NASA Contractor Report 166069

(XASA-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,)

Gerald H. Mall
COMPUTER SCIENCES CORPORATION
Hampton, VA 23666

Contract NAS1-16078
April 1983

NASA
National Aeronautics and Space Administration
Langley Research Center
Hampton, Virginia 23665

Library Copy
APR 9.5 1983

LANGLEY RESEARCH CENTER
LIBRARY, NASA
HAMPTON, VIRGINIA
# Table of Contents

List of Figures .......................... 111
List of Tables ........................... v

Introduction .............................. 1

1. Added Program Capabilities ......... 3
   1.1 Active Control Landing Gear .... 3
   1.2 Implementation of the Active Control Code .......... 4
   1.3 Stroke Dependent Metering Pin Area' ................. 23
   1.4 Variable Aerodynamic Coefficients ................. 23
   1.5 Restart Capability .................. 24

2. Chronological History of Program Modifications ...... 26
   2.1 Structures and Dynamics Division Modifications ....... 26
   2.2 Computer Sciences Corporation Modifications .......... 28

3. User Information ........................ 113
   3.1 Data Preparation ................... 113
   3.2 Operating Instructions .............. 131

4. References ............................. 137

Appendix A - Program Modifications .................. A-1
List of Figures

Figure 1  FATOLA passive Landing Gear
          Subroutine Linkage                   5

2  Active Control Landing Gear
       Subroutine Linkage                6

3  Sample Input Deck                    115

4  Control Card Decks                    132
List of Tables

Table 1 New Input Variables  

2 New Program Variables
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 $C_A$ and $C_N$. 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:

\[
WFORT = f(F_{cg})
\]

\[
SD2 = f(F_{FORST})
\]
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**

<table>
<thead>
<tr>
<th>VARIABLE NAME</th>
<th>TYPE</th>
<th>UNITS</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMUH</td>
<td>LBF-SEC/FT$^2$</td>
<td>Dynamic viscosity of hydraulic fluid</td>
<td></td>
</tr>
<tr>
<td>APINT</td>
<td>ARRAY</td>
<td>FT$^2$</td>
<td>Area of metering pin as function of strut stroke</td>
</tr>
<tr>
<td>AREA1</td>
<td>ARRAY</td>
<td>FT$^2$</td>
<td>Area of strut hydraulic chamber (piston)</td>
</tr>
<tr>
<td>AREA2</td>
<td>ARRAY</td>
<td>FT$^2$</td>
<td>Area of strut pneumatic chamber (cylinder)</td>
</tr>
<tr>
<td>AREA3</td>
<td>ARRAY</td>
<td>FT$^2$</td>
<td>Area of chamber between piston and cylinder</td>
</tr>
<tr>
<td>AREMO</td>
<td>ARRAY</td>
<td>FT$^2$</td>
<td>Area of strut main orifice</td>
</tr>
<tr>
<td>AREO3</td>
<td>ARRAY</td>
<td>FT$^2$</td>
<td>Orifice area of chamber between piston and cylinder</td>
</tr>
<tr>
<td>BETA</td>
<td>LBF/IN$^2$</td>
<td>Bulk modulus of hydraulic fluid</td>
<td></td>
</tr>
<tr>
<td>BLMU</td>
<td>ARRAY</td>
<td>--</td>
<td>Coefficient of friction for strut lower bearing</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>BUMU</td>
<td>ARRAY</td>
<td>Coefficient of friction for strut upper bearing</td>
<td></td>
</tr>
<tr>
<td>CDMOC</td>
<td>ARRAY</td>
<td>Strut main orifice compression discharge coefficient</td>
<td></td>
</tr>
<tr>
<td>CDMOE</td>
<td>ARRAY</td>
<td>Strut main orifice extension discharge coefficient</td>
<td></td>
</tr>
<tr>
<td>CDSV</td>
<td>ARRAY</td>
<td>Servovalve orifice discharge coefficient</td>
<td></td>
</tr>
<tr>
<td>CD3</td>
<td>ARRAY</td>
<td>Discharge coefficient for chamber between piston and cylinder</td>
<td></td>
</tr>
<tr>
<td>CFFOR</td>
<td>ARRAY</td>
<td>Coulomb friction force between piston and cylinder bearings</td>
<td></td>
</tr>
<tr>
<td>DIOTA</td>
<td>DEGREES</td>
<td>Angle between body x-axis and wing main chord</td>
<td></td>
</tr>
<tr>
<td>DSV</td>
<td>ARRAY</td>
<td>Diameter of servovalve spool</td>
<td></td>
</tr>
<tr>
<td>EPSILO</td>
<td>ARRAY</td>
<td>Tolerance about impact control limit force</td>
<td></td>
</tr>
<tr>
<td>EPSROL</td>
<td>ARRAY</td>
<td>Tolerance about rollout control limit force</td>
<td></td>
</tr>
<tr>
<td>EPSSLP</td>
<td>LBF/SEC</td>
<td>Time rate of change of tolerance about transition control limit force</td>
<td></td>
</tr>
<tr>
<td>ETASV</td>
<td>1/SEC</td>
<td>Damping coefficient in servovalve transfer function</td>
<td></td>
</tr>
<tr>
<td>FWORK</td>
<td>ARRAY</td>
<td>Coefficient for selecting value of impact limit force</td>
<td></td>
</tr>
</tbody>
</table>
GAMAH ARRAY LBF/FT$^3$ Specific weight of hydraulic fluid

GNR -- Constant — set to 0.0

IMODE INTEGER ARRAY -- 0=Off 1=On Switch to select active control for each gear

IRST -- 0=None 1=Write 2=Read 3=Read/ Write Restart indicator

KAPT INTEGER ARRAY -- 0=No Pin 1=Constant Area 2=Variable Area Indicator for metering pin

