA(I)
      XMA11(I)=XMA(I)
      XSV(I)=XKSV(I)*XMA11(I)+XBIAS(I)
      CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      IPASS(I)=0
      IXSVL(I)=0
      IXSVH(I)=0
      IIXSVL(I)=0
      IIXSVH(I)=0
      IOPCO(I)=1
      CALL PHLOZ2(PS(I),PR(I),XSV(I),OC(I),XLPSV1(I),XLPSV3(I),RCLSV(I),
     & DSV(I),CSV1(I),CSV3(I),XMU,QTOLER,NITER,P1(I),QS1(I),QS3(I))
C
  295 IF(ISTROK(I) .EQ. 1 .AND. S(1,I) .GT. 0.0)IOPCO(I)=0
C
      ENUP(I)=.5*AMASS*ZG77F1(1)**2
      ENUP(I)=ENUP(I)/(NSTPUT-1)
      IF(HMM(I) .EQ. 1.0)GO TO 130
      SA(I)=0.
      IF(WFORT(I).LT.0.) XSTOT(I)=ENUP(I)/((-WFORT(I))*COPA)
      IF(WFORT(I).GE.0.) XSTOT(I)=1.E20
C     ZSSC IS A PERCENTAGE OF SB(I) FOR ACTIVATING CONTROL-CDMOC(I) IS USE
      ZSSC=0.6*CDMOC(I)*SB(I)
      IF(XSTOT(I) .LE. (ZSSC-S(1,I)) .OR. RESA(I) .EQ. 1.0)SA(I)=1.0
      RESA(I)=SA(I)
      IF(SA(I).EQ.0. .OR. HMM(I).EQ.1.) GO TO 130
      WLFOR(I)=-WFORT(I)
      VELDEC=((WLFOR(I)+WLFORR)/2.*(WLFOR(I)-WLFORR))/(AMASS*RFDSLP(I))
```

```
      WRITE(6,121)TIME,WLFOR(I),VELDEC
  121 FORMAT(50H ACTIVE CONTROL INITIATED...TIME, WLFOR, VELDEC = ,
     1       3E13.5)
      HMM(I)=1.
  130 IF(S(1,I) .GT. 0.0)ISET(I)=0
      IF(HMM(I).EQ.0.) GO TO 451
      IF(-WFORT(I).GT.(WLFOR(I)+EPSILO(I))) DF(I)=(WLFOR(I)+
     &  EPSILO(I))-(-WFORT(I))
      IF(-WFORT(I).LT.(WLFOR(I)-EPSILO(I))) DF(I)=(WLFOR(I)-
     &  EPSILO(I))-(-WFORT(I))
      IF(-WFORT(I).LE.(WLFOR(I)+EPSILO(I)).AND.
     & -WFORT(I).GE.(WLFOR(I)-EPSILO(I)))
     : 457,456
  457 IF(S(1,I) .LE. 0.0)GO TO 456
  453 IF(WFORT(I) .GT. 0.0 .AND. OSVCU(I) .LT. 0.0)454,455
  454 DF(I)=WLFOR(I)-(-WFORT(I))
      GO TO 456
  455 DF(I)=0.0
  456 DELTX(I)=S(1,I)-XSCOM(I)
      IF(S(1,I) .LE. 0.0 .AND. ISET(I) .EQ. 0)GO TO 451
      GO TO 452
  451 DF(I)=0.
      DELTX(I)=0.
  452 XMA(I)=(DF(I)+DELTX1(I))*XKA(I)
      IF(GNR.EQ.1. .AND. XMA(I).GT.0.) XMA(I)=
     :  XMA(I)*SQRT((PGA1T(I)-PGALAC(I))
     X /(PGAHAC(I)-PGA1T(I)))
C
C       NOTE: SUBROUTINE 'FLOZE2' COMPUTES THE FLOWS FROM THE PRESSURES
C       IN UNITS OF INCHES.
      P1(I)=PGA1T(I)/144.
C     COMPUTATION OF HIGH PRESSURE ACCUMULATOR NITROGEN VOLUME
C     AND ACCUMULATOR PRESSURE
      VOLANT(I)=VOLANT(I)+QSVN(I)*DELT-QPUMPS(I)*DELT
      PS(I)=(((PGAHAC(I)+PATM)*(VOLANI(I)/VOLANT(I))**GAMAN)-PATM)/144.0
      IF(PS(I) .GE. 3000.0)464,465
  464 PS(I)=3000.0
      VOLANT(I)=VOLANI(I)
  465 VOLAHT(I)=VOLACI(I)-VOLANT(I)
      IF(VOLAHT(I) .LE. 0.0)466,467
  466 WRITE(6,1050)TIME
 1050 FORMAT(1HO//45H ACCUMULATOR OIL VOLUME INSUFFICENT AT TIME=,E16.8
     1      //)
      CALL LGEAR6
      STOP 500
  467 CONTINUE
      CALL FLOZE2(PS(I),PR(I),P1(I),XLPSV1(I),XLPSV3(I),RCLSV(I),DSV(I),
     &  XSV(I),QS1(I),QS3(I),CSV1(I),CSV3(I),XMU)
      QSV1(I)=QS1(I)/1728.
      QSV3(I)=QS3(I)/1728.
C
```

```
 450 CONTINUE
     OSV1(1) = 0.0
     OSV3(1) = 0.0
     OSV(I)=OSV1(I)-OSV3(I)
     IF(OSV(I) .LT. 0.0)NAC(I)=1
     IF(OSV(I) .GT. 0.0)NAC(I)=2
     IF(NAC(I) .EQ. 2)461,462
 461 OSVN(I)=OSV(I)
     GO TO 463
 462 OSVN(I)=0.0
 463 IF(SD1(1,I).LT. 0.0 .AND.PGA1T(I).LE. -1600.0)PGA1T(I)=-1600.0
 274 CONTINUE
     I1 = I+3*NSTRUT
     I2 = I+4*NSTRUT
     I3 = I+5*NSTRUT
     CALL INTEG(LA(I1),DP1(I))
     CALL INTEG(LA(I2),QQ(I))
     CALL INTEG(LA(I3),OSV(I))
C*********************************************************************
     IF(SD1(1,I).EQ.0.0.AND.FT(I).LE.ABS(FORSST(I)))FORSST(I)=-FT(I)
     AA(I) = (FT(I)+FORSST(I))/MASS(I)
     SD2(1,I) = SR(I)+AA(I)-GZ(I)
     HT1=HT
     IF(SD1(1,I))76,77,78
 76  TTIME=S(1,I)/ABS(SD1(1,I))
 79  IF(TTIME.GE.HT)GO TO 77
     HT1=TTIME
     GO TO 77
 78  TTIME=(SR(I)-S(1,I))/SD1(1,I)
     GO TO 79
 77  CONTINUE
     IF(S(1,I).GT.(-ES(I)))GO TO 50
     WRITE(6,49)I,I,S(1,I)
  49 FORMAT(58X,4H-ES(,I1,20H) EXCEEDED IN ALGEAR/
    C58X,2HS(,I1,4H) = E15.7)
     S(1,I)=-0.5*ES(I)
     SD1(1,I)=-1.0E-10
     SD2(1,I)=-1.0E-10
 50  IF(S(1,I).LE.ES(I))GO TO 51
     IF(S(1,I).LE.(SB(I)-ES(I)))GO TO 55
     IF(S(1,I).LE.(SR(I)+ES(I)))GO TO 52
     WRITE(6,53)I,I,S(1,I)
 53  FORMAT(58X,4H ES(,I1,20H) EXCEEDED IN ALGEAR/
    C58X,2HS(,I1,4H) = E15.7)
     S(1,I)=0.5*ES(I)
 52  IF(SD1(1,I).GT.0.)SD1(1,I)=0.
     IF(SD2(1,I).LT.0.)GO TO 55
     SD2(1,I)=0.
     GO TO 55
  51 IF(SD2(1,I).LT.0.)GO TO 30
     IF(SD1(1,I).LT.0.)SD1(1,I)=0.
     GO TO 55
```

```
   30 SD2(1,I)=0.
      SD1(1,I)=0.
 55   CONTINUE
      I2=2*I+NSTRUT-1
      I1=I2+1
      CALL INTEG(LA(I2),SD2(1,I))
      CALL INTEG(LA(I1),SD1(1,I))
C     RE-CHECK SHOCK STRUT FORCE FOR RE-CONDITIONED FULLY EXTENDED STATE
      IF(SD1(1,I).EQ.0.0.AND.FT(I).LE.ABS(FORSST(I)))FORSST(I)=-FT(I)
      TMP(1)=RZERO(I)-DELTA(I)
      IF(CASK(I).GT.1.E-10)GO TO 200
      IF(INDNWS.EQ.1.AND.I.EQ.1)GO TO 301
      MA(I)=-FTRY(I)*TMP(1)*RI(2,1,I)+FTRX(I)*TMP(1)
     C*RI(2,2,I)
      GO TO 201
 301  MA(I)=-FTRY(I)*TMP(1)*SIN((PSIPD+ETADES)*DEGRAD)+
     1  FTRX(I)*TMP(1)*COS((PSIPD+ETADES)*DEGRAD)
      GO TO 201
 200  MA(I)=TMP(1)*SQRT(FTRY(I)*FTRY(I)+FTRX(I)*FTRX(I))
      MA(I)=SIGN(MA(I),-VAXLE(I)-OMET(1,I)*TMP(1))
 201  AMA(I)=MA(I)
      IF(IB(I).NE.(-1))GO TO 48
      OMETD1(1,I)=0.
      OMET(1,I)=0.
      GO TO 21
 48   TMP(1)=0.
      IF(OMET(1,I).NE.0.)TMP(1)=OMET(1,I)/ABS(OMET(1,I))
      OMETD1(1,I)=(MA(I)-MB(I)*TMP(1))/(NTIRES(I)*MOMENT(I))
 21   CALL INTEG(LA(I),OMETD1(1,I))
C*********************************************************************
   65  CONTINUE
C     CALCULATION OF FTRA, FTRB, AND FTRC
      SFTRX=0.
      SFTRY=0.
      SFTRZ=0.
      DO 70 I=1,NSTRUT
      SFTRX=SFTRX+FTRX(I)
      SFTRY=SFTRY+FTRY(I)
 70   SFTRZ=SFTRZ+FTRZ(I)
      FTRA=RL(1,1)*SFTRX+RL(1,2)*SFTRY+RL(1,3)*SFTRZ
      FTRB=RL(2,1)*SFTRX+RL(2,2)*SFTRY+RL(2,3)*SFTRZ
      FTRC=RL(3,1)*SFTRX+RL(3,2)*SFTRY+RL(3,3)*SFTRZ
C     CALCULATION OF MTX, MTY, AND MTZ
      SMTRX=0.
      SMTRY=0.
      SMTRZ=0.
      DO 75 I=1,NSTRUT
      SMTRX=SMTRX+MTRX(I)
      SMTRY=SMTRY+MTRY(I)
 75   SMTRZ=SMTRZ+MTRZ(I)
      MTX=RL(1,1)*SMTRX+RL(1,2)*SMTRY+RL(1,3)*SMTRZ
```

```
      MTY=RL(2,1)*SMTRX+RL(2,2)*SMTRY+RL(2,3)*SMTRZ
      MTZ=RL(3,1)*SMTRX+RL(3,2)*SMTRY+RL(3,3)*SMTRZ
C     CALCULATION OF FXM, FYM, FZM, LM, MM, AND NM
      FYM=0.
      FZM=0.
      FXM=0.
      LM=0.
      MM=0.
      NM=0.
      BFX=0.
      BFY=0.
      BFZ=0.
      BLM=0.
      BMM=0.
      BNM=0.
      DO 82 I=1,NSTRUT
      DFXM=0.
      DFYM=0.
      DFZM=0.
      DLM=0.
      DMM=0.
      DNM=0.
      DO 14 IL4=1,NMODE
      NBB=(IL4-1)*NSTRUT+I
      DFXM=DFXM-MASS(I)*SXMOD(NBB)*GOD2(IL4)
      DFYM=DFYM-MASS(I)*SYMOD(NBB)*GOD2(IL4)
      DFZM=DFZM-MASS(I)*SZMOD(NBB)*GOD2(IL4)
      NBH=(I-1)*NMODE+IL4
      DLM=DLM-MASS(I)*GFORC2(NBH)*GOD2(IL4)
      DMM=DMM-MASS(I)*GFORC3(NBH)*GOD2(IL4)
   14 DNM=DNM-MASS(I)*GFORC4(NBH)*GOD2(IL4)
      BFX=BFX+DFXM
      BFY=BFY+DFYM
      BFZ=BFZ+DFZM
      BLM=BLM+DLM
      BMM=BMM+DMM
      BNM=BNM+DNM
      TMP(1)=MASS(I)*SD2(1,I)
      FXM=FXM+TMP(1)*A31(I)
      FYM=FYM
      FZM=FZM+TMP(1)*A33(I)
      LM=LM+TMP(1)*A11(I)*RY(I)
      MM=MM-TMP(1)*RRCGX(I)
   82 NM=NM+TMP(1)*A13(I)*RY(I)
      FXM=FXM+BFX+FTRA
      FYM=FYM+BFY+FTRB
      FZM=FZM+BFZ+FTRC
      LM=LM+BLM+MTX
      MM=MM+BMM+MTY
      NM=NM+BNM+MTZ
C
      ENTRY SETUP
```

```
C
      DO 28 I=1,NSTRUT
      IF(SD1(1,I) .LE. 0.0) GO TO 2C
      DP1(I)=(-QD(I)+QSV1(I)-QSV3(I)+(AREA1(I)-APINT(I))*SD1(1,I))
     :   *BETA/VOL1T(I)
      IF(PGA1T(I) .LE. -1600.0) 10,20
   10 PGA1T(I) = -1600.0
      IF (DP1(I) .LT. 0.0) DP1(I)=0.0
   20 CONTINUE
C     II=0
C     CALL VIRK4(II,N,NT,CI,SPEC,CIMAX,IERR,VAR,CUVAR,DER,ELE1,ELE2,
C    1           ELT,ERRVAL,DERSUB,CHSUB,ITEXT)
      IF(ICOSV(I) .EQ. 1)GO TO 26
      XMA5(I)=XMA(I)+(2.*ZETAC2*WC1)*XMA1(I)+(WC1**2)*XMA2(I)
     &        -(2.*ZETAC1*WC1)*XMA3(I)-(WC1**2)*XMA4(I)
      XMA8(I)=(TC1/TC2)*XMA5(I)+(1./TC2)*XMA6(I)-(1./TC2)*XMA7(I)
      XMA11(I)=(TC3/TC4)*XMA8(I)+(1./TC4)*XMA9(I)-(1./TC4)*XMA10(I)
C     IF(S(1,I).LE.XSTHR) XMA11(I)=0.
      XSVDDD(I)=(XKSV(I)*XMA11(I)-BCON*XSVDD(I)-CCON*XSVDOT(I)-
     :   DCON*(XSV(I)-XBIAS(I)))/ACON
      DLTX1D(I)=(XKF(I)*DELTX(I)-DELTX1(I))/TAUF
   26 CONTINUE
      CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      CALL LIMITS(XSVDOT(I),XSVDD(I),XSVDMX(I),XSVDMN(I))
      CALL LIMITS(XSVDD(I),XSVDDD(I),XDDMAX(I),XDDMIN(I))
      I1 = 3*I+6*NSTRUT-2
      I2 = I1+1
      I3 = I1+2
      CALL INTEG(LA(I1),XSVDDD(I))
      CALL INTEG(LA(I2),XSVDD(I))
      CALL INTEG(LA(I3),XSVDOT(I))
      I1 = I+9*NSTRUT
      CALL INTEG(LA(I1),DLTX1D(I))
   28 CONTINUE
      RETURN
      END


      SUBROUTINE PHLOZ2(PS,PR,X,QC,LAP1,LAP3,RCL,D,COEF1,COEF3,MU,QTOLER,
     &,NITER,P1,Q1,Q3)
C     'PHLOZ2'.....R. D. EDSON
C
C     THIS SUBROUTINE CALCULATES THE STEADY-STATE CHAMBER PRESSURE (P1)
C     AND FLOW RATES (Q1 & Q3) FOR A TWO-WAY NONSYMMETRICAL SPOOL VALVE
C     WITH RECTANGULAR WINDOW SLOTS, GIVEN THE STROKE (X) AND THE LOAD
C     FLOW (QC).  THE PARAMETERS REQUIRED IN THE 'CALL' STATEMENT ARE
C     THE SAME AS DESCRIBED IN SUBROUTINE 'FLOZE2', WITH THE FOLLOWING
C     ADDITIONAL PARAMETERS:
C        QC     = THE FLOW RATE TO THE LOAD
C        QTOLER = THE TOLERANCE ALLOWED IN CALCULATING FLOW RATES, FOR
C                 DETERMINING WHETHER OR NOT THE SOLUTION HAS CONVERGED
C                 (.0001 IS TYPICAL)
C        NITER  = THE NUMBER OF ITERATIONS REQUIRED TO CONVERGE TO A
C                 SOLUTION (INTEGER)
```

```
C
C
      IMPLICIT REAL(L,M)
      PFN(X1,X2,Y1,Y2,Y3,Y4)=X1+(X2-X1)*(Y2-Y1)/(-Y1+Y2-Y3+Y4)
      NITER=0.
      FLAG=-1.
      P1FLAG=-1.
C
      P1A=PR
      P1B=PS
      CALL FLOZE2(PS,PR,P1A,LAP1,LAP3,RCL,D,X,Q1A,Q3A,COEF1,COEF3,MU)
      CALL FLOZE2(PS,PR,P1B,LAP1,LAP3,RCL,D,X,Q1B,Q3B,COEF1,COEF3,MU)
      IF(Q1A.EQ.0. .AND. Q3B.EQ.0.) GO TO 51
      Q3A=Q3A+QC
      Q3B=Q3B+QC
      GO TO 50
   51 P1=(PS+PR)/2.
      Q1=0.
      Q3=0.
      GO TO 400
C
   50 P1=PFN(P1A,P1B,Q3A,Q1A,Q1B,Q3B)
      CALL FLOZE2(PS,PR,P1,LAP1,LAP3,RCL,D,X,Q1,Q3,COEF1,COEF3,MU)
      Q3=Q3+QC
      IF(FLAG.LT.0.) GO TO 55
      IF(P1.EQ.P1I) GO TO 100
   55 P1I=P1
      IF(Q1.EQ.0. .AND. Q3.EQ.0.) GO TO 100
      IF(ABS(Q1) .GE. ABS(Q3)) QDEN=Q1
      IF(ABS(Q3) .GT. ABS(Q1)) QDEN=Q3
      IF(ABS((Q1-Q3)/QDEN) .LT. QTOLER) GO TO 100
      IF(Q1.LT.Q3) GO TO 90
      P1A=P1
      Q1A=Q1
      Q3A=Q3
      GO TO 150
   90 P1B=P1
      Q1B=Q1
      Q3B=Q3
      GO TO 150
  100 P1FLAG=1.
  150 FLAG=1.
      NITER=NITER+1
      IF(P1FLAG.GT.0.) GO TO 300
      GO TO 50
  300 CONTINUE
      Q3=Q3-QC
  400 RETURN
      END
```

```
      SUBROUTINE FLOZE2(PS,PR,P1,LAP1,LAP3,RCL,D,X,Q1,Q3,COEF1,COEF3,MU)
C     'FLOZE2'.....R. D. EDSON
C
C     THIS SUBROUTINE CALCULATES THE STEADY-STATE FLOW RATES (Q1 AND
C     Q3) FOR A TWO-WAY NONSYMMETRICAL SPOOL VALVE WITH RECTANGULAR
C     WINDOW SLOTS, GIVEN THE LOAD CHAMBER PRESSURE (P1) AND STROKE
C     (X).  THE PARAMETERS REQUIRED IN THE 'CALL' STATEMENT ARE AS
C     FOLLOWS:
C        X      = VALVE STROKE
C        P1     = PRESSURE IN CHAMBER 1 (TO LOAD)
C        Q1     = FLOW RATE FROM SUPPLY LINE TO CHAMBER 1
C        Q3     = FLOW RATE FROM CHAMBER 1 TO RETURN LINE
C        PS     = SUPPLY PRESSURE
C        PR     = RETURN PRESSURE
C        LAP1   = OVERLAPPED OR UNDERLAPPED LENGTH BETWEEN THE SPOOL
C                 AND SLEEVE AT NULL, FOR FLOW Q1.  A POSITIVE NUMBER
C                 IS USED FOR OVERLAP, A NEGATIVE NUMBER FOR UNDERLAP.
C        LAP3   = OVERLAPPED OR UNDERLAPPED LENGTH BETWEEN THE SPOOL
C                 AND SLEEVE AT NULL, FOR FLOW Q3.  A POSITIVE NUMBER
C                 IS USED FOR OVERLAP, A NEGATIVE NUMBER FOR UNDERLAP.
C        RCL    = RADIAL CLEARANCE BETWEEN THE SPOOL AND SLEEVE
C        D      = DIAMETER OF SPOOL
C        COEF1  = FLOW COEFFICIENT OF ORIFICE 1 (SUPPLY TO CHAMBER 1)
C               = CD*W1*SQRT(2.*GC/RHO)
C        COEF3  = FLOW COEFFICIENT OF ORIFICE 3 (CHAMBER 1 TO RETURN)
C               = CD*W3*SQRT(2.*GC/RHO)
C                 WHERE W1 = TOTAL WINDOW WIDTH OF ORIFICE 1
C                       W3 = TOTAL WINDOW WIDTH OF ORIFICE 3
C                       CD = DISCHARGE COEFFICIENT
C                       GC = GRAVITATIONAL ACCELERATION CONSTANT
C                      RHO = DENSITY OF HYDRAULIC FLUID
C        MU     = VISCOSITY OF HYDRAULIC FLUID, CENTIPOISE
C
C     THE METHOD OF SOLUTION UTILIZES THE TURBULENT ORIFICE EQUATION
C     AND THE EQUATION FOR FULLY-DEVELOPED LAMINAR FLOW THROUGH AN
C     ANNULUS, WITH FULL ECCENTRICITY ASSUMED.  FOR ORIFICE OPENINGS
C     WHERE SOME OVERLAPPED LENGTH EXISTS, THE PROCEDURE IS TO CALC-
```

```
C     ULATE THE FLOW RATE BY BOTH EQUATIONS, AND THEN USE THE ONE
C     THAT GIVES THE SMALLEST ABSOLUTE VALUE AS THE ANSWER.  FOR
C     OPENINGS WHERE NO OVERLAPPED LENGTH EXISTS, ONLY THE TURBULENT
C     ORIFICE EQUATION APPLIES.
C
      IMPLICIT REAL(L,M)
      Q12(T1,T2)=SIGN(1.,(T1-T2))*SQRT(ABS(T1-T2))
      Q34(T3,T4)=4.5E06*(T3-T4)*D*RCL**3/MU
C
C     ************************* CALCULATE  Q1 ************************
      X2=LAP1-X
      X4=SQRT(X2**2+RCL**2)
      IF(LAP1 .LE. 0.) GO TO 99
C
C     POSITIVE LAPS:
      IF(X .GE. LAP1) GO TO 65
      Q1=RCL*COEF1*Q12(PS,P1)
      Q1L=Q34(PS,P1)/X2
      IF(ABS(Q1L) .LT. ABS(Q1)) Q1=Q1L
      GO TO 20
   65 Q1=X4*COEF1*Q12(PS,P1)
      GO TO 20
C
C     NEGATIVE LAPS:
   99 IF(X .LT. LAP1) GO TO 10
      Q1=X4*COEF1*Q12(PS,P1)
      GO TO 20
   10 Q1=RCL*COEF1*Q12(PS,P1)
      Q1L=Q34(PS,P1)/X2
      IF(ABS(Q1L) .LT. ABS(Q1)) Q1=Q1L
C
C     ************************* CALCULATE Q3 ************************
   20 X1=LAP3+X
      X3=SQRT(X1**2+RCL**2)
      IF(LAP3 .LE. 0.) GO TO 199
C
```

```
C       POSITIVE LAPS:
        IF(X .LE. -LAP3) GO TO 165
        Q3=RCL*COEF3*Q12(P1,PR)
        Q3L=Q34(P1,PR)/X1
        IF(ABS(Q3L) .LT. ABS(Q3)) Q3=Q3L
        GO TO 120
  165   Q3=X3*COEF3*Q12(P1,PR)
        GO TO 120
C
C       NEGATIVE LAPS:
  199   IF(X .GT. -LAP3) GO TO 110
        Q3=X3*COEF3*Q12(P1,PR)
        GO TO 120


  110   Q3=RCL*COEF3*Q12(P1,PR)
        Q3L=Q34(P1,PR)/X1
        IF(ABS(Q3L) .LT. ABS(Q3)) Q3=Q3L
  120   RETURN
        END


        SUBROUTINE LIMITS(X,XDOT,XMAX,XMIN)
C
C          STATEMENTS FOR THIS SUBROUTINE OBTAINED BY PHONE BY   BD020177
C          JOHN R. MCGEHEE ON  2/1/77.                           BD020177
C
        IF(X .GE. XMAX) GO TO 10
        IF(X .LE. XMIN) GO TO 20
        GO TO 30
```

```
10 X=XMAX
   IF(XDOT .GT. 0.0) XDOT = 0.0
   GO TO 30
20 X=XMIN
   IF(XDOT .LT. 0.0) XDOT = 0.0
30 RETURN
   END
```

The ident CSCMODS contains the code to interface the active routines with the remainder of the program. In subroutine EXE, a call to the comdeck ACOBLK was inserted before the LOGICAL statement and the call to the landing gear routine was changed as follows.

```
    CALL LGEAR2
    IF (IABS(INDLG).NE.3) GO TO 594
    CALL ALGEAR1
    GO TO 596
594 CONTINUE
    CALL LGEAR3
596 CONTINUE
    CALL FLEX2
```

The following code was added to subroutine INUPD as the first executable statements to monitor changes in the number of integration variables.

```
    NNUM = NUM+N
    WRITE(6,600) NNUM
600 FORMAT(5X,32HNUMBER OF INTEGRATED VARIABLES =,I3)
```

To allow for a greater number of integration variables the common block LGDE was expanded in LGDET, LGEAR1, LGEA3C, and SDFLGP as follows.

```
COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
```

Subroutine LINES was changed as follows since 51 lines are too many for 8 1/2 inch paper.

```
IF(LONG.LE.41)RETURN
```

Subroutine OPT1 was changed by calling the active common block before the REAL statement, by adding the following EQUIVALENCE statement

```
EQUIVALENCE (DM155(133), INDLG)
```

to make the variable INDLG available and the call to the landing gear code was changed as follows.

```
2062 CALL SACS3
     IF (IABS(INDLG).NE.3) GO TO 2070
     CALL ALGEAR1
     GO TO 2075
2070 CONTINUE
2125 CALL LGEAR3
2075 CONTINUE
     CALL FLEX3
```

In subroutine LGEAR1, the number of integration variables was increased by adding the following line of code before the call to INUPD, and

```
IF (IABS(INDLG) .EQ. 3) NDEQ=10*NSTRUT
```

the active control initialization is accomplished by adding the following immediately before the statement labeled 3.

```
IF (IABS(INDLG).EQ.3) CALL ACTINIT
```

In subroutine LGEA3C, the following EQUIVALENCE

```
EQUIVALENCE (DM14(155),INDLG)
```

statement was added to make the variable INDLG available. The following test was added after

```
IF (IABS(INDLG).EQ.3) GO TO 31
```

the statement labeled 27 and the similar test below

```
IF (IABS(INDLG).EQ.3) GO TO 46
```

was added after the fifth line below the statement labeled 4. A call to the active gear common block was added to SDFLGP. Program output capabilities were expanded to include the active control parameters by including the following DIMENSION and DATA statements in subroutine SDFLGP.

```
      DIMENSION ACOVAR1(8), ACOVAR2(8), ACOVAR3(8), ACOVAR4(8),
     * ACOVAR5(8), ACOVAR6(8), ACOVAR7(8), ACOVAR8(8),ACOVAR9(8)
      DATA(ACOVAR1(I),I=1,8 ) /5HVOL1T,5HVOL2T,5HVOL3T,5HPGA1T,5HPGA2T,
     * 5HPGA3T,5HGAMAH,5HCOEFO/
      DATA(ACOVAR2(I),I=1,8 ) /2HQQ,4HVCUM,6HBLFORT,6HBUFORT,5HFFORT,
     * 6HFORCHT,6HFOAHST,6HFONHST/
      DATA(ACOVAR3(I), I=1,8) /5HCFFOR,5HFSTOP,6HFORSST,6HXVALVE,3HXSV,
     * 5HDELTX,6HDELTX1,6HDLTX1D/
      DATA(ACOVAR4(I), I=1,8) /2HDF,3HDP1,2HPS,2HP1,3HQSV,4HQSVN,4HQSV1,
     * 4HQSV3/
      DATA(ACOVAR5(I),I=1,8)/6HEPSILO,3HHMM,6HDMTANH,5HXSTOT,5HICOSV,
     * 4HENUP,5HWLFOR,5HIOPCO/
      DATA(ACOVAR6(I), I=1,8) /6HIIXSVH,6HIIXSVL,5HIXSVH,5HIXSVL,5HIPASS
     *,3HNAC,4HRESA,2HSA/
      DATA(ACOVAR7(I), I=1,8) /6HVELDEC,6HWLFORR,6HIDEACT,4HZSSC,4HCOPA,
     * 4HSIPA,5HDSTOP,6HFSTOPK/
      DATA(ACOVAR8(I), I=1,8) /5HWFORT,6HVOLAHT,6HVOLANT,5HQSVCU,3HXMA,
```

```
     * 4HXMA5,4HXMA8,5HXMA11/
      DATA(ACOVAR9(I), I=1,8) /6HXSVDDD,5HXSVDD,6HXSVDOT,2HPR,4HIFRI,
     * 6HIFSTOP,6HISTROK,4HISET/
```

The actual output is accomplished by the following code which was added three lines below the statement labeled 32.

```
      IF(IABS(INDLG).NE.3) GO TO 50
      CALL STFL(2,8,ACOVAR1)
      DO 33 I=1,NSTRUT
      CALL STOVAR(8,VOL1T(I),VOL2T(I),VOL3T(I),PGA1T(I),PGA2T(I),
     * PGA3T(I),GAMAH(I),COEFO(I))
   33 CONTINUE
      CALL STFL(2,8,ACOVAR2)
      DO 34 I=1,NSTRUT
      CALL STOVAR(8,OO(I),VCUM(I),BLFORT(I),BUFORT(I),FFORT(I),
     * FORCHT(I),FOAHST(I),FONHST(I))
   34 CONTINUE
      CALL STFL(2,8,ACOVAR3)
      DO 35 I=1,NSTRUT
      CALL STOVAR(8,CFFOR(I),FSTOP(I),FORSST(I),XVALVE(I),XSV(I),
     * DELTX(I),DELTX1(I),DLTX1D(I))
   35 CONTINUE
      CALL STFL(2,8,ACOVAR4)
      DO 36 I=1,NSTRUT
      CALL STOVAR(8,DF(I),DP1(I),PS(I),P1(I),QSV(I),QSVN(I),QSV1(I),
     *QSV3(I))
   36 CONTINUE
      CALL STFL(2,8,ACOVAR8)
      DO 37 I=1,NSTRUT
      CALL STOVAR(8,WFORT(I),VOLAHT(I),VOLANT(I),QSVCU(I),XMA(I),
     * XMA5(I),XMA8(I),XMA11(I))
   37 CONTINUE
      CALL STFL(2,8,ACOVAR9)
      DO 38 I=1,NSTRUT
      CALL STOVAR(8,XSVDDD(I),XSVDD(I),XSVDOT(I),PP(I),FLOAT(IFRI(I)),
     * FLOAT(IFSTOP(I)),FLOAT(ISTROK(I)),FLOAT(ISET(I)))
   38 CONTINUE
      CALL STFL(2,8,ACOVAR5)
      DO 39 I=1,NSTRUT
      CALL STOVAR(8,EPSILO(I),HMM(I),DMTANH(I),XSTOT(I),FLOAT(ICOSV(I)),
     * ENUP(I),WLFOR(I),FLOAT(IOPCO(I)))
   39 CONTINUE
      CALL STFL(2,8,ACOVAR6)
      DO 40 I=1,NSTRUT
      CALL STOVAR(8,FLOAT(IIXSVH(I)),FLOAT(IIXSVL(I)),FLOAT(IXSVH(I)),
```

```
     * FLOAT(IXSVL(I)),FLOAT(IPASS(I)),
     * FLOAT(NAC(I)),RESA(I),SA(I))
  40 CONTINUE
     CALL STFL(2,8,ACOVAR7)
     CALL STOVAR(8,VELDEC,WLFORR,FLOAT(IDEACT),ZSSC,COPA,SIPA,DSTOP,
     * FSTOPK)
  50 CONTINUE
```

The active control variables are placed in the proper arrays for subsequent numerical integration by the following code which was inserted in SDFLGP three lines below the statement labeled 5.

```
IF(IABS(INDLG) .NE. 3) GO TO 6
I1 = I+3*NSTRUT
I2 = I+4*NSTRUT
I3 = I+5*NSTRUT
CALL UPDAT(1,LA(I1),PGAIT(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I2),VCUM(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I3),OSVCU(I),DU,DU,DU,DU)
I1 = 3*I+6*NSTRUT-2
I2 = I+9*NSTRUT
CALL UPDAT(3,LA(I1),XSVDD(I),XSVDOT(I),XSV(I),DU,DU)
CALL UPDAT(1,LA(I2),DELTX1(I),DU,DU,DU,DU)
```

The following code was added immediately after the statement labeled 16 in subroutine READ to correct an initialization problem revealed by an operating system release which does not preset memory to zero.

```
SLTSYM = 0.0
IBC = 0
INXQ= 0
JBC = 0
```

For the same reason, the following code was added below the statement labeled 2 in subroutine AUTS.

```
DELPI = 0.0
TR = 0.0
```

The next phase in the development of the active control capability was the interface of the active code with the input and graphic output modules of the FATOLA program. This was accomplished with the correction sets GMMODS, JMMODS, DIRACT, and ACTIN. A new deck ACTIN was inserted after the deck READ and a new deck DIRACT was added following the deck DIR3DA. The contents of these new decks is shown below.

```
      SUBROUTINE ACTIN
      COMMON/ACTDIR/XNAME(71),LOC(71)
      COMMON/ACTIVE/DATA(646)
      DIMENSION IRA(55),MSG(58),IDATA(646)
      EQUIVALENCE (MSG(1),SYM),(MSG(2),OP)
      EQUIVALENCE (MSG(3),IRA(1)),(MSG(58),INC)
      EQUIVALENCE (DATA(1),IDATA(1))
      INTEGER COMMA,POINT,E,BLANK
      DATA REMARK,COMMA,BLANK,POINT,E,ENDACT,MINUS,AINT/
     1 6HREM,1R,,1R ,1R.,1RE,6HENDACT,1R-,3HINT/
  100 CONTINUE
    1 FORMAT(A6,1X,A3,1X,55R1,I1)
    2 FORMAT(18X,A6,1X,A3,1X,55R1,I6)
    3 FORMAT(20HOERROR.THE SYMBOL **,A6,
     1 26H** IS NOT IN THE DIRECTORY/1H )
    4 FORMAT(44HOERROR.ILLEGAL CHARACTER IN NUMERIC FIELD **,1R1,2H**/)
      READ(5,1) SYM,OP,IRA,INC
      CALL LINES(1)
      J = 58
      IF(INC.LE.0) J = 57
      WRITE(6,2) (MSG(I),I=1,J)
      IF(SYM.EQ.REMARK) GO TO 100
      IF(SYM.EQ.ENDACT) GO TO 999
      DO 110 I=1,71
      IF(SYM.EQ.XNAME(I)) GO TO 120
  110 CONTINUE
      CALL LINES(3)
      WRITE(6,3) SYM
      GO TO 100
  120 CONTINUE
      INDEX = LOC(I)
      IF(INC.EQ.0) INC = 1
      INDEX = INDEX + INC - 1
      NUMEXP = 0
      NEXP = 0
      IEXP = 0
      NL = 0
      NR = 0
```

```
      NUML = 0
      NUMR = 0
      ISIGN = 0
      JSIGN = 0
      LEFT = 1
      DO 210 I=1,56
      IF(I.EQ.56) GO TO 140
      IF(IRA(I).EQ.BLANK) GO TO 210
      IF(IRA(I).EQ.COMMA) GO TO 140
      IF(IRA(I).EQ.POINT) GO TO 170
      IF(IRA(I).EQ.E) GO TO 180
      IF(IRA(I).EQ.MINUS) GO TO 200
      IF(IRA(I).GT.36) GO TO 130
      IF(IRA(I).LT.27) GO TO 130
      NUM = IRA(I) - 27
      IF(IEXP.EQ.1) GO TO 190
      IF(LEFT.GT.0) NUML = 10*NUML + NUM
      IF(LEFT.GT.0) NL = NL + 1
      IF(LEFT.LT.0) NUMR = 10*NUMR + NUM
      IF(LEFT.LT.0) NR = NR + 1
      GO TO 210
  130 CALL LINES(3)
      WRITE(6,4) IRA(I)
      GO TO 210
  140 CONTINUE
      IF(NL.EQ.0.AND.NR.EQ.0) GO TO 210
      IF(NR.EQ.0) GO TO 160
      X = FLOAT(NUML) + FLOAT(NUMR)/10.**NR
      IF(JSIGN.EQ.1) NUMEXP = -NUMEXP
      IF(IEXP.EQ.1) X = X*(10.)**NUMEXP
      IF(ISIGN.EQ.1) X = -X
      DATA(INDEX) = X
  150 NUML = 0
      NUMR = 0
      NL = 0
      NR = 0
      LEFT = 1
      ISIGN = 0
      JSIGN = 0
      IEXP = 0
      NEXP = 0
      NUMEXP = 0
      INDEX = INDEX + 1
      GO TO 210
  160 CONTINUE
      X = NUML
      IF(JSIGN.EQ.1) NUMEXP = -NUMEXP
      IF(IEXP.EQ.1) X = X*(10.)**NUMEXP
      IF(ISIGN.EQ.1) X = -X
      NUML = X
      IF(OP.EQ.AINT) IDATA(INDEX) = NUML
```

```
      IF(OP.NE.AINT) DATA(INDEX) = X
      GO TO 150
  170 CONTINUE
      LEFT = -1
      GO TO 210
  180 CONTINUE
      IEXP = 1
      GO TO 210
  190 CONTINUE
      NUMEXP = 10*NUMEXP + NUM
      NEXP = NEXP + 1
      GO TO 210
  200 CONTINUE
      IF(IEXP.EQ.0) ISIGN = 1
      IF(IEXP.NE.0) JSIGN = 1
  210 CONTINUE
      GO TO 100
  999 RETURN
      END
```



```
      BLOCK DATA DIRACT
      COMMON/ACTDIR/NAME(71),LOC(71)
      DATA NAME/  6HAMUH  ,  6HAPINT ,  6HAREA1 ,  6HAREA2 ,
     1 6HAREA3 ,  6HAREMO ,  6HAREO3 ,  6HBETA  ,  6HBLMU  ,
     2 6HBUMU  ,  6HCDMOC ,  6HCDMOE ,  6HCDSV  ,  6HCD3   ,
     3 6HCFFOR ,  6HDELT  ,  6HDIOTA ,  6HDSV   ,  6HEPSILO,
     4 6HEPSROL,  6HEPSSLP,  6HETASV ,  6HGAMAH ,  6HGAMAN ,
     5 6HGNR   ,  6HIDEACT,  6HKAPT  ,  6HOMRUN ,  6HPATM  ,
     6 6HPGAHAC,  6HPGALAC,  6HPGA1I ,  6HPGA2I ,  6HPGA3I ,
     7 6HPERCNT,  6HOPUMPS,  6HRCLSV ,  6HRHOH  ,  6HTAUF  ,
     8 6HTC1   ,  6HTC2   ,  6HTC3   ,  6HTC4   ,  6HVOLACI,
     9 6HVOLANI,  6HVOL1I ,  6HVOL2I ,  6HVOL3I ,  6HWC     ,
     1 6HWC1   ,  6HWSV   ,  6HWSV1  ,  6HWSV3  ,  6HWLFOR ,
     2 6HWLFORR,  6HXBIAS ,  6HXDDMAX,  6HXDDMIN,  6HXKA    ,
     3 6HXKF   ,  6HXKSV  ,  6HXLPSV1,  6HXLPSV3,  6HXSCOM ,
     4 6HXSTHR ,  6HXSVDMN,  6HXSVDMX,  6HXSVMAX,  6HXSVMIN,
     5 6HZETAC1,  6HZETAC2/
      DATA LOC/   1,   6,  16,  21,  26,  31,  36,  61,  46,
     1       56,  62,  67,  72,  82,  77, 109, 141, 135, 152,
     2      157, 162, 163, 195, 200, 201, 212, 263, 274, 275,
     3      281, 286, 291, 301, 311, 276, 346, 387, 397, 410,
     4      411, 412, 413, 414, 621, 631, 421, 431, 441, 451,
     5      452, 641, 642, 643, 458, 463, 464, 611, 616, 474,
     6      479, 484, 489, 494, 561, 499, 591, 596, 601, 606,
     7      644, 645/
      END
```

The GMMODS correction set expanded the dimensions of the TITLE, BUF, TBUF, and BMM arrays in the PLTDAT program as illustrated below.

```
      DIMENSION TITLE(20),BUF(550),NDIL(28),TBUF(550)
     1 BMM(2,550),CMMDS(6),DEPVAR(5),LINE(7),NDVA(5)
```

The JMMODS correction set interfaced the new decks with the FATOLA program. All DATA statements in ACTINIT were replaced with the single DATA statement

```
      DATA ICOSV,IFSTOP,II/11*0/
```

Subroutine SDFLGP was modified to output active control parameters to TAPE13, the disk file which communicates with PLTDAT to generate graphic output. The DATA statement

```
      DATA N19/19/
```

was inserted in this routine and the write statement below the statement labeled 115 was replaced with:

```
      IF(ISUM1.NE.0) WRITE(13) N19,ISUM1,DAT3,OP17,ACOVAR8(1),
     1 ACOVAR8(4),ACOVAR4(5),ACOVAR1(4),ACOVAR1(5)
```

Additionally, the final continuation card in each of the tests on ISTPL1, ISTPL2, ISTPL3, ISTPL4, and ISTPL5 were replaced with the following statements respectively.

```
     1 OMET(1,1),WFORT(1),OSVCU(1),OSV(1),PGA1T(1),PGA2T(1)
     1 OMET(1,2),WFORT(2),OSVCU(2),OSV(2),PGA1T(2),PGA2T(2)
     1 OMET(1,3),WFORT(3),OSVCU(3),OSV(3),PGA1T(3),PGA2T(3)
```

```
     1 OMET(1,4),WFORT(4),QSVCU(4),QSV(4),PGA1T(4),PGA2T(4)
     1 OMET(1,5),WFORT(5),QSVCU(5),QSV(5),PGA1T(5),PGA2T(5)
```

The data statement

```
      DATA ACTIVE/6HACTIVE/
```

was added to subroutine READ and statement 19 was replaced with the following.

```
   19 IF(SYM.EQ.ACTIVE) GO TO 805
      CALL DIPLAC(RA1,INC,BLANK)
```

The following code was inserted after the STOP 25

```
  805 CALL ACTIN
      GO TO 100
  810 CALL ACTIN
      GO TO 802
```

and statement 26 was replaced with the following.

```
   26 IF(SYM.EQ.ACTIVE) GO TO 810
      IF(SYM.EQ.STCASE) GO TO 21
```

The CSCMOD2 correction set contained general program corrections and enhancements with emphasis on the active control code. In the ACOBLK comdeck, the ACTIVE common block was modified by replacing FOAHST(5) with FWORK(5), replacing PGA1T(5) with PGA1T1(5), replacing ZZSSC with ZSSC(5), and by adding IMODE(5), CMASNG, and VMASS(5). The common block HTCOM was expanded to

include INDINT(5), in EXE, MIMIN, LGDET, LGEAR1, ACTINIT, ALGEAR, and FLEX1. Subroutine EXE was further modified by adding

```
STOP "FLIGHT TIME LIMIT"
```

to lines below the statement labeled 743. Three lines of code

```
C
      DO 10 I=1,5
   10 INDINT(I) = 1
```

were inserted as the first executable statements in subroutine MIMIN. In subroutine LGEAR1, the following three lines of code were added as the last executable statements in the LGEAR1 entry point, (pre data initialization).

```
      DO 6 I=1,5
         P(I) = 0.0
    6    P2(I) = 0.0
```

The single DATA statement in ACTINIT (inserted by JMMODS) was replaced with the following code,

```
      EQUIVALENCE (DM15(16),GREFF), (DM1(37),AIYYRS)

      DATA DSTOP,FSTOPK /0.004,0.0/
      DATA ENUP,FSTOP,HMM,QSVN,RESA,SA /30*0.0/
      DATA ICOSV,IFRI,IFSTOP,IPASS /20*0/
      DATA IOPCO,ISET /10*1/
      DATA IXSVH,IXSVL,IIXSVH,IIXSVL /20*0/
C
```

the definition of OMRUN was changed to,

```
OMRUN=ERDEG*0.01745329
```

and the following definitions were inserted in the 100 loop.

```
INDINT(I) = 1
VMASS(I)=AMASS
```

Additionally, the line below statement number 80 was changed to

```
PGA1T1(I)=PGA1I(I
```

In order to make the variable GAMA available to the subroutine ALGEAR1, the variable DUM15(13) in the DIRCOM common block was replaced by GAMA, DUM15(12). The following lines of code were removed

```
EQUIVALENCE (DM5(16),GREFF)
DATA IFRI/5*0/,FSTOPK/0.0/,FSTOP/5*0.0/,DSTOP/0.004/
DATA SA,RESA,HMM,IXSVL,IXSVH,IIXSVL,IIXSVH,IPASS
1     /15*0.,25*0/
DATA ENUP/5*0.0/
DATA ISET,IOPCO,OSVN/10*1,5*0.0/
```

and were replaced with

```
EQUIVALENCE (DM5(16),GREFF), (DM2(27),AXP7F), (DM2(28),AX77F),
*(DM2(29),AYP7F),(DM2(30),AY77F),(DM2(31),AZP7F),(DM2(33),AZ77F),
*(DM1(37),AIYY8S)
DIMENSION INDEACT(5),IPSTOP(5),AIC(5),PGA1T(5)
DATA AIC,INDEACT,IPSTOP /5*0.0,10*0/
```

The following new code was inserted beneath statement number 760.

```
C       CMASNG = AIYYBS/AMASS+AMASS*PX(1)*PX(1)/AIYYBS
        CMASNG = 7.9677
        VMASS(1) = AMASS/CMASNG
        ENCG = 0.5*AMASS*ZG77F1(1)*ZG77F1(1)
        ZDANT = ZG77F1(1)-QI77R*PX(1)
```

The following two lines of code

```
  18  IF(IDEACT.EQ.1) GO TO 56
  19  IF(IDEACT.EQ.2) GO TO 80
```

were replaced with

```
C
   1? DO 90 I=1,NSTRUT
      IF(INDEACT(I).EQ.1) GO TO 56
      IF(INDEACT(I).EQ.2) GO TO 80
```

The following lines of code were added after the test on OMRUN and after statement number 25, respectively.

```
IF(I.EQ.1 .AND. ZDANT.GE.VLDEC) GO TO 90
IF(I.EQ.1 .AND. ZDANT.GE.VELDEC+XG77F1(1)*TAN(OMRUN)) GO TO 90
```

The reduction of the control limit force was modified by replacing the three lines of code centered about statement number 56 with the following.