LTAB10 REAL ARRAY -- $C_N$ limiting value 0

LTAB80 REAL ARRAY -- $C_A$ limiting value 0

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
<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PINM</td>
<td>ARRAY</td>
<td>FT²</td>
<td>Main gear metering pin area table</td>
</tr>
<tr>
<td>PINN</td>
<td>ARRAY</td>
<td>FT²</td>
<td>Nose gear metering pin area table</td>
</tr>
<tr>
<td>QPUMPS</td>
<td>ARRAY</td>
<td>FT³/SEC</td>
<td>Maximum hydraulic pump flow rate</td>
</tr>
<tr>
<td>RCLSV</td>
<td>ARRAY</td>
<td>INCHES</td>
<td>Radial clearance between spool and sleeve of servovalve</td>
</tr>
<tr>
<td>RHOH</td>
<td></td>
<td>SLUGS/FT³</td>
<td>Mass density of hydraulic fluid</td>
</tr>
<tr>
<td>RTAB10</td>
<td>ARRAY</td>
<td></td>
<td>C_N rate of change</td>
</tr>
<tr>
<td>RTAB80</td>
<td>ARRAY</td>
<td></td>
<td>C_A rate of change</td>
</tr>
<tr>
<td>STROM</td>
<td>ARRAY</td>
<td>FT</td>
<td>Main gear stroke table for pin area</td>
</tr>
<tr>
<td>ST10N</td>
<td>ARRAY</td>
<td>FT</td>
<td>Nose gear stroke table for pin area</td>
</tr>
<tr>
<td>TAUF</td>
<td></td>
<td>SEC</td>
<td>Time constant in strut position feedback loop</td>
</tr>
<tr>
<td>TC1</td>
<td></td>
<td>SEC</td>
<td>Time constant of electronic compensation network</td>
</tr>
<tr>
<td>TC2</td>
<td></td>
<td>SEC</td>
<td>Same definition as TC1</td>
</tr>
<tr>
<td>TC3</td>
<td></td>
<td>SEC</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>TC4</td>
<td></td>
<td>SEC</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>VOLACI</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Total volume of high pressure accumulator</td>
</tr>
<tr>
<td>VOLANI</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Initial volume of charging nitrogen of high pressure accumulator</td>
</tr>
<tr>
<td>Symbol</td>
<td>Type</td>
<td>Unit</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>VOL1I</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Initial volume of hydraulic fluid in shock strut piston</td>
</tr>
<tr>
<td>VOL2I</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Pneumatic volume of charged strut</td>
</tr>
<tr>
<td>VOL3I</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Volume between strut piston and cylinder</td>
</tr>
<tr>
<td>WC</td>
<td>SEC⁻¹</td>
<td></td>
<td>Corner frequency in active control servovalve transfer function</td>
</tr>
<tr>
<td>WC1</td>
<td>SEC⁻¹</td>
<td></td>
<td>Natural frequency in electronic compensation network</td>
</tr>
<tr>
<td>WLFOR</td>
<td>ARRAY</td>
<td>LBF</td>
<td>Control limit force for impact</td>
</tr>
<tr>
<td>WLFORR</td>
<td></td>
<td>LBF</td>
<td>Control limit force for rollout</td>
</tr>
<tr>
<td>WSV</td>
<td>SEC⁻¹</td>
<td></td>
<td>Natural frequency in active control servovalve transfer function</td>
</tr>
<tr>
<td>WSV1</td>
<td>IN</td>
<td></td>
<td>Window width of servovalve orifice for high pressure</td>
</tr>
<tr>
<td>WSV3</td>
<td>IN</td>
<td></td>
<td>Window width of servovalve orifice for low pressure</td>
</tr>
<tr>
<td>XBIAS</td>
<td>ARRAY</td>
<td>IN</td>
<td>Servovalve spool displacement for controlling strut charging pressure</td>
</tr>
<tr>
<td>XDDMAX</td>
<td>ARRAY</td>
<td>IN/SEC²</td>
<td>Maximum positive acceleration of servovalve spool</td>
</tr>
<tr>
<td>XDDMIN</td>
<td>ARRAY</td>
<td>IN/SEC²</td>
<td>Maximum negative acceleration of servovalve spool</td>
</tr>
<tr>
<td>Symbol</td>
<td>Array</td>
<td>Unit</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>------------</td>
<td>----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>XKA</td>
<td>ARRAY</td>
<td>AMPS/VOLT</td>
<td>Amplifier gain in active control loop</td>
</tr>
<tr>
<td>XKF</td>
<td>ARRAY</td>
<td>VOLT/IN</td>
<td>Position feedback gain in strut position control loop</td>
</tr>
<tr>
<td>XKS V</td>
<td>ARRAY</td>
<td>IN/AMP</td>
<td>Position gain of servovalve in active control loop</td>
</tr>
<tr>
<td>XLPSV1</td>
<td>ARRAY</td>
<td>IN</td>
<td>Overlap or underlap between spool and sleeve at null for flow Q1</td>
</tr>
<tr>
<td>XLPSV3</td>
<td>ARRAY</td>
<td>IN</td>
<td>Overlap or underlap between spool and sleeve at null for flow Q3</td>
</tr>
<tr>
<td>XSCOM</td>
<td>ARRAY</td>
<td>IN</td>
<td>Commanded (static) position of shock strut</td>
</tr>
<tr>
<td>XSTHR</td>
<td></td>
<td>FT</td>
<td>Threshold strut stroke for determining takeoff or landing mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Elec cont. function not used in program</td>
<td></td>
</tr>
<tr>
<td>XSVDMN</td>
<td>ARRAY</td>
<td>IN/SEC</td>
<td>Maximum negative velocity of servovalve spool</td>
</tr>
<tr>
<td>XSVDMX</td>
<td>ARRAY</td>
<td>IN/SEC</td>
<td>Maximum positive velocity of servovalve spool</td>
</tr>
<tr>
<td>XSVMAX</td>
<td>ARRAY</td>
<td>IN</td>
<td>Maximum positive displacement of servovalve spool</td>
</tr>
<tr>
<td>XSVMIN</td>
<td>ARRAY</td>
<td>IN</td>
<td>Maximum negative displacement of servovalve spool</td>
</tr>
<tr>
<td>ZETAC1</td>
<td></td>
<td></td>
<td>Damping coefficient in electronic compensation network</td>
</tr>
<tr>
<td>ZETAC2</td>
<td></td>
<td></td>
<td>Damping coefficient in electronic compensation network</td>
</tr>
</tbody>
</table>
### TABLE 2
OTHER VARIABLES

<table>
<thead>
<tr>
<th>VARIABLE NAME</th>
<th>TYPE</th>
<th>INITIAL UNITS</th>
<th>VALUE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACON</td>
<td>SEC³</td>
<td>0.0</td>
<td></td>
<td>1./Wsv **2*Wc</td>
</tr>
<tr>
<td>AIC</td>
<td>ARRAY</td>
<td>LBF/FT²</td>
<td>0.0</td>
<td>Switch for regulating flow between strut piston and cylinder</td>
</tr>
<tr>
<td>AP2TO</td>
<td>ARRAY</td>
<td>LBF/FT²</td>
<td></td>
<td>Absolute value of pneumatic charging pressure in strut</td>
</tr>
<tr>
<td>BCON</td>
<td>SEC²</td>
<td>0.0</td>
<td></td>
<td>1./Wsv **2+2.*nsv/(Wsv *Wc)</td>
</tr>
<tr>
<td>BLFORT</td>
<td>ARRAY</td>
<td>LBF</td>
<td></td>
<td>Normal force between strut piston and cylinder at lower bearing</td>
</tr>
<tr>
<td>BUFORT</td>
<td>ARRAY</td>
<td>LBF</td>
<td></td>
<td>Normal force between strut piston and cylinder at upper bearing</td>
</tr>
<tr>
<td>CCON</td>
<td>SEC</td>
<td>0.0</td>
<td></td>
<td>2.*nsv/Wsv+1./Wc</td>
</tr>
<tr>
<td>CMASNG</td>
<td>SLUGS</td>
<td></td>
<td></td>
<td>Effective mass at nose gear root</td>
</tr>
<tr>
<td>COEFO</td>
<td>ARRAY</td>
<td>FT²/LBF-SEC²</td>
<td></td>
<td>Strut main orifice flow coefficient modified for dYh/dP</td>
</tr>
<tr>
<td>COEF1</td>
<td>ARRAY</td>
<td>IN²/SEC/LBF</td>
<td></td>
<td>Coefficient of flow from high-pressure accumulator to strut piston</td>
</tr>
<tr>
<td>COEF3</td>
<td>ARRAY</td>
<td>IN²/SEC/LBF</td>
<td></td>
<td>Coefficient of flow from strut piston to low-pressure reservoir</td>
</tr>
<tr>
<td>COPA</td>
<td>--</td>
<td></td>
<td></td>
<td>cos (φ+α)</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
<td>Units</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>CSV1</td>
<td>ARRAY</td>
<td>FT³/SEC/LBF</td>
<td>Servovalve orifice coefficient from accumulator to strut piston</td>
<td></td>
</tr>
<tr>
<td>CSV3</td>
<td>ARRAY</td>
<td>FT³/SEC/LBF</td>
<td>Servovalve orifice coefficient from strut piston to reservoir</td>
<td></td>
</tr>
<tr>
<td>DCON</td>
<td></td>
<td></td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>DELTX</td>
<td>ARRAY</td>
<td>INCHES</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Difference between strut stroke and command stroke</td>
<td></td>
</tr>
<tr>
<td>DELTX1</td>
<td>ARRAY</td>
<td>VOLTS</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Product of DELTX and position feedback gain in piston control loop</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>ARRAY</td>
<td>LBF</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Variation of gear-airframe interface force about control force</td>
<td></td>
</tr>
<tr>
<td>DLTX1D</td>
<td>ARRAY</td>
<td>VOLTS/SEC</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Time rate of change of DELTX1</td>
<td></td>
</tr>
<tr>
<td>DMD1</td>
<td>ARRAY</td>
<td></td>
<td>--</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Restart data block</td>
<td></td>
</tr>
<tr>
<td>DMTANH</td>
<td>ARRAY</td>
<td></td>
<td>--</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Function of hyperbolic tangent</td>
<td></td>
</tr>
<tr>
<td>DP1</td>
<td>ARRAY</td>
<td>LBF-SEC/FT²</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Time rate of change of hydraulic pressure in strut piston</td>
<td></td>
</tr>
<tr>
<td>DSTOP</td>
<td></td>
<td>FT</td>
<td>.004</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Stopping distance for strut extension stop</td>
<td></td>
</tr>
<tr>
<td>DWFORT</td>
<td></td>
<td>LBF</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Earth axes force at aircraft center of gravity</td>
<td></td>
</tr>
<tr>
<td>ENCG</td>
<td></td>
<td>FT-LBF</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Kinetic energy at aircraft center of gravity</td>
<td></td>
</tr>
<tr>
<td>ENUP</td>
<td>ARRAY</td>
<td>FT-LBF</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Kinetic energy of aircraft at landing gear root</td>
<td></td>
</tr>
<tr>
<td>FFORT</td>
<td>ARRAY</td>
<td>LBF</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Strut axial friction force</td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
<td>Unit</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------</td>
<td>------</td>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FONHST</td>
<td>ARRAY</td>
<td>LBF</td>
<td>0.0</td>
<td>Ground force normal to strut axis</td>
</tr>
<tr>
<td>FORCHT</td>
<td>ARRAY</td>
<td>LBF</td>
<td>0.0</td>
<td>Charging force in fully extended strut</td>
</tr>
<tr>
<td>FORSST</td>
<td>ARRAY</td>
<td>LBF</td>
<td>0.0</td>
<td>Strut force along strut axis</td>
</tr>
<tr>
<td>FSTOP</td>
<td>ARRAY</td>
<td>LBF</td>
<td>0.0</td>
<td>Strut extension stop force</td>
</tr>
<tr>
<td>FSTOPK</td>
<td></td>
<td>LBF/FT</td>
<td>0.0</td>