```
      INDEACT(I)=1
   56 IF(INDINT(I).EQ.0) GO TO 58
      WLFOR(I)=WLFOR(I)-REDSLP(I)*HT
      EPSILO(I)=EPSILO(I)+EPSSLP*HT
   58 CONTINUE
      INDINT(I)=0
```

In the line of code IDEACT = 2, IDEACT was changed to INDEACT(I). The four lines of code beginning the 65 loop were replaced with the following.

```
C*** CALCULATION OF THE WING-GEAR INTERFACE FORCE (WFORT)
      UNSPRNG = 0.0
       DO 66 J=1,NSTRUT
       IF(OMETD1(J).NE. 0.0) UNSPRNG = UNSPRNG+MASS(J)
   66 CONTINUE
       DWFORT =(-SQRT(AXP7F*AXP7F+AYP7F*AYP7F+
     &      AZP7F*AZP7F)+GREFF)*(AMASS-UNSPRNG)
      WFORT(1) = DWFORT+FORSST(1)
      DO 67 I=2,NSTRUT
           WFORT(I) = DWFORT/(NSTRUT-1)
   67 CONTINUE
C
      DO 65 I=1,NSTRUT
C
C***
```

The line below the statement labeled 210 was replaced with

```
      PGA1T(I)=PGA1T1(I)
      IF(PGA1T1(I).LE.-1600.0) PGA1T(I)=-1600.0
```

and GAMAN was replaced with GAMA in the definition of PGA2T(I). The two lines of code ending with the definition of QO were replaced with the following.

```
      IF(IMODE(I).EQ.0 .AND. DDELTA(I).LE.0.0) 112,113
  112 QO(I) = 0.0
      GO TO 109
C
  113 QO(I)=COEFO(I)*(AREMO(I)-APINT(I))*01(PGA1T1(I),PGA2T(I))
      IF(QO(I).LT.0.0 .AND. VCUM(I).LE.0.0) GO TO 102
      AIC(I)=0.0
      GO TO 103
  102 IF(PGA1T1(I).LT.PGA2T(I)) GO TO 103
      GO TO 111
  111 QO(I)=0.0
      VCUM(I)=0.0
      AIC(I)=1.0
  103 IF(QO(I).GT.0.0) AIC(I)=0.0
```

The test on I below statement number 150 was changed to

```
      IF(IMODE(I).EQ.0) GO TO 297
```

and the debugging change to 295 below this was changed back to 289. Statement number 284 was replaced with the following.

```
284   IF(S(1,I).LT.0.0) GO TO 160
      GO TO 161
160   IPSTOP(I)=1
161   IF(S(1,I).LE.DSTOP .AND. IPSTOP(I).EQ.1) GO TO 900
```

In statement number 900, the limiting value for the stroke was changed from 0.0001 to 0.005 and the five lines of code from DP1(I)=0.0 to II=0 were removed. Statement number 901 was replaced with the following code.

```
901   IF(PGA1T(I).LE.(PGA1I(I)+500.0) .AND.
     :   PGA1T(I).GT.(PGA1I(I)-500.0)) GO TO 158
      GO TO 159
158   IF(ABS(FT(I)).LE.FORCHT(I) .AND. S(1,I).EQ.0.0) GO TO 500
159   IF(S(1,I).GE.0.0) GO TO 470
```

The test on I below statement 500 was changed to

```
      IF(IMODE(I).EQ.0) GO TO 297
```

and the branch to 295 for debugging purposes was removed. The following code was inserted after statement number 295.

```
297 I2 = 2*I+NSTRUT-1
    I1 = I2+1
    CALL INTEG(LA(I2),SD2(1,I))
    CALL INTEG(LA(I1),SD1(1,I))
C
    IF(IMODE(I).EQ.0) GO TO 450
```

The two-line definition of ENUP(I) was expanded to

```
    IF(I.NE.1) GO TO 119
    ENUP(1) = 0.5*AIYYBS*QI77R*QI77R+(ENCG/CMASNG)*
   *      (ZG77F1(1)/ABS(ZG77F1(1)))
    GO TO 120
119 ENUP(I) = ENCG/(NSTRUT-1)
120 CONTINUE
```

and the second test on WFORT(I) below statement 120 was changed to the following.

```
IF(WFORT(I).GE.0.0 .OR. DDELTA(I).LT.0.0) XSTOT(I)=1.E20
```

The two lines of code beginning with the definition of ZSSC, the definition of VELDEC, and the definition of PS(I) were changed as follows.

```
ZSSC(I)=FWORK(I)*SB(I)
IF(XSTOT(I).LE.(ZSSC(I)-S(1,I)) .OR. RESA(I).EQ.1.0) SA(I)=1.0

VELDEC=((WLFOR(I)+WLFORR)/2.*(WLFOR(I)-WLFORR))/
&    (AMASS*REDSLP(I))
PS(I)=(((PGAHAC(I)+PATM)*(VOLANI(I)/VOLANT(I))**GAMA)-PATM)/144.0
```

The definitions of QSV1(1) and QSV3(1) below statement number 450 were removed and the following new code was inserted in place of the

six lines below statement 274.

```
I4 = I+4*NSTRUT
I5 = I+5*NSTRUT
CALL INTEG(LA(I4),QD(I))
CALL INTEG(LA(I5),QSV(I))
I9 = I+9*NSTRUT
CALL INTEG(LA(I9),DLTX1D(I))
```

The four lines of code below statement 55 were removed to make up for the insertion at statement number 297. The test on the strut velocity at the top of the 28 loop was removed and the test on PGA1T(I) and the following two lines were replaced with the following.

```
      IF(IMODE(I).EQ.0 .AND. DDELTA(I).LE.0.0) GO TO 19
      IF(S(1,I).NE.0.0 .OR. AIC(I).NE.1.0) GO TO 20
      PGA1T1(I)=PGA1I(I)
      PGA1T(I) =PGA1I(I)
   19 DP1(I)=0.0
```

The eight lines of code below the last call to LIMITS were replaced with the following code.

```
I3 = I+3*NSTRUT
CALL INTEG(LA(I3),DP1(I))
I9 = I +9*NSTRUT
CALL INTEG(LA(I9),DLTX1D(I))
I6 = 3*I+6*NSTRUT-2
I7 = I6+1
I8 = I6+2
CALL INTEG(LA(I6),XSVDDD(I))
CALL INTEG(LA(I7),XSVDD(I))
CALL INTEG(LA(I8),XSVDT(I))
```

In subroutine SDFLGP, the DATA statements for ACOVAR1 and ACOVAR2 were modified by substituting PGA1T1 for PGA1T and VMASS for FOAHST,

respectively. The five WRITE statements to unit 13, the call to STOVAR, and the call to UPDAT were also changed by substituting PAG1T1 for PGA1T, and the call to STOVAR for FOAHST was changed to VMASS. In subroutine ACTIN the size of the ACTIVE common block and the IDATA array were expanded from 646 to 656. In subroutine PACK, the size of the I1 array was changed from 1 to 6, a change which has no impact on program execution. Lastly, in BLOCK DATA DIRACT, the 4th, 5th, and 14th continuation lines in the DATA statement for NAME were changed to the following.

```
4 6HEPSROL,  6HEPSSLP,  6HETASV ,  6HFWORK ,6HGAMAH ,
5 6HGNR   ,  6HKAPT  ,  6HOMRUN ,  6HPATM  ,

5 6H7ETAC1,  6H7ETAC2,  6HIMODE /
```

The 2nd and 7th continuation lines in the DATA statement for LOC were accordingly changed as follows.

```
2     157, 162, 163, 170, 195, 201, 263, 274, 275,
7     644, 645, 651/
```

The EOR correction set is empty and does nothing.

The EORPL correction set inserted a new deck, EORPL, after subroutine CTENGL. This deck does nothing except write a record mark on the COMPILE file such that those decks following it, PLTDAT and FIND, are not ordinarily processed by the compiler.

The CSCMOD3 correction set made several modifications to the program, primarily to the active code. In the comdeck ACOBLK, the variable ZDANT was added to the end of the ACTIVE common block. The labeled common blocks TABDIR, READ1, UPDCAL, LGDE, STGT, TABCOM, CLEAUP, STORA, LGE, and ACTDIR were added to subroutine EXE, although the reason for these additions is unclear. None of the variables in these common blocks is required by EXE. In subroutine DEF, the variable NCASE was set to blanks with a DATA statement preceeding the FORMAT. This was done to clean up the printout on the first page of output which is printed before NCASE is defined. A DATA statement was added to subroutine LGEAR1 to provide initial values of zero to the arrays PGA1T1, PGA2T, QSV, QSVCU, and WFORT. A call to the comdeck ACOBLK was added to subroutine LGEA3C to make the ACTIVE common block available. In subroutine ALGEAR, the definitions of CMASNG, provided in CSCMOD2, were changed to the following.

```
CMASNG = 1+(AMASS*RX(1)*RX(1))/AIYYRS
```

Immediately prior to the definition of UNSPRNG, the variable UNSPR was set to zero, and the test inside the 66 loop, introduced by CSCMOD2, was replaced with the following.

```
IF(J.EQ.1 .AND. OMETD1(1).NE.0.0) UNSPR = MASS(1)
IF(J.GT.1 .AND. OMETD1(J).NE.0.0) UNSPRNG = UNSPRNG+MASS(J)
```

Below this, the definition of WFORT(1) was changed as follows.

```
      WFORT(1) = SR(1)*(VMASS(1)-UNSPR)
```

In the continuation line of the definition of VELDEC, also provided by CSCMOD2, the variable AMASS was replaced with VMASS(1). In subroutine SDFLGP, the DATA statement for ACOVAR7, introduced by CSCMOD5, was changed by replacing the 6HIDEACT with 5HZDANT and the first DATA statement for N1, N15, and N14 was removed. The following code was added beneath the write of N18, N1, and DAT2 to unit 13,

```
      IF(IABS(INDLG).NE.3) 110,115
  110 IF(ISUM1.NE.0) WRITE(13) N14,ISUM1,DAT3,OP17
      GO TO 120
  115 CONTINUE
```

and a CONTINUE statement labeled 120 was added after the write of ACOVAR3(3). The following code was inserted after the write of ETADES,

```
  210 CONTINUE
      IF(ISTPL1.NE.0) WRITE(13) FT(1),SF(1),DELTA(1),P(1),P2(1),MA(1),
     *SD2(1,1),SD1(1,1),S(1,1),S2D2(1,1),S2D1(1,1),S2(1,1),OMETD1(1,1),
     *OMET(1,1)
      IF(ISTPL2.NE.0) WRITE(13) FT(2),SF(2),DELTA(2),P(2),P2(2),MA(2),
     *SD2(1,2),SD1(1,2),S(1,2),S2D2(1,2),S2D1(1,2),S2(1,2),OMETD1(1,2),
     *OMET(1,2)
      IF(ISTPL3.NE.0) WRITE(13) FT(3),SF(3),DELTA(3),P(3),P2(3),MA(3),
     *SD2(1,3),SD1(1,3),S(1,3),S2D2(1,3),S2D1(1,3),S2(1,3),OMETD1(1,3),
     *OMET(1,3)
      IF(ISTPL4.NE.0) WRITE(13) FT(4),SF(4),DELTA(4),P(4),P2(4),MA(4),
     *SD2(1,4),SD1(1,4),S(1,4),S2D2(1,4),S2D1(1,4),S2(1,4),OMETD1(1,4),
     *OMET(1,4)
      IF(ISTPL5.NE.0) WRITE(13) FT(5),SF(5),DELTA(5),P(5),P2(5),MA(5),
     *SD2(1,5),SD1(1,5),S(1,5),S2D2(1,5),S2D1(1,5),S2(1,5),OMETD1(1,5),
     *OMET(1,5)
      GO TO 230
  220 CONTINUE
```

and a CONTINUE statement labeled 230 was added after the write of FORSST(5). Finally, in the call to STOVAR, introduced by CSCMODS, the variable FLOAT(IDEACT) was replaced by ZDANT.

The CSCMOD4 correction set, combined with a new deck, ACTNG, removed the nose gear active code from ALGEAR so that the nose gear could be treated independently from the main gears. Other minor changes were made to the active code as well as other parts of the program. The new deck was added after the ALGEAR deck and contained the following.

```
      SUBROUTINE ACTNG
C
C********************* FATOLA VARIABLES *********************************
C
      COMMON/DIRCOM/DM1(115),ALPHD,DM1A(20),AMASS,DM2(147),DCL1,DCM1,
     C DCN1,DCL2,DCM2,DCN2,DCL3,DCM3,DCN3,DM3(99),FXB7P,
     C DUM4(3),FYB7P(4),FZB7P,DM5(17),GXB7F,DM6(8),GZB7F,
     C DM7(218),INDSTE(48),PHIPD,INDSTE1(23),PSIPD,INDSTE2(156),THTPD,
     C INDSTE3(5),TIME,DM8(287),PI77R(2),PI77R1(2),DM9(4),
     C QI77R(2),QI77R1(2),DM10(4),RI77R(2),RI77R1(2),DM11(48),
     C XG77F(2),XG77F1(12),YG77F(2),YG77F1(12),
     C ZG77F(2),ZG77F1(2),DUM13(52),
     C NSTRUT,MASS(5),RX(5),RY(5),RZ(5),THETAD(5),ERDEG,RGR,
     C NTIRES(5),RZERO(5),W(5),DELTAM(5),MOMENT(5),
     C RF(5),VZ  ,IFD,PZERO(5),VZERO(5),A(5),P20(5),V20(5),
     C A2(5),IL,S2T(5),ES2(5),C2L(5),MASS2(5),MUS(5),
     C CC(5),CE(5),C2C(5),C2E(5),NVGPT,NPP,MB(5),PLT,NDELTA,
     C ES(5),SB(5),SD21(2),SD22(2),SD23(2),SD24(2),SD25(2)
C
      COMMON/DIPCOM/
     C SD11(2),SD12(2),SD13(2),SD14(2),SD15(2),
     C S1(2),SS2(2),S3(2),S4(2),S5(2),
     C S2D21(2),S2D22(2),S2D23(2),S2D24(2),S2D25(2),
     C S2D11(2),S2D12(2),S2D13(2),S2D14(2),S2D15(2),
     C S21(2),S22(2),S23(2),S24(2),S25(2),
     C OMTD11(2),OMTD12(2),OMTD13(2),OMTD14(2),OMTD15(2),
     C OMT1(2),OMT2(2),OMT3(2),OMT4(2),OMT5(2),
     C AI(5),BI(5),DELTA1,DELTA2,DELTA3,DELTA4,DELTA5,
     C DDELT1,DDELT2,DDELT3,DDELT4,DDELT5,ISTAGE,
     C PRTMIN,IPLT,ISDF,ISTPL1,ISTPL2,ISTPL3,ISTPL4,ISTPL5,
     C DM14(22),IB(5),DM15(127),INDLG,DM16(107),CASK(44),INDFLX,
```

```
     * NMODE,DM18(40),SXMOD(100),SYMOD(100),SZMOD(100),DM19(1686),
     * GOD2(20),DDM20(20),DDM21(20),SLEN1(5),SLEN2(5),
     * GAMA,DUM15(12),INDNWS,DDM22(10),ETADES,
     * DDM23(5),AH(5),PH(5),DDM24(30)
C
      COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
C
      COMMON/LGF/A11(5),A13(5),A31(5),A33(5),RRCGX(5),
     C RL(3,3),RI(3,3,5),RAX(5),RAY(5),RAZ(5),TMP(3),ZZERO(5),
     C XR(5),YR(5),EPSLON(5),PA(5),FDELTA(5),
     C FTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
     C VTX(5),VTY(5),VTZ(5),GZ(5),      VGPT(5),FTRX(5),FTRY(5),
     C DX(5),DY(5),DZ(5),FT(5),FDX(5),FDY(5),FF(5),AA(5),C2(5),
     C SR(5),SF(5),PSKD(5),MUVP(5),MTRX(5),MTRY(5),
     C MTRZ(5),MA(5),RG11,RG13,RG31,RG33,IPRT,
     C MTX,MTY,MTZ,SFTRX,SFTRY,SFTRZ,FTRA,
     C FTRB,FTRC,SMTRX,SMTRY,SMTRZ
C
      COMMON /HTCOM/ HT, HT1, HT2, INDINT(5)
C
*CALL ACOBLK
C
      REAL MASS,MOMENT,MASS2,MUS,NTIRES,MB
      REAL MUVP,MTRX,MTRY,MTRZ,MA,MTX,MTY,MTZ
C
      DIMENSION IPSTOP(5),AIC(5),PGA1T(5)
      DIMENSION DELTA(5),DDELTA(5),DLGDE(47),
     *      SD2(2,5),SD1(2,5),S(2,5)
C
      EQUIVALENCE (DLGDE(1),LA(1))
      EQUIVALENCE (DELTA(1),DELTA1),(DDELTA(1),DDELT1)
      EQUIVALENCE (SD21(1),SD2(1,1)),(SD11(1),SD1(1,1)),(S1(1),S(1,1))
      EQUIVALENCE (DM15(1),ITO)
      EQUIVALENCE (DM5(16),GPEFF),
     * (DM1(37),AIYYBS)
C
      DATA AIC,IPSTOP /5*0.0,5*0/
C
      Q1(T1,T2)=SIGN(1.,(T1-T2))*SQRT(ABS(T1-T2))
C*****************************************************************************
C
      I = 1
C
      IF(INDEACT(I).EQ.1) GO TO 56
      IF(INDEACT(I).EQ.2) GO TO 80
      IF(HMM(I) .EQ. 0.) GO TO 90
      IF(OMRUN .GT. 0.0)GO TO 25
      IF(ZDANT.GE.VELDEC) 90,40
   25 IF(ZDANT.GE.VELDEC+XG77F1(1)*TAN(OMRUN)) GO TO 90
  .40 WRITE(6,1014)TIME
1014  FORMAT (1H036H REDUCE CONTROL LIMIT FORCE AT TIME=,E16.8)
      INDEACT(I)=1
   56 IF(INDINT(I).EQ.0) GO TO 58
      WLFOR(I)=WLFOR(I)-PEDSLP(I)*HT
      EPSILO(I)=EPSILO(I)+EPSSLP*HT
```

```
   58 CONTINUE
      INDINT(I)=0
      IF(WLFOP(I) .GT. WLFORR) GO TO 90
      WRITE(6,1015)TIME
 1015 FORMAT (1H027H CONTROL AT WLFORR AT TIME=,F16.8)
      INDEACT(I)=2
   80 WLFOR(I)=WLFORR
      EPSILO(I)=EPSROL(I)
   90 CONTINUE
C*****************************************************************************
C
      IF(KAPT(I).NE.0)GO TO 210
      APINT(I)=0.0
  210 CONTINUE
      PGA1T(I)=PGA1T1(I)
      IF(PGA1T1(I).LE.-1600.0) PGA1T(I)=-1600.0
      VOL1T(I)=VOL1I(I)-(AREA1(I)-APINT(I))*S(1,I)
      VOL3T(I)=VOL3I(I)+AREA3(I)*S(1,I)
      VOL2T(I)=VOL2I(I)-(AREA2(I)-AREA1(I)+APINT(I))*S(1,I)+(VOL3T(I)
     X -VOL3I(I))-VCUM(I)
      PGA2T(I)=AP2TO(I)*((VOL2I(I)/VOL2T(I))**GAMA)-PATM
      IF(SD1(1,I) .EQ. 0.0)GO TO 104
      PGA3T(I)=((COEF3(I)*AREO3(I))**2*PGA2T(I)-SD1(1,I)/ABS(SD1(1,I))
     X *(SD1(1,I)*AREA3(I))**2)
     E/((COEF3(I)*AREO3(I))**2)
      GO TO 105
  104 PGA3T(I)=PGA2T(I)
  105 IF(PGA1T(I) .GE. PGA2T(I))GO TO 106
      GO TO 107
  106 GAMAH(I)=RHOH*GREFF*(1.0+(PGA1T(I)*3.04E-08)-
     * (PGA1T(I)**2*2.72E-15))
      GO TO 108
  107 GAMAH(I)=RHOH*GREFF*(1.0+(PGA2T(I)*3.04E-08)-
     * (PGA2T(I)**2*2.72E-15))
  108 IF(PGA1T(I) .GE. PGA2T(I))COEFO(I)=
     *  CDMOC(I)*SQRT(ABS(2.*GREFF/GAMAH(I)))
      IF(PGA2T(I) .GT. PGA1T(I))COEFO(I)=
     *  CDMOE(I)*SQRT(ABS(2.*GREFF/GAMAH(I)))
      IF(IMODE(I).NE.0 .OR. DDELTA(I).GT.0.0) GO TO 113
      QO(I) = 0.0
      GO TO 109
C
  113 QO(I)=COEFO(I)*(AREMO(I)-APINT(I))*Q1(PGA1T1(I),PGA2T(I))
      IF(QO(I).LT.0.0 .AND. VCUM(I).LE.0.0) GO TO 102
      AIC(I)=0.0
      GO TO 103
  102 IF(PGA1T1(I).LT.PGA2T(I)) GO TO 103
      QO(I)=0.0
      VCUM(I)=0.0
      AIC(I)=1.0
  103 IF(QO(I).GT.0.C) AIC(I)=0.0
  109 IF(PGA2T(I) .LE. -1600.0)PGA2T(I)=-1600.0
      IF(PGA3T(I) .LE. -1600.0)PGA3T(I)=-1600.0
      IF(DELTA(I) .LE. 0.0 .AND. TIME .GT. DELT)GO TO 101
      GO TO 110
```

```
  101  FFORT(I)=0.0
       GO TO 140
   110  CONTINUE
C      COMPUTE STRUT AXIAL RINDING FRICTION FORCE
       BLFORT(I)=FONHST(I)*((SLEN2(I)-S(1,I))/(SLEN1(I)+S(1,I))+1.0)
       BUFORT(I)=FONHST(I)*(SLEN2(I)-S(1,I))/(SLEN1(I)+S(1,I))
       FFORT(I)=BUMU(I)*ABS(BUFORT(I))+BLMU(I)*ABS(BLFORT(I))
  140  CONTINUE
C      COMPUTE SHOCK STRUT CHARGING FORCE
       IF(S(1,I) .GT. 0.0)GO TO 142
       FORCHT(I)=PGA1T(I)*AREA1(I)+PGA2T(I)*(AREA2(I)-AREA1(I))-PGA3T(I)
      X *AREA3(I)+FFORT(I)+CFFOR(I)
C      COMPUTE NORMAL AND AXIAL HUB TO SHOCK STRUT FORCES AT HUB
  142  FONHST(I)=SQRT(FDX(I)**2+FDY(I)**2)-MASS(I)*GREFF*SIPA+SBFOT
       IF(ABS(FT(I)) .LE. FORCHT(I) .AND. S(1,I) .EQ. 0.0)GO TO 150
       GO TO 801
  150  CONTINUE
       FORSST(I)=FT(I)
       SD1(1,I)=0.0
       IF(IMODE(I).EQ.0) GO TO 297
       ISTROK(I)=1
       GO TO 289
C      COMPRESSION VELOCITY OF SHOCK STRUT IS POSITIVE
  801  IF(SD1(1,I).LE.0.8 .AND. IFR(I).EQ.0) GO TO 2
       GO TO 3
    2  DMTANH(I)=1.0
       GO TO 284
    3  DMTANH(I)=ABS(TANH(2.0*SD1(1,I)))
       IFR(I) = 1
  284  IF(S(1,I).LT.0.0) GO TO 160
       GO TO 161
  160  IPSTOP(I)=1
  161  IF(S(1,I).LE.DSTOP .AND. IPSTOP(I).EQ.1) GO TO 900
       GO TO 902
  900  IF(S(1,I).LE.0.005) GO TO 903
       GO TO 904
  903  SD2(1,I)=0.0
       SD1(1,I)=0.0
       S(1,I)=0.0
       IPSTOP(I)=0
       GO TO 902
  904  CONTINUE
       IF(IFSTOP(I) .NE. 0)GO TO 906
       DSTOP=S(1,I)
       FSTOPK=2.0*MASS(I)*SD1(1,I)**2/DSTOP**2
  906  IF(S(1,I) .LE. DSTOP/2.0)GO TO 908
       FSTOP(I)=-FSTOPK*(DSTOP-S(1,I))
       GO TO 909
  908  FSTOP(I)=-FSTOPK*S(1,I)
  909  IFSTOP(I)=1
       IFR(I) = 0
       GO TO 901
  902  FSTOP(I)=0.0
```

```
 901 IF(PGA1T(I).LE.(PGA1I(I)+500.0) .AND.
     :  PGA1T(I).GT.(PGA1I(I)-500.0)) GO TO 158
     GO TO 159
 158 IF(ABS(FT(I)).LE.FORCHT(I) .AND. S(1,I).EQ.0.0) GO TO 500
 159 IF(S(1,I).GE.0.0) GO TO 470
     IF(SD1(1,I) .LT. 0.0)GO TO 470
     GO TO 471
 470 FFORT(I)=-FFORT(I)
     CFFOR(I)=-CFFOR(I)
 471 FORSST(I)=-((PGA1T(I)-PGA2T(I))*(AREA1(I)-APINT(I))
    & +PGA2T(I)*AREA2(I)
    X -PGA3T(I)*AREA3(I)+(FFORT(I)
    1       +CFFOR(I))*DMTANH(I)+FSTOP(I))
 500 IF(INDFLX.GE. 1)GO TO 295
     IF(IMODE(I).EQ.0) GO TO 297
     ISTROK(I)=1
 289 IF(S(1,I) .GT. 0.0) GO TO 295
     IF(IOPCO(I) .EQ. 1) GO TO 295
     IF((PGA1I(I)-1000.0).LT.PGA1T(I).AND.
    &  PGA1T(I).LT.(PGA1I(I)+1000.0))299,298
 299 IF(XVALVE(I) .NE. 0.0)GO TO 311
     IF(IPASS(I) .EQ. 1)GO TO 296
     XVALVE(I)=XKSV(I)*XMA11(I)+XBIAS(I)
     IPASS(I)=1
     GO TO 294
 298 IF(ICOSV(I) .EQ. 1)GO TO 291
     IOPCO(I)=0
     IF(XSV(I) .LT. 0.002 .AND. XSV(I) .GT. -0.002)291,295
 291 IF(SD2(1,I) .LE. 0.0 .AND. ICOSV(I) .EQ. 1)GO TO 311
     IF(IOPCO(I) .EQ. 1)GO TO 295
     IF(PGA1T(I) .LE. PGA1I(I)) GO TO 293
     IF(IXSVL(I) .EQ. 1)GO TO 294
     XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
     IF(XVALVE(I) .GT. -0.1) GO TO 294
     XVALVE(I)=-0.1
     IXSVL(I)=1
     GO TO 294
 293 IF(IXSVH(I) .EQ. 1)GO TO 294
     XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
     IF(XVALVE(I) .LT. 0.1) GO TO 294
     XVALVE(I)=0.1
     IXSVH(I)=1
 294 CONTINUE
     DLTX1D(I)=0.0
     ICOSV(I)=1
 296 IF(WFORT(I) .GT. 0.0 .AND. S(1,I) .LE. 0.0)GO TO 410
 311 IF(NAC(I) .EQ. 1)GO TO 307
     IF(IIXSVH(I) .EQ. 1)GO TO 305
     XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
```

```
      IF(XVALVE(I) .LE. 0.0)305,400
  305 XVALVE(I)=0.0
      IIXSVH(I)=1
      GO TO 400
  307 IF(IIXSVL(I) .EQ. 1)GO TO 308
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
      IF(XVALVE(I) .GE. 0.0)308,400
  308 XVALVE(I)=0.0
      IIXSVL(I)=1
  400 CONTINUE
  410 IF(XVALVE(I) .NE. 0.0)GO TO 295
      ICOSV(I)=0
      DELTX1(I)=DELTX(I)*XKF(I)
      XMA(I)=(DF(I)+DELTX1(I))*XKA(I)
      XMA11(I)=XMA(I)
      XSV(I)=XKSV(I)*XMA11(I)+XBIAS(I)
      CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      IPASS(I)=0
      IXSVL(I)=0
      IXSVH(I)=0
      IIXSVL(I)=0
      IIXSVH(I)=0
      IOPCO(I)=1
      CALL PHLOZ2(PS(I),PR(I),XSV(I),QC(I),XLPSV1(I),XLPSV3(I),PCLSV(I),
     & OSV(I),CSV1(I),CSV3(I),XMU,QTOLER,NITER,P1(I),QS1(I),QS3(I))
C
  295 IF(ISTROK(I) .EQ. 1 .AND. S(1,I) .GT. 0.0)IOPCO(I)=0
 .297 I2 = 2*I+NSTRUT-1
      I1 = I2+1
      CALL INTEG(LA(I2),SD2(1,I))
      CALL INTEG(LA(I1),SD1(1,I))
C
      IF(IMODE(I).EQ.0) GO TO 450
C
      ENUP(1) = 0.5*AIYYBS*QI77R(1)*QI77R(1)+(ENCG/CMASNG)*
     *      (ZG77F1(1)/ABS(ZG77F1(1)))
      IF(HMM(I) .EQ. 1.0)GO TO 130
      SA(I)=0.
      IF(WFORT(I).LT.0.) XSTOT(I)=ENUP(I)/((-WFORT(I))*COPA)
      IF(WFORT(I).GE.0.0 .OR. DDELTA(I).LT.0.0) XSTOT(I)=1.E20
C     ZSSC IS A PERCENTAGE OF SB(I) FOR ACTIVATING CONTROL-CDMOC(I) IS U
      ZSSC(I)=FWORK(I)*SB(I)
      IF(XSTOT(I).LE.(ZSSC(I)-S(1,I)) .OR. RESA(I).EQ.1.0) SA(I)=1.0
      RESA(I)=SA(I)
      IF(SA(I).EQ.0. .OR. HMM(I).EQ.1.) GO TO 130
      WLFOR(I)=-WFORT(I)
C
      VELDEC=((WLFOR(I)+WLFORR)/2.*(WLFOR(I)-WLFORR))/
     &   (VMASS(1)*REDSLP(I))
```

```
      WRITE(6,121)TIME,WLFOR(I),VELDEC
  121 FORMAT(50H ACTIVE CONTROL INITIATED...TIME, WLFOR, VELDEC = ,
     1       3E13.5)
      HMM(I)=1.
  130 IF(S(1,I) .GT. 0.0)ISET(I)=0
      IF(HMM(I).EQ.0.) GO TO 451
      IF(-WFORT(I).GT.(WLFOR(I)+EPSILO(I))) DF(I)=(WLFOR(I)+
     &  EPSILO(I))-(-WFORT(I))
      IF(-WFORT(I).LT.(WLFOR(I)-EPSILO(I))) DF(I)=(WLFOR(I)-
     &  EPSILO(I))-(-WFORT(I))
      IF(-WFORT(I).LE.(WLFOR(I)+EPSILO(I)).AND.
     & -WFORT(I).GE.(WLFOR(I)-EPSILO(I)))
     : 457,456
  457 IF(S(1,I) .LE. 0.0)GO TO 456
      IF(WFORT(I) .GT. 0.0 .AND. QSVCU(I) .LT. 0.0)454,455
  454 DF(I)=WLFOR(I)-(-WFORT(I))
      GO TO 456
  455 DF(I)=0.0
  456 DELTX(I)=S(1,I)-XSCOM(I)
      IF(S(1,I) .LE. 0.0 .AND. ISET(I) .EQ. 0)GO TO 451
      GO TO 452
  451 DF(I)=0.
      DELTX(I)=0.
  452 XMA(I)=(DF(I)+DELTX1(I))*XKA(I)
      IF(GNR.EQ.1. .AND. XMA(I).GT.0.) XMA(I)=
     :  XMA(I)*SQRT((PGA1T(I)-PGALAC(I))
     X /(PGAHAC(I)-PGA1T(I)))
C
C       NOTE: SUBROUTINE 'FLOZE2' COMPUTES THE FLOWS FROM THE PRESSURES
C       IN UNITS OF INCHES.
      P1(I)=PGA1T(I)/144.
C     COMPUTATION OF HIGH PRESSURE ACCUMULATOR NITROGEN VOLUME
C     AND ACCUMULATOR PRESSURE
      VOLANT(I)=VOLANT(I)+QSVN(I)*DELT-QPUMPS(I)*DELT
      PS(I)=(((PGAHAC(I)+PATM)*(VOLANI(I)/VOLANT(I))**GAMA)-PATM)/144.0
      IF(PS(I) .GE. 3000.0)464,465
  464 PS(I)=3000.0
      VOLANT(I)=VOLANI(I)
  465 VOLAHT(I)=VOLACI(I)-VOLANT(I)
      IF(VOLAHT(I) .LE. 0.0)466,467
  466 WRITE(6,1050)TIME
 1050 FORMAT(1HO//45H ACCUMULATOR OIL VOLUME INSUFFICENT AT TIME=,E16.8
     1      //)
      CALL LGEAR6
      STOP 500
  467 CONTINUE
      CALL FLOZE2(PS(I),PR(I),P1(I),XLPSV1(I),XLPSV3(I),RCLSV(I),DSV(I),
     &  XSV(I),QS1(I),QS3(I),CSV1(I),CSV3(I),XMU)
      QSV1(I)=QS1(I)/1728.
```

```
      OSV3(I)=OS3(I)/1728.
C
  450 CONTINUE
      OSV(I)=QSV1(I)-OSV3(I)
      IF(OSV(I) .LT. 0.0)NAC(I)=1
      IF(QSV(I) .GT. 0.0)NAC(I)=2
      IF(NAC(I) .NE. 2) GO TO 462
      QSVN(I)=OSV(I)
      GO TO 463
  462 QSVN(I)=0.0
  463 IF(SD1(1,I).LT. 0.0 .AND.PGA1T(I).LE. -1600.0)PGA1T(I)=-1600.0
C
      RETURN
      END
```

In the comdeck ACOBLK, the variable IDEACT in the ACTIVE common block was changed to INITSW, the array IFRI was changed to IFR, the variable VELDEC was changed to ENCG, and the array INDEACT(5) and the simple variable VELDEC were added at the end. In subroutine EXE, the variable TPD was changed to TIME in the test below statement 412 and the return was changed to

```
STOP "EXECUTIVE ROUTINE"
```

The following declaration statements were added to subroutine ACTINIT to make the variables S

```
DIMENSION S(2,5)
EQUIVALENCE (S1(1),S(1,1))
EQUIVALENCE (DM15(1),ITO)
```

and ITO available, and the DATA statement introduced by CSCMOD2 was changed by substituting IFR for IFRI. The variable INITSW was

initialized to unity following the definition of OMRUN, and the following code was inserted after the definition of VMASS(I).

```
INDEACT(I) = 0
IF(ITO.EQ.1) INDEACT(I) = 2
```

The call to SETUP was changed to a call to ALGEAR. In subroutine ALGEAR1 the following declaration statement was added to make the variable ITO available.

```
EQUIVALENCE (DM15(1),ITO)
```

The call to LGEA3C was replaced with the following code,

```
C
      IF(INITSW.EQ.1) GO TO 16
```

and the following code was inserted after the definition of SIPA.

```
C
      CALL LGEA3C
C
```

Statement number 18, which had been modified by CSCMOD2, was changed as follows.

```
   18 CALL ACTNG
C
      DO 274 I=2,NSTRUT
```

The two tests on I and ZDANT introduced by CSCMOD2 were removed as

were the lines of code running from the definition of UNSPR through statement number 67. The DO 65 statement introduced by CSCMOD2 was removed and in statement number 801, IFR was substituted for IFRI, with a similar substitution in the statement below the statement labeled 3. The four calls to VIRK4 were removed and IFR was substituted for another IFRI below statement 909. The first four lines of code to modify ENUP(I), which had been introduced by CSCMOD2, were removed, as was statement number 120. The continuation line in the definition of VELDEC(I), which had been modified by CSCMOD3, was further modified as follows.

```
     &    VMASS(I)*REDSLP(I))
```

The following code was inserted below statement number 274,

```
C
      DO 65 I=1,NSTRUT
```

the SETUP entry point was replaced with the following,

```
   16 CONTINUE
      INITSW = 0
```

and the definition of II below statement number 20 was removed. In subroutine SDFLGP, the DATA statement for ACOVAR9 was changed by replacing 4HIFRI with 3HIFR. The DATA statement for N19 was replaced with the following,

```
      DATA N20 /20/
```

and the write to unit 13 was changed by replacing N19 with N20 and by adding ACOVAR3(3) to the list. The five write statements to unit 13 below statement 220 were modified by adding the appropriate element of FORSST to each. Finally, the call to STOVAR was changed by substituting IFR for IFRI.

The CSCMOD5 correction set contained the final corrections necessary to make the active code operational. In the ACOBLK common deck the ACTIVE common block was modified by replacing the variables COEF and GAMAN with ALGDUM1 and ALGDUM2, respectively, replacing the simple variable VELDEC with the array VELDEC(5), and by adding the two arrays COEF1(5) and LMODE(5). In subroutine EXE, the variable TIME was changed back to TPD in the test below statement 412, cancelling the effect of the CSCMOD4 change. In subroutine MIMIN a problem with the flight time limit was corrected by changing the test below statement 45 to the following.

```
IF((XF-XO).GT.1.E-10) GO TO 211
```

The DATA statement for NCASE in subroutine DEF was removed. In subroutine ACTINIT the 14th continuation line in the second declaration statement for the common block DIRCOM was changed to the following,

```
*,DDM21(20),SLENDUM(10),GAMA,DUM15(12),INDNWS,DDM22(10),ETADES,
```

and the LGDE common block was removed. The DATA statement for IOPCO was changed to the following.

```
DATA IOPCO,ISET,ISTROK,NAC /10*1,10*0/
```

The following new DATA statements were added

```
DATA INDFACT,INDINT,INITSW /5*0,5*1,1/
DATA FFORT,FONHST,FORCHT,FORSST,WFORT /25*0.0/
DATA QC,QD,QSV1,QSV3,QSVCU,QTOLER /25*0.,0.0001/
DATA DCON,DMTANH,DF,DP1,DELTX,DELTX1,DLTX1D /6*1.0,25*0.0/
DATA XSVDOT,XSVDD,XSVDDD /15*0.0/
DATA REDSLP,SBFOT,VCUM,VELDEC /5*100000.,0.,10*0.0/
DATA XMA1,XMA2,XMA3,XMA4,XMA6,XMA7,XMA9,XMA10 /40*0.0/
DATA XMA,XMA5,XMA8,XMA11 /20*0.0/
```

Assignment statements for the variables DCON, QTOLER, SBFOT, VELDEC, INITSW, REDSLP(I), INDINT(I), ISTROK(I), NAC(I), QSVCU(I), DF(I), DELTX(I), DELTX1(I), DLTX1D(I), XMA(I), XMA5(I), XMA8(I), XMA11(I), XMA1(I), XMA2(I), XMA3(I), XMA4(I), XMA6(I), XMA7(I), XMA9(I), XMA10(I), XSVDD(I), XSVDDD(I), XSVDOT(I), DP1(I), COEF, CSV1(I), and CSV3(I) were removed. The following code was added after the definition of COEF3(I),

```
COEF1(I) = CDSV(I)*SQRT(2.*GREFF/GAMAH(I))*144.
CSV1(I) = COEF1(I)*WSV1
CSV3(I) = COEF1(I)*WSV3
```

and the definition of QC(I) was changed to the following.

```
QSV1(I) = 0.0
QSV3(I) = 0.0
IF(ITD.EQ.1) GO TO 60
IF(IMODE(I).EQ.0) GO TO 80
```

The following new code was added beneath the second definition of QSV3(I).

```
      GO TO 80
C
   60 CONTINUE
      HMM(I) = 1.0
      INDEACT(I) = 2
      LMODE(I) = IMODE(I)
      IMODE(I) = 0
      AP2TO(I) = PGA2I(I)+PATM
      VOL1T(I) = VOL1I(I)-(AREA1(I)-APINT(I))*S(1,I)
      VOL3T(I) = VOL3I(I)+AREA3(I)*S(1,I)
      VOL2T(I) = VOL2I(I)-(AREA2(I)-AREA1(I)+APINT(I))*S(1,I)
     .           +(VOL3T(I)-VOL3I(I))-VCUM(I)
      PGA2I(I) = AP2TO(I)*((VOL2I(I)/VOL2T(I))**GAMA)-PATM
      PGA1I(I) = PGA2I(I)
      PGA3I(I) = PGA2I(I)
      FORSST(I) = -(PGA2I(I)*AREA2(I)-PGA3I(I)*AREA3(I)
     .            +DMTANH(I)*(FFORT(I)+CFFOR(I))+FSTOP(I))
      WFORT(I) = FORSST(I)
C
   80 CONTINUE
```

Assignment statements for the variables QO(I), VCUM(I), FFORT(I), FORSST(I), WFORT(I), FONHST(I), FORCHT(I), and INDEACT(I) were removed. The test on ITO was removed, and the call to ALGEAR, from CSCMOD4, was changed to a call to ALGEAR1. In subroutine ALGEAR1, the variable XCGRF was made available by the following statement.

```
      EQUIVALENCE (DM8(79),XCGRF)
```

The DIMENSION and DATA statements involving the variable IPSTOP were changed as follows.

```
      DIMENSION IPSTOP(5),AIC(5),PGA1T(5)
      DATA AIC,IPSTOP /5*0.0,5*0/
```

The definition of CMASNG was changed to the following,

```
      RXCG1 = RX(1)-XCGRF
      CMASNG = 1+(AMASS*RXCG1*RXCG1)/AIYYBS
```

and in the definition of ZDANT, the variable QI77R was changed to the array element QI77R(1). The statement labeled 18 was changed to the following.

```
      UNSPRNG = 0.0
      DO 22 I=1,NSTRUT
         IF(ITC.EQ.1 .AND. TIME.GE.2.0) IMODE(I) = LMODE(I)
         IF(OMETD1(I).NE.0.0) UNSPRNG = UNSPRNG+MASS(I)
   22 CONTINUE
C
      DWFORT = (-SQRT(AXP7F*AXP7F+AYP7F*AYP7F+AZP7F*AZP7F)+GREFF)*
     .         (AMASS-UNSPRNG)
C
      CALL ACTNG
C
```

The following code was added at the top of the 274 loop.

```
C
      WFORT(I) = DWFORT/(NSTRUT-1)
C
```

The four occurrences of VELDEC were changed to VELDEC(I), and the two definitions of II, left over from earlier VIRK4 calls, were removed. Statement 296 was changed by modifying the branch from 410 to 400, and the statement label of 410 was eliminated. The test on IMODE below statement 297 was changed as follows.

```
      IF(IMODE(I).NE.0.0) GO TO 119
      OSV1(I) = 0.0
      OSV3(I) = 0.0
      GO TO 450
```

The continuation line in the definition of VELDEC(I) was corrected as follows.

```
     (VMASS(I)*REDSLP(I))
```

In subroutine ACTNG, the DIMENSION statement for SD2 was expanded to include OMETD1(2,5), and OMETD1 was equivalenced to OMTD11. The four occurrences of VELDEC were changed to VELDEC(I), and the following code was inserted below statement 90.

```
UNSPRNG = 0.0
IF(OMETD1(1).NE.0.0) UNSPRNG = MASS(1)
WFORT(1) = SR(1)*(VMASS(1)-UNSPRNG)
```

The branch in statement 296 was changed from 410 to 400 and the 410 label was eliminated. The test on IMODE below the statement labeled 297 was changed to the following.

```
    IF(IMODE(I).NE.0.0) GO TO 119
    QSV1(I) = 0.0
    QSV3(I) = 0.0
    GO TO 450
119 CONTINUE
```

In subroutine DECOMP, the array IPS was moved from blank common to labeled common, IPSCOM, with an identical change in subroutine SOLVE. Finally, extensive comments were added to subroutine ACTIN, but no changes were made to the executable code.

The correction set PINARY was primarily intended to introduce logic which would allow the area of the metering pin to vary as a function of strut stroke. In the comdeck ACOBLK, the ACTIVE common block was expanded to include the four arrays PINN(30), PINM(30), STRON(30), and STROM(30), and in subroutine EXE, the arrays NAME and LOC in the ACTDIR common block were expanded from 71 to 75. In subroutine ACTINIT, the following new code was inserted after the initialization of QSV3.

```
      IF(KAPT(I).EQ.1) GO TO 50
      APINT(I) = 0.
      IF(KAPT(I).EQ.0) GO TO 50
      IF(I.GT.1) GO TO 25
      STROK = -1.
      DO 10 J=1,29
      IF(STRON(J).LT.STROK) GO TO 15
      STROK = STRON(J)
      IF(S(1,I).GE.STRON(J).AND.S(1,I).LE.STRON(J+1)) GO TO 20
   10 CONTINUE
   15 J = J - 1
   20 APINT(I) = PINN(J)
      GO TO 50
   25 STROK = -1.
      DO 30 J=1,29
      IF(STROM(J).LT.STROK) GO TO 35
      STROK = STROM(J)
      IF(S(1,I).GE.STROM(J).AND.S(1,I).LE.STROM(J+1)) GO TO 40
   30 CONTINUE
   35 J = J - 1
   40 APINT(I) = PINM(J)
   50 CONTINUE
```

In both ALGEAR1 and ACTNG the three lines of code beginning with a test on KAPT(I) were removed. In subroutine ACTIN, the arrays XNAME and LOC in the ACTDIR common block were expanded from 71 to 75, and the arrays DATA and IDATA, representing the ACTIVE common block were expanded form 656 to 802. The following code was inserted as the

first executable statements.

```
C
C       ZERO OUT PIN AND STROKE ARRAYS
C
      DO 10 J=683,802
      DATA(J) = 0.
   10 CONTINUE
```

The FORMAT statement labeled 1 was changed by replacing I1 with I2, and the upper limit on the 110 loop was changed from 71 to 75. In the DIRACT block data, the NAME and LOC arrays were enlarged from 71 to 75. The final continuation card in the DATA statement for NAME was changed to the following,

```
    5 6HZETAC1,  6HZETAC2,  6HIMODE ,  6HPINN  ,  6HPINM  ,
    6 6HSTRON ,  6HSTROM /
```

and the final continuation card in the DATA statement was changed as follows.

```
7      644, 645, 651, 683, 713, 743, 773/
```

The KLUGEZ correction set was intended to prohibit secondary piston calculations for the nose gear. The following code was inserted at the top of the 100 loop in subroutine LGEAR1.

```
C
C*********
C
C********** TEMPORARY FIX TO FREEZE NOSE SECONDARY
C
      IF(I.EQ.1) GO TO 59
C
C**********
C
```

The correction set SECFIX was meant to provide constraints prohibiting over-extension or over-compression of the secondary piston in a manner similar to earlier corrections made by SDD for the main piston. In subroutine LGEAR1, the following code was added after statement 57.

```
S2(1,I) = -0.5*ES2(I)
S2D1(1,I) = -1.E-10
S2D2(1,I) = -1.E-10
```

The two lines of code beginning with statement 61 were replaced with the following,

```
 61 IF(S2D2(1,I).LT.0.) GO TO 140
    IF(S2D1(1,I).LT.0.) S2D1(1,I) = 0.
```

and the following line was added below the statement labeled 140.

```
S2D1(1,I) = 0.
```

For debugging purposes, the following code was inserted in subroutine LGEA3C below the calculation of TMP(1).

```
    IF(TMP(1).LT.0.) WRITE(6,1234) I,S2(1,I),S2D1(1,I),
   1 S2D2(1,I)
1234 FORMAT(1X,7H*-*-*-*,I5,3E16.8)
    IF(TMP(1).LT.0.) GO TO 31
```

The TABFIX correction set corrected a table look-up problem associated with the secondary piston. In subroutine LGEA3C, the

last two calls to HIHO were changed by substituting S2(1,I) for S(1,I).

The correction set CSCMOD6 was intended to introduce a new variable, HT3, to be used from anywhere in the program to limit the integration step size in critical conditions and to correct a problem with the reduction of the control limit force. The HTCOM common block was modified in subroutines EXE, MIMIN, LGEAR1, and ALGEAR1 as follows.