<td>Strut extension stop spring rate</td>
</tr>
<tr>
<td>HMM</td>
<td>ARRAY</td>
<td>--</td>
<td>0.0</td>
<td>Active control operation indicator 0=off, 1=on</td>
</tr>
<tr>
<td>IA1</td>
<td>ARRAY</td>
<td>--</td>
<td>0</td>
<td>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</td>
</tr>
<tr>
<td>IA2</td>
<td>ARRAY</td>
<td>--</td>
<td>0</td>
<td>Same as IA1</td>
</tr>
<tr>
<td>IA3</td>
<td>ARRAY</td>
<td>--</td>
<td>0</td>
<td>Same as IA1</td>
</tr>
<tr>
<td>IA4</td>
<td>ARRAY</td>
<td>--</td>
<td>0</td>
<td>Same as IA1</td>
</tr>
<tr>
<td>IA6</td>
<td>ARRAY</td>
<td>--</td>
<td>0</td>
<td>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</td>
</tr>
<tr>
<td>IA7</td>
<td>ARRAY</td>
<td>--</td>
<td>0</td>
<td>Same as IA6</td>
</tr>
<tr>
<td>Indicator</td>
<td>Array Type</td>
<td>Value</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
</tbody>
</table>
| IA9           | ARRAY      | 0     | Indicator for defining the return of second lead-lag network compensation current to zero when gear is fully-extended during rebound: 
|               |            |       | =0, non-zero current; =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: 
|               |            |       | =0, has not returned; =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: 
|               |            |       | =0, gear compressed; =1, gear fully-extended                                |
| IGO           | ARRAY      | 0     | Indicator for defining accelerated return of servo-controller parameters to fully-extended gear values: 
<p>|               |            |       | =0, inactive; =1, active                                                   |
| IIXSVH        | INTEGER ARRAY | 0   | Indicator for servovalve returning to null from high pressure                |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IIXSVL</td>
<td>INTEGER ARRAY</td>
<td>0</td>
<td>Indicator for servovalve returning to null from low pressure</td>
</tr>
<tr>
<td>ITRIP</td>
<td>ARRAY</td>
<td>0</td>
<td>Indicator for defining the return of gear and servovalve parameters to fully-extended gear values: =0, inactive; =1, active</td>
</tr>
<tr>
<td>INDEACT</td>
<td>INTEGER ARRAY</td>
<td>0</td>
<td>Indicator for control regime 0=off, 1=impact, 2=rollout</td>
</tr>
<tr>
<td>INITSW</td>
<td>INTEGER</td>
<td>1</td>
<td>Switch indicating first pass through program l=first pass</td>
</tr>
<tr>
<td>IOPCO</td>
<td>INTEGER ARRAY</td>
<td>1</td>
<td>Indicator for control operation during return to null position</td>
</tr>
<tr>
<td>IPASS</td>
<td>INTEGER ARRAY</td>
<td>0</td>
<td>&quot;</td>
</tr>
<tr>
<td>IPSTOP</td>
<td>INTEGER</td>
<td>0</td>
<td>&quot;</td>
</tr>
<tr>
<td>IQCU</td>
<td>ARRAY</td>
<td>0</td>
<td>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</td>
</tr>
<tr>
<td>ISET</td>
<td>INTEGER ARRAY</td>
<td>1</td>
<td>See IOPCO</td>
</tr>
<tr>
<td>ISTROK</td>
<td>INTEGER ARRAY</td>
<td>0</td>
<td>Indicator for strut stroke initialization</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IXS</td>
<td>ARRAY</td>
<td>0</td>
<td>Indicator defining the condition of servovalve spool displacement equal to bias value for the fully-extended strut during rebound: =0, unequal =1, equal</td>
</tr>
<tr>
<td>IXSVH</td>
<td>INTEGER ARRAY</td>
<td>0</td>
<td>Indicator for control operation during return to null position</td>
</tr>
<tr>
<td>IXSVL</td>
<td>INTEGER ARRAY</td>
<td>0</td>
<td>&quot;</td>
</tr>
<tr>
<td>NAC</td>
<td>INTEGER ARRAY</td>
<td>0</td>
<td>Fluid flow indicator 0=off; 1=from strut; 2=to strut</td>
</tr>
<tr>
<td>NITER</td>
<td>INTEGER</td>
<td>—</td>
<td>Number of iterations required to converge to solution</td>
</tr>
<tr>
<td>PGA1T</td>
<td>ARRAY LBF/FT²</td>
<td></td>
<td>Gauge pressure of hydraulic fluid in strut piston</td>
</tr>
<tr>
<td>PGA2T</td>
<td>ARRAY LBF/FT²</td>
<td></td>
<td>Gauge pressure of nitrogen in strut cylinder</td>
</tr>
<tr>
<td>PGA3T</td>
<td>ARRAY LBF/FT²</td>
<td></td>
<td>Gauge pressure of hydraulic fluid in volume 3</td>
</tr>
<tr>
<td>PGA1T1</td>
<td>ARRAY LBF/FT²</td>
<td></td>
<td>See PGA1T</td>
</tr>
<tr>
<td>PR</td>
<td>ARRAY LBF/IN²</td>
<td></td>
<td>Servovalve return pressure</td>
</tr>
<tr>
<td>PS</td>
<td>ARRAY LBF/IN²</td>
<td></td>
<td>Servovalve supply pressure</td>
</tr>
<tr>
<td>P1</td>
<td>ARRAY LBF/IN²</td>
<td></td>
<td>Strut piston pressure</td>
</tr>
<tr>
<td>QC</td>
<td>ARRAY IN³/SEC 0.0</td>
<td></td>
<td>Flow rate to the load</td>
</tr>
<tr>
<td>Q0</td>
<td>ARRAY FT³/SEC 0.0</td>
<td></td>
<td>Flow rate through strut main orifice</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
<td>Units</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>--------</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td>QS1</td>
<td>ARRAY</td>
<td>IN³/SEC</td>
<td>Flow rate from accumulator to strut piston</td>
</tr>
<tr>
<td>QS3</td>
<td>ARRAY</td>
<td>IN³/SEC</td>
<td>Flow rate from strut piston to reservoir</td>
</tr>
<tr>
<td>QSV</td>
<td>ARRAY</td>
<td>FT³/SEC</td>
<td>Fluid flow rate through servovalve, positive from supply to strut</td>
</tr>
<tr>
<td>QSVCU</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Cumulative volume of fluid added to or removed from strut</td>
</tr>
<tr>
<td>QSVN</td>
<td>ARRAY</td>
<td>FT³/SEC</td>
<td>Volume expansion or compression rate of nitrogen in high pressure accumulator</td>
</tr>
<tr>
<td>QSV1</td>
<td>ARRAY</td>
<td></td>
<td>See QS1</td>
</tr>
<tr>
<td>QSV3</td>
<td>ARRAY</td>
<td></td>
<td>See QS3</td>
</tr>
<tr>
<td>QTOLER</td>
<td>IN³/SEC</td>
<td>.0001</td>
<td>Tolerance allowed in calculating flow rates</td>
</tr>
<tr>
<td>REDSLP</td>
<td>ARRAY</td>
<td>LBF/SEC</td>
<td>Slope for reducing control limit force from impact value to rollout value</td>
</tr>
<tr>
<td>RESA</td>
<td>ARRAY</td>
<td>0.0</td>
<td>Control activation switch</td>
</tr>
<tr>
<td>SA</td>
<td>ARRAY</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>SBFOT</td>
<td></td>
<td>0.0</td>
<td>Strut elastic bending force</td>
</tr>
<tr>
<td>SIPA</td>
<td></td>
<td></td>
<td>sin (φ+α)</td>
</tr>
<tr>
<td>UNSPRNG</td>
<td>SLUGS</td>
<td></td>
<td>Mass of strut piston and gear components attached</td>
</tr>
<tr>
<td>VCUM</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Cumulative volume of fluid flowed from piston to cylinder</td>
</tr>
<tr>
<td>VELDEC</td>
<td>ARRAY</td>
<td>FT/SEC</td>
<td>Velocity for initiating transition from impact limit force to rollout value</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
<td>Unit</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>VMASS</td>
<td>ARRAY</td>
<td>SLUGS</td>
<td>Vehicle mass</td>
</tr>
<tr>
<td>VOLAHT</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Volume of hydraulic fluid in high pressure accumulator</td>
</tr>
<tr>
<td>VOLANT</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Volume of nitrogen in high pressure accumulator</td>
</tr>
<tr>
<td>VOL1T</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Volume of hydraulic fluid in strut piston</td>
</tr>
<tr>
<td>VOL2T</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Pneumatic volume in strut cylinder</td>
</tr>
<tr>
<td>VOL3T</td>
<td>ARRAY</td>
<td>FT³</td>
<td>Volume between piston and cylinder</td>
</tr>
<tr>
<td>WFORNT</td>
<td>ARRAY</td>
<td>LBF</td>
<td>Gear-airframe interface force</td>
</tr>
<tr>
<td>XMA</td>
<td>ARRAY</td>
<td>AMPERES</td>
<td>Input signal to electronic compensation networks</td>
</tr>
<tr>
<td>XMA1 ... 4</td>
<td>ARRAY</td>
<td>AMPERES</td>
<td>Modification of input signal to the notch and lead-lag networks</td>
</tr>
<tr>
<td>XMA5</td>
<td>ARRAY</td>
<td>AMPERES</td>
<td></td>
</tr>
<tr>
<td>XMA6,7</td>
<td>ARRAY</td>
<td>AMPERES</td>
<td></td>
</tr>
<tr>
<td>XMA8</td>
<td>ARRAY</td>
<td>AMPERES</td>
<td></td>
</tr>
<tr>
<td>XMA9,10</td>
<td>ARRAY</td>
<td>AMPERES</td>
<td></td>
</tr>
<tr>
<td>XMA11</td>
<td>ARRAY</td>
<td>AMPERES</td>
<td>Output signal from electronic compensation networks to servovalve</td>
</tr>
<tr>
<td>XMU</td>
<td>CENTIPOISE</td>
<td></td>
<td>Viscosity of hydraulic fluid</td>
</tr>
<tr>
<td>XSTOT</td>
<td>ARRAY</td>
<td>FT</td>
<td>Instantaneous stroke required to dissipate remaining kinetic energy at simultaneous force level</td>
</tr>
<tr>
<td>Symbol</td>
<td>Type</td>
<td>Unit</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>--------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>XSV</td>
<td>ARRAY</td>
<td>INCHES</td>
<td>Servovalve spool displacement</td>
</tr>
<tr>
<td>XSVDD</td>
<td>ARRAY</td>
<td>IN/SEC$^2$</td>
<td>0.0</td>
</tr>
<tr>
<td>XSVDDD</td>
<td>ARRAY</td>
<td>IN/SEC$^3$</td>
<td>0.0</td>
</tr>
<tr>
<td>XSVDOT</td>
<td>ARRAY</td>
<td>IN/SEC</td>
<td>0.0</td>
</tr>
<tr>
<td>XVALVE</td>
<td>ARRAY</td>
<td>IN</td>
<td>Analytically controlled servovalve spool displace-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ment</td>
</tr>
<tr>
<td>ZDANT</td>
<td></td>
<td>FT/SEC</td>
<td>Velocity of nose gear root</td>
</tr>
<tr>
<td>ZSSC</td>
<td>ARRAY</td>
<td>FT</td>
<td>Allowable strut stroke for activating control</td>
</tr>
</tbody>
</table>
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 $C_A$ and $C_N$ 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 extremely 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 C82411, C11178, 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 OPTI 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,