```
COMMON/HTCOM/HT,HT1,HT2,HT3,INDINT(5),H
```

The ACTDIR common block was removed from subroutine EXE where it was not needed. The initialization of INDINT in subroutine MIMIN was changed from a 10 loop to a 445 loop and moved to beneath statement 40. In subroutine LGEAR1, the following code was introduced beneath statement 4,

```
 INTFLAG = 0
```

and the call to INUPD was changed to the following.

```
IF(INTFLAG.EQ.0) CALL INUPD(NDEQ,LA)
INTFLAG = 1
```

In subroutine ALGEAR1, the definition of WLFOR(I) above statement 56 was changed by substituting H for HT.

The SECFIX1 correction set made a minor addition to the SECFIX changes to control the secondary piston over-compression. The following statement was added to subroutine LGEAR1 below statement 140

```
S2(1,I) = 0.
```

Additional modifications to provide for a stroke dependent metering pin area were provided in the PINARYX correction set. the following new code was inserted in ALGEAR1 below the initialization of the RL array.

```
      DO 760 I=1,NSTRUT
      IF(KAPT(I).EQ.1) GO TO 750
      APINT(I) = 0.
      IF(KAPT(I).EQ.0) GO TO 750
      IF(I.GT.1) GO TO 725
      STROK = -1.
      DO 710 J=1,29
      IF(STRON(J).LT.STROK) GO TO 715
      STROK = STRON(J)
      IF(S(1,I).GE.STRON(J).AND.S(1,I).LE.STRON(J+1)) GO TO 720
  710 CONTINUE
  715 J = J - 1
  720 APINT(I) = PINN(J)
      GO TO 750
  725 STROK = -1.
      DO 730 J=1,29
      IF(STROM(J).LT.STROK) GO TO 735
      STROK = STROM(J)
      IF(S(1,I).GE.STROM(J).AND.S(1,I).LE.STROM(J+1)) GO TO 740
  730 CONTINUE
  735 J = J - 1
  740 APINT(I) = PINM(J)
  750 CONTINUE
  760 CONTINUE
```

The CSCMOD6 correction set was refined with the addition of the HTTRY mods. In subroutine EXE, the new variable HT3 was initialized to HT above statement 302. In subroutine MIMIN, HT3 was added to the list in the AMIN1 statement labeled 30 and the statement was moved to below statement 40.

A restart capability was added to the program with the REST correction set which placed most all local variables into labeled common blocks and declared all common blocks in the main program to assure contiguous storage of all program variables. The unit TAPE7 is used to hold the restart information. In program TOLA, TAPE7 was added to the program card. The following new code was added below the READ1 common declaration,

```
COMMON/STOPIT/DM2(2)
COMMON/DIRCOM/DM3(4059)
COMMON/TABSRC/DM4(110)
COMMON/EXEAUT/DM5(9)
COMMON/LG/DM6(7)
COMMON/AUTSC/DM7(40)
COMMON/AUTPRC/DM8(63)
COMMON/LGAUTS/DM9(14)
COMMON/FLXOP/DM10(608)
COMMON/AUTSAC/DM11(6)
COMMON/HTCOM/DM12(10)
COMMON/CONTRO/DM13(4)
COMMON/UPDCAL/DM14(181)
COMMON/LGDE/DM15(72)
COMMON/STGT/DM16(10)
COMMON/TABCOM/DM17(230)
COMMON/CLEAUP/DM18(3)
COMMON/STORA/DM19(67)
COMMON/LGE/DM20(299)
COMMON/ACTIVE/DM21(802)
COMMON/IPSCOM/DM23(20)
COMMON/XEXE/DM24(3),BLANK,STCOM2,STCOM3,DM25(5)
COMMON/XMIMIN/DM26(908)
COMMON/XARRAY/DM27
COMMON/XTFFS8/DM28(2)
COMMON/XSACS1/DM29(106)
COMMON/XOPT1/DM30(97)
COMMON/XLGER1/DM31(21)
```

```
COMMON/XLGF3C/DM32(12),IFRI(5),DM33(19)
COMMON/XALGEA/AIC(5),DM34(14),IPSTOP(5),DM34X(10)
COMMON/XACTNG/AICX(5),IPSTOX(5),DM35(6)
COMMON/XFLEX1/DM36(2077)
COMMON/XSDFLG/DM37(179)
COMMON/XAUTS/DM38(8)
COMMON/XFLARE/DM39(20)
COMMON/XAUTPR/DM40(4)
COMMON/RESTRT/IRST
```

and the following code was inserted just above the first executable statement.

```
DATA STCOM2,STCOM3,BLANK/4HTMAX,5HSTAGE,6H      /
DATA IFRI/0,0,0,0,0/
DATA AIC,IPSTOP/5*0.0,5*0/
DATA DM34Y,DM347,DM34A/5*0.,5*31000.,65*0./
DATA AICX,IPSTOX/5*0.0,5*0/
```

The following read sequence was placed immediately before the call to EXE.

```
  READ(5,2) IRST
2 FORMAT(I1)
```

In subroutine EXE, the following declaration statements were added,

```
 COMMON/XEXE/END,SWT2,SWT3,BLANK,STCOM2,STCOM3,
1 MIM,NDEFS,TIMEA,TIMEP,TPD
 COMMON/RESTRT/IRST
 DIMENSION DMD1(1)
 EQUIVALENCE(DMD1(1),TABLE(1))
```

and the DATA statement was removed. The FORMAT labeled 7, which was not referenced, was replaced with the following.

```
5 FORMAT(1HO,16X,*JOB RESTARTED AT*,E16.8)
7 FORMAT(1HO,16X,*RESTART TAPE WRITTEN AT*,E16.8)
```

The following statement was inserted as the first executable statement,

```
IF(IRST.GT.1) GO TO 800
```

and the new code below was inserted at the top of the staging logic below statment 416.

```
    IF(IRST.EQ.1.OR.IRST.EQ.3) GO TO 810
    GO TO 511
800 CONTINUE
    READ(7) (DMD1(IJ),IJ=1,11002)
    CALL LINES(2)
    WRITE(6,5) TIMES
    GO TO 511
810 CONTINUE
    WRITE(7) (DMD1(IJ),IJ=1,11002)
    CALL LINES(2)
    WRITE(6,7) TIMES
```

Local variables in subroutine MIMIN were transferred to a labeled common block, XMIMIN, as follows.

```
    COMMON/XMIMIN/ACH,ERR,J,K,KF,PO,R,S,XF,XK,XO,
   1 YMAX,YP,YO,Y1,Z
```

The variable NMAX in subroutine ARRAY was transferred to common XARRAY, and in subroutine TFFS8, the variables TH1 and TH2 were put in common block XTFFS8. The local variables in SACS1 and OPT1 were relocated to common blocks XSACS1 and XOPT1 as follows.

```
    COMMON/XSACS1/AERO2,BETADE,HG,IGO1,IGO2,IGO3,
   1 TC,TMP,CN
    COMMON/XOPT1/ALPHD1,B,BETAD1,C,D,DRAGC,E,F,FDC,
   1 G,H,INDER,J,K,L,LA,LTHTRR,M,MDX,MDY,MDZ,P,THTRR,
   2 THTRR1,TMP
```

In subroutine LGEAR1, local variables were moved to XLGER1,

```
      COMMON/XLGER1/BFX,BFY,BFZ,BLM,BMM,BNM,DFXM,DFYM,
     1 DFZM,DLM,DMM,ERR,IL4,INTFLAG,I1,I2,NBB,NBH,NDEQ,
     2 THETAR,TTIME
     3 TEMP3,TMPETA,YAWPMM,YAWPRM,DELNM
```

and the DATA statement for PGA1T1 was removed. Local variables in LGEA3C were moved to XLGE3C,

```
      COMMON/XLGE3C/CRNPRM,CRNPWR,DELMN,DFTRX,DFTRXM,
     1 DFTRY,DFTRYM,ETAVE,ETAVEM,FGPYM,FI,HYPTAN,IFRI,
     2 IL1,IL2,IL3,ND,NDO,NOD,SIDEMU,SIDMUM,TEMP1,TEMP2,
```

and the DATA statement for IFRI was removed. In ALGEAR1 local variables were relocated to XALGEA,

```
      COMMON/XALGEA/AIC,BFX,BFY,BFZ,BLM,BMM,BNM,DFXM,
     1 DFYM,DFZM,DLM,DMM,DNM,DWFORT,IL4,IPSTOP,NBB,NBH,
     2 PGA1T,RXCG1,TTIME,UNSPRNG
```

and in ACTNG they were moved to XACTNG

```
      COMMON/XACTNG/AIC,IPSTOP,UNSPRNG,NIN,TIMEL
```

and the DATA statement for AIC was removed. In subroutines FLEX1, SDFLGP, AUTS, FLARE1, and AUTPR1, the following common blocks were created, respectively.

```
      COMMON/XFLEX1/COEF,COPMAS,CTMP1,DIFF,FDC,GF,
     1 GFORC1,GMASS,GSMOD,GTF,HDR,IG,II,LA,OMXD1M,
     2 OMYD1M,OM7D1M,PTN,QS,QS1,RKSY,RVAR,RXA13,RZA11,
     3 SD0,SD1,SD2,SMASS,SUM1,SUM3,TITL,VARY1
      COMMON/XSDFLG/ACOVAR1,ACOVAR2,ACOVAR3,ACOVAR4,
     1 ACOVAR5,ACOVAR6,ACOVAR7,ACOVAR8,ACOVAR9,DAT1,
```

```
2 DAT2,DAT3,DAT4,DAT5,ISUM1,ISUM2,LASTPT,NHL,N1,
3 N14,N15,N18,N20,OP16,OP17,OP18,OP19,OP20,OP21
 COMMON/XAUTS/DELPI,DELRDI,GAMMPR,SWT2,TMP1,
1 TMP2,TMP3,TMP5
 COMMON/XFLARE/AE,AH,AX,D1,ERRAD,GAMAPD,GAMAPR,
1 GAMERR,GAMMAD,L1,RSP,SWT1,TAE,TMP1,TMP2,TMP3,
2 TMP4,TMP5,TMP6,TTH
```

The correction set PGAPRT changed the printed output. In subroutine SDFLGP, the array PGA1T was made available with the following declaration.

```
COMMON/XALGFA/DMX(26),PGA1T(5),DMXX(3)
```

Additionally, the seven occurrences of PGA1T1, introduced by CSCMOD2, were changed back to PGA1T.

The DELTFIX mods were intended to once again address a problem with the reduction of the control limit force. In subroutine EXE, the variable H was initialized to 0.0001 above statement 302. In ALGEAR1, the test on TIME above statement 15 was changed by changing DELT to 0., and the following code was added at the top of the 274 loop.

```
DELT = 0.
IF(INDINT(I).NE.0) DELT = H
INDINT(I) = 0
```

The two statements beginning with statement 56 were replaced by a CONTINUE, and the next three lines were replaced with the following.

```
WLFOR(I)=WLFOR(I)-REDSLP(I)*DELT
EPSILO(I)=EPSILO(I)+EPSSLP*DELT
```

In statement 100, DELT was replaced by 0.0001. In subroutine ACTNG, the new HTCOM common block introduced by CSCMOD6 replaced the original declaration and PGA1T was moved from XACTNG to XALGEA. The following code was inserted at the top of the subroutine,

```
DELT = 0.
IF(INDINT(I).NE.0) DELT = H
INDINT(I) = 0
```

and the five lines of code beginning with statement 56 were replaced with the following.

```
 56 CONTINUE
    WLFOR(I) = WLFOR(I) - REDSLP(I)*DELT
    EPSILO(I) = EPSILO(I) - EPSSLP*DELT
```

The test on TIME below statement 109 was changed by substituting 0.0001 for DELT.

A deficiency was noted in the active code in that the input signal and its modifications to the electronic compensation networks were not being treated as integration variables. The XMAFIX correction set was intended to correct this deficiency. In program TOLA, the size of the LGDE common block was increased from 72 to 112 elements, and the ACTIVE common block was increased from 802 to 842 elements. In the ACOBLK comdeck, the final two continuation lines of the COMMON declaration were replaced with the following.

```
1,COEF1(5)  ,LMODE(5)  ,PINN(30)  ,PINM(30)  ,STRON(30) ,STROM(30)
2,XMA1DT(5) ,XMA2DT(5) ,XMA3DT(5) ,XMA4DT(5) ,XMA6DT(5) ,XMA7DT(5)
3,XMA9DT(5) ,XMA10D(5)
```

In subroutine EXE, the size of the LA array in the LGDE common block was increased from 50 to 90, the number of words being read from and written to TAPE7 was increased from 11002 to 11077, and an ENDFILE 7 was added after the WRITE to unit 7. The size of the LA array in the LGDE common block was also increased from 50 to 90 in LGDET, LGEAR1, LGEA3C, ALGEAR1, ACTNG, and SDFLGP. In subroutine LGEAR1, the second test on INDLG was changed as follows,

```
IF (IABS(INDLG) .EQ. 3) NDEQ=10*NSTRUT
```

and in ACTINIT the following DATA statement was added.

```
DATA XMA1DT,XMA2DT,XMA3DT,XMA4DT /20*0.0/,
     XMA6DT,XMA7DT,XMA9DT,XMA10D /20*0.0/
```

In subroutine ALGEAR1, the following code was inserted after the definition of XMA5(I),

```
XMA1DT(I) = XMA(I)
XMA2DT(I) = XMA1(I)
XMA3DT(I) = XMA5(I)
XMA4DT(I) = XMA3(I)
```

the following was added after the definition of XMA8(I),

```
XMA6DT(I) = XMA5(I)
XMA7DT(I) = XMA8(I)
```

and the following was added after the definition of XMA11(I).

```
XMA9DT(I) = XMA8(I)
XMA10D(I) = XMA11(I)
```

The following new code was inserted after the last call to INTEG.

```
I10 = I+10*NSTRUT
I11 = I+11*NSTRUT
I12 = I+12*NSTRUT
I13 = I+13*NSTRUT
I14 = I+14*NSTRUT
I15 = I+15*NSTRUT
I16 = I+16*NSTRUT
I17 = I+17*NSTRUT
CALL INTEG(LA(I10),XMA1DT(I))
CALL INTEG(LA(I11),XMA2DT(I))
CALL INTEG(LA(I12),XMA3DT(I))
CALL INTEG(LA(I13),XMA4DT(I))
CALL INTEG(LA(I14),XMA6DT(I))
CALL INTEG(LA(I15),XMA7DT(I))
CALL INTEG(LA(I16),XMA9DT(I))
CALL INTEG(LA(I17),XMA10D(I))
```

In subroutine SDFLGP, the following new code was added after the last call to UPDAT.

```
I10 = I+10*NSTRUT
I11 = I+11*NSTRUT
I12 = I+12*NSTRUT
I13 = I+13*NSTRUT
I14 = I+14*NSTRUT
I15 = I+15*NSTRUT
I16 = I+16*NSTRUT
I17 = I+17*NSTRUT
CALL UPDAT(1,LA(I10),XMA1(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I11),XMA2(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I12),XMA3(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I13),XMA4(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I14),XMA6(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I15),XMA7(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I16),XMA9(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I17),XMA10(I),DU,DU,DU,DU)
```

It was found that portions of the active code were referencing the controlled servovalve spool displacement when they should have

referenced the servovalve spool displacement. This problem was addressed with the XSVMOD correction set. Identical corrections were made to both ALGEAR1 and ACTNG. Statement number 500 was replaced with a CONTINUE. The following statement was introduced below statement 290,

```
XVALVE(I) = XSV(I)
```

the following code replaced statement 299,

```
299 CONTINUE
    IF(XVALVE(I).NE.0.) GO TO 311
```

and the following was inserted after statement 400.

```
XSV(I) = XVALVE(I)
```

The GENFIX mods made several small general changes to the program. The XACTNG common block was corrected in both TOLA and ACTNG. In TOLA, the size was reduced to reflect the removal of PGA1T in the REST correction set, and in ACTNG the name was corrected from ACTNG (an error in REST) to XACTNG. Two new elements, NIN and TIMEL, were added for later use. In subroutine EXE, the variable NCASE was made available by changing the third continuation line of the DIRCOM common declaration to the following,

```
*      DM43      ,INDVPC     ,DM44  ( 7),NCASE      ,DM45      ,
```

and NCASE was initialized to be blank. The upper limit on the 35 loop was changed from 4036 to 4059 to reflect the correct size of the DIRCOM common block. A rewind of unit 7 was added after the read from 7 to prepare the unit to receive new restart information. In subroutine MIMIN, the 45 loop was moved to below statement 205.

The DELTFXX mods represented another attempt to resolve the control limit force reduction. New variables NIN and TIMEL were initialized with a DATA statement to zero in both ALGEAR1 and ACTNG. In subroutine ALGEAR1, the following code was inserted as the first executable statements,

```
IF(NIN.EQ.0) TIMEL = TIME
NIN = 1
DELT = TIME - TIMEL
TIMEL = TIME
```

and the three lines at the top of the 274 loop were removed. In ACTNG, the 2nd through 4th executable statements were replaced with code identical to that above.

The MOD282 correction set finalized solution of the control limit force problem. The common block XALGEA was expanded in both TOLA and LGEAR1 to include NIN and TIMEL. The READ and WRITE statements to unit 7 in EXE were altered to transfer 11081 words. In both ALGEAR1 and ACTNG, the three statements beginning with statement 159 were replaced with the following.

```
159 IF(SD1(1,I).LT.0.) GO TO 471
```

The MOD296 correction set changed the active code. In subroutine ALGEAR1, the statement above statement 112 was changed to the following,

```
IF(IMODE(I).EQ.0.AND.S(1,I).EQ.0.) 112,113
```

and the test on IMODE above statement 19 was removed. In subroutine ACTNG, the test on IMODE below statement 108 was changed as follows.

```
IF(S(1,I).GT.0..OR.IMODE(I).EQ.1) GO TO 113
```

The REST1 mods corrected a problem with the program restart. The common block XAUTS was expanded in TOLA and AUTS to accommodate three new variables, DELTS, ERROR, and IPR. The size of the data array transferred to or from TAPE7 in EXE was increased by three words to 11084.

The primary purpose of the MOD329 correction set was to rearrange some logic flow in the active code. Additionally, the AMIN1 evaluation in subroutine MIMIN which had been moved by HTTRY was restored to its original position. In subroutine ALGEAR1, the definition of I9 and subsequent call to INTEG were moved to inside the 28 loop. The definition of I3 and the subsequent call to INTEG were removed from the 28 loop and the following code was inserted after statement 21.

```
      DP1(I) = (-QQ(I)+QSV1(I)-QSV3(I)+(AREA1(I) - APINT(I))
     1 *SD1(1,I))*BETA/VOLIT(I)
      IF(S(1,I).NE.0..OR.AIC(I).NE.1.) GO TO 64
      DP1(I) = 0.
   64 CONTINUE
      I3 = I + 3*NSTRUT
      CALL INTEG(LA(I3),DP1(I))
```

The preexisting definition of DP1 was removed and the following statement was added just inside the 28 loop

```
IF(IMODE(I).EQ.0) GO TO 28
```

In subroutine SDFLGP, the following statement was added after the call to UPDAT for QSVCU.

```
IF(IMODE(I).EQ.0) GO TO 6
```

The MOD351 mods simply added the following statements above statement 64 in ALGEAR1.

```
PGA1T1(I) = PGA1I(I)
PGA1T(I) = PGA1I(I)
```

Following the addition of 40 new integration variables by XMAFIX, it was necessary to increase the hard-wired maximum limit of such variables from 90 to 100. This was accomplished with the MOD1029 correction set. In program TOLA, the size of the UPDCAL common block was increased from 181 to 201, and the XMIMIN common

block was increased from 908 to 1008. The P and Y arrays were each increased in size from 90 to 100 in common block UPDCAL in subroutines EXE, INUPD, LNUPD, INPUZ, INTEG, UPDAT, MIMIN, and LGDET. In EXE, the size of the data block of information transferred to and from TAPE7 was increased from 11084 to 11204. In subroutine INUPD, both the test on NUM+N and the FORMAT were modified to reflect the new maximum limit. In subroutine INPUZ, the upper limit on the loop was increased to 100. Finally, in subroutine MIMIN the DIMENSION statement for YMAX was changed as follows.

```
      DIMENSION YMAX(100),YO(100),PO(100),S(100),YP(100),Y1(100),
     1 Z(100),XK(100,3)
```

The MOD1040 correction set corrected a problem with the pitch autopilot by inserting the following statement below statement 93 in subroutine AUTS.

```
      DELQI = DELQDE
```

Since it is the variable H and not HT which represents the integration step size, a change was made to MIMIN by the MODMIM correction set to enhance the program output. The second write of HT was changed as follows.

```
      WRITE(6,701) HT,H
  701 FORMAT(* INTEG RTN.       HT = *,E15.8,* H = *,E15.8)
```

A units problem in the active code was corrected by MOD1048. In both ALGEAR1 and ACTNG the value of the stroke in statement 4456 was converted from feet to inches by multiplying by 12.

The MOD1103 correction set changed the LGEAR1 code to allow for a special feature of the F4 gear design. The following new code was added after the test on IL.

```
C  MODIFICATION TO ACCOMODATE SECONDARY PISTON OF F4 MAIN GEAR
      IF(S(1,I).LE.0.) GO TO 59
      IF(S(1,I).GE.SB(I)-(S2T(I)-S2(1,I))) 83,85
C  SECONDARY PISTON IN CONTACT WITH ORIFICE TUBE
   83 IF(SD1(1,I)+1.E-4.GE.S2D1(1,I)) 84,85
   84 S2D1(1,I) = SD1(1,I)
      S2D2(1,I) = SD2(1,I)
      GO TO 60
   85 CONTINUE
```

A method for slowly and continuously varying the aerodynamic coefficients $C_{A_0}$ and $C_{N_0}$ was introduced with the AERAT correction set. Two new common blocks AEROCO and XAERO of lengths 8 and 2, respectively, were added to program TOLA. In EXE, the length of the restart data block transferred to and from TAPE7 was increased from 11204 to 11214. A call to AERO4 was added to subroutine OPT1 below statement 621, and the following new subroutine was added after OPT1.

```
SUBROUTINE AERO4
COMMON/DIRCOM/DM1(2),X
COMMON/TABCOM/LOCS(115),ST(115)
COMMON/TABDIR/TABLE(800)
COMMON/AEROCO/RTAB10(2),RTAB80(2),LTAB10(2),LTAB80(2)
COMMON/XAERO/NIN,TIMEL
REAL LTAB10,LTAB80
DATA TIMEL,NIN/0.,0/
DATA RTAB10,RTAB80,LTAB10,LTAB80/8*0./
IF(NIN.EQ.0) TIMEL = X
```

```
      NIN = 1
      DELT = X - TIMEL
      TIMEL = X
      IND1 = LOCS(45)
      IND2 = LOCS(114)
      ATAB11 = TABLE(IND1)
      ATAB12 = TABLE(IND1+1)
      ATAB81 = TABLE(IND2)
      ATAB82 = TABLE(IND2+1)
      ATAB11 = ATAB11 + RTAB10(1)*DELT
      ATAB12 = ATAB12 + RTAB10(2)*DELT
      ATAB81 = ATAB81 + RTAB80(1)*DELT
      ATAB82 = ATAB82 + RTAB80(2)*DELT
      IF(RTAB10(1).GT.0..AND.ATAB11.GE.LTAB10(1)) GO TO 40
      IF(RTAB10(1).LT.0..AND.ATAB11.LE.LTAB10(1)) GO TO 40
   10 IF(RTAB10(2).GT.0..AND.ATAB12.GE.LTAB10(2)) GO TO 50
      IF(RTAB10(2).LT.0..AND.ATAB12.LE.LTAB10(2)) GO TO 50
   20 IF(RTAB80(1).GT.0..AND.ATAB81.GE.LTAB80(1)) GO TO 60
      IF(RTAB80(1).LT.0..AND.ATAB81.LE.LTAB80(1)) GO TO 60
   30 IF(RTAB80(2).GT.0..AND.ATAB82.GE.LTAB80(2)) GO TO 70
      IF(RTAB80(2).LT.0..AND.ATAB82.LE.LTAB80(2)) GO TO 70
      GO TO 80
   40 ATAB11 = LTAB10(1)
      RTAB10(1) = 0.
      GO TO 10
   50 ATAB12 = LTAB10(2)
      RTAB10(2) = 0.
      GO TO 20
   60 ATAB81 = LTAB80(1)
      RTAB80(1) = 0.
      GO TO 30
   70 ATAB82 = LTAB80(2)
      RTAB80(2) = 0.
   80 TABLE(IND1) = ATAB11
      TABLE(IND1+1) = ATAB12
      TABLE(IND2) = ATAB81
      TABLE(IND2+1) = ATAB82
      RETURN
      END
```

In subroutine READ the following code was inserted after statement 19,

```
      IF(SYM.EQ.6HRTAB10) GO TO 905
      IF(SYM.EQ.6HRTAB80) GO TO 905
      IF(SYM.EQ.6HLTAB10) GO TO 905
      IF(SYM.EQ.6HLTAB80) GO TO 905
```

and the following statements were added below statement 810.

```
905 CALL AEROIN(SYM,RA)
    GO TO 100
```

The following new subroutine was added after subroutine READ.

```
    SUBROUTINE AEROIN(SYM,RA)
    DIMENSION RA(55)
    COMMON/AEROCO/DATAX(8)
    DATA DATAX/8*0./
    CALL LINES(1)
    WRITE(6,1) SYM,RA
  1 FORMAT(18X,A6,5X,55A1)
  2 FORMAT(*0ERROR.ILLEGAL CHARACTER IN NUMERIC FIELD*,1R1,2H**/)
    IF(SYM.EQ.6HRTAB10) INDEX = 1
    IF(SYM.EQ.6HRTAB80) INDEX = 3
    IF(SYM.EQ.6HLTAB10) INDEX = 5
    IF(SYM.EQ.6HLTAB80) INDEX = 7
    NUMEXP = 0
    NEXP = 0
    IEXP = 0
    NL = 0
    NR = 0
    NUML = 0
    NUMR = 0
    ISIGN = 0
    JSIGN = 0
    LEFT = 1
    DO 210 I=1,56
    IF(I.EQ.56) GO TO 140
    IF(RA(I).EQ.1H ) GO TO 210
    IF(RA(I).EQ.1H,) GO TO 140
    IF(RA(I).EQ.1H.) GO TO 170
    IF(RA(I).EQ.1HE) GO TO 180
    IF(RA(I).EQ.1H-) GO TO 200
    NUM = SHIFT(RA(I),6)
    NUM = NUM.AND.00000000000000000778
    IF(NUM.GT.36) GO TO 130
    IF(NUM.LT.27) GO TO 130
    NUM = NUM - 27
    IF(IEXP.EQ.1) GO TO 190
    IF(LEFT.GT.0) NUML = 10*NUML + NUM
    IF(LEFT.GT.0) NL = NL + 1
```

```
      IF(LEFT.LT.0) NUMR = 10*NUMR + NUM
      IF(LEFT.LT.0) NR = NR + 1
      GO TO 210
  130 CALL LINES(3)
      WRITE(6,2) RA(I)
      GO TO 210
  140 IF(NL.EQ.0.AND.NR.EQ.0) GO TO 210
      IF(NR.EQ.0) GO TO 160
      X = FLOAT(NUML) + FLOAT(NUMR)/(10.)**NR
      IF(JSIGN.EQ.1) NUMEXP = -NUMEXP
      IF(IEXP.EQ.1) X = X*(10.)**NUMEXP
      IF(ISIGN.EQ.1) X = -X
  150 DATAX(INDEX) = X
      NUML = 0
      NUMR = 0
      NL = 0
      NR = 0
      LEFT = 1
      ISIGN = 0
      JSIGN = 0
      IEXP = 0
      NEXP = 0
      NUMEXP = 0
      INDEX = INDEX + 1
      GO TO 210
  160 X = NUML
      IF(JSIGN.EQ.1) NUMEXP = -NUMEXP
      IF(IEXP.EQ.1) X = X*(10.)**NUMEXP
      IF(ISIGN.EQ.1) X = -X
      GO TO 150
  170 LEFT = -1
      GO TO 210
  180 IEXP = 1
      GO TO 210
  190 NUMEXP = 10*NUMEXP + NUM
      NEXP = NEXP + 1
      GO TO 210
  200 IF(IEXP.EQ.0) ISIGN = 1
      IF(IEXP.NE.0) JSIGN = 1
  210 CONTINUE
      RETURN
      END
```

The MOD2056 correction set corrected two problems with the program. The DATA statement for initializing the AEROCO common block was moved from subroutine AEROIN to program TOLA. In subroutine ACTINIT, the EQUIVALENCE statement for GREFF was corrected by replacing DM15(16) with DM5(16). The size of the ACTDIR common block was increased from 75 to 77 and the variables REDSLP and DSTOP were added to the list of active input variables.

The MOD2075 correction set remedied a sign problem with the strut force. In subroutine ALGEAR1, the statement label 159 was replaced with the following code.

```
159 SIGX = 0.
    IF(SD1(1,I).EQ.0.) GO TO 471
    SIGX = SD1(1,I)/ABS(SD1(1,I))
```

The final continuation line in the definition of FORSST was replaced with the following.

```
1 + CFFOR(I))*SIGX*DMTANH(I) + FSTOP(I))
```

Identical changes were made in the ACTNG routine.

During conduct of the experimental program reported in reference 9, it became apparent that the original control philosophy (see reference 10) was not adequate to control the gear during more realistic landing simulations as opposed to restrained vertical drop testing. For example, if the airplane rebounded from the initial touchdown impact and the gear shock strut fully extended, the

original control laws would permit the control to add fluid to the strut and result in the development of excessive strut pressure. Consequently the control laws were modified to deactivate the control if the gear should become fully extended. As a result the logic and equations programmed in the active gear, flexible airframe takeoff and landing analysis computer program had to be modified to control the servovalve in order to return the strut and servovalve parameters to initial conditions to accommodate subsequent impacts. The MOD2203 and MOD2235 correction sets incoporated the new logic into the program. The size of the XALGEA and XACTNG common blocks were increased to provide for several logic control flags and the size of the restart common block was increased to 11364. The following code replaced the definition of ISTROK below statement 500.

```
      IF(S(1,I).GT.ES(I))287,289
  287 ISTROK(I)=1
      ICU(I)=0
      IQCU(I)=0
      IXS(I)=0
C     THE FOLLOWING LOGIC RETURNS THE GEARS,DURING REBOUND, TO INITIAL
C     CONDITIONS IN THE EVENT THE GEAR CONTACTS THE SURFACE BEFORE
C     THE LOGIC BETWEEN STATEMENTS 226 AND 421 IS FULLY EFFECTIVE
  289 IF(ISTROK(I).EQ.1.AND.DDELTA(I).LT.0.0)IGO(I)=1
      IF(IGE(I).EQ.1.OR.ITRIP(I).EQ.1)227,297
  227 IF(IGO(I).EQ.0)GO TO 226
      IF(DELTA(I).GT.0.0.AND.WLFOR(I).EQ.0.0)220,226
  220 IF(VCUM(I).GT.0.00001.OR.VCUM(I).LT.-0.00001)221,222
  221 QO(I)=-VCUM(I)/DSTOP
      GO TO 226
  222 PGA2T(I)=PGA1I(I)
      QO(I)=0.0
      IGO(I)=0
  226 IF(IGE(I).EQ.1)290,297
```

The code between statements 299 and 400 was replaced with the following.

```
 312  IF(ICOSV(I).NE.1)313,314
 314  IF(XVALVE(I).NE.XBIAS(I))GO TO 311
 298  IF(ICOSV(I).EQ.1)GO TO 311
 313  IF(PGA1T(I).GT.PGA1I(I)+2000.)292,293
 292  IF(IXSVL(I).EQ.1)GO TO 311
      XSVDOT(I)=XSVDMN(I)*PERCNT(I)
      IPASS(I)=1
      XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
      IF(XVALVE(I).LE.-0.13)300,303
 303  XSV(I)=XVALVE(I)
      GO TO 297
 300  XVALVE(I)=-0.13
      XSVDOT(I)=0.0
      IXSVL(I)=1
      GO TO 294
 293  IF(PGA1T(I).LT.PGA1I(I)-2000.)295,294
 295  IF(IXSVH(I).EQ.1)GO TO 311
      XSVDOT(I)=XSVDMX(I)*PERCNT(I)
      IPASS(I)=2
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
      IF(XVALVE(I).GE.0.13)302,304
 304  XSV(I)=XVALVE(I)
      GO TO 297
 302  XVALVE(I)=0.13
      IXSVH(I)=1
 294  CONTINUE
      XSVDOT(I)=0.0
      ICOSV(I)=1
 311  IF(PGA2T(I).GT.PGA1I(I)+4000.0)316,315
 316  IF(QSV(I).LT.QO(I).AND.ICOSV(I).EQ.1)317,318
 317  XSVDOT(I)=0.0

      XSV(I)=XVALVE(I)
      GO TO 297
 318  XSVDOT(I)=XSVDMN(I)*PERCNT(I)
      XSV(I)=XVALVE(I)
      GO TO 297
 315  CONTINUE
      IF(NAC(I).EQ.1)GO TO 307
 320  IF(IIXSVH(I).EQ.1)GO TO 305
      XSVDOT(I)=XSVDMN(I)
      IPASS(I)=3
```

```
      XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT
      IF(XVALVE(I).LE.XBIAS(I))305,306
306   XSV(I)=XVALVE(I)
      GO TO 297
305   XVALVE(I)=XBIAS(I)
      XSVDOT(I)=0.0
      IIXSVH(I)=1
      GO TO 400
307   IF(IIXSVL(I).EQ.1)GO TO 308
      XSVDOT(I)=XSVDMX(I)
      IPASS(I)=4
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT
      IF(XVALVE(I).GE.XBIAS(I))308,309
309   XSV(I)=XVALVE(I)
      GO TO 297
308   XVALVE(I)=XBIAS(I)
      XSVDOT(I)=0.0
      IIXSVL(I)=1
```

The following new code was added near the call to LIMITS.

```
      DELTX1(I)=0.0
      ICOSV(I)=0
      QO(I)=0.0
      QSVCU(I)=0.0
      QSV1(I)=0.0
      QSV3(I)=0.0
      VCUM(I)=0.0
      VOL1T(I)=VOL1I(I)
      VOL2T(I)=VOL2I(I)
      VOL3T(I)=VOL3I(I)
      ITRIP(I)=1
      PGA1T1(I)=PGA1I(I)
      PGA1T(I)=PGA1I(I)
      PGA2T(I)=PGA1I(I)
      PGA3T(I)=PGA1I(I)
```

The 5 lines of code beginning with the call to PHLOZ2 were replaced with the following.

```
297   IF(ITRIP(I).EQ.1)430,421
430   IF(VCUM(I).GT.0.00001.OR.VCUM(I).LT.-0.00001)GO TO 431
      QO(I)=0.0
      ICU(I)=1
      GO TO 440
431   IF(VCUM(I).LT.-0.00001)GO TO 432
      QO(I)=-VCUM(I)/DSTOP
      GO TO 440
432   QO(I)=-VCUM(I)/DSTOP
440   IF(QSVCU(I).GT.0.00001.OR.QSVCU(I).LT.-0.00001)GO TO 433
      QSV(I)=0.0
      IQCU(I)=1
      GO TO 420
433   IF(QSVCU(I).LT.-0.00001)GO TO 434
      QSV(I)=-QSVCU(I)/DSTOP
      GO TO 420
434   QSV(I)=-QSVCU(I)/DSTOP
420   IF(XSV(I).GT.XBIAS(I)+.000001)422,423
422   IF(XSV(I).GT. 0.0)XSVDOT(I)=-XSV(I)/DSTOP
      IF(XSV(I).LT. 0.0)XSVDOT(I)=XSV(I)/DSTOP
      GO TO 600
423   IF(XSV(I).LT.XBIAS(I)-.000001)424,425
424   XSVDOT(I)=-XSV(I)/DSTOP
      GO TO 600
425   XSVDOT(I)=0.0
      IXS(I)=1
600   IF(XMA1(I).GT.0.00001 .OR.XMA1(I).LT.-0.00001)GO TO 601
      XMA1DT(I)=0.0
      IA1(I)=1
      GO TO 602
601   XMA1DT(I)=-XMA1(I)/DSTOP
602   IF(XMA2(I).GT.0.00001 .OR.XMA2(I).LT.-0.00001)GO TO 603
      XMA2DT(I)=0.0
      IA2(I)=1
      GO TO 604
603   XMA2DT(I)=-XMA2(I)/DSTOP
604   IF(XMA3(I).GT.0.00001 .OR.XMA3(I).LT.-0.00001)GO TO 605
      XMA3DT(I)=0.0
      IA3(I)=1
      GO TO 606
605   XMA3DT(I)=-XMA3(I)/DSTOP
606   IF(XMA4(I).GT.0.00001 .OR.XMA4(I).LT.-0.00001)GO TO 607
      XMA4DT(I)=0.0
      IA4(I)=1
      GO TO 608
607   XMA4DT(I)=-XMA4(I)/DSTOP
608   IF(XMA6(I).GT.0.00001 .OR.XMA6(I).LT.-0.00001)GO TO 609
      XMA6DT(I)=0.0
      IA6(I)=1
      GO TO 610
609   XMA6DT(I)=-XMA6(I)/DSTOP
```

```
 610  IF(XMA7(I).GT.0.00001 .OR.XMA7(I).LT.-0.00001)GO TO 611
      XMA7DT(I)=0.0
      IA7(I)=1
      GO TO 612
 611  XMA7DT(I)=-XMA7(I)/DSTOP
 612  IF(XMA9(I).GT.0.00001 .OR.XMA9(I).LT.-0.00001)GO TO 613
      XMA9DT(I)=0.0
      IA9(I)=1
      GO TO 614
 613  XMA9DT(I)=-XMA9(I)/DSTOP
 614  IF(XMA10(I).GT.0.00001 .OR.XMA10(I).LT.-0.00001)GO TO 615
      XMA10D(I)=0.0
      IA10(I)=1
      GO TO 421
 615  XMA10D(I)=-XMA10(I)/DSTOP
C     THESE SWITCHES ARE EITHER ZERO OR ONE
 421  IF(ICU(I)+IQCU(I)+IXS(I)+IA1(I)+IA2(I)+IA3(I)+IA4(I)+IA6(I)
     1   +IA7(I)+IA9(I)+IA10(I) .EQ. 11)ISTROK(I)=0
      I2=2*I+NSTRUT-1
```

The test on HMM(I) below the definition of VELDEC(I) was replaced with the following code.

```
      IF(IGE(I).EQ.0)GO TO 131
      WLFOR(I)=WLFORR
      INDEACT(I)=2
      EPSILO(I)=EPSROL(I)
      GO TO 451
131   IF(S(1,I).LE.ES(I))GO TO 451
      IF(HMM(I).EQ.0)GO TO 451
```

The following test was inserted before statement 452,

```
      IF(S(1,I).LE.ES(I).AND.ITRIP(I).EQ.1)GO TO 458
```

and the following code was added after the definition of P1(I).

```
 458  P1(I)=PGA1T(I)/144.
 459  VOLANT(I)=VOLANT(I)+QSVN(I)*DELT-QPUMPS(I)*DELT
```

The first line of the call to FLOZE2 was replaced with the following.

```
      IF(S(1,I).LE.ES(I).AND.ITRIP(I).EQ.1)462,410
 410  CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      CALL FLOZE2(PS(I),PR(I),P1(I),XLPSV1(I),XLPSV3(I),RCLSV(I),DSV(I),
```

The following initializations were added below statement 50,

```
      IGE(I)=0
      ITRIP(I)=0
```

and the following tests were inserted above statement 55.

```
      IF(IMODE(I).EQ.0)GO TO 55
      IF(ISTROK(I).EQ.1)IGE(I)=1
```

The following calls were inserted below the call to INTEG for XSVDOT(I).

```
      CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      CALL LIMITS(XSVDOT(I),XSVDD(I),XSVDMX(I),XSVDMN(I))
      CALL LIMITS(XSVDD(I),XSVDDD(I),XDDMAX(I),XDDMIN(I))
```

Similar modifications were required in the ACTING subroutine.

The MOD2235 correction set corrected an error in the MOD2203 logic by replacing the definition of IGO(I) below statement 222 with the following.

```
IF(DELTA(I).GT.0.) IGO(I) = 0
```

# 3. USER INFORMATION

In order to exercise the new program options, the user needs to be aware of the new data preparation requirements and operating instructions. These two areas are discussed below.

3.1 Data Preparation - The overall input data format is changed by the introduction of a new data card which must precede all other data. The first card of each data deck controls the restart option and must contain a single integer in the first card column. This card must be present even if the restart capability is not selected. allowed values of the integer restart flag are presented below:

| Value | Action |
|---|---|
| 0 | Reject the restart option - This run will read all input data from cards and a subsequent restart will not be possible. |
| 1 | This run will read all input data from cards and will create a file such that the job may be restarted from any point at which data is staged. |
| 2 | This run is a restart of an earlier run. All program variables will be initialized from the restart file to the values corresponding to the desired stage. Subsequent data cards on the input file are limited to stage data only. This run may not be restarted at subsequent data stages. |

3 This run is the same as for a value of 2 except this run will create a file such that the job may be restarted again from any subsequent point at which data is staged.

Active control simulations are run simply by inserting active control variables into the input file and setting the landing gear mode indicator switch (INDLG) to -3. The input variables are described in Table 1 and a sample listing of these inputs are given in Figure 3. The active control variable list must be preceded by the word ACTIVE starting in column 1. All following variables are entered according to the same format:

| Column Number | |
|---|---|
| 1-6 | VARIABLE NAME |
| 8-10 | DATA TYPE - DEC or blank = REAL<br>INT = INTEGER |
| 12-66 | DATA VALUES, LEFT JUSTIFIED, SEPARATED BY COMMAS, NO EMBEDDED BLANKS |

Unlike other program data, the integer counter for continuation of array data must be right justified in column 68 instead of left justified in column 67. The arrays PINM, PINN, STROM, and STRON are only required if KAPT = 2. If included, these arrays must be placed in the active variable list and are each allowed a maximum length of 30. The independent variable (strut stroke)

```
1
STCASE TAB
ATAB01      2
ATAB02      2
ATAB05      2
ATAB06      2
ATAB08      2
ATAB10      2
ATAB11      2
ATAB12      2
ATAB15      2
ATAB16      2
ATAB51      2
ATAB52      2
ATAB53      2
ATAB56      2
ATAB57      2
ATAB59      2
ATAB80      2
CTAB01      207
CTAB02      207
CTAB03      11
CTAB04      11
FTAB02      99
FTAB03      25
TTAB10      41
VTAB01      5
VTAB02      5
VTAB03      5
VTAB04      5
VTAB06      5
        TRA
STCASE
REM     BCD 4F4 ACTIVE GEAR PROGRAM
NCASE   BCD 1LRPG01
REM     BCD 4INTEGRATION INFORMATION
TVARBH      0
TMAX        .31
AMINER      .00005
PRTMIN      0.
AMAXER      .001
DELTS       .01
PRINT       .01
REM     BCD 4DATA REQD FOR SDF-2
AMASS       1115.7
HGCTF       8.23743
THTRD       7.1
GAMZD       -1.14576
VGZZF       250.0
```

Figure 3-a. Full Input Deck

```
OI77R       0.
XC77F       0.
RWHGR       0.
TNDAPC      1
TNDADD      1
TNDEPA      1
TNDPLA      1
TNDACM      1
TNDCRT      1
TNDWGT      0
REM     BCD 5VEHICLE PHYSICAL PROP. DATA
TNDVPC      1
XCGRF       0.
VTAB01      2,0.,0.,5000.,0.
VTAB02      2,0.,24873.,1200.,24873.
VTAB03      2,0.,144516.,1200.,144516.
VTAB04      2,0.,154897.,1200.,154897.
TNDX7S      1
VTAB06      2,0.,3361.,1200.,3361.
REM     BCD 4AERODYNAMIC INPUT DATA
TNDAER      1
AREFF       547.87
D1REF       16.042
D2REF       38.667
TNDA01      1
ATAB01      -0.000709,-0.000709
TNDA02      1
ATAB02      0.000423,0.000423
TNDA05      1
ATAB05      -0.001027,-0.001027
TNDA06      1
ATAB06      0.000136,0.000136
TNDA08      0
ATAB08      0.,0.
TNDA10      1
ATAB10      .379202,.379202
TNDA11      1
ATAB11      .063627,.063627
TNDA12      1
ATAB12      -.00018,-.00018
TNDA15      1
ATAB15      .0016,.0016
TNDA16      1
ATAB16      -.0002,-.0002
TNDA51      1
ATAB51      -.083516,-.083516
TNDA52      1
ATAB52      -.005969,-.005969
TNDA53      1
```

Figure 3-a. Cont'd

```
ATAB53      -.000216,-.000216
INDA56      1
ATAB56      -.010317,-.010317
INDA57      1
ATAB57      -.000005,-.000005
INDA58      0
ATAB58      .0,.0
INDAB0      1
ATABB0      0.141245,0.141245
REM     BCD 3ENGINE THRUST DATA
INDTEF      3
INDTSO      1
TT10X       6
TT10W       5
TTAB10      -2.,-1.,0.,1.,2.
TTAB10      0.,.05,.1,.15,.2,.25                                        6
TTAB10      0.,0.,0.,470.,17600.                                        12
TTAB10      0.,0.,0.,300.,17300.                                        17
TTAB10      0.,0.,0.,220.,17200.                                        22
TTAB10      0.,0.,0.,160.,17080.                                        27
TTAB10      0.,0.,0.,0.,17080.                                          32
TTAB10      0.,0.,0.,0.,17100.                                          37
REM     BCD 3LANDING GEAR DATA
ACTIVE
AMUH        .00018
STRON       0.0,.1,.18333,.26667,.35,.43333,.51667,.6,.68333,.76667
STRON       .85,.93333,1.01667,1.1,1.18333,1.26667,1.35,1.43333         11
STRON       1.51667,1.6,1.68333,1.76667,1.85,1.93333,2.0                19
PINN        .00231,.00231,.00231,.00204,.00242,.003,.00339,.00352
PINN        .00396,.00435,.00454,.00473,.00492,.00511,.00511,.00511     9
PINN        .00531,.00531,.0055,.00569,.00601,.00627,.00646,.00665      17
PINN        .00665                                                      25
STRDM       0.0,.06604,.10771,.14938,.19104,.23271,.27438,.31604
STRDM       .35771,.39938,.44104,.48271,.52438,.56604,.60771,.64938     9
STRDM       .69104,.73271,.77438,.81604,.85771,.89938,.94104,.9827117
STRDM       1.02438,1.06604,1.10771,1.14938,1.32333                     25
PINM        .00603,.00595,.00572,.0055,.00518,.00492,.00458,.00424
PINM        .0039,.00373,.00356,.00366,.00375,.00409,.00443,.0046       9
PINM        .00486,.00511,.00528,.00554,.00571,.00588,.00614,.0063117
PINM        .00648,.00665,.00674,.00682,.00682                          25
AREA1       .08705,.1134,.1134
AREA2       .12962,.16082,.16082
AREA3       .01651,.00819,.00819
AREMO       .00684,.00684,.00684
AREO2       .000021,.000668,.000668
BETA        14400000.
BLMU        .15,.15,.15
BUMU        .15,.15,.15
CDMOC       .9,.9,.9
```

Figure 3-a. Cont'd