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

the following code was inserted before statement number 50

\[
\begin{align*}
S(1,I) &= -0.5*ES(I) \\
SD1(1,I) &= 1.0E-10 \\
SD2(1,I) &= 1.0E-10
\end{align*}
\]

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

\[
\begin{align*}
51 & \text{ IF(SD2(1,I),LT,0.)GO TO 30} \\
& \text{ IF(SD1(1,I),LT,0.)SD1(1,I)=0.} \\
& \text{ GO TO 55} \\
30 & \text{ SD2(1,I)=0.}
\end{align*}
\]

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

\[
\begin{align*}
\text{RE-CHECK SHOCK STRUT FORCE FOR RE-CONDITIONED FULLY EXTENDED STATE} \\
\text{IF(SD1(1,I),EQ,0,0.0,AND,FT(I),LE,ABS(SF(I))))SF(I)=FT(I)}
\end{align*}
\]

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
original page is of poor quality

(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 ACOLBLK, CSCMODS, ACTINIT, ALGEAR, PHL02Z, FLOZE2, and LIMITS contained the code required for a first cut at implementing the active control. A comdeck, ACOLBLK, 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,AREO(5) ,AREO3(5) ,BLFORT(5) ,BLMU(5) ,BUFORT(5) ,Bム(5)
3,BETA ,CDMC(5) ,CDMO(5) ,CSV(5) ,CFFOR(5) ,CD3(5)
4,COEF ,COEFO(5) ,COEF3(5) ,COPA
5,CSV(5) ,CSV3(5) ,DELT ,DELTX(5) ,DELTX1(5) ,DELTX1D(5)
6,DF(5) ,DPI(5) ,DSV(5) ,DSTOP ,DQTA ,DUMTANH(5)
7,ENUP(5) ,EPSILO(5) ,EPSRO(5) ,EPSPL ,ETASV ,FSTOPK
8,FFORT(5) ,FOAHS(5) ,FONHS(5) ,FORCHT(5) ,FORSTT(5) ,FSTOP(5)
9,GAMAH(5) ,GAMAN ,GAMN ,HMM(5) ,ICOVS(5) ,IDEACT
*,IFRI(5) ,IXSVH(5) ,IXSVL(5) ,IOPCO(5) ,IPASS(5) ,ISET(5)
1,IXSV(5) ,IXSVL(5) ,IFSTOP(5) ,ISTROK(5) ,KAPT(5)
2,NAC(5) ,NITER ,OMRUN ,PAM ,PAM
3,PERCN(5) ,PGAHA(5) ,PGALAC(5) ,PGAII(5) ,PGA1T(5)
4,PGA2I(5) ,PGA2T(5) ,PGA3I(5) ,PGA3T(5)
5,PR(5) ,PS(5) ,P1(5) ,QC(5) ,QR(5) ,QPUMPS(5)
6,PSV(5) ,QSVCM(5) ,QSV(5) ,QSV1(5) ,QSV3(5) ,QSI(5)
7,PS3(5) ,QTOLER ,RCLS(5) ,REDSLV(5) ,RHOH ,SBFOT
8,SIPA ,SA(5) ,RESA(5) ,RAUF ,TC1 ,TC2
COMMON /ACTIVE/ TC3 ,TC4 ,VCUM(5) ,VELDEC ,VOLLI(5)
1,VALI(5) ,VAL2I(5) ,VAL2T(5) ,VAL3I(5) ,VAL3T(5) ,WC
2,WCT ,WFTO(5) ,WLFORT(5) ,WLMR ,WLMR ,XBIAS(5) ,XVALV(5)
3,XKA(5) ,XXF(5) ,XKV(5) ,XLPSV(5) ,XLPSV3(5) ,XSTHR
4,XMA(5) ,XMAI(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) ,XSVDC(5) ,XSVOD(5)
```
New decks ACTINIT, for initializing the active gear variables, and ALGEAR, PHLOZ2, FLOZ2, and LIMITS, containing the active gear logic, were added after the LGEA3C deck. These new decks are provided below.

SUBROUTINE ACTINIT

C***************************************************************************
FATOLA VARIABLES***************************************************************************

COMMON/DIRCOM/DM1(115),ALPHD,DM1A(20),AMASS,DM2(147),DCL1,DCLM,
CDO CN1,DCL2,DCN2,DCN3,DCN4,DM3(99),FZB7P,
CDUM(4),FYB7P(4),FZB7P,DM5(17),GX87F,DM6(8),GZB7F,
CDM7(218),INDSTE48,PHPD,INDSTE1(23),PSIPD,INDSTE2(156),THPD,
CDNP(3),TIME,DM8(267),PI77R(2),PI77R1(2),DM9(4),
CQ177R(2),QI77R(2),DM10(4),RI77R(2),RI77R1(2),DM11(48),
CXR77F(2),XG77F1(12),YG77F(2),YG77F1(12),
CGZ77F(2),ZG77F1(2),DUM13(52),
CNSTRUT,MASS(5),RX(5),RY(5),RZ(5),THETAD(5),ERDEG,GR,
CMT(4),ZERO(5),W(5),DELTA4(5),MOMENT(5),
CC(5),CE(5),CC(5),C2E(5),NVGPT,NPP,MB(5),RLT,NDETA,
CE(5),SB(5),SD21(2),SD22(2),SD23(2),SD24(2),SD25(2),