```
CDMOF       .9,.9,.9
CDSV        .62,.62,.62
CD3         .6,.6,.6
CFFOR       50.,50.,50.
DELT        .0001
DTOTA       1.
DSV         1.1875,1.1875,1.1875
EPSILO      200.,200.,200.
EPSROL      2000.,2000.,2000.
EPSSLP      0.
ETASV       .436
FWORK       .9,.9,.9
GAMAH       52.36,52.36,52.36
GNP         0.
KAPT    INT 2,2,2
OMRUN       0.
PATM        2116.8
PGAHAC      432000.,432000.,432000.
PGALAC      0.,0.,0.
PGA1I       16272.,28800.,28800.
PGA2I       16272.,28800.,28800.
PGA3I       16272.,28800.,28800.
PERCNT      1.66115,1.66115,1.66115
PPUMPS      0.05793,0.05793,0.05793
RCLSV       4.75E-5,4.75E-5,4.75E-5
RHOH        1.626
TAUF        0.1
TC1         .281
TC2         .141
TC3         .001
TC4         .0001
VOLACI      1.33333,1.33333,1.33333
VOLANI      .26667,.26667,.26667
VOL1I       .11733,.11507,.11507
VOL2I       .28923,.26239,.26239
VOL3I       0.,0.,0.
WC          1263.
WC1         251.3
WSV         655.5
WSV1        3.48
WSV3        3.48
WLFOR       500.,500.,500.
WLFORP      0.,0.,0.
XBIAS       -.000227,-.000171,-.000171
XDDMAX      1.0E20,1.0E20,1.0E20
XDDMIN      -1.0E20,-1.0E20,-1.0E20
XKA         .04,.04,.04
XKF         60.,60.,60.
XKSV        .0025,.0025,.0025
```

Figure 3-a. Cont'd

```
     YLPSV1        0.,0.,0.
     XLPSV2        0.,0.,0.
     XSCOM         1.54667,1.15667,1.15667
     XSTHR         .02083,.02083,.02083
     XSVDMN        -30.1,-30.1,-30.1
     XSVDMX        30.1,30.1,30.1
     XSVMAX        .1,.1,.1
     XSVMIN        -.1,-.1,-.1
     ZETAC1        5.1
     ZETAC2        .1
     IMODE   INT   0,1,1
     ENDACT
     NSTRUT        3
     MASS          4.28971,13.98819,13.98819
     RX            19.57503,-4.34308,-4.34308
     RY            0.,-8.95333,8.95333
     RZ            1.40617,1.41425,1.41425
     THETAD        0.,4.29137,4.29137
     ERDEG         0.0
     CONEM         23228.5
     CTWOM         69883.8
     CTHREM        3180.1
     CFOURM        6581.4
     CONERM        .1683
     DELPPM        .4173
     VPOWRM        0.0
     AH            .08705,.11020,.11020
     RGR           0.
     NTIRES        2.,1.,1.
     RZERO         .73333,1.21875,1.21875
     W             .46042,.9375,.9375
     DELTAM        .21666,.36875,.36875
     RLT           1.5E+4
     IED           1
     AT            1.06148E+5,1.94542E+5,1.94542E+5
     RT            1.3,1.2,1.2
     SLEN1         1.04037,.78723,.78723
     SLEN2         3.0122,3.095,3.095
     INDATM        1
     GAMA          1.06
     CONE          6483.9
     CTWO          32406.4
     CTHREE        534.12
     CFOUR         1837.1
     CONERI        .1683
     DELPWR        .4173
     DTIRE         1.95208
     ETAMAX        25.
     ETAMIN        -25.
```

Figure 3-a. Cont'd

```
EA          30.
ETANOS      0.0
ETART1      20.
INDNWS      0
KANWS       0
KPNWS       0
ISTER       0
TRUDD       0
TNWPUD      0
PCTETA      1
PSIDES      0.0
RPSI        0.0
RYP         10.
VPOWP       0.0
YRDES       0.0
ETAR02      2,0.,0.,1.5E+5,0.
ETAR03      11,0.,.0005,.03,.16,.06,.32,.12,.64,.15,.8
ETAR03      .2,.8,.4,.72,.6,.52,.8,.36,1.,.336,100.,.336                    12
MOMENT      .53703,8.019,8.019
MB          0.,0.,0.
PF          6.0108,5.1525,5.1525
V7          0.
P7ERD       16272.,28800.,28800.
V7ERD       .09343,.22793,.22793
A           .08705,.11020,.11020
TL          1
INDC01      1
NSMAIN      51
CTAR01      .0,.066,.1,.108,.149,.183,.191,.233,.267,.274,.316
CTAR01      .35,.358,.399,.433,.441,.483,.517,.524,.566,.6,.608          12
CTAR01      .649,.683,.691,.733,.767,.774,.816,.85,.858,.899,.933        23
CTAR01      .941,.983,1.017,1.024,1.066,1.1,1.108,1.149,1.183            34
CTAR01      1.267,1.323,1.351,1.433,1.517,1.6,1.683,1.767,2.0            43
CTAR01      1.,2.,3.,32.3,32.3,29.1,29.1,29.1,27.4,27.4                  52
CTAR01      27.4,28.1,28.1,28.1,33.9,33.9,33.9,44.9,44.9,44.9,55.4       62
CTAR01      55.4,55.4,62.1,62.1,62.1,79.2,79.2,79.2,106.2,106.2          73
CTAR01      106.2,124.7,124.7,124.7,148.4,148.4,148.4,179.5,179.5        83
CTAR01      179.5,221.6,221.6,221.6,221.6,221.6,221.6,280.5,280.5        92
CTAR01      366.3,498.6,1121.9,1994.5,17950.5                            101
CTAR01      2090.,1690.,1690.,1184.,773.,773.,482.,366.,366.,264.        106
CTAR01      199.,199.,155.,139.,139.,125.,133.,133.,141.,172.,172.       116
CTAR01      231.,290.,290.,342.,451.,451.,555.,795.,795.,1054.           127
CTAR01      1463.,1463.,2725.,4756.,4756.,10320.,37168.,37168.           137
CTAR01      122381.,3559184.,3559184.,3559184.,3559184.,3559184.         145
CTAR01      3559184.,0.,0.,0.,0.,0.,2090.,1690.,1690.,1184.              151
CTAR01      773.,773.,482.,366.,366.,264.,199.,199.,155.,139.,139.       161
CTAR01      125.,133.,133.,141.,172.,172.,231.,290.,290.,342.,451.       172
CTAR01      451.,555.,795.,795.,1054.,1463.,1463.,2725.,4756.,4756.183
CTAR01      10320.,37168.,37168.,122381.,3559184.,3559184.,3559184.193
```

Figure 3-a. Cont'd

```
CTABO1      3559184.,3559184.,3559184.,0.,0.,0.,0.,0.                    200
INDCO2      1
CTABO2      .0,.066,.1,.108,.149,.183,.191,.233,.267,.274,.316
CTABO2      .35,.358,.399,.433,.441,.483,.517,.524,.566,.6,.608          12
CTABO2      .649,.683,.691,.733,.767,.774,.816,.85,.858,.899,.933        23
CTABO2      .941,.983,1.017,1.024,1.066,1.1,1.108,1.149,1.183            34
CTABO2      1.267,1.323,1.351,1.433,1.517,1.6,1.683,1.767,2.0            43
CTABO2      1.,2.,3.,32.3,32.3,29.1,29.1,29.1,27.4,27.4                  52
CTABO2      27.4,28.1,28.1,28.1,33.9,33.9,33.9,44.9,44.9,44.9,55.4       62
CTABO2      55.4,55.4,62.1,62.1,62.1,79.2,79.2,79.2,106.2,106.2          73
CTABO2      106.2,124.7,124.7,124.7,148.4,148.4,148.4,179.5,179.5        83
CTABO2      179.5,221.6,221.6,221.6,221.6,221.6,221.6,280.5,280.5        92
CTABO2      366.3,498.6,1121.9,1994.5,17950.5                            101
CTABO2      2090.,1690.,1690.,1184.,773.,773.,482.,366.,366.,264.        106
CTABO2      199.,199.,155.,139.,139.,125.,133.,133.,141.,172.,172.       116
CTABO2      231.,290.,290.,342.,451.,451.,555.,795.,795.,1054.           127
CTABO2      1463.,1463.,2725.,4756.,4756.,10320.,37168.,37168.           137
CTABO2      122381.,3559184.,3559184.,3559184.,3559184.,3559184.         145
CTABO2      3559184.,0.,0.,0.,0.,0.,2090.,1690.,1690.,1184.              151
CTABO2      773.,773.,482.,366.,366.,264.,199.,199.,155.,139.,139.       161
CTABO2      125.,133.,133.,141.,172.,172.,231.,290.,290.,342.,451.       172
CTABO2      451.,555.,795.,795.,1054.,1463.,1463.,2725.,4756.,4756.183
CTABO2      10320.,37168.,37168.,122381.,3559184.,3559184.,3559184.193
CTABO2      3559184.,3559184.,3559184.,0.,0.,0.,0.,0.                    200
INDCO3      1
NS2NDY      2
CTABO3      0.,.1542,1.,2.,3.
CTABO3      0.,0.,28.3,28.3,28.3,28.3                                    6
INDCO4      1
CTABO4      0.,.1542,1.,2.,3.
CTABO4      0.,0.,28.3,28.3,28.3,28.3                                    6
MASS2       .0001,.0986,.0986
ES          .02,.013,.013
MUS         .15,.15,.15
SB          2.,1.323,1.323
REM    BCD 4FLEXIBLE AIRFRAME DATA
INDFLX      0
NMODE       5
GMASS1      22.68,2.88,14.28,5.88,19.92
GFREQ       50.868,64.684,80.384,116.180,133.764
GDAMP       28.8574,4.6597,27.3655,17.0862,9.5505
SZMOD       .535,.347,.347,-.333,.333,.333,-.411,.115,.115
SZMOD       .294,-.282,-.282,.443,.117,.117                              10
ARMODE      0.,0.,.05,0.,0.,0.,0.,0.,.037,0.,0.,0.
ARMODE      0.,0.,-.075,0.,0.,0.,0.,0.,-.050,0.,0.,0.                    13
ARMODE      0.,0.,-.001,0.,0.,0.                                         25
DCMODE      0.,0.,.987,0.,0.,-.88,0.,0.,-.964
DCMODE      0.,0.,.541,0.,0.,-.647                                       10
PF          0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.
```

Figure 3-a. Cont'd

```
PF          0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.                          13
PF          0.,0.,0.,0.,0.,0.                                            25
NPTS        5
OUTMOD      0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,.533,-.333,-.411,.294
OUTMOD      .443,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,-.5,-.083,-.23,.059       15
OUTMOD      .135,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,-.333,.308,.172           30
OUTMOD      -.294,-.004,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,.347,.333          44
OUTMOD      .115,-.282,.117,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,.7,.067        58
OUTMOD      -.326,-.062,-.241                                            73
ROTS        19.575,0.,1.40617,15.56617,0.,-.10,0.,0.,0.,-4.34758
ROTS        -8.95333,1.40617,-24.71125,0.,3.025                          11
REM     BCD 3PLOT TAPE DATA
IPLT        1
ISDF        1
ISTPL1      1
ISTPL2      1
ISTPL3      1
IFLX    INT 1,1,1,1,1
REM     BCD 3AUTOPILOT DATA
REM     BCD 3A. ENGINE DATA
INDAUT      1
TN          2
ZN          .04025,.04025
YN          -2.7,2.7
N           0.,0.
REM     BCD 3B. DRAG CHUTE DATA
ICS         0
CDCH        .244
SSH         160.8
XCH         -36.575
YCH         0.
ZCH         -2.564
REM     BCD 5C. PHASE BEGIN-TERMINATE
TTD         0
HF          500.0
NF          0
XRF         0.
HRF         8.23743
DELTAH      3.
NLRI        0
TT          0.0
KP          1
VS          10.0
XS          6000.
TS          60.
HS          0.
REM     BCD 4G. HOLD MANEUVER DATA
ALPDES      8.1
TTD         470.0
```

Figure 3-a. Cont'd

```
KE      INT 0,0
PM          10.0
REM     BCD 5H. LANDING ROLL MANEUVER DATA
TSP         .5
TRV         .1
TCH         4.
TRK         3.
TSS         0.
TLR         0
TRS         0
REM     BCD 5I. ENGINE FAILURE STAGE DATA
TC      INT 1,1
YREF1       0.
IT1     INT 1,1
YREF2       0.
IT2     INT 1,1
H1          0.
IH1     INT 1,1
H2          0.
IH2     INT 1,1
HR1         0.
IHR1    INT 1,1
HR2         0.
IHR2    INT 1,1
TR1         0.
ITR1    INT 1,1
TR2         0.
ITR2    INT 1,1
REM     BCD 5J. BRAKE COND.STAGE DATA
IB      INT 0,0,0
TRK1        100.
IRK1    INT 0,1,1
TRK2        100.
IRK2    INT 0,1,1
REM     BCD 4K. PITCH AUTOPILOT DATA
TST         0.
ALPDL       2.
REALPH      0.
DELALA      .01
PSH         0.
PSH2        0.
REALP2      0.
DELQF       0.
DELQTC      0.
DELQU       7.
DELQL       -21.
DELFD1      0.
REM     BCD 4L. YAW AUTOPILOT DATA
REB         .5
```

Figure 3-a. Cont'd

```
DELRA       .05
PSP         10.
DPSTA       .05
REPSI       .5
PSPSI       0.
DELRL       -30.
DELRU       30.
REM     BCD 4M. ROLL AUTOPILOT DATA
REPHI       .5
DPHIA       .05
PSA         -10.
DELPL       20.
DELPU       -1.
REM     BCD 5N. THROTTLE AUTOPILOT DATA
NDF         1.,1.
TF      INT 1,1
TR      INT 0,0
NR          0.,0.
NLR         0.,0.
NTD         2.,2.
K2          .5
REM     BCD 40. BRAKE AUTOPILOT DATA
MBC         0.,0.,0.
PD          .155
DELTAW      .1
OMEGD1      20.
MBL         0.,0.,0.
MBU         1.0E+4,1.0E+4,1.0E+4
REM     BCD 5P. CONTROL RESPONSE DATA
DELHS       25.
DELPPD      30.
DELA        120.
NED1        .1
REM     BCD 40. INITIALIZATION
TAP         4
HR          8.23743
DELON       -12.
DELODE      -12.
DELOD       -12.
DELOU       7.
DELOL       -21.
DELOF       -21.0
DELPD       0.
DELRD       0.
MANLOG      1
PITCHP      1
ROLLAP      1
THRDAP      1
CONTRP      1
```

Figure 3-a. Cont'd

```
AUXICP      1
REM     BCD 2STAGING DATA
REM     BCD 4A.GEARS INTO PROGRAM
TNDLC       -3
AINCRS  BCD 1TIME
STEST       .005
        TRA
STCASE
DELED1      20.
AINCRS  BCD 1TIME
STEST       .02
        TRA
STCASE
PRINT       .001
AINCRS  BCD 1TIME
STEST       .04
        TRA
STCASE
PRINT       .01
AINCRS  BCD 1TIME
STEST       .3
        TRA
STCASE
PRINT       .01
AINCRS  BCD 1DELTA1
STEST       .05
        TRA
STCASE
ICS         1
DELOF       0.
DELED1      25.
AINCRS  BCD 1TIME
STEST       .60
        TRA
STCASE
ATAB10      .35,.35
ATABRO      .15,.15
AINCRS  BCD 1TIME
STEST       .70
        TRA
STCASE
AMAXER      .0001
ATAB10      .325,.325
ATABRO      .16,.16
AINCRS  BCD 1TIME
STEST       .80
        TRA
STCASE
ATAB10      .30,.30
```

Figure 3-a. Cont'd

```
ATARRO      .17,.17
ATNCRS BCD 1TIME
STEST       .90
        TRA
STCASE
ATAB10      .275,.275
ATARRO      .18,.18
AINCRS BCD 1TIME
STEST       1.0
        TRA
STCASE
ATAB10      .25,.25
ATARRO      .19,.19
ATNCRS BCD 1TIME
STEST       1.1
        TRA
STCASE
ATARRO      .20,.20
AINCRS BCD 1TIME
STEST       1.2
        TRA
STCASE
ATARRO      .21,.21
AINCRS BCD 1TIME
STEST       1.3
        TRA
STCASE
ATARRO      .22,.22
AINCRS BCD 1TIME
STEST       1.4
        TRA
STCASE
ATARRO      .23,.23
AINCRS BCD 1TIME
STEST       1.5
        TRA
STCASE
ATARRO      .24,.24
AINCRS BCD 1TIME
STEST       1.6
        TRA
STCASE
ATARRO      .25,.25
AINCRS BCD 1TIME
STEST       4.0
        TRA
STCASE
AMINER      .00005
AMAXER      .001
```

Figure 3-a. Cont'd

```
3
STCASE
AMAXER      .0005
ATAB10      .35,.35
ATAB80      .15,.15
ATNCRS BCD 1TIME
STEST       .70
        TRA
STCASE
AMAXER      .0001
ATAB10      .325,.325
ATAB80      .16,.16
ATNCRS BCD 1TIME
STEST       .80
        TRA
STCASE
ATAB10      .30,.30
ATAB80      .17,.17
ATNCRS BCD 1TIME
STEST       .90
        TRA
STCASE
ATAB10      .275,.275
ATAB80      .18,.18
ATNCRS BCD 1TIME
STEST       1.0
        TRA
STCASE
ATAB10      .25,.25
ATAB80      .19,.19
ATNCRS BCD 1TIME
STEST       1.1
        TRA
STCASE
ATAB80      .20,.20
ATNCRS BCD 1TIME
STEST       1.2
        TRA
```

Figure 3-b. Short Input Staging Deck for a restart at 0.60 second.

```
STCASE
ATABRO      .21,.21
AINCRS BCD  1TIME
STEST       1.3
       TRA
STCASE
ATABRO      .22,.22
AINCRS BCD  1TIME
STEST       1.4
       TRA
STCASE
ATABRO      .23,.23
AINCRS BCD  1TIME
STEST       1.5
       TRA
STCASE
ATABRO      .24,.24
AINCRS BCD  1TIME
STEST       1.6
       TRA
STCASE
ATABRO      .25,.25
AINCRS BCD  1TIME
STEST       4.0
       TRA
STCASE
AMTNER      .00005
AMAXER      .001
AINCRS BCD  1TIME
STEST       10.
       TRA
```

Figure 3-b. Cont'd

for the nose and main gears are stored in STRON and STROM, respectively. The dependent variable (metering pin area) for the nose and main gears are stored in PINN and PINM, respectively. The active variable list must be terminated with the word ENDACT, which also begins in card column 1.

The INDLG switch is located at the beginning of the staging data section and follows the same format as outlined above. It is also recommended that the minimum integration interval (AMINER) and the maximum integration interval (AMAXER) be set to .00005 and .001 respectively to permit accurate numerical integration during ground contact stages. These variables are located in the integration information section near the beginning of the data set.

In order to specify time rates of change for the aerodynamic coefficients $C_{A_0}$ and $C_{N_0}$, four new symbols have been added to the program: RTAB10, RTAB80, LTAB10, and LTAB80. These symbols, unlike ATAB10 and ATAB80,should not be included in table size data but may appear anywhere else in the input deck. As with other aerodynamic tables, the new symbols have a size of two. The first data point is for full ground effect and the second is for no ground effect. The use of these symbols is illustrated by an example.

```
ATAB10      .379202,.379202
      .
      .
ATAB80      .141245,.141245
      .
      .
AINCRS BCD 1TIME
STEST       .005
       TRA
STCASE
RTAB10      -.01,-.01
LTAB10      .36,.36
RTAB80      .005,.005
LTAB80      .15,.15
      .
      .
      .
```

In this example, the initial values of $C_{N_0}$ and $C_{A_0}$ are .379202 and .141245, respectively. At time t=.005, rates of change and limiting values for the parameters are specified. The value of $C_{N_0}$ (ATAB10) will decrease at a rate of .01 (RTAB10) until it has reached a value of .36 (LTAB10). At this time (1.9209 seconds after the stage) the rate of change will be set to zero and $C_{N_0}$ will remain at .36 until it is redefined by a new ATAB10 card or by new values of RTAB10 and LTAB10. Similarly, the value of $C_{A_0}$ (ATAB80)

will increase at a rate of .005 until it has reached a value of .15 (LTAB80). At this time (1.751 seconds after the stage) the rate of change will be set to zero and $C_{A_0}$ will remain at .15 until it is redefined.

3.2 Operating Instructions - The program resides as an UPDATE program library on an indirect access file named ACTOLA. The relocatable binaries are on a file named BTOLAR. The plot program source and binaries are on files NEWPLOT and NAPBN, respectively.

The user is responsible for maintaining the files necessary for restart runs and for plotting the cumulative results of several restart runs. The program files of concern to the user are TAPE7, TAPE16, and TAPE13. The TAPE7 file contains the restart data with one record for each restart generated (i.e., one record for each time data is staged). In subsequent runs, it is the responsibility of the user to correctly position TAPE7. The TAPE16 file also contains restart information, but contains a single record. The file TAPE13 contains data for the plot postprocessor and contains a single record. The use, storage, and manipulation of these files is best illustrated by a set of example runs.

Figure 4-a illustrates the simplest use of the FATOLA program. This run will execute the program without the restart option and no plots will be generated.

```
JOB,...
USER,...
CHARGE,...
GET,BTOLAR/UN=585787N.
MAP,OFF.
BTOLAR.
---EOR
   Full data deck (similar to Figure 3-a)
---EOF
```

Figure 4-a. Deck setup for simple execution of the program.

```
JOB,...
USER,...
CHARGE,...
GET,BTOLAR/UN=585787N.
MAP,OFF.
BTOLAR.
GET,NAPBN/UN=585787N.
ATTACH,LRCGOSF/UN=LIBRARY.
REWIND,TAPE13.
COPYEI,TAPE13,TAPE3.
RETURN,TAPE13.
REWIND,TAPE3.
LDSET,LIB=LRCGOSF.
NAPBN.
PLOT.VARIAN
---EOR
   Full data deck (similar to Figure 3-a)
---EOR
   Plot instructions
---EOF
```

Figure 4-b. Deck setup for simple execution of the program with plotting.

```
JOB,...
USER,...
CHARGE,...
GET,OLDPL=ACTOLA/UN=585787N.
UPDATE.
FTN,I.
REWIND,LGO.
GET,OLDBIN=BTOLAR/UN=585787N.
COPYL,OLDBIN,LGO,BTOLAR.
MAP,OFF.
BTOLAR.
REPLACE,TAPE13=T134C.
---EOR
   UPDATE correction set
---EOR
   Full data deck (similar to Figure 3-a)
---EOF

JOB,...
USER,...
CHARGE,...
GET,TAPE3=T134C.
GET,NAPBN/UN=585787N.
ATTACH,LRCGOSF/UN=LIBRARY.
LDSET,LIB=LRCGOSF,MAP=SBEX/OTT.
NAPBN.
PLOT.VARIAN
---EOR
   Plot instructions
---EOF.
```

Figure 4-c. Decks for executing the program with corrections and plotting the results at a later time.

```
JOB,...
USER,...
CHARGE,...
GET,BTOLAR/UN=585787N.
MAP,OFF.
BTOLAR.
REPLACE,TAPE7=T74D.
REPLACE,TAPE16=T164D.
REPLACE,TAPE13=T134D.
EXIT.
REPLACE,TAPE7=T74DF.
REPLACE,TAPE16=T164DF.
REPLACE,TAPE13=T134DF.
---EOR
   Full data deck (similar to Figure 3-a with a restart flag of 1.)
---EOF
```

Figure 4-d. Deck setup for an initialization run with the restart option.

```
JOB,...
USER,...
CHARGE,...
GET,BTOLAR/UN=585787N.
MAP,OFF.
GET,TAPE16=T164DF.
GET,TAPE7=T74DF.
SKIPR,TAPE7,5.
BTOLAR.
REPLACE,TAPE7=T74E.
REPLACE,TAPE16=T164E.
REPLACE,TAPE13=T134E.
EXIT.
REPLACE,TAPE7=T74EF.
REPLACE,TAPE16=T164EF.
REPLACE,TAPE13=T134EF.
---EOR
   Short data deck (similar to Figure 3-b with a restart flag of 3.)
---EOF
```

Figure 4-e. Deck setup for restarting example from Figure 4-d at time = 0.60 sec.

```
JOB,...
USER,...
CHARGE,...
GET,NAPBN/UN=585787N.
ATTACH,LRCGOSF/UN=LIBRARY.
MAP,OFF.
GET,T134DF.
GET,T134E.
COPYBR,T134DF,TAPE3.
COPYBR,T134E,TAPE3.
REWIND,TAPE3.
LDSET,LIB=LRCGOSF.
NAPBN.
PLOT.VARIAN
---EOR
   Plot instructions
---EOF
```

Figure 4-f. Plotting cumulative results of restart runs.

When plots are desired, they can either be generated at the same time as the program is executed, or they can be generated at a later time after the user has examined the printed output. Figure 4-b shows the deck setup for executing the program and generating plots at the same time. Figure 4-c contains the deck setups for executing the program with UPDATE modifications and plotting the output at a later time.

When the restart option is selected, the first run is known as the initialization run and the first data card should contain a 1 in the first card column. The user should save the files TAPE7 and TAPE16, and if plots will be desired TAPE13. Figure 4-d contains the deck setup for an initialization run. Note that important files are saved if the program terminates normally, and these files are also saved if the program fails. Assume, as an example, the program fails at time = 0.65 seconds, using the data of Figure 3-a. An analysis of the program output suggests that if a smaller integration step were being used, the failure might have been avoided.

To correct the problem, it is first necessary to determine the TAPE7 record structure. From our sample data or from the program output it can be determined that the first restart record was written at time = 0.005, the second was written at time = 0.02, etc. Thus, the record written at

0.60 was the sixth restart record generated. Figure 4-e illustrates a deck setup for restarting this run. Note that 5 records are skipped on TAPE7, positioning the file to the sixth record. Note also the following points:

- It is not required to get TAPE13 before executing the program.
- The permanent file names T164DF and T74DF correspond to the file names under which the files TAPE16 and TAPE7 were replaced (after the EXIT.) in Figure 4-d
- Following execution, the restart and plot files are saved with new (unique) names. In subsequent runs, T74DF could be used to restart the program prior to time = 0.60 and T74E (or T74EF) could be used to restart the program after time = 0.60.

Assume, to continue our example, the restart run terminates normally and we desire to plot the cumulative results of the two runs. Figure 4-f presents a deck setup to plot the results of our example. Notice that T134DF (after the exit) and T134E (before the exit) correspond to the file names under which TAPE13 was saved in Figures 4-d and 4-e. This example is easily extended to as many restart runs as necessary to complete the simulation. A GET is required for each TAPE13 saved and a COPYBR is also required, with the COPYBR's in the same order in which the files were created.

## REFERENCES


1. Carden, Huey D.; and McGehee, John R.: Validation of a Flexible Aircraft Takeoff and Landing Analysis (FATOLA). NASA TP-1025, 1977.

2. Carden, Huey D.; and McGehee, John R.: Improvements to the FATOLA Computer Program Including Nosewheel Steering. NASA TM-78768, 1978.

3. Dick, J. W.; and Benda, B. J.: Addition of Flexible Body Option to the TOLA Computer Program, Part I - Final Report; Part II - User Programmer Documentation. NASA CR-132732-1, Part I and NASA CR-132732-2, Part II, 1975.

4. Lynch, Urban H. D.: Takeoff and Landing Analysis (TOLA) Computer Program. Part I: Capabilities of the Takeoff and Landing Analysis Computer Program. AFFDL-TR-71-155 Part 1. 1972.

5. Lynch, Urban H. D.; and Dueweke, John J.: Takeoff and Landing Analysis (TOLA) Computer Program. Part II: Problem Formulation, Technical Report AFFDL-TR-71-155, May, 1974.

6. _____: Takeoff and Landing Analysis (TOLA) Computer Program. Part III: User's Manual. AFFDL-TR-71-155, 1974.

7. McGehee, John R.; and Carden, Huey D.: A Mathematical Model of an Active Control Landing Gear for Load Control During Impact and Rollout. NASA TN D-8080, 1976.

8. Young, Fay O.; and Dueweke, John J.: Takeoff and Landing Analysis (TOLA) Computer Program; Part IV - Programmer's Manual. AFFDL-TR-71-155, Part 4, 1975.

9. McGehee, John R.; and Dreher, Robert C.: Experimental Investigation of Active Loads Control for Aircraft Landing Gears. NASA TP 2042, 1982.

10. Ross, Irving; and Edson, Ralph: An Electronic Control for an Electrohydraulic Active Control Aircraft Landing Gear. NASA CR-3113, 1979.

Appendix A
Program Modifications

```
*IDENT    D81877
*DELETE       PLTDAT.3,PLTDAT.3
      DIMENSION TITLE(18), BUF(400), NDIL(28), TBUF(400),
*DELETE       PLTDAT.125,PLTDAT.125
   50 DO 51 I=1,18
*IDENT    C82477
*INSERT          TOLA.6
      COMMON/NWSTER/CFOUR,CONE,CONFRI,CTHREE,CTWO,DELPWR,DTIRE
     1  ,EA,ETAMAX,ETAMIN,ETANOS,ETART1,INDNWS,KANWS,KPNWS,PCTETA
     2  ,PSIDES,RPSI,RYR,VPOWR,YRDES
     3     ,DELN,ETADES,FGPY
*INSERT          TOLA.9
     1     ,CFOUR,CONE,CONFRI,CTHREE,CTWO,DELPWR,DTIRE,EA,ETAMAX
     2     ,ETAMIN,ETANOS,ETART1,INDNWS,KANWS,KPNWS,PCTETA,PSIDES,RPSI
     3     ,RYR,VPOWR,YRDES
*INSERT          TOLA.19
      READ(5,5004)CFOUR,CONE,CONFRI,CTHREE,CTWO,DELPWR,DTIRE,EA
     1     ,ETAMAX,ETAMIN,ETANOS,ETART1,PCTETA,PSIDES,RPSI,RYR,VPOWR
     2     ,YRDES
*INSERT          TOLA.20
      READ(5,5005)INDNWS,KANWS,KPNWS
 5005 FORMAT(3I5)
*INSERT       LGEA3C.2
      EXTERNAL ATAN2
*INSERT       LGEA3C.34
      COMMON/NWSTER/CFOUR,CONE,CONFRI,CTHREE,CTWO,DELPWR,DTIRE
     1  ,EA,ETAMAX,ETAMIN,ETANOS,ETART1,INDNWS,KANWS,KPNWS,PCTETA
     2  ,PSIDES,RPSI,RYR,VPOWR,YRDES
     3     ,DELN,ETADES,FGPY
*INSERT       LGEA3C.59
      EQUIVALENCE (DM18(72),PSIPD)
*INSERT       LGEA3C.61
      DATA RADDEG,DEGRAD/57.2957795,.01745329/
*DELETE       LGEA3C.143,LGEA3C.143
      IF(I.EQ.1 .AND. INDNWS.EQ.1)GO TO 200
      GO TO 7
 200  VTX(I)=VTX(I)-TMP(2)
      VTY(I)=RDYG(I)
   7  VTZ(I)=RG31*RDXG(I)+RG33*RDZG(I)
*INSERT       LGEA3C.167
      IF(DELTA1 .EQ. 0.0) FGPY=0.0
      IF(DELTA1 .EQ. 0.0) GO TO 41
      IF(INDNWS.EQ.0)GO TO 41
      IF(I.GT.1)GO TO 41
      ETAVE= ATAN2(VTY(I),VTX(I))
      DELN=PSIPD+ETADES-ETAVE*RADDEG
      IF(DELN.EQ.0.0.OR.XG77F1.EQ.0.0)GO TO 110
      SIDEMU=CONFRI*(ABS(DELN))**DELPWR*XG77F1**VPOWR
      GO TO 180
```

```
 110  SIDEMU=1.0E-6
 180  IF((DELTA1/DTIRE).GT.0.0875)GO TO 130
      CRNPWR=CONE*DELTA1-CTWO*DELTA1**2.
      GO TO 170
 130  CRNPWR=CTHREE-CFOUR*DELTA1
  170 YAWPRM=ABS(CRNPWR*DELN/(SIDEMU*FTRZ(I)))
      IF(YAWPRM.LE.1.5)GO TO 150
      FGPY=SIDEMU*FTRZ(I)*COS(DELN*DEGRAD)
      GO TO 160
 150  FGPY=((YAWPRM-(4.0/27.0)*YAWPRM**3.0)*SIDEMU*FTRZ(I))
     1     *COS(DELN*DEGRAD)
 160  FGPY=PCTETA*FGPY
      IF(DELN.GT.0.0)FGPY=ABS(FGPY)
      IF(DELN.LT.0.0)FGPY=-ABS(FGPY)
      DFTRX=FGPY*SIN(ETAVE)
      DFTRY=FGPY*COS(ETAVE)
      FTRX(I)=FTRX(I)+DFTRX
      FTRY(I)=FTRY(I)+DFTRY
*INSERT       LGEA3C.170
      FGPY = 0.0
*INSERT       SDFLGP.34
      COMMON/NWSTER/CFOUR,CONE,CONFRI,CTHREE,CTWO,DELPWR,DTIRE
     1  ,EA,ETAMAX,ETAMIN,ETANOS,ETART1,INDNWS,KANWS,KPNWS,PCTETA
     2  ,PSIDES,RPSI,RYR,VPOWR,YRDES
     3     ,DELN,ETADES,FGPY
*DELETE       SDFLGP.59,SDFLGP.60
      DIMENSION OP16(18),OP17(8),OP18(8),OP19(7),OP20(8),OP21(8),
     1   DAT2(18),DAT3(6),DAT4(15)
*DELETE       SDFLGP.67,SDFLGP.67
     *(OP19(I),I=1,7)/3HFZM,2HLM,2HMM,2HNM,4HFGPY,4HDELN,6HETADES/,
*DELETE       SDFLGP.73,SDFLGP.73
      DATA DAT1/4HTIME/,(DAT2(I),I=1,18)/2HLM,2HMM,2HNM,5HQI77R,
*DELETE       SDFLGP.75,SDFLGP.75
     *5HXG77F,5HYG77F,5HAX77F,6HXG77F1,4HFGPY,4HDELN,6HETADES/,
*INSERT       SDFLGP.80
      DATA N1/1/,N15/15/,N14/14/,N18/18/
*DELETE       SDFLGP.95,SDFLGP.95
      IF(ISDF.NE.0)WRITE(13)N18,N1,DAT2
*INSERT       SDFLGP.102
     2   ,FGPY,DELN,ETADES
*DELETE       SDFLGP.164,SDFLGP.165
      CALL STFL(2,7,OP19)
      CALL STOVAR(7,FZM,LM,MM,NM,FGPY,DELN,ETADES,DU)
*INSERT         AUTS.74
      COMMON/NWSTER/CFOUR,CONE,CONFRI,CTHREE,CTWO,DELPWR,DTIRE
     1  ,EA,ETAMAX,ETAMIN,ETANOS,ETART1,INDNWS,KANWS,KPNWS,PCTETA
     2  ,PSIDES,RPSI,RYR,VPOWR,YRDES
     3     ,DELN,ETADES,FGPY
*INSERT         AUTS.81
```

```
      DATA ILIM,INOSE/0,0/
*INSERT        AUTS.102
      IF(DELTA1 .EQ. 0.0) GO TO 8
      IF(INDNWS.EQ.0)GO TO 8
      IF(KANWS.EQ.1)GO TO 7
C        CURRENTLY, NOSEWHEEL STEERING IS FOR DISTANCE ERROR
C                   NOT ANGLE ERROR
      IF(KPNWS.EQ.1)GO TO 9
      ETADES=ETANOS
      GO TO 8
   7  ERROR=(PSIPD-PSIDES)+RPSI*PSIPD1*RADDEG
      GO TO 3
   9  ERROR=(YR-YRDES)+RYR*YRD1
    3 IF(ILIM .EQ. 1 .AND. ETADES .EQ. 0.0) INOSE=0
      IF(INOSE .GT. 0 .AND. ETADES .EQ. 0.0) GO TO 8
      IF(ABS(ERROR) .GT. EA) GO TO 30
      GO TO 32
   30 IF(ERROR .GT. 0.0) GO TO 37
      ETADES=ETADES+ETART1*DELTS
      INOSE=1
      GO TO 31
   37 ETADES=ETADES-ETART1*DELTS
      INOSE=2
   31 IF(ETADES .GT. ETAMAX) ETADES=ETAMAX
      IF(ETADES .LT. ETAMIN) ETADES=ETAMIN
      GO TO 8
   32 IF(INOSE .EQ. 2) GO TO 38
      GO TO 39
   38 ETADES=ETADES+ETART1*DELTS
      ILIM=1
      GO TO 99
   39 ETADES=ETADES-ETART1*DELTS
      ILIM=1
   99 IF(ABS(ETADES) .LT. 1.0) ETADES = 0.0
   8  CONTINUE
*IDENT    C11778
*DELETE         TOLA.4,C82477.4
*DELETE         TOLA.9,C82477.7
*DELETE         TOLA.15,TOLA.21
*INSERT          EXE.32
     *,DM64(30),GAMA,DM65(15),PCTETA,DM66(13),AH(5),DM67(35)
*DELETE         EXE.54,EXE.54
  34  DO 35 II=1,4029
*INSERT          EXE.68
      DO 30 I=1,5
   30 AH(I)=1.0
*INSERT          EXE.79
      PCTETA=1.0
      GAMA=1.0
```

```
*INSERT        INUPD.4
     1,DM1(100)
*INSERT        UPDAT.3
     1,DM2(100)
*INSERT        MIMIN.6
     *,DM8(100)
*INSERT        LGDET.8
     C,DM9(100)
*DELETE        LGDET.33,LGDET.33
   49 FORMAT(58X,4H-ES(,I1,19H) EXCEEDED IN LGDET/
*INSERT        LGDET.34
       Y(J)=-.5*ES(I)
       Y(JJ)=-1.0E-10
       P(JJ)=-1.0E-10
*DELETE        LGDET.47,LGDET.50
   51 IF(P(JJ).LT.0.)GO TO 30
       IF(Y(JJ).LT.0.)Y(JJ)=0.
       GO TO 55
   30 P(J)=0.
       P(JJ)=0.
*INSERT       STGTSI.7
     *,DM8(100)
*INSERT       STGTST.6
     4,DM8(100)
*INSERT          DEF.5
     *,DM5(100)
*INSERT        LINES.4
     C,D4(100)
*INSERT        ERROR.4
     C,DM3(100)
*INSERT        EXERR.3
     C,DM3(100)
*INSERT         ATMS.10
     4,DM8(100)
*INSERT        TFFS1.16
     *,DM16(100)
*INSERT        TFFS8.16
     *,DM16(100)
*DELETE        VPCS1.23,VPCS1.23
     9XCGBF      ,XCGRF      ,DM19( 930),DM20(2068),DM21(100)
*INSERT        SACS1.63
     *,DDM78(100)
*INSERT         OPT1.72
     *,DM158(100)
*INSERT       LGEAR1.30
     *,DDM21(60),AH(5),PH(5),DDM22(30)
*INSERT       LGEAR1.203
       IF(P2(I).GT.0.)GO TO 1000
       PH(I)=(P(I)*AH(I)-FC2(I))/AH(I)
```

```
      IF(PH(I).LE.-1600.)GO TO 1003
*DELETE      LGEAR1.204,LGEAR1.204
 1000 SF(I)=-P(I)*(A(I)-A2(I))-P2(I)*A2(I)+FC2(I)-S2D1(1,I)*
*INSERT      LGEAR1.205
      GO TO 1002
 1003 SF(I)=-P(I)*(A(I)-AH(I))+1600.*AH(I)-FF(I)*TMP(2)
*DELETE      LGEAR1.206,LGEAR1.206
 1002 IF(SD1(1,I).EQ.0.0.AND.FT(I).LE.ABS(SF(I)))SF(I)=-FT(I)
*DELETE      LGEAR1.222,LGEAR1.222
   49 FORMAT(58X,4H-ES(,I1,20H) EXCEEDED IN LGEAR1/
*INSERT      LGEAR1.223
      S(1,I)=-0.5*ES(I)
      SD1(1,I)=-1.0E-10
      SD2(1,I)=-1.0E-10
*INSERT      LGEAR1.229
      S(1,I)=0.5*ES(I)
*DELETE      LGEAR1.234,LGEAR1.235
   51 IF(SD2(1,I).LT.0.)GO TO 30
      IF(SD1(1,I).LT.0.)SD1(1,I)=0.
      GO TO 55
   30 SD2(1,I)=0.
      SD1(1,I)=0.
*INSERT      LGEAR1.240
C     RE-CHECK SHOCK STRUT FORCE FOR RE-CONDITIONED FULLY EXTENDED STATE
      IF(SD1(1,I).EQ.0.0.AND.FT(I).LE.ABS(SF(I)))SF(I)=-FT(I)
*INSERT      LGEA3C.30
     *,DDM30(20),SLEN1(5),SLEN2(5),GAMA,CFOUR,CONE,CONFRI,CTHREE,CTWO,
     *DELPWR,DTIRE,DDM31(5),INDNWS,DDM32(2),PCTETA,DDM33(3),VPOWR,
     *DDM34,FGPY,DELN,ETADES,DDM35(45)
*DELETE      LGEA3C.32,LGEA3C.33
*DELETE      C82477.14,C82477.17
*INSERT      LGEA3C.59
      EQUIVALENCE (DM18(31),PA77P)
*DELETE      LGEA3C.105,LGEA3C.106
      P(I) = (PZERO(I)+PA77P)*(VZERO(I)/(VZERO(I)+A2(I)*S2(1,I)-
     *S(1,I)*A(I)))**GAMA-PA77P
*DELETE      LGEA3C.110,LGEA3C.110
      P2(I) = (P20(I)+PA77P)*(V20(I)/TMP(1))**GAMA-PA77P
*DELETE      C82477.22,C82477.23
  200 IF(ABS((OMET(1,I)*TMP(1))*COS((PSIPD+ETADES)*DEGRAD)*RI(2,2,I))
     1  .GE. (RG11*RDXG(I)+RG13*RDZG(I)))GO TO 201
      VTX(I)=VTX(I)-TMP(2)+(OMET(1,I)*TMP(1))*COS((PSIPD+ETADES)*
     1  DEGRAD)*RI(2,2,I)
      VTY(I)=RDYG(I)-(OMET(1,I)*TMP(1))*SIN((PSIPD+ETADES)*DEGRAD)*
     1 RI(2,1,I)
*INSERT      C82477.24
      GO TO 203
  201 VTX(I)=1.E-10
      OMET(1,I)=-(RG11*RDXG(I)+RG13*RDZG(I))/(RZERO(I)-DELTA(I))
```

```
      VTY(I)=RDYG(I)-(OMET(1,I)*TMP(1))*SIN((PSIPD+ETADES)*DEGRAD)*
     1  RI(2,1,I)
      VTZ(I)=RG31*RDXG(I)+RG33*RDZG(I)
*DELETE       LGEA3C.145,LGEA3C.145
  203 TMP(1)=RG11*RDXG(I)+RDZG(I)*RG13
*DELETE       C82477.25,C82477.27
*DELETE       C82477.29,C82477.29
      ETAVE=ATAN2(RDYG(I),TMP(1))
*INSERT       C82477.30
      IF(DELTA1 .EQ. 0.0) FGPY = 0.0
      IF(DELTA1 .EQ. 0.0)GO TO 41
*INSERT        FLEX1.17
     *,GDAMP(20),DDM25(80)
*DELETE        FLEX1.18,FLEX1.18
*INSERT       SDFLGP.31
     C,DM18(52),FGPY,DELN,ETADES,DM19(45)
*DELETE       C82477.53,C82477.56
*DELETE        STORE.5,STORE.5
      COMMON/DIRCOM/DATA(4029)
*DELETE       DSERCH.3,DSERCH.3
      COMMON/FIXDIR/ANAME(1000),LOC(1000),NCOUNT
*DELETE       DIRODA.3,DIRODA.3
      COMMON/FIXDIR/NAME(1000),LOC(1000),NCOUNT
*DELETE       DIR1DA.3,DIR1DA.3
      COMMON/FIXDIR/NAME(1000),LOC(1000),NCOUNT
*DELETE       DIR2DA.3,DIR2DA.4
      COMMON/FIXDIR/NAME(1000),LOC(1000),NCOUNT
      DATA NCOUNT/929/
*DELETE       DIR3DA.3,DIR3DA.4
      COMMON/FIXDIR/NAME(1000),LOC(1000),NCOUNT
      DATA (NAME(K3),K3=876,929)/
*DELETE       DIR3DA.7,DIR3DA.8
     *   6HPF     ,6HGQ     ,6HGQD1   ,6HGQD2   ,6HIFLX   ,6HGDAMP ,6HSLEN1 ,
     *   6HSLEN2 ,6HGAMA   ,6HCFOUR ,6HCONE   ,6HCONFRI,6HCTHREE,6HCTWO  ,
     *   6HDELPWR,6HDTIRE ,6HEA     ,6HETAMAX,6HETAMIN,6HETANOS,6HETART1,
     *   6HINDNWS,6HKANWS ,6HKPNWS ,6HPCTETA,6HPSIDES,6HRPSI   ,6HRYR    ,
     *   6HVPOWR ,6HYRDES ,6HFGPY  ,6HDELN   ,6HETADES,6HILIM  ,6HINOSE ,
     *   6HISTER ,6HIRUDD ,6HINWRUD,6HAH     ,6HPH     /
      DATA (LOC(K4),K4=876,929)/
*DELETE       DIR3DA.11,DIR3DA.11
     *   3760     ,3880     ,3900     ,3920     ,3940     ,3960     ,3980     ,
     *   3985     ,3990     ,3991     ,3992     ,3993     ,3994     ,3995     ,
     *   3996     ,3997     ,3998     ,3999     ,4000     ,4001     ,4002     ,
     *   4003     ,4004     ,4005     ,4006     ,4007     ,4008     ,4009     ,
     *   4010     ,4011     ,4012     ,4013     ,4014     ,4015     ,4016     ,
     *   4017     ,4018     ,4019     ,4020     ,4025     /
*INSERT         AUTS.43
     *,DDM41(38),EA,ETAMAX,ETAMIN,ETANOS,ETART1,INDNWS,KANWS,KPNWS,
     *DDM42,PSIDES,RPSI,RYR,DDM43,YRDES,DDM44(2),ETADES,ILIM,INOSE,
```

```
     *ISTER,IRUDD,INWRUD,DDM45(40)
*DELETE       C82477.67,C82477.70
*DELETE       C82477.71,C82477.71
*DELETE       C82477.72,C82477.72
      IF(DELTA1.EQ.0.0)GO TO 130
*INSERT       C82477.77
  130 IF(ISTER.EQ.1)GO TO 29
      IF(INWRUD.EQ.1)GO TO 82
      GO TO 8
   82 IF(DELRDE.GT.0.0)GO TO 86
      ETADES = DELRDE*(ETAMAX/DELRL)
      GO TO 8
   86 ETADES = DELRDE*(ETAMIN/DELRU)
*DELETE       C82477.78,C82477.78
*INSERT       C82477.79
   29 IF(TR.LE.TST)GO TO 35
      ETADES=ETANDS+ETART1*(TR-TST)
      GO TO 31
   35 ETADES=ETANDS
      GO TO 31
*DELETE       C82477.93,C82477.95
      GO TO 31
*INSERT       C82477.103
   31 IF(ETADES .GT. ETAMAX) ETADES=ETAMAX
      IF(ETADES .LT. ETAMIN) ETADES=ETAMIN
*DELETE         AUTS.447,AUTS.448
   34 IF(IAP.GT.2)GO TO 500
      DELRN=DELRNN
*DELETE         AUTS.454,AUTS.454
  500 IF(IRUDD.EQ.1)GO TO 5
      GO TO 150
    5 IF(INWRUD.EQ.1)GO TO 6
      IF(TR.LE.TST)GO TO 27
      DELRDE=DELRDI+DELRRD*(TR-TST)
      GO TO 101
   27 DELRDI=DELRD
      GO TO 101
  150 DELRN=DELRNN
      PSIE=PSIPD
*INSERT         AUTS.457
      GO TO 101
    6 IF(TR.LE.TST)GO TO 110
      DELRDE=DELRDI+DELRRD*(TR-TST)
      GO TO 101
  110 IF(ABS(DELRDE).GT.0.0)GO TO 120
      DELRDI=DELRD
      GO TO 101
  120 DELRDI=DELRDE
*INSERT         AUTS.460
```

```
  106 CONTINUE
*DELETE         AUTS.582,AUTS.583
       IF(IRUDD.EQ.1)GO TO 49
       IF(DELRDE.LT.DELRD)DELRD1=-DELRRD
   49 DELPD1=DELA
*DELETE         AUTS.607,AUTS.607
   61 IF(IAP.EQ.4)GO TO 62
       IF(ABS(DELRD1*DELTS).GE.ABS(DELRDE-DELRD))GO TO 62
*INSERT        FLARE1.38
      *,DM21(100)
*INSERT        AUTPR1.35
      *,DM41(100)
*INSERT        THAUTS.32
      *,DDM9(100)
*INSERT         ENGFL.29
      *,DM8(100)
*INSERT         CENGL.9
      *,DM5(100)
*IDENT    SDDMODS
*DELETE        C11778.2,C11778.2
   34 DO 35 II=1,4036
*DELETE        C11778.32,C11778.32
      *,DDM21(43),INDNWS,DDM22(10),ETADES,DDM23(5),AH(5),PH(5),DDM24(30)
*INSERT        LGEAR1.61
       EQUIVALENCE (INDSTE(73),PSIPD)
       DATA RADDEG,DEGRAD/57.2957795,.01745329/
*INSERT        LGEAR1.242
       IF(INDNWS.EQ.1.AND.I.EQ.1)GO TO 301
*INSERT        LGEAR1.245
 301   MA(I)=-FTRY(I)*TMP(1)*SIN((PSIPD+ETADES)*DEGRAD)+
     1   FTRX(I)*TMP(1)*COS((PSIPD+ETADES)*DEGRAD)
       GO TO 201
*DELETE        C11778.54,C11778.54
      *DDM34,FGPY,DELN,ETADES,DDM35(15),CONFRM,DELPPM,VPOWRM,CONEM,CTWOM,
      *CTHREM,CFOURM,DDM36(23)
*INSERT        LGEA3C.142
       TMPETA = ETADES
*DELETE        C82477.21,C82477.21
       TMPETA = 0.0
*DELETE        C11778.59,C11778.59
 200   IF(ABS(OMET(1,I)*TMP(1)*COS((PSIPD+TMPETA)*DEGRAD))
*DELETE        C11778.61,C11778.64
       VTX(I)=VTX(I)-TMP(2)+OMET(1,I)*TMP(1)*COS((PSIPD+TMPETA)*DEGRAD)
       VTY(I) = RDYG(I)+OMET(1,I)*TMP(1)*SIN((PSIPD+TMPETA)*DEGRAD)
*DELETE        C11778.67,C11778.69
       OMET(1,I)=-((RG11*RDXG(I)+RG13*RDZG(I))/(COS((PSIPD+TMPETA)*
     1    DEGRAD)))/(RZERO(I)-DELTA(I))
       VTY(I) = RDYG(I)+OMET(1,I)*TMP(1)*SIN((PSIPD+TMPETA)*DEGRAD)
*DELETE        C11778.74,C11778.74
```

```
      IF(DELTA1 .EQ. 0.0)GO TO 48
*DELETE      C82477.32,C82477.32
      SIDEMU = CONFRI*(ABS(DELN))**DELPWR*VAXLE(I)**VPOWR
*DELETE      LGEA3C.168,LGEA3C.168
      GO TO 48
*DELETE      C82477.52,C82477.52
      IF(I .GT. 1)GO TO 48
      FGPY=0.0
*DELETE      LGEA3C.171,LGEA3C.171
      GO TO 48
   41 ETAVEM=ATAN2(RDYG(I),TMP(1))
      DELNM=PSIPD-ETAVEM*RADDEG
      IF(DELTA(I).EQ.0.0)FGPYM=0.0
      IF(DELTA(I).EQ.0.0)GO TO 48
      IF(DELNM.EQ.0.0.OR.XG77F1.EQ.0.0)GO TO 1100
      SIDMUM =CONFRM*(ABS(DELNM))**DELPRM*VAXLE(I)**VPOWRM
      GO TO 1800
 1100 SIDMUM =1.0E-6
 1800 IF((DELTA(I)/2.*RZERO(I)).GT.0.0875)GO TO 1300
      CRNPRM = CONEM*DELTA(I)-CTWOM*DELTA(I)**2.
      GO TO 1700
 1300 CRNPRM = CTHREM-CFOURM*DELTA(I)
 1700 YAWPMM = ABS(CRNPRM*DELNM/(SIDMUM*FTRZ(I)))
      IF(YAWPMM .LE.1.5)GO TO 1500
      FGPYM = SIDMUM*FTRZ(I)*COS(DELNM*DEGRAD)
      GO TO 1600
 1500 FGPYM = ((YAWPMM-(4.0/27.0)*YAWPMM**3.0)*SIDMUM*FTRZ(I))
     1    *COS(DELNM*DEGRAD)
 1600 IF(DELNM.GT.0.0)FGPYM=ABS(FGPYM)
      IF(DELNM.LT.0.0)FGPYM=-ABS(FGPYM)
      DFTRXM=FGPYM*SIN(ETAVEM)
      DFTRYM=FGPYM*COS(ETAVEM)
      FTRX(I)=FTRX(I)+DFTRXM
      FTRY(I)=FTRY(I)+DFTRYM
   48 CONTINUE
*DELETE      C11778.77,C11778.77
      COMMON/DIRCOM/DATA(4036)
*DELETE      C11778.82,C11778.82
      DATA NCOUNT/936/
*DELETE      C11778.84,C11778.84
      DATA(NAME(K3),K3=876,936)/
*DELETE      C11778.90,C11778.90
     *   6HISTER ,6HIRUDD ,6HINWRUD,6HAH    ,6HPH    ,6HCONFRM,6HDELPRM,
     *   6HVPOWRM,6HCONEM ,6HCTWOM ,6HCTHREM,6HCFOURM/
*DELETE      C11778.91,C11778.91
      DATA(LOC(K4),K4=876,936)/
*DELETE      C11778.97,C11778.97
     *   4017    ,4018    ,4019    ,4020    ,4025    ,4030    ,4031    ,
     *   4032    ,4033    ,4034    ,4035    ,4036    /
```

```
*IDENT    ACOBLK
*INSERT         TOLA.27
*COMDECK ACOBLK
C**************************** ACTIVE VARIABLES ************************
      COMMON /ACTIVE/ AMUH   ,ACON       ,BCON       ,CCON       ,DCON
     1,APINT(5)   ,AP2TO(5)  ,AREA1(5)   ,AREA2(5)   ,AREA3(5)
     2,AREMO(5)   ,AREO3(5)  ,BLFORT(5)  ,BLMU(5)    ,BUFORT(5)  ,BUMU(5)
     3,BETA       ,CDMOC(5)  ,CDMOE(5)   ,CDSV(5)    ,CFFOR(5)   ,CD3(5)
     4,COEF       ,COEFO(5)  ,COEF3(5)   ,COPA
     5,CSV1(5)    ,CSV3(5)   ,DELT       ,DELTX(5)   ,DELTX1(5)  ,DLTX1D(5)
     6,DF(5)      ,DP1(5)    ,DSV(5)     ,DSTOP      ,DIOTA      ,DMTANH(5)
     7,ENUP(5)    ,EPSILO(5) ,EPSROL(5)  ,EPSSLP     ,ETASV      ,FSTOPK
     8,FFORT(5)   ,FOAHST(5) ,FONHST(5)  ,FORCHT(5)  ,FORSST(5)  ,FSTOP(5)
     9,GAMAH(5)   ,GAMAN     ,GNR        ,HMM(5)     ,ICOSV(5)   ,IDEACT
     *,IFRI(5)    ,IIXSVH(5) ,IIXSVL(5)  ,IOPCO(5)   ,IPASS(5)   ,ISET(5)
     1,IXSVH(5)   ,IXSVL(5)  ,IFSTOP(5)  ,ISTROK(5)  ,KAPT(5)
     2,NAC(5)     ,NITER     ,OMRUN      ,PATM
     3,PERCNT(5)  ,PGAHAC(5) ,PGALAC(5)  ,PGA1I(5)   ,PGA1T(5)
     4,PGA2I(5)   ,PGA2T(5)  ,PGA3I(5)   ,PGA3T(5)
     5,PR(5)      ,PS(5)     ,P1(5)      ,QC(5)      ,QD(5)      ,QPUMPS(5)
     6,QSV(5)     ,QSVCU(5)  ,QSVN(5)    ,QSV1(5)    ,QSV3(5)    ,QS1(5)
     7,QS3(5)     ,QTOLER    ,RCLSV(5)   ,REDSLP(5)  ,RHOH       ,SBFOT
     8,SIPA       ,SA(5)     ,RESA(5)    ,TAUF       ,TC1        ,TC2
      COMMON /ACTIVE/ TC3     ,TC4        ,VCUM(5)    ,VELDEC     ,VOL1I(5)-
     1,VOL1T(5)   ,VOL2I(5)  ,VOL2T(5)   ,VOL3I(5)   ,VOL3T(5)   ,WC
     2,WC1        ,WFORT(5)  ,WLFOR(5)   ,WLFORR     ,XBIAS(5)   ,XVALVE(5)
     3,XKA(5)     ,XKF(5)    ,XKSV(5)    ,XLPSV1(5)  ,XLPSV3(5)  ,XSTHR
     4,XMA(5)     ,XMA1(5)   ,XMA2(5)    ,XMA3(5)    ,XMA4(5)    ,XMA5(5)
     5,XMA6(5)    ,XMA7(5)   ,XMA8(5)    ,XMA9(5)    ,XMA10(5)   ,XMA11(5)
     6,XMU        ,XSCOM(5)  ,XSTOT(5)   ,XSV(5)     ,XSVDOT(5)  ,XSVDD(5)
     7,XSVDDD(5)  ,XSVDMN(5) ,XSVDMX(5)  ,XSVMAX(5)  ,XSVMIN(5)  ,XDDMAX(5)
     8,XDDMIN(5)  ,VOLACI(5) ,VOLAHT(5)  ,VOLANI(5)  ,VOLANT(5)
     9,WSV        ,WSV1      ,WSV3       ,ZETAC1     ,ZETAC2     ,ZSSC
C**********************************************************************
*IDENT    CSCMODS
*INSERT          EXE.42
*CALL ACOBLK
*INSERT          EXE.253
      IF (IABS(INDLG).NE.3) GO TO 594
      CALL ALGEAR1
      GO TO 596
  594 CONTINUE
*INSERT          EXE.254
  596 CONTINUE
*INSERT        INUPD.5
      NNUM = NUM+N
      WRITE(6,600) NNUM
  600 FORMAT(5X,32HNUMBER OF INTEGRATED VARIABLES =,I3)
*DELETE        LGDET.4,LGDET.4
```

```
      COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
*DELETE       LGDET.39,LGDET.39
   53 FORMAT(58X,4H ES(,I1,20H) EXCEEDED IN LGDET /
*DELETE       LINES.6,LINES.6
       IF(LONG.LE.41)RETURN
*INSERT       OPT1.75
*CALL ACOBLK
*INSERT       OPT1.77
      EQUIVALENCE (DM155(133), INDLG)
*INSERT       OPT1.354
      IF (IABS(INDLG).NE.3) GO TO 2070
      CALL ALGEAR1
      GO TO 2075
 2070 CONTINUE
*INSERT       OPT1.355
 2075 CONTINUE
*DELETE       LGEAR1.34,LGEAR1.34
      COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
*INSERT       LGEAR1.121
      IF (IABS(INDLG) .EQ. 3) NDEQ=10*NSTRUT
*INSERT       LGEAR1.122
      IF (IABS(INDLG).EQ.3) CALL ACTINIT
*DELETE       LGEAR1.228,LGEAR1.228
   53 FORMAT(58X,4H ES(,I1,20H) EXCEEDED IN LGEAR1/
*DELETE       LGEA3C.35,LGEA3C.35
      COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
*INSERT       LGEA3C.53
      EQUIVALENCE (DM14(155),INDLG)
*INSERT       LGEA3C.104
      IF (IABS(INDLG).EQ.3) GO TO 31
*INSERT       LGEA3C.196
      IF (IABS(INDLG).EQ.3) GO TO 46
*DELETE       SDFLGP.36,SDFLGP.36
      COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
*INSERT       SDFLGP.47
*CALL ACOBLK
*INSERT       C82477.58
      DIMENSION ACOVAR1(8), ACOVAR2(8), ACOVAR3(8), ACOVAR4(8),
     * ACOVAR5(8), ACOVAR6(8), ACOVAR7(8), ACOVAR8(8),ACOVAR9(8)
*INSERT       SDFLGP.78
      DATA(ACOVAR1(I),I=1,8 ) /5HVOL1T,5HVOL2T,5HVOL3T,5HPGA1T,5HPGA2T,
     * 5HPGA3T,5HGAMAH,5HCOEFO/
      DATA(ACOVAR2(I),I=1,8 ) /2HQO,4HVCUM,6HBLFORT,6HBUFORT,5HFFORT,
     * 6HFORCHT,6HFOAHST,6HFONHST/
      DATA(ACOVAR3(I), I=1,8) /5HCFFOR,5HFSTOP,6HFORSST,6HXVALVE,3HXSV,
     * 5HDELTX,6HDELTX1,6HDLTX1D/
      DATA(ACOVAR4(I), I=1,8) /2HDF,3HDP1,2HPS,2HP1,3HQSV,4HQSVN,4HQSV1,
     * 4HQSV3/
      DATA(ACOVAR5(I),I=1,8)/6HEPSILO,3HHMM,6HDMTANH,5HXSTOT,5HICOSV,
```

```
     * 4HENUP,5HWLFOR,5HIOPCO/
      DATA(ACOVAR6(I), I=1,8) /6HIIXSVH,6HIIXSVL,5HIXSVH,5HIXSVL,5HIPASS
     *,3HNAC,4HRESA,2HSA/
      DATA(ACOVAR7(I), I=1,8) /6HVELDEC,6HWLFORR,6HIDEACT,4HZSSC,4HCOPA,
     * 4HSIPA,5HDSTOP,6HFSTOPK/
      DATA(ACOVAR8(I), I=1,8) /5HWFORT,6HVOLAHT,6HVOLANT,5HQSVCU,3HXMA,
     * 4HXMA5,4HXMA8,5HXMA11/
      DATA(ACOVAR9(I), I=1,8) /6HXSVDDD,5HXSVDD,6HXSVDOT,2HPR,4HIFRI,
     * 6HIFSTOP,6HISTROK,4HISET/
*INSERT      C82477.66
      IF(IABS(INDLG).NE.3) GO TO 50
      CALL STFL(2,8,ACOVAR1)
      DO 33 I=1,NSTRUT
      CALL STOVAR(8,VOL1T(I),VOL2T(I),VOL3T(I),PGA1T(I),PGA2T(I),
     * PGA3T(I),GAMAH(I),COEFO(I))
   33 CONTINUE
      CALL STFL(2,8,ACOVAR2)
      DO 34 I=1,NSTRUT
      CALL STOVAR(8,QQ(I),VCUM(I),BLFORT(I),BUFORT(I),FFORT(I),
     * FORCHT(I),FOAHST(I),FONHST(I))
   34 CONTINUE
      CALL STFL(2,8,ACOVAR3)
      DO 35 I=1,NSTRUT
      CALL STOVAR(8,CFFOR(I),FSTOP(I),FORSST(I),XVALVE(I),XSV(I),
     * DELTX(I),DELTX1(I),DLTX1D(I))
   35 CONTINUE
      CALL STFL(2,8,ACOVAR4)
      DO 36 I=1,NSTRUT
      CALL STOVAR(8,DF(I),DP1(I),PS(I),P1(I),QSV(I),QSVN(I),QSV1(I),
     *QSV3(I))
   36 CONTINUE
      CALL STFL(2,8,ACOVAR8)
      DO 37 I=1,NSTRUT
      CALL STOVAR(8,WFORT(I),VOLAHT(I),VOLANT(I),QSVCU(I),XMA(I),
     * XMA5(I),XMA8(I),XMA11(I))
   37 CONTINUE
      CALL STFL(2,8,ACOVAR9)
      DO 38 I=1,NSTRUT
      CALL STOVAR(8,XSVDDD(I),XSVDD(I),XSVDOT(I),PR(I),FLOAT(IFRI(I)),
     * FLOAT(IFSTOP(I)),FLOAT(ISTROK(I)),FLOAT(ISET(I)))
   38 CONTINUE
      CALL STFL(2,8,ACOVAR5)
      DO 39 I=1,NSTRUT
      CALL STOVAR(8,EPSILO(I),HMM(I),DMTANH(I),XSTOT(I),FLOAT(ICOSV(I)),
     * ENUP(I),WLFOR(I),FLOAT(IOPCO(I)))
   39 CONTINUE
      CALL STFL(2,8,ACOVAR6)
      DO 40 I=1,NSTRUT
      CALL STOVAR(8,FLOAT(IIXSVH(I)),FLOAT(IIXSVL(I)),FLOAT(IXSVH(I)),
```

```
     * FLOAT(IXSVL(I)),FLOAT(IPASS(I)),
     * FLOAT(NAC(I)),RESA(I),SA(I))
  40 CONTINUE
     CALL STFL(2,8,ACOVAR7)
     CALL STOVAR(8,VELDEC,WLFORR,FLOAT(IDEACT),ZSSC,COPA,SIPA,DSTOP,
     * FSTOPK)
  50 CONTINUE
*INSERT      SDFLGP.183
      IF(IABS(INDLG) .NE. 3) GO TO 6
      I1 = I+3*NSTRUT
      I2 = I+4*NSTRUT
      I3 = I+5*NSTRUT
      CALL UPDAT(1,LA(I1),PGA1T(I),DU,DU,DU,DU)
      CALL UPDAT(1,LA(I2),VCUM(I),DU,DU,DU,DU)
      CALL UPDAT(1,LA(I3),OSVCU(I),DU,DU,DU,DU)
      I1 = 3*I+6*NSTRUT-2
      I2 = I+9*NSTRUT
      CALL UPDAT(3,LA(I1),XSVDD(I),XSVDOT(I),XSV(I),DU,DU)
      CALL UPDAT(1,LA(I2),DELTX1(I),DU,DU,DU,DU)
*INSERT        READ.35
      SLTSYM = 0.0
      IBC = 0
      INXQ= 0
      JBC = 0
*INSERT        AUTS.87
      DELPI = 0.0
      TR = 0.0
*IDENT    ACTINIT
*INSERT      LGEA3C.227
*DECK ACTINIT
      SUBROUTINE ACTINIT
C
C******************** FATOLA VARIABLES ********************************
      COMMON/DIPCOM/DM1(115),ALPHD,DM1A(20),AMASS,DM2(147),DCL1,DCM1,
     CDCN1,DCL2,DCM2,DCN2,DCL3,DCM3,DCN3,DM3(99),FXB7P,
     CDUM4(3),FYB7P(4),FZB7P,DM5(17),GXB7F,DM6(8),GZB7F,
     CDM7(218),INDSTE(48),PHIPD,INDSTE1(23),PSIPD,INDSTE2(156),THTPD,
     CINDSTE3(5),TIME,DM8(287),PI77R(2),PI77R1(2),DM9(4),
     CQI77R(2),QI77R1(2),DM10(4),RI77R(2),RI77R1(2),DM11(48),
     CXG77F(2),XG77F1(12),YG77F(2),YG77F1(12),
     CZG77F(2),ZG77F1(2),DUM13(52),
     CNSTRUT,MASS(5),RX(5),RY(5),RZ(5),THETAD(5),ERDFG,RGR,
     CNTIRES(5),RZERO(5),W(5),DELTAM(5),MOMENT(5),
     CRF(5),VZ  ,IFD,PZERO(5),VZERO(5),A(5),P20(5),V20(5),
     CA2(5),IL,S2T(5),ES2(5),C2L(5),MASS2(5),MUS(5),
     CCC(5),CE(5),C2C(5),C2E(5),NVGPT,NPP,MB(5),RLT,NDELTA,
     CES(5),SB(5),SD21(2),SD22(2),SD23(2),SD24(2),SD25(2)
      COMMON/DIRCOM/
     CSD11(2),SD12(2),SD13(2),SD14(2),SD15(2),
```

```
     CS1(2),SS2(2),S3(2),S4(2),S5(2),
     CS2D21(2),S2D22(2),S2D23(2),S2D24(2),S2D25(2),
     CS2D11(2),S2D12(2),S2D13(2),S2D14(2),S2D15(2),
     CS21(2),S22(2),S23(2),S24(2),S25(2),
     COMTD11(2),OMTD12(2),OMTD13(2),OMTD14(2),OMTD15(2),
     COMT1(2),OMT2(2),OMT3(2),OMT4(2),OMT5(2),
     CAI(5),BI(5),DELTA1,DELTA2,DELTA3,DELTA4,DELTA5,
     CDDELT1,DDELT2,DDELT3,DDELT4,DDELT5,ISTAGE,
     CPRTMIN,IPLT,ISDF,ISTPL1,ISTPL2,ISTPL3,ISTPL4,ISTPL5,
     CDM14(22),IB(5),DM15(127),INDLG,DM16(107),CASK(44),INDFLX,
     *NMODE,DM18(40),SXMOD(100),SYMOD(100),SZMOD(100),DM19(1686),
     *GQD2(20),DDM20(20)
     *,DDM21(20),SLEN1(5),SLEN2(5),DUM15(13),INDNWS,DDM22(10),ETADES,
     *DDM23(5),AH(5),PH(5),DDM24(30)
      COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
C
*CALL ACOBLK
      DIMENSION CD3(5), SD1(2,5)
      EQUIVALENCE(SD11(1),SD1(1,1)), (DM5(16),GREFF)
C
C         DATA STATEMENTS TO SET VALUES OF INPUT VARIABLES
C
      DATA AMUH/0.00018/, BETA/14400000./
      DATA BLMU,BUMU/10*0.15/
      DATA AREA1,AREA2,AREA3/5*0.21139,5*0.32167,5*0.00698/
      DATA AREMO,AREO3,APINT/5*0.0066,5*0.19635,5*0.0/
      DATA CDMOC,CDMOE,CD3/5*0.9,5*0.14197,5*0.6/, CFFOR/5*50.0/
      DATA CDSV,DELT,DIOTA/5*0.62,0.0001,0.0/
      DATA EPSILO,EPSSLP,EPSROL,ETASV/5*200.0,0.0,5*2000.0,0.436/
      DATA GAMAN,GAMAH,GNR/1.06,5*52.36,0.0/
      DATA ICOSV,IFSTOP/10*0/
      DATA IDEACT,II,KAPT,OMRUN,PERCNT/7*0,-0.07575,5*1.66115/
      DATA PATM,PGAHAC,PGALAC/2116.8,5*432000.0,5*0.0/
      DATA PGA1I,PGA2I,PGA3I/15*40320.0/
      DATA TAUF,TC1,TC2,TC3,TC4/0.1,0.281,0.141,0.001,0.0001/
      DATA VOL1I,VOL2I,VOL3I/5*0.36379,5*0.47164,5*0.0/
      DATA RHOH,WLFOR,WLFORR/1.626,5*500.0,0.0/
      DATA DSV,RCLSV,WSV1,WSV3/5*1.1875,5*0.0000475,2*3.480/
      DATA WC,WC1,WSV,XKSV,XSTHR/1263.0,251.3,655.5,5*0.00250,0.020833/
      DATA XKA,XKF,XLPSV1,XLPSV3/5*0.04,5*60.0,5*0.0,5*0.0/
      DATA XBIAS,XSCOM,XSVDMN,XSVDMX/5*-0.00014,5*1.206,5*-30.1,5*30.1/
      DATA XDDMAX,XDDMIN/5*1.0E20,5*-1.0E20/
      DATA XSVMAX,XSVMIN,ZETAC1,ZETAC2/5*0.100,5*-0.100,5.1,0.1/
      DATA VOLACI,VOLANI,QPUMPS/5*1.33333,5*0.26667,5*0.02005/
C*****************************************************************
      IF (TIME .GT. DELT) RETURN
C
      ACON=1./(WSV*WSV*WC)
      BCON=1./(WSV*WSV)+2*ETASV/(WSV*WC)
```

```
      CCON=2.*ETASV/WSV+1./WC
      DCON=1.
      XMU=AMUH/.0000209
      QTOLER=.0001
      SBFOT=0.0
      VELDEC=0.
      OMRUN=OMRUN*0.01745329
      II = 0
C
      DO 100 I=1,NSTRUT
      REDSLP(I) = 100000.0
      ISTROK(I)=0
      NAC(I)=0
      VOLANT(I) = VOLANI(I)
      VOL1T(I)=VOL1I(I)
      VOL2T(I)=VOL2I(I)
      VOL3T(I)=VOL3I(I)
      QSVCU(I)=0.0
      DF(I)=0.
      DELTX(I)=0.
      DELTX1(I)=DELTX(I)*XKF(I)
      DLTX1D(I)=0.
      XMA(I)=(DF(I)+DELTX1(I))*XKA(I)
      XMA5(I)=XMA(I)
      XMA8(I)=XMA5(I)
      XMA11(I)=XMA8(I)
      XMA1(I)=0.
      XMA2(I)=0.
      XMA3(I)=0.
      XMA4(I)=0.
      XMA6(I)=0.
      XMA7(I)=0.
      XMA9(I)=0.
      XMA10(I)=0.
      XSVDD(I)=0.
      XSVDDD(I)=0.
      XSV(I)=XKSV(I)*XMA11(I)+XBIAS(I)
      XSVDOT(I)=0.0
      DP1(I)=0.
      COEF3(I)=CD3(I)*SQRT(2.*GREFF/GAMAH(I))
C
C      NOTE: SUBROUTINE 'PHLOZ2' COMPUTES INITIAL PRESSURES AND FLOWS IN
C      UNITS OF INCHES.
      COEF=CDSV(I)*SQRT(2.*GREFF/GAMAH(I))*144.
      CSV1(I)=COEF*WSV1
      CSV3(I)=COEF*WSV3
      PS(I)=PGAHAC(I)/144.
      PR(I)=PGALAC(I)/144.
      QC(I)=0.
```

```
      CALL PHLOZ2(PS(I),PR(I),XSV(I),QC(I),XLPSV1(I),XLPSV3(I),RCLSV,DSV
     & ,CSV1(I),CSV3(I),XMU,QTOLER,NITER,P1(I),QS1(I),QS3(I))
      PGA1I(I)=P1(I)*144.
      PGA2I(I)=PGA1I(I)
      PGA3I(I)=PGA2I(I)
      QSV1(I)=QS1(I)/1728.
      QSV3(I)=QS3(I)/1728.
C
      PGA1T(I)=PGA1I(I)
      PGA2T(I)=PGA2I(I)
      PGA3T(I)=PGA3I(I)
      QD(I)=0.
      QSV(I)=QSV1(I)-QSV3(I)
      VCUM(I)=0.
C
      AP2TO(I)=PGA2I(I)+PATM
      FFORT(I) = 0.0
      FORSST(I)=0.0
      WFORT(I)=0.0
      FONHST(I)=0.0
      FORCHT(I)=0.0
      XVALVE(I)=XSV(I)
  100 CONTINUE
C
      CALL SETUP
      RETURN
      END
*IDENT    ALGEAR
*INSERT     ACTINIT.145
*DECK ALGEAR
      SUBROUTINE ALGEAR1
C
********************** FATOLA VARIABLES ********************************
      COMMON/DIRCOM/DM1(115),ALPHD,DM1A(20),AMASS,DM2(147),DCL1,DCM1,
     CDCN1,DCL2,DCM2,DCN2,DCL3,DCM3,DCN3,DM3(99),FXB7P,
     CDUM4(3),FYB7P(4),FZB7P,DM5(17),GXB7F,DM6(8),GZB7F,
     CDM7(218),INDSTE(48),PHIPD,INDSTE1(23),PSIPD,INDSTE2(156),THTPD,
     CINDSTE3(5),TIME,DM8(287),PI77R(2),PI77R1(2),DM9(4),
     CQI77R(2),QI77R1(2),DM10(4),RI77R(2),RI77R1(2),DM11(48),
     CXG77F(2),XG77F1(12),YG77F(2),YG77F1(12),
     CZG77F(2),ZG77F1(2),DUM13(52),
     CNSTRUT,MASS(5),RX(5),RY(5),RZ(5),THETAD(5),ERDEG,RGR,
     CNTIRES(5),RZERO(5),W(5),DELTAM(5),MOMENT(5),
     CRF(5),VZ   ,IFD,PZERO(5),VZERO(5),A(5),P20(5),V20(5),
     CA2(5),IL,S2T(5),ES2(5),C2L(5),MASS2(5),MUS(5),
     CCC(5),CE(5),C2C(5),C2E(5),NVGPT,NPP,MB(5),RLT,NDELTA,
     CES(5),SB(5),SD21(2),SD22(2),SD23(2),SD24(2),SD25(2)
      COMMON/DIRCOM/
     CSD11(2),SD12(2),SD13(2),SD14(2),SD15(2),
```

```
      CS1(2),SS2(2),S3(2),S4(2),S5(2),
      CS2D21(2),S2D22(2),S2D23(2),S2D24(2),S2D25(2),
      CS2D11(2),S2D12(2),S2D13(2),S2D14(2),S2D15(2),
      CS21(2),S22(2),S23(2),S24(2),S25(2),
      COMTD11(2),OMTD12(2),OMTD13(2),OMTD14(2),OMTD15(2),
      COMT1(2),OMT2(2),OMT3(2),OMT4(2),OMT5(2),
      CAI(5),BI(5),DELTA1,DELTA2,DELTA3,DELTA4,DELTA5,
      CDDELT1,DDELT2,DDELT3,DDELT4,DDELT5,ISTAGE,
      CPRTMIN,IPLT,ISDF,ISTPL1,ISTPL2,ISTPL3,ISTPL4,ISTPL5,
      CDM14(22),IB(5),DM15(127),INDLG,DM16(107),CASK(44),INDFLX,
      *NMODE,DM18(40),SXMOD(100),SYMOD(100),SZMOD(100),DM19(1686),
      *GOD2(20),DDM20(20)
      *,DDM21(20),SLEN1(5),SLEN2(5),DUM15(13),INDNWS,DDM22(10),ETADES,
      *DDM23(5),AH(5),PH(5),DDM24(30)
       REAL MASS,MOMENT,   MASS2,MUS,NTIRES,MB
       DIMENSION DLGAUT(14),DLGDE(47),DLG(7),DLGE(299)
       COMMON/LGAUTS/ARG11,ARG13,ARG31,ARG33,AMA(5),VAXLE(5)
       COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
       COMMON/LG/FXM,FYM,FZM,LM,MM,NM,EPSLO2
       COMMON/LGE/A11(5),A13(5),A31(5),A33(5),RRCGX(5),
      CRL(3,3),RI(3,3,5),RAX(5),RAY(5),RAZ(5),TMP(3),ZZERO(5),
      CXR(5),YR(5),EPSLON(5),PA(5),FDELTA(5),
      CFTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
      CVTX(5),VTY(5),VTZ(5),GZ(5),      VGPT(5),FTRX(5),FTRY(5),
      CDX(5),DY(5),DZ(5),FT(5),FDX(5),FDY(5),FF(5),AA(5),C2(5),
      CSR(5),SF(5),PSKD(5),MUVP(5),MTRX(5),MTRY(5),
      CMTRZ(5),MA(5),RG11,RG13,RG31,RG33,IPRT,
      CMTX,MTY,MTZ,SFTRX,SFTRY,SFTRZ,FTRA,
      CFTRB,FTRC,SMTRX,SMTRY,SMTRZ
       COMMON/FLXOP/GFORC2(100),GFORC3(100),GFORC4(100),BH1(308)
       EQUIVALENCE (DLGAUT(1),ARG11),(DLGDE(1),LA(1)),
      *(DLG(1),FXM),(DLGE(1),A11(1))
       COMMON/TABSRC/DUMM1(103),LOC(7)
       COMMON/HTCOM/HT,HT1,HT2
C
*CALL ACOBLK
C
       REAL MUVP,MTRX,MTRY,MTRZ,MA,
      CMTX,MTY,MTZ,LM,MM,NM
       DIMENSION DELTA(5),DDELTA(5),P(5),
      C       SD2(2,5),SD1(2,5),S(2,5),S2D2(2,5),S2D1(2,5),S2(2,5),
      COMETD1(2,5),OMET(2,5)
       EQUIVALENCE (P(1),PRES(1))
       EQUIVALENCE (DELTA(1),DELTA1),(DDELTA(1),DDELT1)
       EQUIVALENCE
      C(SD21(1),SD2(1,1)),(SD11(1),SD1(1,1)),(S1(1),S(1,1)),
      C(S2D21(1),S2D2(1,1)),(S2D11(1),S2D1(1,1)),(S21(1),S2(1,1)),
      C(OMTD11(1),OMETD1(1,1)),(OMT1(1),OMET(1,1))
C
```

```
C**************ACTIVE CODE*****************************************
      EQUIVALENCE (DM5(16),GREFF)
      DATA IFRI/5*0/,FSTOPK/0.0/,FSTOP/5*0.0/,DSTOP/0.004/
      DATA SA,RESA,HMM,IXSVL,IXSVH,IIXSVL,IIXSVH,IPASS
     1     /15*0.,25*0/
      DATA ENUP/5*0.0/
      DATA ISET,IOPCO,OSVN/10*1,5*0.0/
      Q1(T1,T2)=SIGN(1.,(T1-T2))*SQRT(ABS(T1-T2))
C*****************************************************************
C
      DATA RADDEG,DEGRAD/57.2957795,.01745329/
C
C     MAIN COMPUTATIONAL AREA
C     RL MATRIX ELEMENTS
      RL(1,1)=DCL1*RG11+DCL3*RG13
      RL(1,2)=DCL2
      RL(1,3)=DCL1*RG31+DCL3*RG33
      RL(2,1)=DCM1*RG11+DCM3*RG13
      RL(2,2)=DCM2
      RL(2,3)=DCM1*RG31+DCM3*RG33
      RL(3,1)=DCN1*RG11+DCN3*RG13
      RL(3,2)=DCN2
      RL(3,3)=DCN1*RG31+DCN3*RG33
      CALL LGEA3C
C
C******************* ACTIVE CODE *********************************
C   START ACTIVE GEAR CALCULATIONS
      COPA = COS((THTPD+DIOTA)*DEGRAD)
      SIPA = SIN((THTPD+DIOTA)*DEGRAD)
      IF(TIME .GT. DELT)GO TO 18
  15  WRITE(6,1013)
1013  FORMAT (1H020H ACTIVE CONTROL GEAR)
  18  IF(IDEACT.EQ.1) GO TO 56
  19  IF(IDEACT.EQ.2) GO TO 80
      IF(HMM(I) .EQ. 0.) GO TO 90
      IF(OMRUN .GT. 0.0)GO TO 25
      IF(ZG77F1(1) .GE. VELDEC) GO TO 90
      GO TO 40
  25  IF(ZG77F1(1) .GE. VELDEC+XG77F1(1)*TAN(OMRUN))GO TO 90
  40  WRITE(6,1014)TIME
1014  FORMAT (1H036H REDUCE CONTROL LIMIT FORCE AT TIME=,E16.8)
      IDEACT=1
  56  WLFOR(I)=WLFOR(I)-REDSLP(I)*DELT
      EPSILO(I)=EPSILO(I)+EPSSLP*DELT
      IF(WLFOR(I) .GT. WLFORR) GO TO 90
  60  WRITE(6,1015)TIME
1015  FORMAT (1H027H CONTROL AT WLFORR AT TIME=,E16,8)
      IDEACT=2
  80  WLFOR(I)=WLFORR
```

```
      EPSILO(I)=EPSROL(I)
   90 CONTINUE
C*****************************************************************************
      DO 65 I=1,NSTRUT
      WFORT(I) = (SQRT(FXB7P*FXB7P+FYB7P*FYB7P+FZB7P*FZB7P))/(NSTRUT-1)
     &     +(FORSST(I)*COPA)
C************************** ACTIVE CODE ***********************************
      IF(KAPT(I).NE.0)GO TO 210
      APINT(I)=0.0
  210 CONTINUE
      IF(PGA1T(I) .LE. -1600.0) PGA1T(I) =  -1600.0
      VOL1T(I)=VOL1I(I)-(AREA1(I)-APINT(I))*S(1,I)
      VOL3T(I)=VOL3I(I)+AREA3(I)*S(1,I)
      VOL2T(I)=VOL2I(I)-(AREA2(I)-AREA1(I)+APINT(I))*S(1,I)+(VOL3T(I)
     X -VOL3I(I))-VCUM(I)
      PGA2T(I)=AP2TO(I)*((VOL2I(I)/VOL2T(I))**GAMAN)-PATM
      IF(SD1(1,I) .EQ. 0.0)GO TO 104
      PGA3T(I)=((COEF3(I)*AREO3(I))**2*PGA2T(I)-SD1(1,I)/ABS(SD1(1,I))
     X *(SD1(1,I)*AREA3(I))**2)
     &/((COEF3(I)*AREO3(I))**2)
      GO TO 105
  104 PGA3T(I)=PGA2T(I)
  105 IF(PGA1T(I) .GE. PGA2T(I))GO TO 106
      GO TO 107
  106 GAMAH(I)=RHOH*GREFF*(1.0+(PGA1T(I)*3.04E-08)-
     * (PGA1T(I)**2*2.72E-15))
      GO TO 108
  107 GAMAH(I)=RHOH*GREFF*(1.0+(PGA2T(I)*3.04E-08)-
     * (PGA2T(I)**2*2.72E-15))
  108 IF(PGA1T(I) .GE. PGA2T(I))COEFO(I)=
     *  COMOC(I)*SQRT(ABS(2.*GREFF/GAMAH(I)))
      IF(PGA2T(I) .GT. PGA1T(I))COEFO(I)=
     *  COMOE(I)*SQRT(ABS(2.*GREFF/GAMAH(I)))
      IF(SD1(1,I).LE.0.0) GO TO 109
      QO(I)=COEFO(I)*(AREMO(I)-APINT(I))*Q1(PGA1T(I),PGA2T(I))
  109 IF(PGA2T(I) .LE. -1600.0)PGA2T(I)=-1600.0
      IF(PGA3T(I) .LE. -1600.0)PGA3T(I)=-1600.0
  100 IF(DELTA(I) .LE. 0.0 .AND. TIME .GT. DELT)GO TO 101
      GO TO 110
  101 FFORT(I)=0.0
      GO TO 140
  110  CONTINUE
C     COMPUTE STRUT AXIAL BINDING FRICTION FORCE
      BLFORT(I)=FONHST(I)*((SLEN2(I)-S(1,I))/(SLEN1(I)+S(1,I))+1.0)
      BUFORT(I)=FONHST(I)*(SLEN2(I)-S(1,I))/(SLEN1(I)+S(1,I))
      FFORT(I)=BUMU(I)*ABS(BUFORT(I))+BLMU(I)*ABS(BLFORT(I))
  140 CONTINUE
C     COMPUTE SHOCK STRUT CHARGING FORCE
      IF(S(1,I) .GT. 0.0)GO TO 142
```

```
  141 FORCHT(I)=PGA1T(I)*AREA1(I)+PGA2T(I)*(AREA2(I)-AREA1(I))-PGA3T(I)
     X *AREA3(I)+FFORT(I)+CFFOR(I)
C      COMPUTE NORMAL AND AXIAL HUB TO SHOCK STRUT FORCES AT HUB
  142  FONHST(I)=SQRT(FDX(I)**2+FDY(I)**2)-MASS(I)*GREFF*SIPA+SBFOT
       IF(ABS(FT(I)) .LE. FORCHT(I) .AND. S(1,I) .EQ. 0.0)GO TO 150
       GO TO 801
  150  CONTINUE
       FORSST(I)=FT(I)
       SD1(1,I)=0.0
       IF(I.EQ.1) GO TO 450
       ISTROK(I)=1
C****** 289 CHANGED TO 295 FOR DEBUGGING PURPOSES:
       GO TO 295
C      COMPRESSION VELOCITY OF SHOCK STRUT IS POSITIVE
  801  IF(SD1(1,I) .LE. 0.8 .AND. IFRI(I) .EQ. 0)GO TO 2
       GO TO 3
    2  DMTANH(I)=1.0
       GO TO 284
    3  DMTANH(I)=ABS(TANH(2.0*SD1(1,I)))
       IFRI(I)=1
  284  IF(S(1,I) .LE. DSTOP .AND. SD1(1,I) .LT. 0.0)GO TO 900
       GO TO 902
  900  IF(S(1,I) .LE. 0.0001)GO TO 903
       GO TO 904
  903  SD2(1,I)=0.0
       SD1(1,I)=0.0
       S(1,I)=0.0
       DP1(I)=0.0
       FFORT(I)=0.0
       VCUM(I)=0.0
       QD(I)=0.0
       II=0
C      CALL VIRK4(II,N,NT,CI,SPEC,CIMAX,IERR,VAR,CUVAR,DER,ELE1,ELE2,
C     1           ELT,ERRVAL,DERSUB,CHSUB,ITEXT)
       GO TO 902
  904  CONTINUE
       IF(IFSTOP(I) .NE. 0)GO TO 906
  905  DSTOP=S(1,I)
       FSTOPK=2.0*MASS(I)*SD1(1,I)**2/DSTOP**2
  906  IF(S(1,I) .LE. DSTOP/2.0)GO TO 908
  907  FSTOP(I)=-FSTOPK*(DSTOP-S(1,I))
       GO TO 909
  908  FSTOP(I)=-FSTOPK*S(1,I)
  909  IFSTOP(I)=1
       IFRI(I)=0
       GO TO 901
  902  FSTOP(I)=0.0
  901  IF(ABS(FT(I)) .LE. FORCHT(I) .AND. S(1,I) .EQ. 0.0)GO TO 500
       IF(SD1(1,I) .LT. 0.0)GO TO 470
```

```
      GO TO 471
  470 FFORT(I)=-FFORT(I)
      CFFOR(I)=-CFFOR(I)
  471 FORSST(I)=-((PGA1T(I)-PGA2T(I))*(AREA1(I)-APINT(I))
     & +PGA2T(I)*AREA2(I)
     X -PGA3T(I)*AREA3(I)+(FFORT(I)
     1       +CFFOR(I))*DMTANH(I)+FSTOP(I))
  500 IF(INDFLX.GE. 1)GO TO 295
      IF(I.EQ.1) GO TO 450
      ISTROK(I)=1
C************ BRANCH FOR DEBUGGING PURPOSES:
      GO TO 295
  289 IF(S(1,I) .LE. 0.0)290,295
  290 IF(IOPCO(I) .EQ. 1)GO TO 295
      IF((PGA1I(I)-1000.0).LT.PGA1T(I).AND.
     &  PGA1T(I).LT.(PGA1I(I)+1000.0))299,298
  299 IF(XVALVE(I) .NE. 0.0)GO TO 311
      IF(IPASS(I) .EQ. 1)GO TO 296
      XVALVE(I)=XKSV(I)*XMA11(I)+XBIAS(I)
      IPASS(I)=1
      GO TO 294
  298 IF(ICOSV(I) .EQ. 1)GO TO 291
      IOPCO(I)=0
      IF(XSV(I) .LT. 0.002 .AND. XSV(I) .GT. -0.002)291,295
  291 IF(SD2(1,I) .LE. 0.0 .AND. ICOSV(I) .EQ. 1)GO TO 311
      IF(IOPCO(I) .EQ. 1)GO TO 295
      IF(PGA1T(I) .GT. PGA1I(I))292,293
  292 IF(IXSVL(I) .EQ. 1)GO TO 294
      XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
      IF(XVALVE(I) .LE. -0.1)300,294
  300 XVALVE(I)=-0.1
      IXSVL(I)=1
      GO TO 294
  293 IF(IXSVH(I) .EQ. 1)GO TO 294
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
      IF(XVALVE(I) .GE. 0.1)302,294
  302 XVALVE(I)=0.1
      IXSVH(I)=1
  294 CONTINUE
      II=0
C     CALL VIRK4(II,N,NT,CI,SPEC,CIMAX,IERR,VAR,CUVAR,DER,ELE1,ELE2,
C    1          ELT,ERRVAL,DERSUB,CHSUB,ITEXT)
      DLTX1D(I)=0.0
      ICOSV(I)=1
  296 IF(WFORT(I) .GT. 0.0 .AND. S(1,I) .LE. 0.0)GO TO 410
  311 IF(NAC(I) .EQ. 1)GO TO 307
      IF(IIXSVH(I) .EQ. 1)GO TO 305
      XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
      IF(XVALVE(I) .LE. 0.0)305,400
```

```
 305  XVALVE(I)=0.0
      IIXSVH(I)=1
      GO TO 400
 307  IF(IIXSVL(I) .EQ. 1)GO TO 308
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
      IF(XVALVE(I) .GE. 0.0)308,400
 308  XVALVE(I)=0.0
      IIXSVL(I)=1
 400  II=0
C     CALL VIRK4(II,N,NT,CI,SPEC,CIMAX,IERR,VAR,CUVAR,DER,ELE1,ELE2,
C    1           ELT,ERRVAL,DERSUB,CHSUB,ITEXT)
 410  IF(XVALVE(I) .NE. 0.0)GO TO 295
      ICOSV(I)=0
      DELTX1(I)=DELTX(I)*XKF(I)
      XMA(I)=(DF(I)+DELTX1(I))*XKA(I)
      XMA11(I)=XMA(I)
      XSV(I)=XKSV(I)*XMA11(I)+XBIAS(I)
      CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      IPASS(I)=0
      IXSVL(I)=0
      IXSVH(I)=0
      IIXSVL(I)=0
      IIXSVH(I)=0
      IOPCO(I)=1
      CALL PHLOZ2(PS(I),PR(I),XSV(I),QC(I),XLPSV1(I),XLPSV3(I),RCLSV(I),
     & DSV(I),CSV1(I),CSV3(I),XMU,QTOLER,NITER,P1(I),QS1(I),QS3(I))
C
 295  IF(ISTROK(I) .EQ. 1 .AND. S(1,I) .GT. 0.0)IOPCO(I)=0
C
      ENUP(I)=.5*AMASS*ZG77F1(1)**2
      ENUP(I)=ENUP(I)/(NSTRUT-1)
      IF(HMM(I) .EQ. 1.0)GO TO 130
      SA(I)=0.
      IF(WFORT(I).LT.0.) XSTOT(I)=ENUP(I)/((-WFORT(I))*COPA)
      IF(WFORT(I).GE.0.) XSTOT(I)=1.E20
C     ZSSC IS A PERCENTAGE OF SB(I) FOR ACTIVATING CONTROL-CDMOC(I) IS US
      ZSSC=0.6*CDMOC(I)*SB(I)
      IF(XSTOT(I) .LE. (ZSSC-S(1,I)) .OR. RESA(I) .EQ. 1.0)SA(I)=1.0
      RESA(I)=SA(I)
      IF(SA(I).EQ.0. .OR. HMM(I).EQ.1.) GO TO 130
      WLFOR(I)=-WFORT(I)
      VELDEC=((WLFOR(I)+WLFORR)/2.*(WLFOR(I)-WLFORR))/(AMASS*REDSLP(I))
      WRITE(6,121)TIME,WLFOR(I),VELDEC
 121  FORMAT(50H ACTIVE CONTROL INITIATED...TIME, WLFOR, VELDEC = ,
     1       3E13.5)
      HMM(I)=1.
 130  IF(S(1,I) .GT. 0.0)ISET(I)=0
      IF(HMM(I).EQ.0.) GO TO 451
      IF(-WFORT(I).GT.(WLFOR(I)+EPSILO(I))) DF(I)=(WLFOR(I)+
```