COMMON/DIRCOM/
CSO11(2),SO12(2),SO13(2),SO14(2),SO15(2),
CS1(2),SS1(2),S3(2),S4(2),S5(2),
CS2D(2),SD2D(2),SS2D(2),SS2D(2),SS2D(2),
CS2D11(2),SSD2D(2),SS2D11(2),SSD2D(2),SSD2D(2),
CS2S(2),SS2(2),S3(2),S4(2),S5(2),
CS2D(2),SS2D(2),S2D(2),S2D24(2),S2D25(2),
COMT11(2),OMT12(2),OMT13(2),OMT14(2),OMT15(2),
COMT1(2),OMT12(2),OMT12(2),OMT14(2),OMT15(2),
CA(5),BI(5),DELT1A,DELT1B,DELT4,DELT5,DELT6,
CDDEL1,DEDEL2,DEDEL3,DEDEL4,DEDEL5,ISTAGE,
CPRTMIN,IPRT,ISER,ISERPL3,ISERPL4,ISERPL5,
CDM14(22),IB(5),DM15(127),INDL,DM16(107),CASK(44),INFL,
*NMOD,DM18(40),SXMOD(100),SYM0D(100),SYM0D(100),DM19(1868),
*G40(20),DDM20(20),
*DDM21(20),SLEN1(5),SLEN2(5),DM15(13),INDNW,DDM22(10),ETADES,
*DDM23(5),AH(5),PH(5),DDM24(30),
COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LTTP

CALL ACOBLK
DIMENSION C03(5), SD1(2,5)
EQUIVALENCE(SD11(1),SD11(1,1)), (DM5(16),GREFF)

DATA STATEMENTS TO SET VALUES OF INPUT VARIABLES

DATA AMUH/0.00018/, BETA/14400000./
DATA BLMU,BUMU/10*0.15/
DATA AREA1,AREA2,AREA3/5*0.21139,5*0.32167,5*0.00689/
DATA AREO3,ADAO3,APINT/5*0.0066,5*0.19635,5*0.0 /
DATA CDMOE,CDMSE,CD3/5*0.9,5*0.14977,5*0.46/ 
DATA CDFOR,DELTA,DITOTA/5*0.62,0.0001,0.0 /
DATA EPSSL,EPSSL,ETASV/5*200.0,0.0,5*2000.0,0.436/
DATA GAMAN,GAMAH,GNR/1.06,5*52.36,0.0 /
DATA ICOS1,IFSTOP/10*0/
DATA IDEACT,II,KAPT,OMRUN,PERCENT/7*0,-0.07575,5*1.66115/
DATA PATH,PGAHA,PGALAC/2116.8,5*432000.0,5*0.0/
DATA PGI1,PGI2,PG3I/15*40320.0 /
DATA TAUFT,TC1,TC2,TC3,TC4/0.1,0.281,0.141,0.001,0.0001/
DATA VOL1,VOL2,VL3/5*0.36379,5*0.47164,5*0.0(/
DATA RHOH,WLFOR,WLFOR/1.626,5*500.0,0.0 /
DATA DSV,RSCLW,WSVI,WSV3/5*1.1875,5*0.0000475,2*3.480/
DATA WC,WC,+XKSV,XKSV,XSTH/1263.0,5*251.3,655.5,5*0.00250,0.020833 /
DATA XK+XKF,XLPSV1,XLPSV3/5*0.04,5*60.0,5*0.0,5*0.0 /
DATA XBIAS,XSCDM,XSVDMN,XSVDMX/5*0.00014,5*1.206,5*1.5*0.1/ 
DATA XDDMAX,XDDMIN/5*1.0E20,5*0.1E20 /
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****************************************************************************
*C IF (TIME .GT. DELT) RETURN
C****************************************************************************

ACON=1.0*(WSV*WSV*WC)
BCON=1.0*(WSV*WSV)+2*ETASV/(WSV*WC)
CCON=2.*ETASV/WSV+1./WC
DCON=1.0
XMU=AMUH/0.0000209
QTLER=0.001
SBFOT=0.0
VELDEC=0.
OMRUN=OMRUN*0.0174329
II = 0

DO 100 I=1,NSTRUT
REDLSL(I) = 100000.0
ISTROI(I)=0
NACI(I)=0
VOLANT(I) = VOLANI(I)
VOL1(I)=VOL1(I)
VOL2(I)=VOL2(I)
VOL3(I)=VOL3(I)
QSVCU(I)=0.0
DF(I)=0.

C
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)
XSVDDT(I) = 0.0
DP1(I) = 0.
COEF3(I) = CD3(I) * SQRT(2. * GREFF/GAMAH(I))

NOTE: SUBROUTINE 'PHLOZ2' COMPUTES INITIAL PressURES AND FLOWS IN UNITS OF INCHES.
COEF = CSV(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, QSV 
& , CSV1(I), CSV3(I), XMU, QTOLER, NITER, P1(I), QS1(I), QS3(I))
P11(I) = P1(I) * 144.
P12(I) = PGA11(I)
P13(I) = P12(I)
QSV1(I) = QS1(I)/1728.
QSV3(I) = QS3(I)/1728.

PGA1T(I) = PGA11(I)
PGA2T(I) = PGA211(I)
PGA3T(I) = PGA311(I)
QD(I) = 0.
QSV(I) = QSV1(I) - QSV3(I)
VCUM(I) = 0.

AP2TO(I) = PGA211(I) + PATM
FFORT(I) = 0.0
FORSST(I) = 0.0
WFOR1(I) = 0.0
FONHST(I) = 0.0
FORCHT(I) = 0.0
XVALVE(I) = XSV(I)
100 CONTINUE
**ORIGINAL PAGE IS OF POOR QUALITY**