```
     &  EPSILO(I))-(-WFORT(I))
      IF(-WFORT(I).LT.(WLFOR(I)-EPSILO(I))) DF(I)=(WLFOR(I)-
     &  EPSILO(I))-(-WFORT(I))
      IF(-WFORT(I).LE.(WLFOR(I)+EPSILO(I)).AND.
     & -WFORT(I).GE.(WLFOR(I)-EPSILO(I)))
     : 457,456
  457 IF(S(1,I) .LE. 0.0)GO TO 456
  453 IF(WFORT(I) .GT. 0.0 .AND. QSVCU(I) .LT. 0.0)454,455
  454 DF(I)=WLFOR(I)-(-WFORT(I))
      GO TO 456
  455 DF(I)=0.0
  456 DELTX(I)=S(1,I)-XSCOM(I)
      IF(S(1,I) .LE. 0.0 .AND. ISET(I) .EQ. 0)GO TO 451
      GO TO 452
  451 DF(I)=0.
      DELTX(I)=0.
  452 XMA(I)=(DF(I)+DELTX1(I))*XKA(I)
      IF(GNR.EQ.1. .AND. XMA(I).GT.0.) XMA(I)=
     :  XMA(I)*SQRT((PGA1T(I)-PGALAC(I))
     X /(PGAHAC(I)-PGA1T(I)))
C
C       NOTE: SUBROUTINE 'FLOZE2' COMPUTES THE FLOWS FROM THE PRESSURES
C       IN UNITS OF INCHES.
      P1(I)=PGA1T(I)/144.
C     COMPUTATION OF HIGH PRESSURE ACCUMULATOR NITROGEN VOLUME
C     AND ACCUMULATOR PPESSURE
      VOLANT(I)=VOLANT(I)+QSVN(I)*DELT-QPUMPS(I)*DELT
      PS(I)=(((PGAHAC(I)+PATM)*(VOLANI(I)/VOLANT(I))**GAMAN)-PATM)/144.0
      IF(PS(I) .GE. 3000.0)464,465
  464 PS(I)=3000.0
      VOLANT(I)=VOLANI(I)
  465 VOLAHT(I)=VOLACI(I)-VOLANT(I)
      IF(VOLAHT(I) .LE. 0.0)466,467
  466 WRITE(6,1050)TIME
 1050 FORMAT(1HO//45H ACCUMULATOR OIL VOLUME INSUFFICENT AT TIME=,E16.8
     1      //)
      CALL LGEAR6
      STOP 500
  467 CONTINUE
      CALL FLOZE2(PS(I),PR(I),P1(I),XLPSV1(I),XLPSV3(I),RCLSV(I),DSV(I),
     &  XSV(I),QS1(I),QS3(I),CSV1(I),CSV3(I),XMU)
      QSV1(I)=QS1(I)/1728.
      QSV3(I)=QS3(I)/1728.
C
  450 CONTINUE
      QSV1(1) = 0.0
      QSV3(1) = 0.0
      QSV(I)=QSV1(I)-QSV3(I)
      IF(QSV(I) .LT. 0.0)NAC(I)=1
```

```
      IF(QSV(I) .GT. 0.0)NAC(I)=2
      IF(NAC(I) .EQ. 2)461,462
  461 QSVN(I)=QSV(I)
      GO TO 463
  462 QSVN(I)=0.0
  463 IF(SD1(1,I).LT. 0.0 .AND.PGA1T(I).LE. -1600.0)PGA1T(I)=-1600.0
  274 CONTINUE
      I1 = I+3*NSTRUT
      I2 = I+4*NSTRUT
      I3 = I+5*NSTRUT
      CALL INTEG(LA(I1),DP1(I))
      CALL INTEG(LA(I2),QD(I))
      CALL INTEG(LA(I3),QSV(I))
C*********************************************************************
      IF(SD1(1,I).EQ.0.0.AND.FT(I).LE.ABS(FORSST(I)))FORSST(I)=-FT(I)
      AA(I) = (FT(I)+FORSST(I))/MASS(I)
      SD2(1,I) = SR(I)+AA(I)-GZ(I)
      HT1=HT
      IF(SD1(1,I))76,77,78
   76 TTIME=S(1,I)/ABS(SD1(1,I))
   79 IF(TTIME.GE.HT)GO TO 77
      HT1=TTIME
      GO TO 77
   78 TTIME=(SB(I)-S(1,I))/SD1(1,I)
      GO TO 79
   77 CONTINUE
      IF(S(1,I).GT.(-ES(I)))GO TO 50
      WRITE(6,49)I,I,S(1,I)
   49 FORMAT(58X,4H-ES(,I1,20H) EXCEEDED IN ALGEAR/
     C58X,2HS(,I1,4H) = E15.7)
      S(1,I)=-0.5*ES(I)
      SD1(1,I)=-1.0E-10
      SD2(1,I)=-1.0E-10
   50 IF(S(1,I).LE.ES(I))GO TO 51
      IF(S(1,I).LE.(SB(I)-ES(I)))GO TO 55
      IF(S(1,I).LE.(SB(I)+ES(I)))GO TO 52
      WRITE(6,53)I,I,S(1,I)
   53 FORMAT(58X,4H ES(,I1,20H) EXCEEDED IN ALGEAR/
     C58X,2HS(,I1,4H) = E15.7)
      S(1,I)=0.5*ES(I)
   52 IF(SD1(1,I).GT.0.)SD1(1,I)=0.
      IF(SD2(1,I).LT.0.)GO TO 55
      SD2(1,I)=0.
      GO TO 55
   51 IF(SD2(1,I).LT.0.)GO TO 30
      IF(SD1(1,I).LT.0.)SD1(1,I)=0.
      GO TO 55
   30 SD2(1,I)=0.
      SD1(1,I)=0.
```

```
 55   CONTINUE
      I2=2*I+NSTRUT-1
      I1=I2+1
      CALL INTEG(LA(I2),SD2(1,I))
      CALL INTEG(LA(I1),SD1(1,I))
C     RE-CHECK SHOCK STRUT FORCE FOR RE-CONDITIONED FULLY EXTENDED STATE
      IF(SD1(1,I).EQ.0.0.AND.FT(I).LE.ABS(FORSST(I)))FORSST(I)=-FT(I)
      TMP(1)=RZERO(I)-DELTA(I)
      IF(CASK(I).GT.1.E-10)GO TO 200
      IF(INDNWS.EQ.1.AND.I.EQ.1)GO TO 301
      MA(I)=-FTRY(I)*TMP(1)*RI(2,1,I)+FTRX(I)*TMP(1)
     C*RI(2,2,I)
      GO TO 201
 301  MA(I)=-FTRY(I)*TMP(1)*SIN((PSIPD+ETADES)*DEGRAD)+
     1  FTRX(I)*TMP(1)*COS((PSIPD+ETADES)*DEGRAD)
      GO TO 201
 200  MA(I)=TMP(1)*SQRT(FTRY(I)*FTRY(I)+FTRX(I)*FTRX(I))
      MA(I)=SIGN(MA(I),-VAXLE(I)-OMET(1,I)*TMP(1))
 201  AMA(I)=MA(I)
      IF(IB(I).NE.(-1))GO TO 48
      OMETD1(1,I)=0.
      OMET(1,I)=0.
      GO TO 21
 48   TMP(1)=0.
      IF(OMET(1,I).NE.0.)TMP(1)=OMET(1,I)/ABS(OMET(1,I))
      OMETD1(1,I)=(MA(I)-MB(I)*TMP(1))/(NTIRES(I)*MOMENT(I))
 21   CALL INTEG(LA(I),OMETD1(1,I))
C*****************************************************************
  65  CONTINUE
C     CALCULATION OF FTRA, FTRB, AND FTRC
      SFTRX=0.
      SFTRY=0.
      SFTRZ=0.
      DO 70 I=1,NSTRUT
      SFTRX=SFTRX+FTRX(I)
      SFTRY=SFTRY+FTRY(I)
 70   SFTRZ=SFTRZ+FTRZ(I)
      FTRA=RL(1,1)*SFTRX+RL(1,2)*SFTRY+RL(1,3)*SFTRZ
      FTRB=RL(2,1)*SFTRX+RL(2,2)*SFTRY+RL(2,3)*SFTRZ
      FTRC=RL(3,1)*SFTRX+RL(3,2)*SFTRY+RL(3,3)*SFTRZ
C     CALCULATION OF MTX, MTY, AND MTZ
      SMTRX=0.
      SMTRY=0.
      SMTRZ=0.
      DO 75 I=1,NSTRUT
      SMTRX=SMTRX+MTRX(I)
      SMTRY=SMTRY+MTRY(I)
 75   SMTRZ=SMTRZ+MTRZ(I)
      MTX=RL(1,1)*SMTRX+RL(1,2)*SMTRY+RL(1,3)*SMTRZ
```

```
      MTY=RL(2,1)*SMTRX+RL(2,2)*SMTRY+RL(2,3)*SMTRZ
      MTZ=RL(3,1)*SMTRX+RL(3,2)*SMTRY+RL(3,3)*SMTRZ
C     CALCULATION OF FXM, FYM, FZM, LM, MM, AND NM
      FYM=0.
      FZM=0.
      FXM=0.
      LM=0.
      MM=0.
      NM=0.
      BFX=0.
      BFY=0.
      BFZ=0.
      BLM=0.
      BMM=0.
      BNM=0.
      DO 82 I=1,NSTRUT
      DFXM=0.
      DFYM=0.
      DFZM=0.
      DLM=0.
      DMM=0.
      DNM=0.
      DO 14 IL4=1,NMODE
      NBB=(IL4-1)*NSTRUT+I
      DFXM=DFXM-MASS(I)*SXMOD(NBB)*GOD2(IL4)
      DFYM=DFYM-MASS(I)*SYMOD(NBB)*GOD2(IL4)
      DFZM=DFZM-MASS(I)*SZMOD(NBB)*GOD2(IL4)
      NBH=(I-1)*NMODE+IL4
      DLM=DLM-MASS(I)*GFORC2(NBH)*GOD2(IL4)
      DMM=DMM-MASS(I)*GFORC3(NBH)*GOD2(IL4)
   14 DNM=DNM-MASS(I)*GFORC4(NBH)*GOD2(IL4)
      BFX=BFX+DFXM
      BFY=BFY+DFYM
      BFZ=BFZ+DFZM
      BLM=BLM+DLM
      BMM=BMM+DMM
      BNM=BNM+DNM
      TMP(1)=MASS(I)*SD2(1,I)
      FXM=FXM+TMP(1)*A31(I)
      FYM=FYM
      FZM=FZM+TMP(1)*A33(I)
      LM=LM+TMP(1)*A11(I)*RY(I)
      MM=MM-TMP(1)*RRCGX(I)
   82 NM=NM+TMP(1)*A13(I)*RY(I)
      FXM=FXM+BFX+FTRA
      FYM=FYM+BFY+FTRB
      FZM=FZM+BFZ+FTRC
      LM=LM+BLM+MTX
      MM=MM+BMM+MTY
```

```
      NM=NM+BNM+MTZ
C
      ENTRY SETUP
C
      DO 28 I=1,NSTRUT
      IF(SD1(1,I) .LE. 0.0) GO TO 20
      DP1(I)=(-QO(I)+QSV1(I)-QSV3(I)+(AREA1(I)-APINT(I))*SD1(1,I))
     $   *BETA/VOL1T(I)
      IF(PGA1T(I) .LE. -1600.0) 10,20
   10 PGA1T(I) = -1600.0
      IF (DP1(I) .LT. 0.0) DP1(I)=0.0
   20 CONTINUE
C     II=0
C     CALL VIRK4(II,N,NT,CI,SPEC,CIMAX,IERR,VAR,CUVAR,DER,ELE1,ELE2,
C    1           ELT,ERRVAL,DERSUB,CHSUB,ITEXT)
      IF(ICOSV(I) .EQ. 1)GO TO 26
      XMA5(I)=XMA(I)+(2.*ZETAC2*WC1)*XMA1(I)+(WC1**2)*XMA2(I)
     &        -(2.*ZETAC1*WC1)*XMA3(I)-(WC1**2)*XMA4(I)
      XMA8(I)=(TC1/TC2)*XMA5(I)+(1./TC2)*XMA6(I)-(1./TC2)*XMA7(I)
      XMA11(I)=(TC3/TC4)*XMA8(I)+(1./TC4)*XMA9(I)-(1./TC4)*XMA10(I)
C     IF(S(1,I).LE.XSTHR) XMA11(I)=0.
      XSVDDD(I)=(XKSV(I)*XMA11(I)-BCON*XSVDD(I)-CCON*XSVDOT(I)-
     $   DCON*(XSV(I)-XBIAS(I)))/ACON
      DLTX1D(I)=(XKF(I)*DELTX(I)-DELTX1(I))/TAUF
   26 CONTINUE
      CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      CALL LIMITS(XSVDOT(I),XSVDD(I),XSVDMX(I),XSVDMN(I))
      CALL LIMITS(XSVDD(I),XSVDDD(I),XDDMAX(I),XDDMIN(I))
      I1 = 3*I+6*NSTRUT-2
      I2 = I1+1
      I3 = I1+2
      CALL INTEG(LA(I1),XSVDDD(I))
      CALL INTEG(LA(I2),XSVDD(I))
      CALL INTEG(LA(I3),XSVDOT(I))
      I1 = I+9*NSTRUT
      CALL INTEG(LA(I1),DLTX1D(I))
   28 CONTINUE
      RETURN
      END
*IDENT    PHLOZ2
*INSERT       PHLOZ2.1
*DECK PHLOZ2
      SUBROUTINE PHLOZ2(PS,PR,X,QC,LAP1,LAP3,RCL,D,COEF1,COEF3,MU,QTOLER
     &,NITER,P1,Q1,Q3)
C     'PHLOZ2'.....................................R. D. EDSON
C
C     THIS SUBROUTINE CALCULATES THE STEADY-STATE CHAMBER PRESSURE (P1)
C     AND FLOW RATES (Q1 & Q3) FOR A TWO-WAY NONSYMMETRICAL SPOOL VALVE
C     WITH RECTANGULAR WINDOW SLOTS, GIVEN THE STROKE (X) AND THE LOAD
```

```
C     FLOW (QC).  THE PARAMETERS REQUIRED IN THE 'CALL' STATEMENT ARE
C     THE SAME AS DESCRIBED IN SUBROUTINE 'FLOZE2', WITH THE FOLLOWING
C     ADDITIONAL PARAMETERS:
C        QC      = THE FLOW RATE TO THE LOAD
C        QTOLER  = THE TOLERANCE ALLOWED IN CALCULATING FLOW RATES, FOR
C                  DETERMINING WHETHER OR NOT THE SOLUTION HAS CONVERGED
C                  (.0001 IS TYPICAL)
C        NITER   = THE NUMBER OF ITERATIONS REQUIRED TO CONVERGE TO A
C                  SOLUTION (INTEGER)
C
C
      IMPLICIT REAL(L,M)
      PFN(X1,X2,Y1,Y2,Y3,Y4)=X1+(X2-X1)*(Y2-Y1)/(-Y1+Y2-Y3+Y4)
      NITER=0.
      FLAG=-1.
      P1FLAG=-1.
C
      P1A=PR
      P1B=PS
      CALL FLOZE2(PS,PR,P1A,LAP1,LAP3,RCL,D,X,Q1A,Q3A,COEF1,COEF3,MU)
      CALL FLOZE2(PS,PR,P1B,LAP1,LAP3,RCL,D,X,Q1B,Q3B,COEF1,COEF3,MU)
      IF(Q1A.EQ.0. .AND. Q3B.EQ.0.) GO TO 51
      Q3A=Q3A+QC
      Q3B=Q3B+QC
      GO TO 50
   51 P1=(PS+PR)/2.
      Q1=0.
      Q3=0.
      GO TO 400
C
   50 P1=PFN(P1A,P1B,Q3A,Q1A,Q1B,Q3B)
      CALL FLOZE2(PS,PR,P1,LAP1,LAP3,RCL,D,X,Q1,Q3,COEF1,COEF3,MU)
      Q3=Q3+QC
      IF(FLAG.LT.0.) GO TO 55
      IF(P1.EQ.P1I) GO TO 100
   55 P1I=P1
      IF(Q1.EQ.0. .AND. Q3.EQ.0.) GO TO 100
      IF(ABS(Q1) .GE. ABS(Q3)) QDEN=Q1
      IF(ABS(Q3) .GT. ABS(Q1)) QDEN=Q3
      IF(ABS((Q1-Q3)/QDEN) .LT. QTOLER) GO TO 100
      IF(Q1.LT.Q3) GO TO 90
      P1A=P1
      Q1A=Q1
      Q3A=Q3
      GO TO 150
   90 P1B=P1
      Q1B=Q1
      Q3B=Q3
      GO TO 150
```

```
  100  P1FLAG=1.
  150  FLAG=1.
       NITER=NITER+1
       IF(P1FLAG.GT.0.) GO TO 300
       GO TO 50
  300  CONTINUE
       Q3=Q3-QC
  400  RETURN
       END
*IDENT    FLOZE2
*INSERT      PHLOZ2.66
*DECK FLOZE2
       SUBROUTINE FLOZE2(PS,PR,P1,LAP1,LAP3,RCL,D,X,Q1,Q3,COEF1,COEF3,MU)
C      'FLOZE2'.....................................R. D. EDSON
C
C      THIS SUBROUTINE CALCULATES THE STEADY-STATE FLOW RATES (Q1 AND
C      Q3) FOR A TWO-WAY NONSYMMETRICAL SPOOL VALVE WITH RECTANGULAR
C      WINDOW SLOTS, GIVEN THE LOAD CHAMBER PRESSURE (P1) AND STROKE
C      (X).  THE PARAMETERS REQUIRED IN THE 'CALL' STATEMENT ARE AS
C      FOLLOWS:
C         X      = VALVE STROKE
C         P1     = PRESSURE IN CHAMBER 1 (TO LOAD)
C         Q1     = FLOW RATE FROM SUPPLY LINE TO CHAMBER 1
C         Q3     = FLOW RATE FROM CHAMBER 1 TO RETURN LINE
C         PS     = SUPPLY PRESSURE
C         PR     = RETURN PRESSURE
C         LAP1   = OVERLAPPED OR UNDERLAPPED LENGTH BETWEEN THE SPOOL
C                  AND SLEEVE AT NULL, FOR FLOW Q1.  A POSITIVE NUMBER
C                  IS USED FOR OVERLAP, A NEGATIVE NUMBER FOR UNDERLAP.
C         LAP3   = OVERLAPPED OR UNDERLAPPED LENGTH BETWEEN THE SPOOL
C                  AND SLEEVE AT NULL, FOR FLOW Q3.  A POSITIVE NUMBER
C                  IS USED FOR OVERLAP, A NEGATIVE NUMBER FOR UNDERLAP.
C         RCL    = RADIAL CLEARANCE BETWEEN THE SPOOL AND SLEEVE
C         D      = DIAMETER OF SPOOL
C         COEF1  = FLOW COEFFICIENT OF ORIFICE 1 (SUPPLY TO CHAMBER 1)
C                = CD*W1*SQRT(2.*GC/RHO)
C         COEF3  = FLOW COEFFICIENT OF ORIFICE 3 (CHAMBER 1 TO RETURN)
C                = CD*W3*SQRT(2.*GC/RHO)
C                  WHERE W1 = TOTAL WINDOW WIDTH OF ORIFICE 1
C                        W3 = TOTAL WINDOW WIDTH OF ORIFICE 3
C                        CD = DISCHARGE COEFFICIENT
C                        GC = GRAVITATIONAL ACCELERATION CONSTANT
C                       RHO = DENSITY OF HYDRAULIC FLUID
C         MU     = VISCOSITY OF HYDRAULIC FLUID, CENTIPOISE
C
C      THE METHOD OF SOLUTION UTILIZES THE TURBULENT ORIFICE EQUATION
C      AND THE EQUATION FOR FULLY-DEVELOPED LAMINAR FLOW THROUGH AN
C      ANNULUS, WITH FULL ECCENTRICITY ASSUMED.  FOR ORIFICE OPENINGS
C      WHERE SOME OVERLAPPED LENGTH EXISTS, THE PROCEDURE IS TO CALC-
```

```
C     ULATE THE FLOW RATE BY BOTH EQUATIONS, AND THEN USE THE ONE
C     THAT GIVES THE SMALLEST ABSOLUTE VALUE AS THE ANSWER.  FOR
C     OPENINGS WHERE NO OVERLAPPED LENGTH EXISTS, ONLY THE TURBULENT
C     ORIFICE EQUATION APPLIES.
C
      IMPLICIT REAL(L,M)
      Q12(T1,T2)=SIGN(1.,(T1-T2))*SQRT(ABS(T1-T2))
      Q34(T3,T4)=4.5E06*(T3-T4)*D*RCL**3/MU
C
C     ************************* CALCULATE  Q1 ***********************
      X2=LAP1-X
      X4=SQRT(X2**2+RCL**2)
      IF(LAP1 .LE. 0.) GO TO 99
C
C     POSITIVE LAPS:
      IF(X .GE. LAP1) GO TO 65
      Q1=RCL*COEF1*Q12(PS,P1)
      Q1L=Q34(PS,P1)/X2
      IF(ABS(Q1L) .LT. ABS(Q1)) Q1=Q1L
      GO TO 20
   65 Q1=X4*COEF1*Q12(PS,P1)
      GO TO 20
C
C     NEGATIVE LAPS:
   99 IF(X .LT. LAP1) GO TO 10
      Q1=X4*COEF1*Q12(PS,P1)
      GO TO 20
   10 Q1=RCL*COEF1*Q12(PS,P1)
      Q1L=Q34(PS,P1)/X2
      IF(ABS(Q1L) .LT. ABS(Q1)) Q1=Q1L
C
C     ************************* CALCULATE Q3 ************************
   20 X1=LAP3+X
      X3=SQRT(X1**2+RCL**2)
      IF(LAP3 .LE. 0.) GO TO 199
C
C     POSITIVE LAPS:
      IF(X .LE. -LAP3) GO TO 165
      Q3=RCL*COEF3*Q12(P1,PR)
      Q3L=Q34(P1,PR)/X1
      IF(ABS(Q3L) .LT. ABS(Q3)) Q3=Q3L
      GO TO 120
  165 Q3=X3*COEF3*Q12(P1,PR)
      GO TO 120
C
C     NEGATIVE LAPS:
  199 IF(X .GT. -LAP3) GO TO 110
      Q3=X3*COEF3*Q12(P1,PR)
      GO TO 120
```

```
 110  Q3=RCL*COEF3*Q12(P1,PR)
      Q3L=Q34(P1,PR)/X1
      IF(ABS(Q3L) .LT. ABS(Q3)) Q3=Q3L
 120  RETURN
      END
*IDENT    LIMITS
*INSERT       FLOZE2.92
*DECK LIMITS
      SUBROUTINE LIMITS(X,XDOT,XMAX,XMIN)
C
C        STATEMENTS FOR THIS SUBROUTINE OBTAINED BY PHONE BY
C        JOHN R. MCGEHEE ON  2/1/77.
C
      IF(X .GE. XMAX) GO TO 10
      IF(X .LE. XMIN) GO TO 20
      GO TO 30
   10 X=XMAX
      IF(XDOT .GT. 0.0) XDOT = 0.0
      GO TO 30
   20 X=XMIN
      IF(XDOT .LT. 0.0) XDOT = 0.0
   30 RETURN
      END
*IDENT    GMMODS
*DELETE       D81877.1,D81877.1
      DIMENSION TITLE(20),BUF(550),NDIL(28),TBUF(550)
*DELETE       PLTDAT.6,PLTDAT.6
     1 BMM(2,550),CMMDS(6),DEPVAR(5),LINE(7),NDVA(5)
*IDENT    JMMODS
*DELETE      ACTINIT.40,ACTINIT.65
      DATA ICOSV,IFSTOP,II/11*0/
*INSERT       SDFLGP.80
      DATA N19/19/
*DELETE       SDFLGP.96,SDFLGP.96
      IF(ISUM1.NE.0) WRITE(13) N19,ISUM1,DAT3,OP17,ACOVAR8(1),
     1 ACOVAR8(4),ACOVAR4(5),ACOVAR1(4),ACOVAR1(5)
*DELETE       SDFLGP.105,SDFLGP.105
     1 OMET(1,1),WFORT(1),QSVCU(1),QSV(1),PGA1T(1),PGA2T(1)
*DELETE       SDFLGP.108,SDFLGP.108
     1 OMET(1,2),WFORT(2),QSVCU(2),QSV(2),PGA1T(2),PGA2T(2)
*DELETE       SDFLGP.111,SDFLGP.111
     1 OMET(1,3),WFORT(3),QSVCU(3),QSV(3),PGA1T(3),PGA2T(3)
*DELETE       SDFLGP.114,SDFLGP.114
     1 OMET(1,4),WFORT(4),QSVCU(4),QSV(4),PGA1T(4),PGA2T(4)
*DELETE       SDFLGP.117,SDFLGP.117
     1 OMET(1,5),WFORT(5),QSVCU(5),QSV(5),PGA1T(5),PGA2T(5)
*INSERT         READ.14
      DATA ACTIVE/6HACTIVE/
*DELETE         READ.42,READ.42
```

```
   19 IF(SYM.EQ.ACTIVE) GO TO 805
      CALL DIPLAC(RA1,INC,BLANK)
*INSERT         READ.183
  805 CALL ACTIN
      GO TO 100
  810 CALL ACTIN
      GO TO 802
*DELETE         READ.184,READ.184
   26 IF(SYM.EQ.ACTIVE) GO TO 810
      IF(SYM.EQ.STCASE) GO TO 21
*IDENT    DIRACT
*INSERT       DIR3DA.12
*DECK DIRACT
      BLOCK DATA DIRACT
      COMMON/ACTDIR/NAME(71),LOC(71)
      DATA NAME/  6HAMUH  ,  6HAPINT ,  6HAREA1 ,  6HAREA2 ,
     1 6HAREA3 ,  6HAREMO ,  6HARE03 ,  6HBETA  ,  6HBLMU  ,
     2 6HBUMU  ,  6HCDMOC ,  6HCDMOE ,  6HCDSV  ,  6HCD3   ,
     3 6HCFFOR ,  6HDELT  ,  6HDIOTA ,  6HDSV   ,  6HEPSILO,
     4 6HEPSROL,  6HEPSSLP,  6HETASV ,  6HGAMAH ,  6HGAMAN ,
     5 6HGNR   ,  6HIDEACT,  6HKAPT  ,  6HOMRUN ,  6HPATM  ,
     6 6HPGAHAC,  6HPGALAC,  6HPGA1I ,  6HPGA2I ,  6HPGA3I ,
     7 6HPERCNT,  6HQPUMPS,  6HRCLSV ,  6HRHOH  ,  6HTAUF  ,
     8 6HTC1   ,  6HTC2   ,  6HTC3   ,  6HTC4   ,  6HVOLACI,
     9 6HVOLANI,  6HVOL1I ,  6HVOL2I ,  6HVOL3I ,  6HWC    ,
     1 6HWC1   ,  6HWSV   ,  6HWSV1  ,  6HWSV3  ,  6HWLFOR ,
     2 6HWLFORR,  6HXBIAS ,  6HXDDMAX,  6HXDDMIN,  6HXKA   ,
     3 6HXKF   ,  6HXKSV  ,  6HXLPSV1,  6HXLPSV3,  6HXSCOM ,
     4 6HXSTHR ,  6HXSVDMN,  6HXSVDMX,  6HXSVMAX,  6HXSVMIN,
     5 6HZETAC1,  6HZETAC2/
      DATA LOC/   1,   6,  16,  21,  26,  31,  36,  61,  46,
     1       56,  62,  67,  72,  82,  77, 109, 141, 135, 152,
     2      157, 162, 163, 195, 200, 201, 212, 263, 274, 275,
     3      281, 286, 291, 301, 311, 276, 346, 387, 397, 410,
     4      411, 412, 413, 414, 621, 631, 421, 431, 441, 451,
     5      452, 641, 642, 643, 458, 463, 464, 611, 616, 474,
     6      479, 484, 489, 494, 561, 499, 591, 596, 601, 606,
     7      644, 645/
      END
*IDENT    ACTIN
*INSERT         READ.194
*DECK ACTIN
      SUBROUTINE ACTIN
      COMMON/ACTDIR/XNAME(71),LOC(71)
      COMMON/ACTIVE/DATA(646)
      DIMENSION IRA(55),MSG(58),IDATA(646)
      EQUIVALENCE (MSG(1),SYM),(MSG(2),OP)
      EQUIVALENCE (MSG(3),IRA(1)),(MSG(58),INC)
      EQUIVALENCE (DATA(1),IDATA(1))
```

```
      INTEGER COMMA,POINT,E,BLANK
      DATA REMARK,COMMA,BLANK,POINT,E,ENDACT,MINUS,AINT/
     1 3HREM,1R,,1R ,1R.,1RE,6HENDACT,1R-,3HINT/
  100 CONTINUE
    1 FORMAT(A6,1X,A3,1X,55R1,I1)
    2 FORMAT(18X,A6,1X,A3,1X,55R1,I6)
    3 FORMAT(20HOERROR.THE SYMBOL **,A6,
     1 26H** IS NOT IN THE DIRECTORY/1H )
    4 FORMAT(44HOERROR.ILLEGAL CHARACTER IN NUMERIC FIELD **,1R1,2H**/)
      READ(5,1) SYM,OP,IRA,INC
      CALL LINES(1)
      J = 58
      IF(INC.LE.0) J = 57
      WRITE(6,2) (MSG(I),I=1,J)
      IF(SYM.EQ.REMARK) GO TO 100
      IF(SYM.EQ.ENDACT) GO TO 999
      DO 110 I=1,71
      IF(SYM.EQ.XNAME(I)) GO TO 120
  110 CONTINUE
      CALL LINES(3)
      WRITE(6,3) SYM
      GO TO 100
  120 CONTINUE
      INDEX = LOC(I)
      IF(INC.EQ.0) INC = 1
      INDEX = INDEX + INC - 1
      NUMEXP = 0
      NEXP = 0
      IEXP = 0
      NL = 0
      NR = 0
      NUML = 0
      NUMR = 0
      ISIGN = 0
      JSIGN = 0
      LEFT = 1
      DO 210 I=1,56
      IF(I.EQ.56) GO TO 140
      IF(IRA(I).EQ.BLANK) GO TO 210
      IF(IRA(I).EQ.COMMA) GO TO 140
      IF(IRA(I).EQ.POINT) GO TO 170
      IF(IRA(I).EQ.E) GO TO 180
      IF(IRA(I).EQ.MINUS) GO TO 200
      IF(IRA(I).GT.36) GO TO 130
      IF(IRA(I).LT.27) GO TO 130
      NUM = IRA(I) - 27
      IF(IEXP.EQ.1) GO TO 190
      IF(LEFT.GT.0) NUML = 10*NUML + NUM
      IF(LEFT.GT.0) NL = NL + 1
```

```
      IF(LEFT.LT.0) NUMR = 10*NUMR + NUM
      IF(LEFT.LT.0) NR = NR + 1
      GO TO 210
  130 CALL LINES(3)
      WRITE(6,4) IRA(I)
      GO TO 210
  140 CONTINUE
      IF(NL.EQ.0.AND.NR.EQ.0) GO TO 210
      IF(NR.EQ.0) GO TO 160
      X = FLOAT(NUML) + FLOAT(NUMR)/10.**NR
      IF(JSIGN.EQ.1) NUMEXP = -NUMEXP
      IF(IEXP.EQ.1) X = X*(10.)**NUMEXP
      IF(ISIGN.EQ.1) X = -X
      DATA(INDEX) = X
  150 NUML = 0
      NUMR = 0
      NL = 0
      NR = 0
      LEFT = 1
      ISIGN = 0
      JSIGN = 0
      IEXP = 0
      NEXP = 0
      NUMEXP = 0
      INDEX = INDEX + 1
      GO TO 210
  160 CONTINUE
      X = NUML
      IF(JSIGN.EQ.1) NUMEXP = -NUMEXP
      IF(IEXP.EQ.1) X = X*(10.)**NUMEXP
      IF(ISIGN.EQ.1) X = -X
      NUML = X
      IF(OP.EQ.AINT) IDATA(INDEX) = NUML
      IF(OP.NE.AINT) DATA(INDEX) = X
      GO TO 150
  170 CONTINUE
      LEFT = -1
      GO TO 210
  180 CONTINUE
      IEXP = 1
      GO TO 210
  190 CONTINUE
      NUMEXP = 10*NUMEXP + NUM
      NEXP = NEXP + 1
      GO TO 210
  200 CONTINUE
      IF(IEXP.EQ.0) ISIGN = 1
      IF(IEXP.NE.0) JSIGN = 1
  210 CONTINUE
```

```
      GO TO 100
  999 RETURN
      END
*IDENT    CSCMOD2
*DELETE       ACOBLK.11,ACOBLK.11
     8,FFORT(5)  ,FWORK(5)  ,FONHST(5) ,FORCHT(5) ,FORSST(5) ,FSTOP(5)
*DELETE       ACOBLK.16,ACOBLK.16
     3,PERCNT(5) ,PGAHAC(5) ,PGALAC(5) ,PGA1I(5)  ,PGA1T1(5)
*DELETE       ACOBLK.31,ACOBLK.31
     9,WSV       ,WSV1      ,WSV3      ,ZETAC1    ,ZETAC2    ,ZSSC(5)
     *,IMODE(5)  ,CMASNG    ,VMASS(5)
*DELETE          EXE.41,EXE.41
      COMMON /HTCOM/ HT, HT1, HT2, INDINT(5)
*INSERT          EXE.280
      STOP "FLIGHT TIME LIMIT"
*DELETE        MIMIN.8,MIMIN.8
      COMMON /HTCOM/ HT, HT1, HT2, INDINT(5)
*INSERT        MIMIN.10
C
      DO 10 I=1,5
   10 INDINT(I) = 1
*DELETE        LGDET.9,LGDET.9
      COMMON /HTCOM/ HT, HT1, HT2, INDINT(5)
*DELETE       LGEAR1.50,LGEAR1.50
      COMMON /HTCOM/ HT, HT1, HT2, INDINT(5)
*INSERT       LGEAR1.81
      DO 6 I=1,5
         P(I) = 0.0
    6   P2(I) = 0.0
*INSERT      ACTINIT.35
      COMMON /HTCOM/ HT, HT1, HT2, INDINT(5)
*DELETE      ACTINIT.38,ACTINIT.39
*DELETE       JMMODS.1,JMMODS.1
      EQUIVALENCE (DM15(16),GREFF), (DM1(37),AIYYBS)
C
      DATA DSTOP,FSTOPK /0.004,0.0/
      DATA ENUP,FSTOP,HMM,QSVN,RESA,SA /30*0.0/
      DATA ICOSV,IFRI,IFSTOP,IPASS /20*0/
      DATA IOPCO,ISET /10*1/
      DATA IXSVH,IXSVL,IIXSVH,IIXSVL /20*0/
C
*INSERT      ACTINIT.75
      OMRUN=ERDEG*0.01745329
*DELETE      ACTINIT.76,ACTINIT.77
*INSERT      ACTINIT.80
      INDINT(I) = 1
*DELETE      ACTINIT.127,ACTINIT.127
      PGA1T1(I)=PGA1I(I)
*INSERT      ACTINIT.140
```

```
      VMASS(I)=AMASS
*DELETE      ALGEAR.32,ALGEAR.33
     *GQD2(20),DDM20(20),DDM21(20),SLEN1(5),SLEN2(5),
     *GAMA,DUM15(12),INDNWS,DDM22(10),ETADES,
*DELETE      ALGEAR.54,ALGEAR.54
      COMMON /HTCOM/ HT, HT1, HT2, INDINT(5)
*DELETE      ALGEAR.71,ALGEAR.76
      EQUIVALENCE (DM5(16),GREFF), (DM2(27),AXP7F), (DM2(28),AX77F),
     *(DM2(29),AYP7F),(DM2(30),AY77F),(DM2(31),AZP7F),(DM2(33),AZ77F),
     *(DM1(37),AIYYBS)
      DIMENSION INDEACT(5),IPSTOP(5),AIC(5),PGA1T(5)
      DATA AIC,INDEACT,IPSTOP /5*0.0,10*0/
*INSERT      ALGEAR.96
C     CMASNG = AIYYBS/AMASS+AMASS*RX(1)*RX(1)/AIYYBS
      CMASNG = 7.9677
      VMASS(1) = AMASS/CMASNG
      ENCG = 0.5*AMASS*ZG77F1(1)*ZG77F1(1)
      ZDANT = ZG77F1(1)-QI77P*RX(1)
*DELETE      ALGEAR.102,ALGEAR.103
C
   18 DO 90 I=1,NSTRUT
      IF(INDEACT(I).EQ.1) GO TO 56
      IF(INDEACT(I).EQ.2) GO TO 80
*INSERT      ALGEAR.105
      IF(I.EQ.1 .AND. ZDANT.GE.VELDEC) GO TO 90
*INSERT      ALGEAR.108
      IF(I.EQ.1 .AND. ZDANT.GE.VELDEC+XG77F1(1)*TAN(OMRUN)) GO TO 90
*DELETE      ALGEAR.111,ALGEAR.113
      INDEACT(I)=1
   56 IF(INDINT(I).EQ.0) GO TO 58
      WLFOR(I)=WLFOR(I)-REDSLP(I)*HT
      EPSILO(I)=EPSILO(I)+EPSSLP*HT
   58 CONTINUE
      INDINT(I)=0
*DELETE      ALGEAR.117,ALGEAR.117
      INDEACT(I)=2
*DELETE      ALGEAR.122,ALGEAR.125
C*** CALCULATION OF THE WING-GEAR INTERFACE FORCE (WFORT)
      UNSPRNG = 0.0
       DO 66 J=1,NSTRUT
       IF(OMETD1(J).NE. 0.0) UNSPRNG = UNSPRNG+MASS(J)
   66 CONTINUE
       DWFORT =(-SQRT(AXP7F*AXP7F+AYP7F*AYP7F+
     &     AZP7F*AZP7F)+GREFF)*(AMASS-UNSPRNG)
      WFORT(1) = DWFORT+FORSST(1)
      DO 67 I=2,NSTRUT
           WFORT(I) = DWFORT/(NSTRUT-1)
   67 CONTINUE
C
```

```
      DO 65 I=1,NSTRUT
C
C***
*DELETE       ALGEAR.129,ALGEAR.129
      PGA1T(I)=PGA1T1(I)
      IF(PGA1T1(I).LE.-1600.0) PGA1T(I)=-1600.0
*DELETE       ALGEAR.134,ALGEAR.134
      PGA2T(I)=AP2TO(I)*((VOL2I(I)/VOL2T(I))**GAMA)-PATM
*DELETE       ALGEAR.152,ALGEAR.153
      IF(IMODE(I).EQ.0 .AND. DDELTA(I).LE.0.0) 112,113
  112 QO(I) = 0.0
      GO TO 109
C
  113 QO(I)=COEFO(I)*(AREMO(I)-APINT(I))*Q1(PGA1T1(I),PGA2T(I))
      IF(QO(I).LT.0.0 .AND. VCUM(I).LE.0.0) GO TO 102
      AIC(I)=0.0
      GO TO 103
  102 IF(PGA1T1(I).LT.PGA2T(I)) GO TO 103
      GO TO 111
  111 QO(I)=0.0
      VCUM(I)=0.0
      AIC(I)=1.0
  103 IF(QO(I).GT.0.0) AIC(I)=0.0
*DELETE       ALGEAR.177,ALGEAR.177
      IF(IMODE(I).EQ.0) GO TO 297
*DELETE       ALGEAR.179,ALGEAR.180
      GO TO 289
*DELETE       ALGEAR.188,ALGEAR.188
  284 IF(S(1,I).LT.0.0) GO TO 160
      GO TO 161
  160 IPSTOP(I)=1
  161 IF(S(1,I).LE.DSTOP .AND. IPSTOP(I).EQ.1) GO TO 900
*DELETE       ALGEAR.190,ALGEAR.190
  900 IF(S(1,I).LE.0.005) GO TO 903
*DELETE       ALGEAR.195,ALGEAR.199
      IPSTOP(I)=0
*DELETE       ALGEAR.215,ALGEAR.215
  901 IF(PGA1T(I).LE.(PGA1I(I)+500.0) .AND.
     :  PGA1T(I).GT.(PGA1I(I)-500.0)) GO TO 158
      GO TO 159
  158 IF(ABS(FT(I)).LE.FORCHT(I) .AND. S(1,I).EQ.0.0) GO TO 500
  159 IF(S(1,I).GE.0.0) GO TO 470
*DELETE       ALGEAR.225,ALGEAR.225
      IF(IMODE(I).EQ.0) GO TO 297
*DELETE       ALGEAR.227,ALGEAR.228
*INSERT       ALGEAR.293
  297 I2 = 2*I+NSTRUT-1
      I1 = I2+1
      CALL INTEG(LA(I2),SD2(1,I))
```

```
      CALL INTEG(LA(I1),SD1(1,I))
C
      IF(IMODE(I).EQ.0) GO TO 450
*DELETE       ALGEAR.295,ALGEAR.296
      IF(I.NE.1) GO TO 119
      ENUP(1) = 0.5*AIYYBS*QI77R*QI77R+(ENCG/CMASNG)*
     *      (ZG77F1(1)/ABS(ZG77F1(1)))
      GO TO 120
  119 ENUP(I) = ENCG/(NSTRUT-1)
  120 CONTINUE
*DELETE       ALGEAR.300,ALGEAR.300
      IF(WFORT(I).GE.0.0 .OR. DDELTA(I).LT.0.0) XSTOT(I)=1.E20
*DELETE       ALGEAR.302,ALGEAR.303
      ZSSC(I)=FWORK(I)*SB(I)
      IF(XSTOT(I).LE.(ZSSC(I)-S(1,I)) .OR. RESA(I).EQ.1.0) SA(I)=1.0
*DELETE       ALGEAR.307,ALGEAR.307
C
      VELDEC=((WLFOR(I)+WLFORR)/2.*(WLFOR(I)-WLFORR))/
     &    (AMASS*REDSLP(I))
*DELETE       ALGEAR.342,ALGEAR.342
      PS(I)=(((PGAHAC(I)+PATM)*(VOLANI(I)/VOLANT(I))**GAMA)-PATM)/144.0.
*DELETE       ALGEAR.360,ALGEAR.361
*DELETE       ALGEAR.371,ALGEAR.376
      I4 = I+4*NSTRUT
      I5 = I+5*NSTRUT
      CALL INTEG(LA(I4),QD(I))
      CALL INTEG(LA(I5),QSV(I))
      I9 = I+9*NSTRUT
      CALL INTEG(LA(I9),DLTX1D(I))
*DELETE       ALGEAR.414,ALGEAR.417
*DELETE       ALGEAR.516,ALGEAR.516
*DELETE       ALGEAR.519,ALGEAR.521
      IF(IMODE(I).EQ.0 .AND. DDELTA(I).LE.0.0) GO TO 19
      IF(S(1,I).NE.0.0 .OR. AIC(I).NE.1.0) GO TO 20
      PGA1T1(I)=PGA1I(I)
      PGA1T(I) =PGA1I(I)
   19 DP1(I)=0.0
*DELETE       ALGEAR.539,ALGEAR.546
      I3 = I+3*NSTRUT
      CALL INTEG(LA(I3),DP1(I))
      I6 = 3*I+6*NSTRUT-2
      I7 = I6+1
      I8 = I6+2
      CALL INTEG(LA(I6),XSVDDD(I))
      CALL INTEG(LA(I7),XSVDD(I))
      CALL INTEG(LA(I8),XSVDOT(I))
*DELETE        FLEX1.25,FLEX1.25
      COMMON /HTCOM/ HT, HT1, HT2, INDINT(5)
*DELETE     CSCMODS.32,CSCMODS.32
```

```
      DATA(ACOVAR1(I),I=1,8 ) /5HVOL1T,5HVOL2T,5HVOL3T,6HPGA1T1,5HPGA2T,
*DELETE      CSCMODS.35,CSCMODS.35
     * 6HFORCHT,5HVMASS,6HFONHST/
*DELETE       JMMODS.5,JMMODS.5
     1 OMET(1,1),WFORT(1),QSVCU(1),QSV(1),PGA1T1(1),PGA2T(1)
*DELETE       JMMODS.6,JMMODS.6
     1 OMET(1,2),WFORT(2),QSVCU(2),QSV(2),PGA1T1(2),PGA2T(2)
*DELETE       JMMODS.7,JMMODS.7
     1 OMET(1,3),WFORT(3),QSVCU(3),QSV(3),PGA1T1(3),PGA2T(3)
*DELETE       JMMODS.8,JMMODS.8
     1 OMET(1,4),WFORT(4),QSVCU(4),QSV(4),PGA1T1(4),PGA2T(4)
*DELETE       JMMODS.9,JMMODS.9
     1 OMET(1,5),WFORT(5),QSVCU(5),QSV(5),PGA1T1(5),PGA2T(5)
*DELETE      CSCMODS.53,CSCMODS.53
      CALL STOVAR(8,VOL1T(I),VOL2T(I),VOL3T(I),PGA1T1(I),PGA2T(I),
*DELETE      CSCMODS.59,CSCMODS.59
     * FORCHT(I),VMASS(I),FONHST(I))
*DELETE      CSCMODS.100,CSCMODS.100
      CALL UPDAT(1,LA(I1),PGA1T1(I),DU,DU,DU,DU)
*DELETE        ACTIN.4,ACTIN.5
      COMMON /ACTIVE/ DATA(656)
      DIMENSION IRA(55), MSG(58), IDATA(656)
*DELETE         PACK.3,PACK.3
      DIMENSION I1(6)
*DELETE       DIRACT.8,DIRACT.9
     4 6HEPSROL,  6HEPSSLP,  6HETASV ,  6HFWORK ,6HGAMAH ,
     5 6HGNR   ,  6HKAPT  ,  6HOMRUN ,  6HPATM  ,
*DELETE       DIRACT.18,DIRACT.18
     5 6HZETAC1,  6HZETAC2,  6HIMODE /
*DELETE       DIRACT.21,DIRACT.21
     2     157, 162, 163, 170, 195, 201, 263, 274, 275,
*DELETE       DIRACT.26,DIRACT.26
     7     644, 645, 651/
*IDENT    EOR
*IDENT    EORPL
*INSERT       CTENGL.65
*DECK EORPL
*WEOR
*IDENT    CSCMOD3
*DELETE      CSCMOD2.4,CSCMOD2.4
     *,IMODE(5)  ,CMASNG    ,VMASS(5)  ,ZDANT
*INSERT          EXE.42
      COMMON/TABDIR/TABLE(800)
      COMMON/READ1/DUM1(64),JBC,INXO
      COMMON /UPDCAL/ NUM , P( 90),Y( 90)
      COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
      COMMON/STGT/ICOUNT,KCOUNT,LOCAIN(4),LOCADE(4)
      COMMON/TABCOM/LOCS(115),STABLE(115)
      COMMON/CLEAUP/I2,CLEAN,INTEG
```

```
      COMMON/STORA/ARG(48),ALIST(8),GETARG(8),NENT,LENT,K
      COMMON/LGE/A11(5),A13(5),A31(5),A33(5),RRCGX(5),
     CRL(3,3),RI(3,3,5),RAX(5),RAY(5),RAZ(5),TMP(3),ZZERO(5),
     CXR(5),YR(5),EPSLON(5),PA(5),FDELTA(5),
     CFTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
     CVTX(5),VTY(5),VTZ(5),GZ(5),      VGPT(5),FTRX(5),FTRY(5),
     CDX(5),DY(5),DZ(5),FT(5),FDX(5),FDY(5),FF(5),AA(5),C2(5),
     CSR(5),SF(5),PSKD(5),MUVP(5),MTRX(5),MTRY(5),
     CMTRZ(5),MA(5),RG11,RG13,RG31,RG33,IPRT,
     CMTX,MTY,MTZ,SFTRX,SFTRY,SFTRZ,FTRA,
     CFTRB,FTRC,SMTRX,SMTRY,SMTRZ
      COMMON/ACTDIR/NAME(71),LOC(71)
*INSERT      C11778.22
      DATA NCASE /10H          /
*INSERT      SDDMODS.4
      DATA PGA1T1,PGA2T,OSV,OSVCU,WFORT /25*0.0/
*INSERT      LGEA3C.47
*CALL ACOBLK
*DELETE      CSCMOD2.44,CSCMOD2.45
      CMASNG = 1+(AMASS*RX(1)*RX(1))/AIYYBS
*INSERT      CSCMOD2.62
      UNSPR = 0.0
*DELETE      CSCMOD2.65,CSCMOD2.65
      IF(J.EQ.1 .AND. OMETD1(1).NE.0.0) UNSPR = MASS(1)
      IF(J.GT.1 .AND. OMETD1(J).NE.0.0) UNSPRNG = UNSPRNG+MASS(J)
*DELETE      CSCMOD2.69,CSCMOD2.69
      WFORT(1) = SR(1)*(VMASS(1)-UNSPR)
*DELETE      CSCMOD2.125,CSCMOD2.125
     &   (VMASS(1)*REDSLP(I))
*DELETE      CSCMODS.44,CSCMODS.44
      DATA(ACOVAR7(I), I=1,8) /6HVELDEC,6HWLFORR,5HZDANT,4HZSSC,4HCOPA,
*DELETE      SDFLGP.80,SDFLGP.80
*INSERT      C82477.63
      IF(IABS(INDLG).NE.3) 110,115
  110 IF(ISUM1.NE.0) WRITE(13) N14,ISUM1,DAT3,OP17
      GO TO 120
  115 CONTINUE
*INSERT      JMMODS.4
  120 CONTINUE
*INSERT      C82477.64
      IF(IABS(INDLG).NE.3) 210,220
  210 CONTINUE
      IF(ISTPL1.NE.0) WRITE(13) FT(1),SF(1),DELTA(1),P(1),P2(1),MA(1),
     *SD2(1,1),SD1(1,1),S(1,1),S2D2(1,1),S2D1(1,1),S2(1,1),OMETD1(1,1),
     *OMET(1,1)
      IF(ISTPL2.NE.0) WRITE(13) FT(2),SF(2),DELTA(2),P(2),P2(2),MA(2),
     *SD2(1,2),SD1(1,2),S(1,2),S2D2(1,2),S2D1(1,2),S2(1,2),OMETD1(1,2),
     *OMET(1,2)
      IF(ISTPL3.NE.0) WRITE(13) FT(3),SF(3),DELTA(3),P(3),P2(3),MA(3),
```

```
     *SD2(1,3),SD1(1,3),S(1,3),S2D2(1,3),S2D1(1,3),S2(1,3),OMETD1(1,3),
     *OMET(1,3)
      IF(ISTPL4.NE.0) WRITE(13) FT(4),SF(4),DELTA(4),P(4),P2(4),MA(4),
     *SD2(1,4),SD1(1,4),S(1,4),S2D2(1,4),S2D1(1,4),S2(1,4),OMETD1(1,4),
     *OMET(1,4)
      IF(ISTPL5.NE.0) WRITE(13) FT(5),SF(5),DELTA(5),P(5),P2(5),MA(5),
     *SD2(1,5),SD1(1,5),S(1,5),S2D2(1,5),S2D1(1,5),S2(1,5),OMETD1(1,5),
     *OMET(1,5)
      GO TO 230
  220 CONTINUE
*INSERT     CSCMOD2.153
  230 CONTINUE
*DELETE     CSCMODS.93,CSCMODS.93
      CALL STOVAR(8,VELDEC,WLFORR,ZDANT,ZSSC,COPA,SIPA,DSTOP,
*IDENT    CSCMOD4
*DELETE      ACOBLK.12,ACOBLK.12
     9,GAMAH(5)  ,GAMAN     ,GNR       ,HMM(5)    ,ICOSV(5)  ,INITSW
*DELETE      ACOBLK.13,ACOBLK.13
     *,IFR(5)    ,IIXSVH(5) ,IIXSVL(5) ,IOPCO(5)  ,IPASS(5)  ,ISET(5)
*DELETE      ACOBLK.22,ACOBLK.22
      COMMON /ACTIVE/ TC3   ,TC4       ,VCUM(5)   ,ENCG      ,VOL1I(5)
*DELETE     CSCMOD3.1,CSCMOD3.1
     *,IMODE(5)  ,CMASNG    ,VMASS(5)  ,ZDANT     ,INDEACT(5),VELDEC
*DELETE          EXE.175,EXE.175
      IF(TIME .LT. TMAX) GO TO 413
*DELETE          EXE.286,EXE.286
      STOP "EXECUTIVE ROUTINE"
*INSERT      JMMODS.1
      DIMENSION S(2,5)
      EQUIVALENCE (S1(1),S(1,1))
      EQUIVALENCE (DM15(1),ITO)
*DELETE     CSCMOD2.21,CSCMOD2.21
      DATA ICOSV,IFR,IFSTOP,IPASS /20*0/
*INSERT     CSCMOD2.25
      INITSW = 1
*INSERT     CSCMOD2.28
      INDEACT(I) = 0
      IF(ITO.EQ.1) INDEACT(I) = 2
*DELETE     ACTINIT.143,ACTINIT.143
      CALL ALGEAR
*DELETE      ALGEAR.70,ALGEAR.70
      EQUIVALENCE (DM15(1),ITO)
*DELETE      ALGEAR.93,ALGEAR.93
C
      IF(INITSW.EQ.1) GO TO 16
*INSERT      ALGEAR.98
C
      CALL LGEA3C
C
```

```
*DELETE     CSCMOD2.50,CSCMOD2.50
   18 CALL ACTNG
C
      DO 274 I=2,NSTRUT
*DELETE     CSCMOD2.53,CSCMOD2.53
*DELETE     CSCMOD2.54,CSCMOD2.54
*DELETE     CSCMOD2.62,CSCMOD2.72
*DELETE     CSCMOD2.74,CSCMOD2.74
*DELETE      ALGEAR.182,ALGEAR.182
  801 IF(SD1(1,I).LE.0.8 .AND. IFR(I).EQ.0) GO TO 2
*DELETE      ALGEAR.187,ALGEAR.187
      IFR(I) = 1
*DELETE      ALGEAR.200,ALGEAR.201
*DELETE      ALGEAR.212,ALGEAR.212
      IFR(I) = 0
*DELETE      ALGEAR.257,ALGEAR.258
*DELETE      ALGEAR.275,ALGEAR.276
*DELETE     CSCMOD2.114,CSCMOD2.117
*DELETE     CSCMOD2.119,CSCMOD2.119
*DELETE     CSCMOD3.29,CSCMOD3.29
     &    VMASS(I)*REDSLP(I))
*INSERT      ALGEAR.370
C
      DO 65 I=1,NSTRUT
*DELETE      ALGEAR.513,ALGEAR.513
   16 CONTINUE
      INITSW = 0
*DELETE      ALGEAR.523,ALGEAR.525
*DELETE     CSCMODS.48,CSCMODS.48
      DATA(ACOVAR9(I), I=1,8) /6HXSVDDD,5HXSVDD,6HXSVDOT,2HPR,3HIFR,
*DELETE      JMMODS.2,JMMODS.2
      DATA N20 /20/
*DELETE      JMMODS.3,JMMODS.3
      IF(ISUM1.NE.0) WRITE(13) N20,ISUM1,DAT3,OP17,ACOVAR8(1),
*INSERT      JMMODS.4
     2,ACOVAR3(3)
*INSERT     CSCMOD2.149
     2,FORSST(1)
*INSERT     CSCMOD2.150
     2,FORSST(2)
*INSERT     CSCMOD2.151
     2,FORSST(3)
*INSERT     CSCMOD2.152
     2,FORSST(4)
*INSERT     CSCMOD2.153
     2,FORSST(5)
*DELETE     CSCMODS.78,CSCMODS.78
      CALL STOVAR(8,XSVDDD(I),XSVDD(I),XSVDOT(I),PR(I),FLOAT(IFR(I)),
*IDENT    ACTNG
```

```
*INSERT      ALGEAR.549
*DECK ACTNG
       SUBROUTINE ACTNG
C
********************** FATOLA VARIABLES ********************************
C
      COMMON/DIRCOM/DM1(115),ALPHD,DM1A(20),AMASS,DM2(147),DCL1,DCM1,
     C DCN1,DCL2,DCM2,DCN2,DCL3,DCM3,DCN3,DM3(99),FXB7P,
     C DUM4(3),FYB7P(4),FZB7P,DM5(17),GXB7F,DM6(8),GZB7F,
     C DM7(218),INDSTE(48),PHIPD,INDSTE1(23),PSIPD,INDSTE2(156),THTPD,
     C INDSTE3(5),TIME,DM8(287),PI77R(2),PI77R1(2),DM9(4),
     C QI77R(2),QI77R1(2),DM10(4),RI77R(2),RI77R1(2),DM11(48),
     C XG77F(2),XG77F1(12),YG77F(2),YG77F1(12),
     C ZG77F(2),ZG77F1(2),DUM13(52),
     C NSTRUT,MASS(5),RX(5),RY(5),RZ(5),THETAD(5),ERDEG,RGR,
     C NTIRES(5),RZERO(5),W(5),DELTAM(5),MOMENT(5),
     C RF(5),VZ  ,IFD,PZERO(5),VZERO(5),A(5),P20(5),V20(5),
     C A2(5),IL,S2T(5),ES2(5),C2L(5),MASS2(5),MUS(5),
     C CC(5),CE(5),C2C(5),C2E(5),NVGPT,NPP,MB(5),RLT,NDELTA,
     C ES(5),SB(5),SD21(2),SD22(2),SD23(2),SD24(2),SD25(2)
C
      COMMON/DIRCOM/
     C SD11(2),SD12(2),SD13(2),SD14(2),SD15(2),
     C S1(2),SS2(2),S3(2),S4(2),S5(2),
     C S2D21(2),S2D22(2),S2D23(2),S2D24(2),S2D25(2),
     C S2D11(2),S2D12(2),S2D13(2),S2D14(2),S2D15(2),
     C S21(2),S22(2),S23(2),S24(2),S25(2),
     C OMTD11(2),OMTD12(2),OMTD13(2),OMTD14(2),OMTD15(2),
     C OMT1(2),OMT2(2),OMT3(2),OMT4(2),OMT5(2),
     C AI(5),BI(5),DELTA1,DELTA2,DELTA3,DELTA4,DELTA5,
     C DDELT1,DDELT2,DDELT3,DDELT4,DDELT5,ISTAGE,
     C PRTMIN,IPLT,ISDF,ISTPL1,ISTPL2,ISTPL3,ISTPL4,ISTPL5,
     C DM14(22),IB(5),DM15(127),INDLG,DM16(107),CASK(44),INDFLX,
     * NMODE,DM18(40),SXMOD(100),SYMOD(100),SZMOD(100),DM19(1686),
     * GOD2(20),DDM20(20),DDM21(20),SLEN1(5),SLEN2(5),
     * GAMA,DUM15(12),INDNWS,DDM22(10),ETADES,
     * DDM23(5),AH(5),PH(5),DDM24(30)
C
      COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTPT
C
      COMMON/LGE/A11(5),A13(5),A31(5),A33(5),RRCGX(5),
     C RL(3,3),RI(3,3,5),RAX(5),RAY(5),RAZ(5),TMP(3),ZZERO(5),
     C XR(5),YR(5),EPSLON(5),PA(5),FDELTA(5),
     C FTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
     C VTX(5),VTY(5),VTZ(5),GZ(5),      VGPT(5),FTRX(5),FTRY(5),
     C DX(5),DY(5),DZ(5),FT(5),FDX(5),FDY(5),FF(5),AA(5),C2(5),
     C SR(5),SF(5),PSKD(5),MUVP(5),MTRX(5),MTRY(5),
     C MTRZ(5),MA(5),RG11,RG13,RG31,RG33,IPRT,
     C MTX,MTY,MTZ,SFTRX,SFTRY,SFTRZ,FTRA,
```

```
     C FTRB,FTRC,SMTRX,SMTRY,SMTRZ
C
      COMMON /HTCOM/ HT, HT1, HT2, INDINT(5)
C
*CALL ACOBLK
C
      REAL MASS,MOMENT,MASS2,MUS,NTIRES,MB
      REAL MUVP,MTRX,MTRY,MTRZ,MA,MTX,MTY,MTZ
C
      DIMENSION IPSTOP(5),AIC(5),PGAIT(5)
      DIMENSION DELTA(5),DDELTA(5),DLGDE(47),
     *      SD2(2,5),SD1(2,5),S(2,5)
C
      EQUIVALENCE (DLGDE(1),LA(1))
      EQUIVALENCE (DELTA(1),DELTA1),(DDELTA(1),DDELT1)
      EQUIVALENCE (SD21(1),SD2(1,1)),(SD11(1),SD1(1,1)),(S1(1),S(1,1))
      EQUIVALENCE (DM15(1),ITO)
      EQUIVALENCE (DM5(16),GREFF),
     * (DM1(37),AIYYBS)
C
      DATA AIC,IPSTOP /5*0.0,5*0/
C
      Q1(T1,T2)=SIGN(1.,(T1-T2))*SQRT(ABS(T1-T2))
C*****************************************************************
C
      I = 1
C
      IF(INDEACT(I).EQ.1) GO TO 56
      IF(INDEACT(I).EQ.2) GO TO 80
      IF(HMM(I) .EQ. 0.) GO TO 90
      IF(OMRUN .GT. 0.0)GO TO 25
      IF(ZDANT.GE.VELDEC) 90,40
   25 IF(ZDANT.GE.VELDEC+XG77F1(1)*TAN(OMRUN)) GO TO 90
   40 WRITE(6,1014)TIME
1014  FORMAT (1H036H REDUCE CONTROL LIMIT FORCE AT TIME=,E16.8)
      INDEACT(I)=1
   56 IF(INDINT(I).EQ.0) GO TO 58
      WLFOR(I)=WLFOR(I)-REDSLP(I)*HT
      EPSILO(I)=EPSILO(I)+EPSSLP*HT
   58 CONTINUE
      INDINT(I)=0
      IF(WLFOR(I) .GT. WLFORR) GO TO 90
      WRITE(6,1015)TIME
1015  FORMAT (1H027H CONTROL AT WLFORR AT TIME=,E16.8)
      INDEACT(I)=2
   80 WLFOR(I)=WLFORR
      EPSILO(I)=EPSROL(I)
   90 CONTINUE
C*****************************************************************
```

```
C
      IF(KAPT(I).NE.0)GO TO 210
      APINT(I)=0.0
  210 CONTINUE
      PGA1T(I)=PGA1T1(I)
      IF(PGA1T1(I).LE.-1600.0) PGA1T(I)=-1600.0
      VOL1T(I)=VOL1I(I)-(AREA1(I)-APINT(I))*S(1,I)
      VOL3T(I)=VOL3I(I)+AREA3(I)*S(1,I)
      VOL2T(I)=VOL2I(I)-(AREA2(I)-AREA1(I)+APINT(I))*S(1,I)+(VOL3T(I)
     X -VOL3I(I))-VCUM(I)
      PGA2T(I)=AP2TO(I)*((VOL2I(I)/VOL2T(I))**GAMA)-PATM
      IF(SD1(1,I) .EQ. 0.0)GO TO 104
      PGA3T(I)=((COEF3(I)*AREO3(I))**2*PGA2T(I)-SD1(1,I)/ABS(SD1(1,I))
     X *(SD1(1,I)*AREA3(I))**2)
     &/((COEF3(I)*AREO3(I))**2)
      GO TO 105
  104 PGA3T(I)=PGA2T(I)
  105 IF(PGA1T(I) .GE. PGA2T(I))GO TO 106
      GO TO 107
  106 GAMAH(I)=RHOH*GREFF*(1.0+(PGA1T(I)*3.04E-08)-
     * (PGA1T(I)**2*2.72E-15))
      GO TO 108
  107 GAMAH(I)=RHOH*GREFF*(1.0+(PGA2T(I)*3.04E-08)-
     * (PGA2T(I)**2*2.72E-15))
  108 IF(PGA1T(I) .GE. PGA2T(I))COEFO(I)=
     *  CDMOC(I)*SQRT(ABS(2.*GREFF/GAMAH(I)))
      IF(PGA2T(I) .GT. PGA1T(I))COEFO(I)=
     *  CDMOE(I)*SQRT(ABS(2.*GREFF/GAMAH(I)))
      IF(IMODE(I).NE.0 .OR. DDELTA(I).GT.0.0) GO TO 113
      QO(I) = 0.0
      GO TO 109
C
  113 QO(I)=COEFO(I)*(AREMO(I)-APINT(I))*Q1(PGA1T1(I),PGA2T(I))
      IF(QO(I).LT.0.0 .AND. VCUM(I).LE.0.0) GO TO 102
      AIC(I)=0.0
      GO TO 103
  102 IF(PGA1T1(I).LT.PGA2T(I)) GO TO 103
      QO(I)=0.0
      VCUM(I)=0.0
      AIC(I)=1.0
  103 IF(QO(I).GT.0.0) AIC(I)=0.0
  109 IF(PGA2T(I) .LE. -1600.0)PGA2T(I)=-1600.0
      IF(PGA3T(I) .LE. -1600.0)PGA3T(I)=-1600.0
      IF(DELTA(I) .LE. 0.0 .AND. TIME .GT. DELT)GO TO 101
      GO TO 110
  101 FFORT(I)=0.0
      GO TO 140
  110  CONTINUE
C     COMPUTE STRUT AXIAL BINDING FRICTION FORCE
```

```
      BLFORT(I)=FONHST(I)*((SLEN2(I)-S(1,I))/(SLEN1(I)+S(1,I))+1.0)
      BUFORT(I)=FONHST(I)*(SLEN2(I)-S(1,I))/(SLEN1(I)+S(1,I))
      FFORT(I)=BUMU(I)*ABS(BUFORT(I))+BLMU(I)*ABS(BLFORT(I))
 140  CONTINUE
C     COMPUTE SHOCK STRUT CHARGING FORCE
      IF(S(1,I) .GT. 0.0)GO TO 142
      FORCHT(I)=PGA1T(I)*AREA1(I)+PGA2T(I)*(AREA2(I)-AREA1(I))-PGA3T(I)
     X *AREA3(I)+FFORT(I)+CFFOR(I)
C     COMPUTE NORMAL AND AXIAL HUB TO SHOCK STRUT FORCES AT HUB
 142  FONHST(I)=SQRT(FDX(I)**2+FDY(I)**2)-MASS(I)*GREFF*SIPA+SBFOT
      IF(ABS(FT(I)) .LE. FORCHT(I) .AND. S(1,I) .EQ. 0.0)GO TO 150
      GO TO 801
 150  CONTINUE
      FORSST(I)=FT(I)
      SD1(1,I)=0.0
      IF(IMODE(I).EQ.0) GO TO 297
      ISTROK(I)=1
      GO TO 289
C     COMPRESSION VELOCITY OF SHOCK STRUT IS POSITIVE
  801 IF(SD1(1,I).LE.0.8 .AND. IFR(I).EQ.0) GO TO 2
      GO TO 3
    2 DMTANH(I)=1.0
      GO TO 284
    3 DMTANH(I)=ABS(TANH(2.0*SD1(1,I)))
      IFR(I) = 1
  284 IF(S(1,I).LT.0.0) GO TO 160
      GO TO 161
  160 IPSTOP(I)=1
  161 IF(S(1,I).LE.DSTOP .AND. IPSTOP(I).EQ.1) GO TO 900
      GO TO 902
  900 IF(S(1,I).LE.0.005) GO TO 903
      GO TO 904
 903  SD2(1,I)=0.0
      SD1(1,I)=0.0
      S(1,I)=0.0
      IPSTOP(I)=0
      GO TO 902
 904  CONTINUE
      IF(IFSTOP(I) .NE. 0)GO TO 906
      DSTOP=S(1,I)
      FSTOPK=2.0*MASS(I)*SD1(1,I)**2/DSTOP**2
 906  IF(S(1,I) .LE. DSTOP/2.0)GO TO 908
      FSTOP(I)=-FSTOPK*(DSTOP-S(1,I))
      GO TO 909
 908  FSTOP(I)=-FSTOPK*S(1,I)
 909  IFSTOP(I)=1
      IFR(I) = 0
      GO TO 901
 902  FSTOP(I)=0.0
```

```
 901 IF(PGA1T(I).LE.(PGA1I(I)+500.0) .AND.
     :  PGA1T(I).GT.(PGA1I(I)-500.0)) GO TO 158
     GO TO 159
 158 IF(ABS(FT(I)).LE.FORCHT(I) .AND. S(1,I).EQ.0.0) GO TO 500
 159 IF(S(1,I).GE.0.0) GO TO 470
     IF(SD1(1,I) .LT. 0.0)GO TO 470
     GO TO 471
 470 FFORT(I)=-FFORT(I)
     CFFOR(I)=-CFFOR(I)
 471 FORSST(I)=-((PGA1T(I)-PGA2T(I))*(AREA1(I)-APINT(I))
    & +PGA2T(I)*AREA2(I)
    X -PGA3T(I)*AREA3(I)+(FFORT(I)
    1        +CFFOR(I))*DMTANH(I)+FSTOP(I))
 500 IF(INDFLX.GE. 1)GO TO 295
     IF(IMODE(I).EQ.0) GO TO 297
     ISTROK(I)=1
 289 IF(S(1,I) .GT. 0.0) GO TO 295
     IF(IOPCO(I) .EQ. 1) GO TO 295
     IF((PGA1I(I)-1000.0).LT.PGA1T(I).AND.
    &  PGA1T(I).LT.(PGA1I(I)+1000.0))299,298
 299 IF(XVALVE(I) .NE. 0.0)GO TO 311
     IF(IPASS(I) .EQ. 1)GO TO 296
     XVALVE(I)=XKSV(I)*XMA11(I)+XBIAS(I)
     IPASS(I)=1
     GO TO 294
 298 IF(ICOSV(I) .EQ. 1)GO TO 291
     IOPCO(I)=0
     IF(XSV(I) .LT. 0.002 .AND. XSV(I) .GT. -0.002)291,295
 291 IF(SD2(1,I) .LE. 0.0 .AND. ICOSV(I) .EQ. 1)GO TO 311
     IF(IOPCO(I) .EQ. 1)GO TO 295
     IF(PGA1T(I) .LE. PGA1I(I)) GO TO 293
     IF(IXSVL(I) .EQ. 1)GO TO 294
     XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
     IF(XVALVE(I) .GT. -0.1) GO TO 294
     XVALVE(I)=-0.1
     IXSVL(I)=1
     GO TO 294
 293 IF(IXSVH(I) .EQ. 1)GO TO 294
     XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
     IF(XVALVE(I) .LT. 0.1) GO TO 294
     XVALVE(I)=0.1
     IXSVH(I)=1
 294 CONTINUE
     DLTX1D(I)=0.0
     ICOSV(I)=1
 296 IF(WFORT(I) .GT. 0.0 .AND. S(1,I) .LE. 0.0)GO TO 410
 311 IF(NAC(I) .EQ. 1)GO TO 307
     IF(IIXSVH(I) .EQ. 1)GO TO 305
     XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
```

```
      IF(XVALVE(I) .LE. 0.0)305,400
  305 XVALVE(I)=0.0
      IIXSVH(I)=1
      GO TO 400
  307 IF(IIXSVL(I) .EQ. 1)GO TO 308
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
      IF(XVALVE(I) .GE. 0.0)308,400
  308 XVALVE(I)=0.0
      IIXSVL(I)=1
  400 CONTINUE
  410 IF(XVALVE(I) .NE. 0.0)GO TO 295
      ICOSV(I)=0
      DELTX1(I)=DELTX(I)*XKF(I)
      XMA(I)=(DF(I)+DELTX1(I))*XKA(I)
      XMA11(I)=XMA(I)
      XSV(I)=XKSV(I)*XMA11(I)+XBIAS(I)
      CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      IPASS(I)=0
      IXSVL(I)=0
      IXSVH(I)=0
      IIXSVL(I)=0
      IIXSVH(I)=0
      IOPCO(I)=1
      CALL PHLOZ2(PS(I),PR(I),XSV(I),QC(I),XLPSV1(I),XLPSV3(I),RCLSV(I),
     & DSV(I),CSV1(I),CSV3(I),XMU,QTOLER,NITER,P1(I),QS1(I),QS3(I))
C
  295 IF(ISTROK(I) .EQ. 1 .AND. S(1,I) .GT. 0.0)IOPCO(I)=0
  297 I2 = 2*I+NSTRUT-1
      I1 = I2+1
      CALL INTEG(LA(I2),SD2(1,I))
      CALL INTEG(LA(I1),SD1(1,I))
C
      IF(IMODE(I).EQ.0) GO TO 450
C
      ENUP(1) = 0.5*AIYYBS*QI77R(1)*QI77R(1)+(ENCG/CMASNG)*
     *       (ZG77F1(1)/ABS(ZG77F1(1)))
      IF(HMM(I) .EQ. 1.0)GO TO 130
      SA(I)=0.
      IF(WFORT(I).LT.0.) XSTOT(I)=ENUP(I)/((-WFORT(I))*COPA)
      IF(WFORT(I).GE.0.0 .OR. DDELTA(I).LT.0.0) XSTOT(I)=1.E20
C     ZSSC IS A PERCENTAGE OF SB(I) FOR ACTIVATING CONTROL-CDMOC(I) IS U
      ZSSC(I)=FWORK(I)*SB(I)
      IF(XSTOT(I).LE.(ZSSC(I)-S(1,I)) .OR. RESA(I).EQ.1.0) SA(I)=1.0
      RESA(I)=SA(I)
      IF(SA(I).EQ.0. .OR. HMM(I).EQ.1.) GO TO 130
      WLFOR(I)=-WFORT(I)
C
      VELDEC=((WLFOR(I)+WLFORR)/2.*(WLFOR(I)-WLFORR))/
     &   (VMASS(1)*REDSLP(I))
```

```
      WRITE(6,121)TIME,WLFOR(I),VELDEC
  121 FORMAT(50H ACTIVE CONTROL INITIATED...TIME, WLFOR, VELDEC = ,
     1       3E13.5)
      HMM(I)=1.
  130 IF(S(1,I) .GT. 0.0)ISET(I)=0
      IF(HMM(I).EQ.0.) GO TO 451
      IF(-WFORT(I).GT.(WLFOR(I)+EPSILO(I))) DF(I)=(WLFOR(I)+
     &  EPSILO(I))-(-WFORT(I))
      IF(-WFORT(I).LT.(WLFOR(I)-EPSILO(I))) DF(I)=(WLFOR(I)-
     &  EPSILO(I))-(-WFORT(I))
      IF(-WFORT(I).LE.(WLFOR(I)+EPSILO(I)).AND.
     & -WFORT(I).GE.(WLFOR(I)-EPSILO(I)))
     : 457,456
  457 IF(S(1,I) .LE. 0.0)GO TO 456
      IF(WFORT(I) .GT. 0.0 .AND. QSVCU(I) .LT. 0.0)454,455
  454 DF(I)=WLFOR(I)-(-WFORT(I))
      GO TO 456
  455 DF(I)=0.0
  456 DELTX(I)=S(1,I)-XSCOM(I)
      IF(S(1,I) .LE. 0.0 .AND. ISET(I) .EQ. 0)GO TO 451
      GO TO 452
  451 DF(I)=0.
      DELTX(I)=0.
  452 XMA(I)=(DF(I)+DELTX1(I))*XKA(I)
      IF(GNR.EQ.1. .AND. XMA(I).GT.0.) XMA(I)=
     :  XMA(I)*SQRT((PGA1T(I)-PGALAC(I))
     X /(PGAHAC(I)-PGA1T(I)))
C
C      NOTE: SUBROUTINE 'FLOZE2' COMPUTES THE FLOWS FROM THE PRESSURES
C      IN UNITS OF INCHES.
      P1(I)=PGA1T(I)/144.
C     COMPUTATION OF HIGH PRESSURE ACCUMULATOR NITROGEN VOLUME
C     AND ACCUMULATOR PRESSURE
      VOLANT(I)=VOLANT(I)+QSVN(I)*DELT-QPUMPS(I)*DELT
      PS(I)=(((PGAHAC(I)+PATM)*(VOLANI(I)/VOLANT(I))**GAMA)-PATM)/144.0
      IF(PS(I) .GE. 3000.0)464,465
  464 PS(I)=3000.0
      VOLANT(I)=VOLANI(I)
  465 VOLAHT(I)=VOLACI(I)-VOLANT(I)
      IF(VOLAHT(I) .LE. 0.0)466,467
  466 WRITE(6,1050)TIME
 1050 FORMAT(1HO//45H ACCUMULATOR OIL VOLUME INSUFFICENT AT TIME=,E16.8
     1       //)
      CALL LGEAR6
      STOP 500
  467 CONTINUE
      CALL FLOZE2(PS(I),PR(I),P1(I),XLPSV1(I),XLPSV3(I),RCLSV(I),DSV(I),
     &  XSV(I),QS1(I),QS3(I),CSV1(I),CSV3(I),XMU)
      QSV1(I)=QS1(I)/1728.
```

```
      QSV3(I)=QS3(I)/1728.
C
  450 CONTINUE
      QSV(I)=QSV1(I)-QSV3(I)
      IF(QSV(I) .LT. 0.0)NAC(I)=1
      IF(QSV(I) .GT. 0.0)NAC(I)=2
      IF(NAC(I) .NE. 2) GO TO 462
      QSVN(I)=QSV(I)
      GO TO 463
 462  QSVN(I)=0.0
 463  IF(SD1(1,I).LT. 0.0 .AND.PGA1T(I).LE. -1600.0)PGA1T(I)=-1600.0
C
      RETURN
      END
*IDENT    CSCMOD5
*DELETE       ACOBLK.7,ACOBLK.7
     4,ALGDUM1    ,COEF0(5)   ,COEF3(5)   ,COPA
*DELETE      CSCMOD4.1,CSCMOD4.1
     9,GAMAH(5)   ,ALGDUM2    ,GNR        ,HMM(5)     ,ICOSV(5)   ,INITSW
*DELETE      CSCMOD4.4,CSCMOD4.4
     *,IMODE(5)   ,CMASNG     ,VMASS(5)   ,ZDANT      ,INDEACT(5),VELDEC(5)
     1,COEF1(5)   ,LMODE(5)
*DELETE      CSCMOD4.5,CSCMOD4.5
      IF(TPD.LT.TMAX) GO TO 413
*DELETE        MIMIN.114,MIMIN.114
      IF((XF-XO).GT.1.E-10) GO TO 211
*DELETE      CSCMOD3.21,CSCMOD3.21
*DELETE      ACTINIT.33,ACTINIT.33
     *,DDM21(20),SLENDUM(10),GAMA,DUM15(12),INDNWS,DDM22(10),ETADES,
*DELETE      ACTINIT.35,ACTINIT.35
*DELETE      CSCMOD2.22,CSCMOD2.22
      DATA IOPCO,ISET,ISTROK,NAC /10*1,10*0/
*INSERT      CSCMOD2.23
      DATA INDEACT,INDINT,INITSW /5*0,5*1,1/
      DATA FFORT,FONHST,FORCHT,FORSST,WFORT /25*0.0/
      DATA QC,QD,QSV1,QSV3,QSVCU,QTOLER /25*0.,0.0001/
      DATA DCON,DMTANH,DF,DP1,DELTX,DELTX1,DLTX1D /6*1.0,25*0.0/
      DATA XSVDOT,XSVDD,XSVDDD /15*0.0/
      DATA REDSLP,SBFOT,VCUM,VELDEC /5*100000.,0.,10*0.0/
      DATA XMA1,XMA2,XMA3,XMA4,XMA6,XMA7,XMA9,XMA10 /40*0.0/
      DATA XMA,XMA5,XMA8,XMA11 /20*0.0/
*DELETE      ACTINIT.71,ACTINIT.71
*DELETE      ACTINIT.73,ACTINIT.75
*DELETE      CSCMOD4.11,CSCMOD4.11
*DELETE      ACTINIT.80,ACTINIT.82
*DELETE      ACTINIT.87,ACTINIT.105
*DELETE      ACTINIT.107,ACTINIT.108
*DELETE      ACTINIT.110,ACTINIT.115
      COEF1(I) = CDSV(I)*SQRT(2.*GREFF/GAMAH(I))*144.
```

```
      CSV1(I) = COEF1(I)*WSV1
      CSV3(I) = COEF1(I)*WSV3
*DELETE     ACTINIT.118,ACTINIT.118
      OSV1(I) = 0.0
      OSV3(I) = 0.0
      IF(ITO.EQ.1) GO TO 60
      IF(IMODE(I).EQ.0) GO TO 80
C
*INSERT     ACTINIT.126
      GO TO 80
C
   60 CONTINUE
      HMM(I) = 1.0
      INDEACT(I) = 2
      LMODE(I) = IMODE(I)
      IMODE(I) = 0
      AP2TO(I) = PGA2I(I)+PATM
      VOL1T(I) = VOL1I(I)-(AREA1(I)-APINT(I))*S(1,I)
      VOL3T(I) = VOL3I(I)+AREA3(I)*S(1,I)
      VOL2T(I) = VOL2I(I)-(AREA2(I)-AREA1(I)+APINT(I))*S(1,I)
     .           +(VOL3T(I)-VOL3I(I))-VCUM(I)
      PGA2I(I) = AP2TO(I)*((VOL2I(I)/VOL2T(I))**GAMA)-PATM
      PGA1I(I) = PGA2I(I)
      PGA3I(I) = PGA2I(I)
      FORSST(I) = -(PGA2I(I)*AREA2(I)-PGA3I(I)*AREA3(I)
     .            +DMTANH(I)*(FFORT(I)+CFFOR(I))+FSTOP(I))
      WFORT(I) = FORSST(I)
C
   80 CONTINUE
*DELETE     ACTINIT.130,ACTINIT.130
*DELETE     ACTINIT.132,ACTINIT.132
*DELETE     ACTINIT.135,ACTINIT.139
*DELETE     CSCMOD4.12,CSCMOD4.13
*DELETE     CSCMOD4.14,CSCMOD4.14
      CALL ALGEAR1
*INSERT      ALGEAR.69
      EQUIVALENCE (DM8(79),XCGRF)
*DELETE     CSCMOD2.42,CSCMOD2.43
      DIMENSION IPSTOP(5),AIC(5),PGA1T(5)
      DATA AIC,IPSTOP /5*0.0,5*0/
*DELETE     CSCMOD3.24,CSCMOD3.24
      RXCG1 = RX(1)-XCGRF
      CMASNG = 1+(AMASS*RXCG1*RXCG1)/AIYYBS
*DELETE     CSCMOD2.48,CSCMOD2.48
      ZDANT = ZG77F1(1)-QI77R(1)*RX(1)
*DELETE     CSCMOD4.21,CSCMOD4.21
   18 CONTINUE
*INSERT     CSCMOD4.22
      UNSPRNG = 0.0
```

```
      DO 22 I=1,NSTRUT
         IF(ITO.EQ.1 .AND. TIME.GE.2.0) IMODE(I) = LMODE(I)
         IF(OMETD1(I).NE.0.0) UNSPRNG = UNSPRNG+MASS(I)
   22 CONTINUE
C
      DWFORT = (-SQRT(AXP7F*AXP7F+AYP7F*AYP7F+AZP7F*AZP7F)+GREFF)*
     .          (AMASS-UNSPRNG)
C
      CALL ACTNG
C
*INSERT     CSCMOD4.23
C
      WFORT(I) = DWFORT/(NSTRUT-1)
C
*DELETE       ALGEAR.106,ALGEAR.106
      IF(ZG77F1(1).GE.VELDEC(I)) GO TO 90
*DELETE       ALGEAR.108,ALGEAR.108
   25 IF(ZG77F1(1).GE.VELDEC(I)+XG77F1(1)*TAN(OMRUN)) GO TO 90
*DELETE       ALGEAR.256,ALGEAR.256
*DELETE       ALGEAR.261,ALGEAR.261
  296 IF(WFORT(I).GT.0.0 .AND. S(1,I).LE.0.0) GO TO 400
*DELETE       ALGEAR.274,ALGEAR.274
  400 CONTINUE
*DELETE       ALGEAR.277,ALGEAR.277
      IF(XVALVE(I).NE.0.0) GO TO 295
*DELETE     CSCMOD2.113,CSCMOD2.113
      IF(IMODE(I).NE.0.0) GO TO 119
      OSV1(I) = 0.0
      OSV3(I) = 0.0
      GO TO 450
*DELETE     CSCMOD2.124,CSCMOD2.124
      VELDEC(I) = ((WLFOR(I)+WLFORR)/2.*(WLFOR(I)-WLFORR))/
*DELETE     CSCMOD4.27,CSCMOD4.27
     .     (VMASS(I)*REDSLP(I))
*DELETE       ALGEAR.308,ALGEAR.308
      WRITE(6,121) TIME,WLFOR(I),VELDEC(I)
*DELETE        ACTNG.60,ACTNG.60
     .   SD2(2,5),SD1(2,5),S(2,5),OMETD1(2,5)
*DELETE        ACTNG.62,ACTNG.62
      EQUIVALENCE (DLGDE(1),LA(1)), (OMTD11(1),OMETD1(1,1))
*DELETE        ACTNG.80,ACTNG.80
      IF(ZDANT.GE.VELDEC(I)) 90,40
*DELETE        ACTNG.81,ACTNG.81
   25 IF(ZDANT.GE.VELDEC(I)+XG77F1(1)*TAN(OMRUN)) GO TO 90
*INSERT        ACTNG.97
      UNSPRNG = 0.0
      IF(OMETD1(1).NE.0.0) UNSPRNG = MASS(1)
      WFORT(1) = SR(1)*(VMASS(1)-UNSPRNG)
*DELETE        ACTNG.241,ACTNG.241
```

```
  296 IF(WFORT(I).GT.0.0 .AND. S(1,I).LE.0.0) GO TO 400
*DELETE        ACTNG.255,ACTNG.255
       IF(XVALVE(I).NE.0.0) GO TO 295
*DELETE        ACTNG.277,ACTNG.277
       IF(IMODE(I).NE.0.0) GO TO 119
       OSV1(I) = 0.0
       OSV3(I) = 0.0
       GO TO 450
  119 CONTINUE
*DELETE        ACTNG.292,ACTNG.292
       VELDEC(I) = ((WLFOR(I)+WLFORR)/2.*(WLFOR(I)-WLFORR))/
*DELETE        ACTNG.294,ACTNG.294
       WRITE(6,121) TIME,WLFOR(I),VELDEC(I)
*DELETE       DECOMP.4,DECOMP.4
       COMMON /IPSCOM/ IPS
*DELETE        SOLVE.4,SOLVE.4
       COMMON /IPSCOM/ IPS
*INSERT        ACTIN.11
C
C-----------------------------------------------------------------
C    THE 100 LOOP PROCESSES DATA CARDS UNTIL ENDACT IS ENCOUNTERED
C-----------------------------------------------------------------
C
*INSERT        ACTIN.17
C
C---------- READ A DATA CARD -------------------------------------
*INSERT        ACTIN.22
C
C---------- IGNORE REMARKS ---------------------------------------
*INSERT        ACTIN.23
C
C---------- TEST FOR END OF ACTIVE DATA --------------------------
*INSERT        ACTIN.24
C
C---------- LOOK FOR A MATCHING MNEUMONIC ------------------------
*INSERT        ACTIN.28
C
C---------- NO MATCH FOUND ---------------------------------------
*INSERT        ACTIN.30
C
C---------- FOUND A MATCH ----------------------------------------
*INSERT        ACTIN.31
C
C-----------------------------------------------------------------
C    SET INDEX TO ADDRESS IN ACTIVE AND CORRECT FOR OFFSET
C    (INC ON DATA CARD)
C-----------------------------------------------------------------
C
*INSERT        ACTIN.34
```

C-3

```
C
C---------- INITIALIZE ALL FLAGS AND COUNTERS ----------------------
*INSERT       ACTIN.44
C
C-------------------------------------------------------------------
C    THE 210 LOOP SCANS DATA COLUMNS ON A SINGLE CARD
C-------------------------------------------------------------------
C
*INSERT       ACTIN.46
C
C---------- IGNORE BLANKS ------------------------------------------
*INSERT       ACTIN.47
C
C---------- COMMA SEPARATES NUMBERS --------------------------------
*INSERT       ACTIN.48
C
C---------- REAL NUMBER --------------------------------------------
*INSERT       ACTIN.49
C
C---------- SCIENTIFIC NOTATION ------------------------------------
*INSERT       ACTIN.50
C
C---------- NEGATIVE -----------------------------------------------
*INSERT       ACTIN.51
C
C---------- NON-NUMERIC CHARACTER ----------------------------------
*INSERT       ACTIN.52
C
C---------- ALPHABETIC CHARACTER -----------------------------------
*INSERT       ACTIN.53
C
C---------- CORRECT FROM DISPLAY CODE TO INTEGER DIGIT -------------
*INSERT       ACTIN.55
C
C---------- LEFT POSITIVE INDICATES NO DECIMAL POINT HAS BEEN FOUND
*INSERT       ACTIN.57
C
C---------- LEFT NEGATIVE INDICATES A DECIMAL POINT HAS BEEN FOUND -
*INSERT       ACTIN.63
C
C-------------------------------------------------------------------
C    A COMMA OR END OF DATA CARD SIGNALS END OF NUMBER
C-------------------------------------------------------------------
C
*INSERT       ACTIN.66
C
C---------- COMBINE LEFT AND RIGHT PORTIONS OF REAL NUMBER ---------
*INSERT       ACTIN.70
C
```

```
C---------- STORE IN ACTIVE BLOCK -------------------------
*INSERT       ACTIN.71
C
C---------- RESET FLAGS AND COUNTERS ----------------------
*INSERT       ACTIN.83
C
C---------- NO DECIMAL POINT FOUND ------------------------
*INSERT       ACTIN.86
C
C---------- DATA GIVEN IN SCIENTIFIC NOTATION -------------
*INSERT       ACTIN.89
C
C---------- INTEGER DATA ----------------------------------
*INSERT       ACTIN.90
C
C---------- REAL DATA -------------------------------------
*INSERT       ACTIN.93
C
C---------- SET FLAG TO SHOW A DECIMAL POINT WAS FOUND -----
*INSERT       ACTIN.96
C
C---------- SET FLAG TO SHOW SCIENTIFIC NOTATION ----------
*INSERT       ACTIN.99
C
C---------- ACCUMULATE EXPONENT ---------------------------
*INSERT       ACTIN.103
C
C---------- NEGATIVE NUMBER -------------------------------
*INSERT       ACTIN.104
C
C---------- NEGATIVE EXPONENT -----------------------------
*INSERT      DIRACT.3
C
C-------------------------------------------------------------
C    ACTIVE CONTROL DATA IDENTIFIED BY THE MTH MNEUMONIC
C    IN ARRAY NAME IS STORED IN THE ACTIVE COMMON BLOCK
C    AT THE ADDRESS CONTAINED IN THE MTH WORD IN ARRAY LOC.
C-------------------------------------------------------------
C
*IDENT    PINARY
*DELETE     CSCMOD5.4,CSCMOD5.4
     1,COEF1(5),LMODE(5),PINN(30),PINM(30),STRON(30),
     2 STROM(30)
*DELETE     CSCMOD3.20,CSCMOD3.20
      COMMON/ACTDIR/NAME(75),LOC(75)
*INSERT     CSCMOD5.21
      IF(KAPT(I).EQ.1) GO TO 50
      APINT(I) = 0.
      IF(KAPT(I).EQ.0) GO TO 50
```

```
      IF(I.GT.1) GO TO 25
      STROK = -1.
      DO 10 J=1,29
      IF(STRON(J).LT.STROK) GO TO 15
      STROK = STRON(J)
      IF(S(1,I).GE.STRON(J).AND.S(1,I).LE.STRON(J+1)) GO TO 20
   10 CONTINUE
   15 J = J - 1
   20 APINT(I) = PINN(J)
      GO TO 50
   25 STROK = -1.
      DO 30 J=1,29
      IF(STROM(J).LT.STROK) GO TO 35
      STROK = STROM(J)
      IF(S(1,I).GE.STROM(J).AND.S(1,I).LE.STROM(J+1)) GO TO 40
   30 CONTINUE
   35 J = J - 1
   40 APINT(I) = PINM(J)
   50 CONTINUE
*DELETE       ALGEAR.126,ALGEAR.128
*DELETE        ACTNG.99,ACTNG.101
*DELETE        ACTIN.3,ACTIN.3
      COMMON/ACTDIR/XNAME(75),LOC(75)
*DELETE     CSCMOD2.34,CSCMOD2.35
      COMMON/ACTIVE/DATA(802)
      DIMENSION IRA(55),MSG(58),IDATA(802)
*INSERT       ACTIN.11
C
C       ZERO OUT PIN AND STROKE ARRAYS
C
      DO 10 J=683,802
      DATA(J) = 0.
   10 CONTINUE
*DELETE       ACTIN.13,ACTIN.13
    1 FORMAT(A6,1X,A3,1X,55R1,I2)
*DELETE       ACTIN.25,ACTIN.25
      DO 110 I=1,75
*DELETE      DIRACT.3,DIRACT.3
      COMMON/ACTDIR/NAME(75),LOC(75)
*DELETE     CSCMOD2.31,CSCMOD2.31
     5 6HZETAC1,  6HZETAC2,  6HIMODE ,  6HPINN  ,  6HPINM  ,
     6 6HSTRON ,  6HSTROM /
*DELETE     CSCMOD2.33,CSCMOD2.33
     7     644, 645, 651, 683, 713, 743, 773/
*IDENT    KLUGEZ
*INSERT      LGEAR1.155
C
C*********
C
```

```
C********** TEMPORARY FIX TO FREEZE NOSE SECONDARY
C
      IF(I.EQ.1) GO TO 59
C
C**********
C
*IDENT    SECFIX
*INSERT       LGEAR1.173
      S2(1,I) = -0.5*ES2(I)
      S2D1(1,I) = -1.E-10
      S2D2(1,I) = -1.E-10
*DELETE       LGEAR1.183,LGEAR1.184
   61 IF(S2D2(1,I).LT.0.) GO TO 140
      IF(S2D1(1,I).LT.0.) S2D1(1,I) = 0.
*INSERT       LGEAR1.187
      S2D1(1,I) = 0.
*INSERT       LGEA3C.109
      IF(TMP(1).LT.0.) WRITE(6,1234) I,S2(1,I),S2D1(1,I),
     1 S2D2(1,I)
 1234 FORMAT(1X,7H*-*-*-*,I5,3E16.8)
      IF(TMP(1).LT.0.) GO TO 31
*IDENT    TABFIX
*DELETE       LGEA3C.205,LGEA3C.205
      CALL HIHO(3,LOC(7),NS2NDY,NSTRUT,DU,DU,S2(1,I),FI,DU,DU,C2(I))
*DELETE       LGEA3C.208,LGEA3C.208
      CALL HIHO(3,LOC(6),NS2NDY,NSTRUT,DU,DU,S2(1,I),FI,DU,DU,C2(I))
*IDENT    XSVMOD
*DELETE       ALGEAR.224,ALGEAR.224
  500 CONTINUE
*INSERT       ALGEAR.230
      XVALVE(I) = XSV(I)
*DELETE       ALGEAR.233,ALGEAR.233
  299 CONTINUE
      IF(XVALVE(I).NE.0.) GO TO 311
*INSERT     CSCMOD5.70
      XSV(I) = XVALVE(I)
*DELETE        ACTNG.209,ACTNG.209
  500 CONTINUE
*INSERT        ACTNG.213
      XVALVE(I) = XSV(I)
*DELETE        ACTNG.216,ACTNG.216
  299 CONTINUE
      IF(XVALVE(I).NE.0.) GO TO 311
*INSERT        ACTNG.254
      XSV(I) = XVALVE(I)
*IDENT    GENFIX
*DELETE         REST.32,REST.32
      COMMON/XACTNG/AICX(5),IPSTOX(5),DM35(3)
*DELETE          EXE.24,EXE.24
```

```
     /*       DM43        ,INDVPC      ,DM44   (   7),NCASE       ,DM45        ,
*DELETE       SDDMODS.1,SDDMODS.1
   34 DO 35 II=1,4059
*INSERT           EXE.55
       NCASE = 1H
*INSERT        XMAFIX.7
       REWIND 7
*DELETE       CSCMOD6.4,CSCMOD6.5
*INSERT         MIMIN.113
       DO 45 I=1,5
   45 INDINT(I) = 1
*DELETE       DELTFIX.11,DELTFIX.11
       COMMON/XACTNG/AIC,IPSTOP,UNSPRNG,NIN,TIMEL
*IDENT    DELTFXX
*INSERT       CSCMOD5.48
       DATA TIMEL,NIN/0.,0/
*INSERT        ALGEAR.83
       IF(NIN.EQ.0) TIMEL = TIME
       NIN = 1
       DELT = TIME - TIMEL
       TIMEL = TIME
*DELETE       DELTFIX.3,DELTFIX.5
*INSERT         ACTNG.67
       DATA TIMEL,NIN/0.,0/
*DELETE       DELTFIX.13,DELTFIX.15
       IF(NIN.EQ.0) TIMEL = TIME
       NIN = 1
       DELT = TIME - TIMEL
       TIMEL = TIME
*IDENT    MOD282
*DELETE          REST.31,REST.31
       COMMON/XALGEA/AIC(5),DM34(14),IPSTOP(5),DM34X(12)
*DELETE        XMAFIX.7,XMAFIX.7
       READ(7) (DMD1(IJ),IJ=1,11081)
*DELETE        XMAFIX.8,XMAFIX.8
       WRITE(7) (DMD1(IJ),IJ=1,11081)
*DELETE          REST.82,REST.82
      2 PGA1T,RXCG1,TTIME,UNSPRNG,NIN,TIMEL
*DELETE       CSCMOD2.106,ALGEAR.217
  159 IF(SD1(1,I).LT.0.) GO TO 471
*DELETE         ACTNG.200,ACTNG.202
  159 IF(SD1(1,I).LT.0.) GO TO 471
*IDENT    MOD296
*DELETE       CSCMOD2.80,CSCMOD2.80
       IF(IMODE(I).EQ.0.AND.S(1,I).EQ.0.) 112,113
*DELETE       CSCMOD2.133,CSCMOD2.133
*DELETE         ACTNG.126,ACTNG.126
       IF(S(1,I).GT.0..OR.IMODE(I).EQ.1) GO TO 113
*IDENT    REST1
```

```
*DELETE         REST.35,REST.35
      COMMON/XAUTS/DM38(11)
*DELETE       MOD282.2,MOD282.2
      READ(7) (DMD1(IJ),IJ=1,11084)
*DELETE       MOD282.3,MOD282.3
      WRITE(7) (DMD1(IJ),IJ=1,11084)
*DELETE         REST.93,REST.93
     1 TMP2,TMP3,TMP5,DELTS,ERROR,IPR
*IDENT    MOD329
*INSERT         MIMIN.17
      H = AMIN1(HT,HT1,HT2,HT3)
*DELETE         HTTRY.2,HTTRY.2
*DELETE      CSCMOD2.131,CSCMOD2.132
*INSERT        ALGEAR.439
      DP1(I) = (-QO(I)+QSV1(I)-QSV3(I)+(AREA1(I) - APINT(I))
     1 *SD1(1,I))*BETA/VOL1T(I)
       IF(S(1,I).NE.0..OR.AIC(I).NE.1.) GO TO 64
       DP1(I) = 0.
   64 CONTINUE
       I3 = I + 3*NSTRUT
       CALL INTEG(LA(I3),DP1(I))
*INSERT        ALGEAR.515
      IF(IMODE(I).EQ.0) GO TO 28
*DELETE        ALGEAR.517,ALGEAR.518
*DELETE      CSCMOD2.138,CSCMOD2.139
       I9 = I +9*NSTRUT
       CALL INTEG(LA(I9),DLTX1D(I))
*INSERT      CSCMODS.102
      IF(IMODE(I).EQ.0) GO TO 6
*IDENT    MOD351
*INSERT        MOD329.5
      PGA1T1(I) = PGA1I(I)
      PGA1T(I) = PGA1I(I)
*IDENT    MOD1029
*DELETE         REST.14,REST.14
      COMMON/UPDCAL/DM14(201)
*DELETE         REST.24,REST.24
      COMMON/XMIMIN/DM26(1008)
*DELETE      CSCMOD3.4,CSCMOD3.4
      COMMON/UPDCAL/NUM,P(100),Y(100)
*DELETE        REST1.2,REST1.2
      READ(7) (DMD1(IJ),IJ=1,11204)
*DELETE        REST1.3,REST1.3
      WRITE(7) (DMD1(IJ),IJ=1,11204)
*DELETE        INUPD.5,INUPD.5
      COMMON/UPDCAL/NUM,P(100),Y(100)
*DELETE        INUPD.6,INUPD.6
      IF(NUM+N .LE. 100) GO TO 5
*DELETE        INUPD.9,INUPD.9
```

```
  700 FORMAT(*ONUMBER OF INTEGRATION VARIABLES EXCEEDS MAX LIMIT 100*)
*DELETE        LNUPD.3,LNUPD.3
      COMMON/UPDCAL/NUM,P(100),Y(100)
*DELETE        INPUZ.3,INPUZ.3
      COMMON/UPDCAL/NUM,P(100),Y(100)
*DELETE        INPUZ.5,INPUZ.5
    9 DO 10 I=1,100
*DELETE        INTEG.3,INTEG.3
      COMMON/UPDCAL/NUM,P(100),Y(100)
*DELETE        UPDAT.4,UPDAT.4
      COMMON/UPDCAL/NUM,P(100),Y(100)
*DELETE        MIMIN.7,MIMIN.7
      COMMON/UPDCAL/N,P(100),Y(100)
*DELETE        MIMIN.9,MIMIN.10
      DIMENSION YMAX(100),YO(100),PO(100),S(100),YP(100),Y1(100),
     1 Z(100),XK(100,3)
*DELETE        LGDET.3,LGDET.3
      COMMON/UPDCAL/NUM,P(100),Y(100)
*IDENT    MOD1040
*INSERT         AUTS.406
      DELQI = DELQDE
*IDENT    MODMIM
*DELETE        MIMIN.173,MIMIN.173
      WRITE(6,701) HT,H
  701 FORMAT(* INTEG RTN.       HT = *,E15.8,* H = *,E15.8)
*IDENT    MOD1048
*DELETE       ALGEAR.326,ALGEAR.326
  456 DELTX(I) = S(1,I)*12.0 - XSCOM(I)
*DELETE        ACTNG.312,ACTNG.312
  456 DELTX(I) = S(1,I)*12.0 - XSCOM(I)
*IDENT    MOD1103
*INSERT       LGEAR1.156
C  MODIFICATION TO ACCOMODATE SECONDARY PISTON OF F4 MAIN GEAR
      IF(S(1,I).LE.0.) GO TO 59
      IF(S(1,I).GE.SB(I)-(S2T(I)-S2(1,I))) 83,85
C  SECONDARY PISTON IN CONTACT WITH ORIFICE TUBE
   83 IF(SD1(1,I)+1.E-4.GE.S2D1(1,I)) 84,85
   84 S2D1(1,I) = SD1(1,I)
      S2D2(1,I) = SD2(1,I)
      GO TO 60
   85 CONTINUE
*IDENT    AERAT
*INSERT         REST.37
      COMMON/AEROCO/DM41(8)
      COMMON/XAERO/DM42(2)
*DELETE      MOD1029.4,MOD1029.4
      READ(7) (DMD1(IJ),IJ=1,11214)
*DELETE      MOD1029.5,MOD1029.5
      WRITE(7) (DMD1(IJ),IJ=1,11214)
```

```
*INSERT        OPT1.204
      CALL AERO4
*INSERT        OPT1.545
      SUBROUTINE AERO4
      COMMON/DIRCOM/DM1(2),X
      COMMON/TABCOM/LOCS(115),ST(115)
      COMMON/TABDIR/TABLE(800)
      COMMON/AEROCO/RTAB10(2),RTAB80(2),LTAB10(2),LTAB80(2)
      COMMON/XAERO/NIN,TIMEL
      REAL LTAB10,LTAB80
      DATA TIMEL,NIN/0.,0/
      DATA RTAB10,RTAB80,LTAB10,LTAB80/8*0./
      IF(NIN.EQ.0) TIMEL = X
      NIN = 1
      DELT = X - TIMEL
      TIMEL = X
      IND1 = LOCS(45)
      IND2 = LOCS(114)
      ATAB11 = TABLE(IND1)
      ATAB12 = TABLE(IND1+1)
      ATAB81 = TABLE(IND2)
      ATAB82 = TABLE(IND2+1)
      ATAB11 = ATAB11 + RTAB10(1)*DELT
      ATAB12 = ATAB12 + RTAB10(2)*DELT
      ATAB81 = ATAB81 + RTAB80(1)*DELT
      ATAB82 = ATAB82 + RTAB80(2)*DELT
      IF(RTAB10(1).GT.0..AND.ATAB11.GE.LTAB10(1)) GO TO 40
      IF(RTAB10(1).LT.0..AND.ATAB11.LE.LTAB10(1)) GO TO 40
   10 IF(RTAB10(2).GT.0..AND.ATAB12.GE.LTAB10(2)) GO TO 50
      IF(RTAB10(2).LT.0..AND.ATAB12.LE.LTAB10(2)) GO TO 50
   20 IF(RTAB80(1).GT.0..AND.ATAB81.GE.LTAB80(1)) GO TO 60
      IF(RTAB80(1).LT.0..AND.ATAB81.LE.LTAB80(1)) GO TO 60
   30 IF(RTAB80(2).GT.0..AND.ATAB82.GE.LTAB80(2)) GO TO 70
      IF(RTAB80(2).LT.0..AND.ATAB82.LE.LTAB80(2)) GO TO 70
      GO TO 80
   40 ATAB11 = LTAB10(1)
      RTAB10(1) = 0.
      GO TO 10
   50 ATAB12 = LTAB10(2)
      RTAB10(2) = 0.
      GO TO 20
   60 ATAB81 = LTAB80(1)
      RTAB80(1) = 0.
      GO TO 30
   70 ATAB82 = LTAB80(2)
      RTAB80(2) = 0.
   80 TABLE(IND1) = ATAB11
      TABLE(IND1+1) = ATAB12
      TABLE(IND2) = ATAB81
```

```
      TABLE(IND2+1) = ATAB82
      RETURN
      END
*INSERT       JMMODS.11
      IF(SYM.EQ.6HRTAB10) GO TO 905
      IF(SYM.EQ.6HRTAB80) GO TO 905
      IF(SYM.EQ.6HLTAB10) GO TO 905
      IF(SYM.EQ.6HLTAB80) GO TO 905
*INSERT       JMMODS.16
  905 CALL AEROIN(SYM,RA)
      GO TO 100
*INSERT         READ.194
      SUBROUTINE AEROIN(SYM,RA)
      DIMENSION RA(55)
      COMMON/AEROCO/DATAX(8)
      DATA DATAX/8*0./
      CALL LINES(1)
      WRITE(6,1) SYM,RA
    1 FORMAT(18X,A6,5X,55A1)
    2 FORMAT(*0ERROR.ILLEGAL CHARACTER IN NUMERIC FIELD*,1R1,2H**/)
      IF(SYM.EQ.6HRTAB10) INDEX = 1
      IF(SYM.EQ.6HRTAB80) INDEX = 3
      IF(SYM.EQ.6HLTAB10) INDEX = 5
      IF(SYM.EQ.6HLTAB80) INDEX = 7
      NUMEXP = 0
      NEXP = 0
      IEXP = 0
      NL = 0
      NR = 0
      NUML = 0
      NUMR = 0
      ISIGN = 0
      JSIGN = 0
      LEFT = 1
      DO 210 I=1,56
      IF(I.EQ.56) GO TO 140
      IF(RA(I).EQ.1H ) GO TO 210
      IF(RA(I).EQ.1H,) GO TO 140
      IF(RA(I).EQ.1H.) GO TO 170
      IF(RA(I).EQ.1HE) GO TO 180
      IF(RA(I).EQ.1H-) GO TO 200
      NUM = SHIFT(RA(I),6)
      NUM = NUM.AND.00000000000000000077B
      IF(NUM.GT.36) GO TO 130
      IF(NUM.LT.27) GO TO 130
      NUM = NUM - 27
      IF(IEXP.EQ.1) GO TO 190
      IF(LEFT.GT.0) NUML = 10*NUML + NUM
      IF(LEFT.GT.0) NL = NL + 1
```

```
      IF(LEFT.LT.0) NUMR = 10*NUMR + NUM
      IF(LEFT.LT.0) NR = NR + 1
      GO TO 210
  130 CALL LINES(3)
      WRITE(6,2) RA(I)
      GO TO 210
  140 IF(NL.EQ.0.AND.NR.EQ.0) GO TO 210
      IF(NR.EQ.0) GO TO 160
      X = FLOAT(NUML) + FLOAT(NUMR)/(10.)**NR
      IF(JSIGN.EQ.1) NUMEXP = -NUMEXP
      IF(IEXP.EQ.1) X = X*(10.)**NUMEXP
      IF(ISIGN.EQ.1) X = -X
  150 DATAX(INDEX) = X
      NUML = 0
      NUMR = 0
      NL = 0
      NR = 0
      LEFT = 1
      ISIGN = 0
      JSIGN = 0
      IEXP = 0
      NEXP = 0
      NUMEXP = 0
      INDEX = INDEX + 1
      GO TO 210
  160 X = NUML
      IF(JSIGN.EQ.1) NUMEXP = -NUMEXP
      IF(IEXP.EQ.1) X = X*(10.)**NUMEXP
      IF(ISIGN.EQ.1) X = -X
      GO TO 150
  170 LEFT = -1
      GO TO 210
  180 IEXP = 1
      GO TO 210
  190 NUMEXP = 10*NUMEXP + NUM
      NEXP = NEXP + 1
      GO TO 210
  200 IF(IEXP.EQ.0) ISIGN = 1
      IF(IEXP.NE.0) JSIGN = 1
  210 CONTINUE
      RETURN
      END
*IDENT    MOD2056
*INSERT         REST.42
      DATA DM41/8*0./
*DELETE      CSCMOD2.17,CSCMOD2.17
      EQUIVALENCE (DM5(16),GREFF),(DM1(37),AIYYBS)
*DELETE        AERAT.64,AERAT.64
*DELETE       PINARY.26,PINARY.26
```

```
      COMMON/ACTDIR/XNAME(77),LOC(77)
*DELETE       PINARY.36,PINARY.36
      DO 110 I=1,77
*DELETE       PINARY.37,PINARY.37
      COMMON/ACTDIR/NAME(77),LOC(77)
*DELETE       PINARY.39,PINARY.39
     6 6HSTRON ,  6HSTROM ,  6HREDSLP,  6HDSTOP /
*DELETE       PINARY.40,PINARY.40
     7     644, 645, 651, 683, 713, 743, 773, 392, 140/
*IDENT    MOD2075
*DELETE       MOD282.5,ALGEAR.219
  159 SIGX = 0.
      IF(SD1(1,I).EQ.0.) GO TO 471
      SIGX = SD1(1,I)/ABS(SD1(1,I))
*DELETE       ALGEAR.223,ALGEAR.223
     1 + CFFOR(I))*SIGX*DMTANH(I) + FSTOP(I))
*DELETE       MOD282.6,ACTNG.204
  159 SIGX = 0.
      IF(SD1(1,I).EQ.0.) GO TO 471
      SIGX = SD1(1,I)/ABS(SD1(1,I))
*DELETE        ACTNG.208,ACTNG.208
     1 + CFFOR(I))*SIGX*DMTANH(I) + FSTOP(I))
*IDENT    MOD2203
*DELETE       MOD282.1,MOD282.1
*DELETE       GENFIX.1,GENFIX.1
      COMMON/XALGEA/AIC(5),DM34(14),IPSTOP(5),DM34X(12),DM34Y(5),
     1 DM34Z(5),DM34A(65)
      COMMON/XACTNG/AICX(5),IPSTOX(5),DM35(3),DM35X(5),DM35Y(5),
     1 DM35Z(65)
*INSERT         REST.41
      DATA DM34Y,DM34Z,DM34A/5*0.,5*31000.,65*0./
*INSERT         REST.42
      DATA DM35X,DM35Y,DM35Z/5*0.,5*31000.,65*0./
*DELETE        AERAT.3,AERAT.3
      READ(7) (DMD1(IJ),IJ=1,11364)
*DELETE        AERAT.4,AERAT.4
      WRITE(7) (DMD1(IJ),IJ=1,11364)
*INSERT       MOD282.4
     3 ,IGE,RDSLP,ITRIP,ICU,IQCU,IXS,IA1,IA2,IA3,IA4,
     4 IA6,IA7,IA9,IA10,IGO
*DELETE      CSCMOD5.47,CSCMOD5.48
      DIMENSION IPSTOP(5),AIC(5),PGA1T(5),IGE(5),RDSLP(5),ITRIP(5),
     1           ICU(5),IQCU(5),IXS(5),IA1(5),IA2(5),IA3(5),IA4(5),
     2           IA6(5),IA7(5),IA9(5),IA10(5),IGO(5)
*DELETE      CSCMOD2.78,CSCMOD2.78
      IF(SD1(1,I).LT.0.0.AND.PGA1T1(I).LE.-1600.0)PGA1T(I)=-1600.0
      IF(ITRIP(I).EQ.1)GO TO 101
*DELETE       ALGEAR.172,ALGEAR.172
      IF(ABS(FT(I)).LE.FORCHT(I).AND.S(1,I).LE.ES(I))GO TO 150
```

```
*DELETE       ALGEAR.175,ALGEAR.175
      FORSST(I)=-FT(I)
*DELETE      CSCMOD2.94,CSCMOD2.94
      IF(IMODE(I).EQ.0.AND.S(1,I).LT.ES(I))GO TO 421
*DELETE       ALGEAR.178,ALGEAR.178
*DELETE      CSCMOD2.95,CSCMOD2.95
*DELETE      CSCMOD4.24,CSCMOD4.24
 801  IF(S(1,I).GT.ES(I))ISTROK(I)=1
      IF(SD1(1,I).LE.0.8.AND.IFR(I).EQ.0)GO TO 2
*INSERT       ALGEAR.184
      IFR(I)=0
*DELETE      CSCMOD2.105,CSCMOD2.105
 158  IF(ABS(FT(I)).LE.FORCHT(I).AND.S(1,I).LE.ES(I))GO TO 500
*DELETE      CSCMOD2.107,CSCMOD2.107
      IF(IMODE(I).EQ.0)GO TO 421
*DELETE       ALGEAR.226,ALGEAR.226
      IF(S(1,I).GT.ES(I))287,289
 287  ISTROK(I)=1
      ICU(I)=0
      IQCU(I)=0
      IXS(I)=0
C     THE FOLLOWING LOGIC RETURNS THE GEARS,DURING REBOUND, TO INITIAL
C     CONDITIONS IN THE EVENT THE GEAR CONTACTS THE SURFACE BEFORE
C     THE LOGIC BETWEEN STATEMENTS 226 AND 421 IS FULLY EFFECTIVE
 289  IF(ISTROK(I).EQ.1.AND.DDELTA(I).LT.0.0)IGO(I)=1
      IF(IGE(I).EQ.1.OR.ITRIP(I).EQ.1)227,297
 227  IF(IGO(I).EQ.0)GO TO 226
      IF(DELTA(I).GT.0.0.AND.WLFOR(I).EQ.0.0)220,226
 220  IF(VCUM(I).GT.0.00001.OR.VCUM(I).LT.-0.00001)221,222
 221  QO(I)=-VCUM(I)/DSTOP
      GO TO 226
 222  PGA2T(I)=PGA1I(I)
      QO(I)=0.0
      IGO(I)=0
 226  IF(IGE(I).EQ.1)290,297
*DELETE       ALGEAR.229,ALGEAR.230
 290  IF(ITRIP(I).EQ.1.AND.IOPCO(I).EQ.1)GO TO 297
      IOPCO(I)=0
      IF(ABS(FORSST(I)).LE.FT(I))FORSST(I)=-FT(I)
      XSVDDD(I)=0.0
      XSVDD(I)=0.0
      IF(DELTX1(I).GT.0.00001.OR.DELTX1(I).LT.-0.00001)GO TO 210
      DLTX1D(I)=0.0
      GO TO 202
 210  DLTX1D(I)=-DELTX1(I)/TAUF
 202  XMA(I)=0.0
      XMA1(I)=0.0
      XMA2(I)=0.0
      XMA5(I)=0.0
```

```
      XMA3(I)=0.0
      XMA4(I)=0.0
      XMA8(I)=0.0
      XMA11(I)=0.0
      XMA1DT(I)=0.0
      XMA2DT(I)=0.0
      XMA3DT(I)=0.0
      XMA4DT(I)=0.0
      XMA6DT(I)=0.0
      XMA7DT(I)=0.0
      XMA9DT(I)=0.0
      XMA10D(I)=0.0
*DELETE       ALGEAR.231,ALGEAR.232
      IF((PGA1I(I)-2000.0).LT.PGA1T(I).AND.
     1  PGA1T(I).LT.(PGA1I(I)+2000.0))299,298
*DELETE       XSVMOD.4,XSVMOD.4
      IF(XVALVE(I).EQ.XBIAS(I).AND.ICOSV(I).EQ.1)400,312
*DELETE       ALGEAR.234,ALGEAR.273
 312  IF(ICOSV(I).NE.1)313,314
 314  IF(XVALVE(I).NE.XBIAS(I))GO TO 311
 298  IF(ICOSV(I).EQ.1)GO TO 311
 313  IF(PGA1T(I).GT.PGA1I(I)+2000.)292,293
 292  IF(IXSVL(I).EQ.1)GO TO 311
      XSVDOT(I)=XSVDMN(I)*PERCNT(I)
      IPASS(I)=1
      XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
      IF(XVALVE(I).LE.-0.13)300,303
 303  XSV(I)=XVALVE(I)
      GO TO 297
 300  XVALVE(I)=-0.13
      XSVDOT(I)=0.0
      IXSVL(I)=1
      GO TO 294
 293  IF(PGA1T(I).LT.PGA1I(I)-2000.)295,294
 295  IF(IXSVH(I).EQ.1)GO TO 311
      XSVDOT(I)=XSVDMX(I)*PERCNT(I)
      IPASS(I)=2
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
      IF(XVALVE(I).GE.0.13)302,304
 304  XSV(I)=XVALVE(I)
      GO TO 297
 302  XVALVE(I)=0.13
      IXSVH(I)=1
 294  CONTINUE
      XSVDOT(I)=0.0
      ICOSV(I)=1
 311  IF(PGA2T(I).GT.PGA1I(I)+4000.0)316,315
 316  IF(QSV(I).LT.QD(I).AND.ICOSV(I).EQ.1)317,318
 317  XSVDOT(I)=0.0
```

```
      XSV(I)=XVALVE(I)
      GO TO 297
 318  XSVDOT(I)=XSVDMN(I)*PERCNT(I)
      XSV(I)=XVALVE(I)
      GO TO 297
 315  CONTINUE
      IF(NAC(I).EQ.1)GO TO 307
 320  IF(IIXSVH(I).EQ.1)GO TO 305
      XSVDOT(I)=XSVDMN(I)
      IPASS(I)=3
      XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT
      IF(XVALVE(I).LE.XBIAS(I))305,306
 306  XSV(I)=XVALVE(I)
      GO TO 297
 305  XVALVE(I)=XBIAS(I)
      XSVDOT(I)=0.0
      IIXSVH(I)=1
      GO TO 400
 307  IF(IIXSVL(I).EQ.1)GO TO 308
      XSVDOT(I)=XSVDMX(I)
      IPASS(I)=4
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT
      IF(XVALVE(I).GE.XBIAS(I))308,309
 309  XSV(I)=XVALVE(I)
      GO TO 297
 308  XVALVE(I)=XBIAS(I)
      XSVDOT(I)=0.0
      IIXSVL(I)=1
*DELETE     CSCMOD5.71,ALGEAR.278
*INSERT      ALGEAR.279
      DELTX1(I)=0.0
*INSERT      ALGEAR.283
      ICOSV(I)=0
      QQ(I)=0.0
      QSVCU(I)=0.0
      QSV1(I)=0.0
      QSV3(I)=0.0
      VCUM(I)=0.0
      VOL1T(I)=VOL1I(I)
      VOL2T(I)=VOL2I(I)
      VOL3T(I)=VOL3I(I)
      ITRIP(I)=1
      PGA1T1(I)=PGA1I(I)
      PGA1T(I)=PGA1I(I)
      PGA2T(I)=PGA1I(I)
      PGA3T(I)=PGA1I(I)
*DELETE      ALGEAR.290,CSCMOD2.108
 297  IF(ITRIP(I).EQ.1)430,421
 430  IF(VCUM(I).GT.0.00001.OR.VCUM(I).LT.-0.00001)GO TO 431
```

```
      QO(I)=0.0
      ICU(I)=1
      GO TO 440
  431 IF(VCUM(I).LT.-0.00001)GO TO 432
      QO(I)=-VCUM(I)/DSTOP
      GO TO 440
  432 QO(I)=-VCUM(I)/DSTOP
  440 IF(QSVCU(I).GT.0.00001.OR.QSVCU(I).LT.-0.00001)GO TO 433
      QSV(I)=0.0
      IQCU(I)=1
      GO TO 420
  433 IF(QSVCU(I).LT.-0.00001)GO TO 434
      QSV(I)=-QSVCU(I)/DSTOP
      GO TO 420
  434 QSV(I)=-QSVCU(I)/DSTOP
  420 IF(XSV(I).GT.XBIAS(I)+.000001)422,423
  422 IF(XSV(I).GT. 0.0)XSVDOT(I)=-XSV(I)/DSTOP
      IF(XSV(I).LT. 0.0)XSVDOT(I)=XSV(I)/DSTOP
      GO TO 600
  423 IF(XSV(I).LT.XBIAS(I)-.000001)424,425
  424 XSVDOT(I)=-XSV(I)/DSTOP
      GO TO 600
  425 XSVDOT(I)=0.0
      IXS(I)=1
  600 IF(XMA1(I).GT.0.00001 .OR.XMA1(I).LT.-0.00001)GO TO 601
      XMA1DT(I)=0.0
      IA1(I)=1
      GO TO 602
  601 XMA1DT(I)=-XMA1(I)/DSTOP
  602 IF(XMA2(I).GT.0.00001 .OR.XMA2(I).LT.-0.00001)GO TO 603
      XMA2DT(I)=0.0
      IA2(I)=1
      GO TO 604
  603 XMA2DT(I)=-XMA2(I)/DSTOP
  604 IF(XMA3(I).GT.0.00001 .OR.XMA3(I).LT.-0.00001)GO TO 605
      XMA3DT(I)=0.0
      IA3(I)=1
      GO TO 606
  605 XMA3DT(I)=-XMA3(I)/DSTOP
  606 IF(XMA4(I).GT.0.00001 .OR.XMA4(I).LT.-0.00001)GO TO 607
      XMA4DT(I)=0.0
      IA4(I)=1
      GO TO 608
  607 XMA4DT(I)=-XMA4(I)/DSTOP
  608 IF(XMA6(I).GT.0.00001 .OR.XMA6(I).LT.-0.00001)GO TO 609
      XMA6DT(I)=0.0
      IA6(I)=1
      GO TO 610
  609 XMA6DT(I)=-XMA6(I)/DSTOP
```

```
 610  IF(XMA7(I).GT.0.00001 .OR.XMA7(I).LT.-0.00001)GO TO 611
      XMA7DT(I)=0.0
      IA7(I)=1
      GO TO 612
 611  XMA7DT(I)=-XMA7(I)/DSTOP
 612  IF(XMA9(I).GT.0.00001 .OR.XMA9(I).LT.-0.00001)GO TO 613
      XMA9DT(I)=0.0
      IA9(I)=1
      GO TO 614
 613  XMA9DT(I)=-XMA9(I)/DSTOP
 614  IF(XMA10(I).GT.0.00001 .OR.XMA10(I).LT.-0.00001)GO TO 615
      XMA10D(I)=0.0
      IA10(I)=1
      GO TO 421
 615  XMA10D(I)=-XMA10(I)/DSTOP
C     THESE SWITCHES ARE EITHER ZERO OR ONE
 421  IF(ICU(I)+IQCU(I)+IXS(I)+IA1(I)+IA2(I)+IA3(I)+IA4(I)+IA6(I)
     1   +IA7(I)+IA9(I)+IA10(I) .EQ. 11)ISTROK(I)=0
      I2=2*I+NSTRUT-1
*DELETE      ALGEAR.301,ALGEAR.301
C     ZSSC IS A PERCENTAGE OF S8(I) FOR ACTIVATING CONTROL
*DELETE     CSCMOD5.77,CSCMOD5.77
     1     (VMASS(I)*RDSLP(I))
*DELETE      ALGEAR.313,ALGEAR.313
      IF(IGE(I).EQ.0)GO TO 131
      WLFOR(I)=WLFORR
      INDEACT(I)=2
      EPSILO(I)=EPSROL(I)
      GO TO 451
 131  IF(S(1,I).LE.ES(I))GO TO 451
      IF(HMM(I).EQ.0)GO TO 451
*INSERT      ALGEAR.330
      IF(S(1,I).LE.ES(I).AND.ITRIP(I).EQ.1)GO TO 458
*DELETE      ALGEAR.338,ALGEAR.338
 458  P1(I)=PGA1T(I)/144.
 459  VOLANT(I)=VOLANT(I)+QSVN(I)*DELT-QPUMPS(I)*DELT
*DELETE      ALGEAR.354,ALGEAR.354
      IF(S(1,I).LE.ES(I).AND.ITRIP(I).EQ.1)462,410
 410  CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      CALL FLOZE2(PS(I),PR(I),P1(I),XLPSV1(I),XLPSV3(I),RCLSV(I),DSV(I),
*INSERT      ALGEAR.397
      IGE(I)=0
      ITRIP(I)=0
*INSERT      ALGEAR.412
      IF(IMODE(I).EQ.0)GO TO 55
      IF(ISTROK(I).EQ.1)IGE(I)=1
*INSERT      ALGEAR.439
      IF(IGE(I).EQ.1.AND.ICOSV(I).EQ.1)GO TO 63
      IF(DELTA(I).GT.0..AND.IGO(I).EQ.1)GO TO 63
```

```
*DELETE       MOD329.4,MOD329.5
      IF(S(1,I).NE.0.0.OR.AIC(I).NE.1.)64,63
 63   DP1(I)=0.0
*INSERT       MOD329.9
      IF(S(1,I).LE.ES(I))GO TO 29
*DELETE       ALGEAR.526,ALGEAR.526
      IF(IGE(I).EQ.1)GO TO 29
*DELETE       ALGEAR.535,ALGEAR.538
*DELETE       MOD329.10,MOD329.10
  29  I9=I+9*NSTRUT
*DELETE      CSCMOD2.140,CSCMOD2.140
 27   I6=3*I+6*NSTRUT-2
*INSERT      CSCMOD2.145
      CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      CALL LIMITS(XSVDOT(I),XSVDD(I),XSVDMX(I),XSVDMN(I))
      CALL LIMITS(XSVDD(I),XSVDDD(I),XDDMAX(I),XDDMIN(I))
*INSERT       GENFIX.8
     2 ,IGE,RDSLP,ITRIP,ICU,IQCU,IXS,IA1,IA2,IA3,IA4,
     3 IA6,IA7,IA9,IA10,IGO
*DELETE        ACTNG.58,ACTNG.58
      DIMENSION IPSTOP(5),AIC(5),PGA1T(5),IGE(5),RDSLP(5),ITRIP(5),
     1          ICU(5),IQCU(5),IXS(5),IA1(5),IA2(5),IA3(5),IA4(5),
     2          IA6(5),IA7(5),IA9(5),IA10(5),IGO(5)
*DELETE        ACTNG.103,ACTNG.103
      IF(SD1(1,I).LT.0.0.AND.PGA1T1(I).LE.-1600.0)PGA1T(I)=-1600.0
      IF(ITRIP(I).EQ.1)GO TO 101
*DELETE        ACTNG.157,ACTNG.157
      IF(ABS(FT(I)).LE.FORCHT(I).AND.S(1,I).LE.ES(I))GO TO 150
*DELETE        ACTNG.160,ACTNG.160
      FORSST(I)=-FT(I)
*DELETE        ACTNG.162,ACTNG.162
      IF(IMODE(I).EQ.0.AND.S(1,I).LT.ES(I))GO TO 421
*DELETE        ACTNG.163,ACTNG.164
*DELETE        ACTNG.166,ACTNG.166
 801  IF(S(1,I).GT.ES(I))ISTROK(I)=1
      IF(SD1(1,I).LE.0.8.AND.IFR(I).EQ.0)GO TO 2
*INSERT        ACTNG.168
      IFR(I)=0
*DELETE        ACTNG.199,ACTNG.199
 158  IF(ABS(FT(I)).LE.FORCHT(I).AND.S(1,I).LE.ES(I))GO TO 500
*DELETE        ACTNG.210,ACTNG.210
      IF(IMODE(I).EQ.0)GO TO 421
*DELETE        ACTNG.211,ACTNG.211
      IF(S(1,I).GT.ES(I))287,289
 287  ISTROK(I)=1
      ICU(I)=0
      IQCU(I)=0
      IXS(I)=0
C     THE FOLLOWING LOGIC RETURNS THE GEARS,DURING REBOUND, TO INITIAL
```

```
C     CONDITIONS IN THE EVENT THE GEAR CONTACTS THE SURFACE BEFORE
C     THE LOGIC BETWEEN STATEMENTS 226 AND 421 IS FULLY EFFECTIVE
 289  IF(ISTROK(I).EQ.1.AND.DDELTA(I).LT.0.0)IGO(I)=1
      IF(IGE(I).EQ.1.OR.ITRIP(I).EQ.1)227,297
 227  IF(IGO(I).EQ.0)GO TO 226
      IF(DELTA(I).GT.0.0.AND.WLFOR(I).EQ.0.0)220,226
 220  IF(VCUM(I).GT.0.00001.OR.VCUM(I).LT.-0.00001)221,222
 221  QO(I)=-VCUM(I)/DSTOP
      GO TO 226
 222  PGA2T(I)=PGA1I(I)
      QO(I)=0.0
      IGO(I)=0
 226  IF(IGE(I).EQ.1)290,297
*DELETE       ACTNG.212,ACTNG.213
 290  IF(ITRIP(I).EQ.1.AND.IOPCO(I).EQ.1)GO TO 297
      IOPCO(I)=0
      IF(ABS(FORSST(I)).LE.FT(I))FORSST(I)=-FT(I)
      XSVDDD(I)=0.0
      XSVDD(I)=0.0
      IF(DELTX1(I).GT.0.00001.OR.DELTX1(I).LT.-0.00001)GO TO 210
      DLTX1D(I)=0.0
      GO TO 202
 210  DLTX1D(I)=-DELTX1(I)/TAUF
 202  XMA(I)=0.0
      XMA1(I)=0.0
      XMA2(I)=0.0
      XMA5(I)=0.0
      XMA3(I)=0.0
      XMA4(I)=0.0
      XMA8(I)=0.0
      XMA11(I)=0.0
      XMA1DT(I)=0.0
      XMA2DT(I)=0.0
      XMA3DT(I)=0.0
      XMA4DT(I)=0.0
      XMA6DT(I)=0.0
      XMA7DT(I)=0.0
      XMA9DT(I)=0.0
      XMA10D(I)=0.0
*DELETE       ACTNG.214,ACTNG.215
      IF((PGA1I(I)-2000.0).LT.PGA1T(I).AND.
     1  PGA1T(I).LT.(PGA1I(I)+2000.0))299,298
*DELETE      XSVMOD.9,XSVMOD.9
      IF(XVALVE(I).EQ.XBIAS(I).AND.ICOSV(I).EQ.1)400,312
*DELETE       ACTNG.217,ACTNG.253
 312  IF(ICOSV(I).NE.1)313,314
 314  IF(XVALVE(I).NE.XBIAS(I))GO TO 311
 298  IF(ICOSV(I).EQ.1)GO TO 311
 313  IF(PGA1T(I).GT.PGA1I(I)+2000.)292,293
```

```
292   IF(IXSVL(I).EQ.1)GO TO 311
      XSVDOT(I)=XSVDMN(I)*PERCNT(I)
      IPASS(I)=1
      XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT*PERCNT(I)
      IF(XVALVE(I).LE.-0.13)300,303
303   XSV(I)=XVALVE(I)
      GO TO 297
300   XVALVE(I)=-0.13
      XSVDOT(I)=0.0
      IXSVL(I)=1
      GO TO 294
293   IF(PGA1T(I).LT.PGA1I(I)-2000.)295,294
295   IF(IXSVH(I).EQ.1)GO TO 311
      XSVDOT(I)=XSVDMX(I)*PERCNT(I)
      IPASS(I)=2
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PERCNT(I)
      IF(XVALVE(I).GE.0.13)302,304
304   XSV(I)=XVALVE(I)
      GO TO 297
302   XVALVE(I)=0.13
      IXSVH(I)=1
294   CONTINUE
      XSVDOT(I)=0.0
      ICOSV(I)=1
311   IF(PGA2T(I).GT.PGA1I(I)+4000.0)316,315
316   IF(QSV(I).LT.QO(I).AND.ICOSV(I).EQ.1)317,318
317   XSVDOT(I)=0.0
      XSV(I)=XVALVE(I)
      GO TO 297
318   XSVDOT(I)=XSVDMN(I)*PERCNT(I)
      XSV(I)=XVALVE(I)
      GO TO 297
315   CONTINUE
      IF(NAC(I).EQ.1)GO TO 307
320   IF(IIXSVH(I).EQ.1)GO TO 305
      XSVDOT(I)=XSVDMN(I)
      IPASS(I)=3
      XVALVE(I)=XVALVE(I)+XSVDMN(I)*DELT
      IF(XVALVE(I).LE.XBIAS(I))305,306
306   XSV(I)=XVALVE(I)
      GO TO 297
305   XVALVE(I)=XBIAS(I)
      XSVDOT(I)=0.0
      IIXSVH(I)=1
      GO TO 400
307   IF(IIXSVL(I).EQ.1)GO TO 308
      XSVDOT(I)=XSVDMX(I)
      IPASS(I)=4
      XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT
```

```
      IF(XVALVE(I).GE.XBIAS(I))308,309
 309  XSV(I)=XVALVE(I)
      GO TO 297
 308  XVALVE(I)=XBIAS(I)
      XSVDOT(I)=0.0
      IIXSVL(I)=1
*DELETE     CSCMOD5.87,ACTNG.256
*INSERT       ACTNG.257
      DELTX1(I)=0.0
*INSERT       ACTNG.261
      ICOSV(I)=0
      QD(I)=0.0
      QSVCU(I)=0.0
      QSV1(I)=0.0
      QSV3(I)=0.0
      VCUM(I)=0.0
      VOL1T(I)=VOL1I(I)
      VOL2T(I)=VOL2I(I)
      VOL3T(I)=VOL3I(I)
      ITRIP(I)=1
      PGA1T1(I)=PGA1I(I)
      PGA1T(I)=PGA1I(I)
      PGA2T(I)=PGA1I(I)
      PGA3T(I)=PGA1I(I)
*DELETE       ACTNG.268,ACTNG.272
 297  IF(ITRIP(I).EQ.1)430,421
 430  IF(VCUM(I).GT.0.00001.OR.VCUM(I).LT.-0.00001)GO TO 431
      QD(I)=0.0
      ICU(I)=1
      GO TO 440
 431  IF(VCUM(I).LT.-0.00001)GO TO 432
      QD(I)=-VCUM(I)/DSTOP
      GO TO 440
 432  QD(I)=-VCUM(I)/DSTOP
 440  IF(QSVCU(I).GT.0.00001.OR.QSVCU(I).LT.-0.00001)GO TO 433
      QSV(I)=0.0
      IQCU(I)=1
      GO TO 420
 433  IF(QSVCU(I).LT.-0.00001)GO TO 434
      QSV(I)=-QSVCU(I)/DSTOP
      GO TO 420
 434  QSV(I)=-QSVCU(I)/DSTOP
 420  IF(XSV(I).GT.XBIAS(I)+.000001)422,423
 422  IF(XSV(I).GT. 0.0)XSVDOT(I)=-XSV(I)/DSTOP
      IF(XSV(I).LT. 0.0)XSVDOT(I)=XSV(I)/DSTOP
      GO TO 600
 423  IF(XSV(I).LT.XBIAS(I)-.000001)424,425
 424  XSVDOT(I)=-XSV(I)/DSTOP
      GO TO 600
```

```
 425  XSVDOT(I)=0.0
      IXS(I)=1
 600  IF(XMA1(I).GT.0.00001 .OR.XMA1(I).LT.-0.00001)GO TO 601
      XMA1DT(I)=0.0
      IA1(I)=1
      GO TO 602
 601  XMA1DT(I)=-XMA1(I)/DSTOP
 602  IF(XMA2(I).GT.0.00001 .OR.XMA2(I).LT.-0.00001)GO TO 603
      XMA2DT(I)=0.0
      IA2(I)=1
      GO TO 604
 603  XMA2DT(I)=-XMA2(I)/DSTOP
 604  IF(XMA3(I).GT.0.00001 .OR.XMA3(I).LT.-0.00001)GO TO 605
      XMA3DT(I)=0.0
      IA3(I)=1
      GO TO 606
 605  XMA3DT(I)=-XMA3(I)/DSTOP
 606  IF(XMA4(I).GT.0.00001 .OR.XMA4(I).LT.-0.00001)GO TO 607
      XMA4DT(I)=0.0
      IA4(I)=1
      GO TO 608
 607  XMA4DT(I)=-XMA4(I)/DSTOP
 608  IF(XMA6(I).GT.0.00001 .OR.XMA6(I).LT.-0.00001)GO TO 609
      XMA6DT(I)=0.0
      IA6(I)=1
      GO TO 610
 609  XMA6DT(I)=-XMA6(I)/DSTOP
 610  IF(XMA7(I).GT.0.00001 .OR.XMA7(I).LT.-0.00001)GO TO 611
      XMA7DT(I)=0.0
      IA7(I)=1
      GO TO 612
 611  XMA7DT(I)=-XMA7(I)/DSTOP
 612  IF(XMA9(I).GT.0.00001 .OR.XMA9(I).LT.-0.00001)GO TO 613
      XMA9DT(I)=0.0
      IA9(I)=1
      GO TO 614
 613  XMA9DT(I)=-XMA9(I)/DSTOP
 614  IF(XMA10(I).GT.0.00001 .OR.XMA10(I).LT.-0.00001)GO TO 615
      XMA10D(I)=0.0
      IA10(I)=1
      GO TO 421
 615  XMA10D(I)=-XMA10(I)/DSTOP
C     THESE SWITCHES ARE EITHER ZERO OR ONE
 421  IF(ICU(I)+IQCU(I)+IXS(I)+IA1(I)+IA2(I)+IA3(I)+IA4(I)+IA6(I
     1   +IA7(I)+IA9(I)+IA10(I) .EQ. 11)ISTROK(I)=0
      I2=2*I+NSTRUT-1
*DELETE        ACTNG.285,ACTNG.285
C     ZSSC IS A PERCENTAGE OF SB(I) FOR ACTIVATING CONTROL
*DELETE        ACTNG.293,ACTNG.293
```

```
     1     (VMASS(I)*RDSLP(I))
*DELETE        ACTNG.299,ACTNG.299
      IF(IGE(I).EQ.0)GO TO 131
      WLFOR(I)=WLFORR
      INDEACT(I)=2
      EPSILO(I)=EPSROL(I)
      GO TO 451
 131  IF(S(1,I).LE.ES(I))GO TO 451
      IF(HMM(I).EQ.0)GO TO 451
*INSERT        ACTNG.316
      IF(S(1,I).LE.ES(I).AND.ITRIP(I).EQ.1)GO TO 458
*DELETE        ACTNG.324,ACTNG.324
 458  P1(I)=PGA1T(I)/144.
 459  VOLANT(I)=VOLANT(I)+QSVN(I)*DELT-QPUMPS(I)*DELT
*DELETE        ACTNG.340,ACTNG.340
      IF(S(1,I).LE.ES(I).AND.ITRIP(I).EQ.1)462,410
 410  CALL LIMITS(XSV(I),XSVDOT(I),XSVMAX(I),XSVMIN(I))
      CALL FLOZE2(PS(I),PR(I),P1(I),XLPSV1(I),XLPSV3(I),RCLSV(I),DSV(I),
*IDENT    MOD2235
*DELETE      MOD2203.41,MOD2203.41
      IF(DELTA(I).GT.0.) IGO(I) = 0
*DELETE      MOD2203.235,MOD2203.235
      IF(ITRIP(I).EQ.1) GO TO 63
*DELETE      MOD2203.277,MOD2203.277
      IF(DELTA(I).GT.0.) IGO(I) = 0
*IDENT    MOD2342
*DELETE      MOD2203.216,MOD2203.216
     1   (VMASS(I)*REDSLP(I))
*IDENT    MOD3028
*DELETE      MOD2203.452,MOD2203.452
     1 (VMASS(I)*REDSLP(I))
```

| 1. Report No. | 2. Government Accession No. | 3. Recipient's Catalog No. |
|---|---|---|
| NASA CR-166069 | | |

| 4. Title and Subtitle | 5. Report Date |
|---|---|
| IMPROVEMENTS TO THE FATOLA COMPUTER PROGRAM INCLUDING ADDED ACTIVELY CONTROLLED LANDING GEAR SUBROUTINES | April 1983 |
| | 6. Performing Organization Code |
| 7. Author(s) | 8. Performing Organization Report No. |
| G. H. Mall | TAO 33842 |
| | 10. Work Unit No. |
| 9. Performing Organization Name and Address | |
| Computer Sciences Corporation<br>3217 N. Armistead Avenue<br>Hampton, VA 23666 | 11. Contract or Grant No.<br>NAS1-16078 |
| | 13. Type of Report and Period Covered |
| 12. Sponsoring Agency Name and Address | Contractor Report |
| National Aeronautics and Space Administration<br>Washington, DC 20546 | 14. Sponsoring Agency Code |

15. Supplementary Notes

Langley technical monitors: John E. Hogge and John R. McGehee
Final Report

16. Abstract

Modifications to a multi-degree-of-freedom flexible aircraft take-off and landing analysis (FATOLA) computer program, including a provision for actively controlled landing gears to expand the programs simulation capabilities, are presented. Supplemental instructions for preparation of data and for use of the modified program are included.

| 17. Key Words (Suggested by Author(s)) | 18. Distribution Statement |
|---|---|
| Landing analysis<br>Flexible aircraft<br>Aircraft landing gear<br>Active controls | ~~FEDD Distribution~~<br>Subject Category 05 |

| 19. Security Classif. (of this report) | 20. Security Classif. (of this page) | 21. No. of Pages | 22. Price |
|---|---|---|---|
| Unclassified | Unclassified | 218 | |

Available: NASA's Industrial Applications Centers

End of Document