```plaintext
CALL SETUP RETURN *IDENT ALGEAR
*INSERT ACTINIT.145
*DECK ALGEAR
SUBROUTINE ALGEA1

**FATOLA VARIABLES**

COMMON/DIRCOM/DM1(115),ALPHAD,DM1A(20),AMASS,DM2(147),DCL1,DCM1,
CUM1,DCL2,DCM2,DCN2,DCL3,DCM3,DCN3,DM3(99),FXB7P,
CDUM4(3),FYB7P(4),FZB7P,DM5(17),GXB7F,DM6(8),GZB7F,
CDMT(218),INDSTE(48),PHIBD,INDSTE3(123),PSIPD,INDSTE2(156),THTPD,
CINDST3(5),TIME,DM8(287),PI17R12),PI17R1(2),DM9(4),
C017R(2),Q17R1(2),DM10(4),R17R(2),R17R1(2),DM11(48),
CXG77F(2),XG77F1(12),YG77F(2),YG77F1(12),
CZG77F(2),ZG77F1(2),DM13(52),
CNSTRT,MASS(5,RX(5),RY(5),RZ(5),THETAD(5),ERDEG,RGR,
CNIRES(5),PZERO(5),W(5),DELTAM(5),MOMENT(5),
CCC(5),CE(5),C2C(5),C2E(5),NGPNT,NPB,MB(5),RLT,NDLTA,
CFS(5),SB(5),SD21(2),SD22(2),SD23(2),SD24(2),SD25(2),
COMMON/DIRCOM/
CSD11(2),S12(2),S13(2),SD14(2),SD15(2),
CS1(2),SS2(2),S3(2),S4(2),S5(2),
CS201(2),SS202(2),SS203(2),SS204(2),SS205(2),
CS201(2),SS202(2),SS203(2),SS204(2),SS205(2),
CS21(2),SS21(2),SS22(2),SS24(2),SS25(2),
COMTD11(2),OMTD12(2),OMTD13(2),OMTD14(2),OMTD15(2),
COMTD1(2),OMCDM1(2),OMCDM2(2),OMCDM3(2),OMCDM4(2),OMCDM5(2),
CAI(5),BI(5),DETA1,DETA2,DETA3,DETA4,DETA5,
CDELT1,CDIEL2,DELTA3,DELT5,DELTA5,ISTAGE,
CPRTMIN,IPLT,ISOF,ISTPL1,ISTPL2,ISTPL3,ISTPL4,ISTPL5,
CDMD4(22),IR(5),DM15(127),INDLG,DM16(107),CASK(44),INDFLX,
*NAMES,DM18(40),SXMOD(100),SYMOD(100),SMOD(100),DM19(1686),
*GQD2(20),DDM20(20)
*,DDM21(20),SLGEN1(5),SLen2(5),DUM15(13),INDNWS,DDM22(10),ETADES,
*DDM23(5),AH(5),PH(5),DDM24(30)
REAL MASS,MOMENT,MASS,MUS,NIRES,MR
DIMENSION DLGAUT(14),DLGDF(47),DLG7(47),DLG(299)
COMMON/LGATS/ARG11,ARG13(ARG33,ARG33,AM1A(5),VAXL(5)
COMMON/LGDF/FA(2),FC(2),P2(5),PPS(5),C(5),IPPT,LHPT
COMMON/LG/FVX,FYX,FZ,LM,LM,LM,MN,EMPS02
CRX(5),VR(5),EPS00(5),PA(5),FDELA(5),
CF/1ZP(5),USD(5),RDY(5),PDZ(5),RDY(5),RDZ(5),PDZ(5),
CV(5),VT(5),VT(5),G7(5),
*VGPT(5),FTRX(5),FTRY(5),
CSR(5),SF(5),PSK(5),PSK(5),MTRX(5),MTRY(5),
```
CMTRZ(5), MA(5), RG11, RG13, RG31, RG33, IPRT,
CMTX, MTY, MTZ, SFTRX, SFTRY, SFTRZ, FTRA,
CFTRB, FTRC, SNTRX, SNTRY, SMTRZ
COMMON/FLXOP/GFORC2(100), GFORC3(100), GFORC4(100), BHI(308)
EQUIVALENCE (DLGALT(1), ARG11), (DLGDE(1), LA(1)),
* (DLG(1), FXH), (DLGE(1), A11(1))
COMMON/TABSRC/DUM1(103), LOC(7)
COMMON/HTCOM/HT, HT1, HT2

CALL ACDBLK

REAL MUVPE, 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),
COMED1(2, 5), OMET(2, 5)
EQUIVALENCE (P(1), PRES(1))
EQUIVALENCE (DELTA(1), DELTA1), (DDELTA(1), DDELTA1)
EQUIVALENCE
C(SD21(1, 1), SD2(1, 1)), (SD11(1, 1), SD1(1, 1)), (S1(1), S(1, 1)),
C(S2D21(1, 1), S2D2(1, 1)), (S2D11(1, 1), S2D1(1, 1)), (S21(1), S2(1, 1)),
C(OMED111, OMET1(1, 1)), (OMT11, OMTET(1, 1))

C***********************************************************************
EQUIVALENCE (DMM(16), GREF)
DATA IFRI/5*0/, FSTOPK/0.0, FSTOP/5*0.0, DSTOP/0.004/
DATA SA, RESA, HMM, IIXSVL, IIXSVH, IIXSVL, IIXSVH, IPASS
1 /15*0., 25*0/
DATA ENUP/5*0.0/
DATA ISET, IOPCO, QSVN/10*1, 5*0.0/
Q1T1(T1, T2) = SIGN(1, (T1-T2)) * SQRT(ABS(T1-T2))

C***********************************************************************

DATA RADDEG, DEGRAD/57.2957795, 01745329/

C MAIN COMPUTATIONAL AREA
C
C RL MATRIX ELEMENTS
C RL(1,1) = DLC1*RG11+DCL3*RG13
C RL(1,2) = DCL2
C RL(1,3) = DLC1*RG31+DCL3*RG33
C RL(2,1) = DCM1*RG11+DCM3*RG13
C RL(2,2) = DCM2
C RL(2,3) = DCM1*RG31+DCM3*RG33
C RL(3,1) = DCN1*RG11+DCN3*RG13
C RL(3,2) = DCN2
C RL(3,3) = DCN1*RG31+DCN3*RG33
CALL LGEA3C

C*************************************************************************

C START ACTIVE GEAR CALCULATIONS
C
COPA = COS((THP TD+DIOTA)*DEGRAD)
SIPA = SIN((THP TD+DIOTA)*DEGRAD)
IF(TIME GT DELT) GO TO 18
WRITE(6,1013)
1013 FORMAT (1HO20H ACTIVE CONTROL GEAP)
18 IF(IDEACT.EQ.1) GO TO 56
19 IF(IDEACT.EQ.2) GO TO 80
20 IF(HM(I) .GE. 0.) GO TO 90
21 IF(OMRUN .GT. 0.0) GO TO 25
22 IF(ZG77F1(I) .GE. VELDEC) GO TO 90
23 GO TO 40
24 IF(ZG77F1(I) .GE. VELDEC+XG77F1(I)*TAN(OMPUN)) GO TO 90
40 WRITE(6,1014)
1014 FORMAT (1HO36H REDUCE CONTROL LIMIT FORCE AT TIME=,E16.8)
IDEACT=1
56 WLFOR(I)=WLFOR(I)-PEDSLP(I)*DELT
57 EPSIL0(I)=EPSIL0(I)+EPSILP*DELT
58 IF(WLFOR(I) .GE. WLFOR) GO TO 90
60 WRITE(6,1015)
1015 FORMAT (1HO27H CONTROL AT WLFOR AT TIME=,E16.8)
IDEACT=2
80 WLFOR(I)=WLFOR
81 EPSIL0(I)=EPSIL0(I)
90 CONTINUE
C*****************************************************************************
 DO 65 I=1,NSTRUT
 WORT(I) = (SODEP(XB7P*FB7P*FB7P*YB7P*FB7P*ZB7P*FB7P)) / (NSTRUT-1)
 E = E + (FORSST(I)*COPA)
C*****************************************************************************
 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)=VOLI1(I)-(AREA1(I)-APINT(I))S(I,I)
 VOL3T(I)=VOL3I(I)+AREA3(I)*S(I,I)
 VOL2T(I)=VOL2I(I)-(AREA2(I)-AREA1(I)+APINT(I))S(I,I)+(VOL3T(I)
 X -VOL3I(I))-VCLM(I)
 PGA2T(I)=AP2TO(I)*((VOL2I(I)/VOL2T(I))**GAMAH)-PATM
 IF(SD(I,I) .EQ. 0.0) GO TO 104
 PGA3T(I)=(CCOEF3(I)*AREA3(I))**2*PGA2T(I)-SD(I,I)/ABS(SD(I,I))
 X *(SD(I,I)*AREA3(I))**2)
 E/(CCOEF3(I)*AREA3(I))**2) GO TO 105
104 PGA3T(I)=PGA2T(I)
105 IF(PGA1T(I) .GE. PGA2T(I)) GO TO 106
106 GAMAH(I)=RHOD*GREFF*(1.0+(PGA1T(I)*3.04E-08)-
 * (PGA1T(I)**2*2.72E-15))
 GO TO 107
107 GAMAH(I)=RHOD*GREFF*(1.0+(PGA2T(I)*3.04E-08)-
 * (PGA2T(I)**2*2.72E-15))
108 IF(PGA1T(I) .GE. PGA2T(I)) COEFO(I)=
 * CDMO(I)*SORT(ABS(2.*GREFF/GAMAH(I)))
 IF(PGA2T(I) .GE. GTA(PGA2T(I))COEFO(I)=
 * CDMO(I)*SORT(ABS(2.*GREFF/GAMAH(I)))
IF(SD1(I,I) .LE. 0.0) GO TO 109

OD(I)=COEPO(I)*(AREM0(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(I,I))/(SLEN1(I)+S(I,I))+1.0)
BUFORT(I)=FONHST(I)*((SLEN2(I)-S(I,I))/(SLEN1(I)+S(I,I))
FFORT(I)=BNUU(I)*ABS(BUFORT(I))+BLMU(I)*ABS(BLFORT(I))

140 CONTINUE

C COMPUTE SHOCK STRUT CHARGING FORCE
FORC4T(I)=PGA1T(I)*APEA1(I)+PGA2T(I)*((AREA2(I)-APEA1(I)))-PGA3T(I)
X*APEA3(I)+FFOPTT(I)+CFFOPT(I)

C COMPUTE NORMAL AND AXIAL HUR TO SHOCK STRUT FORCES AT HUR

141 FORCSTT(I)=FORC4T(I)*SQR(P(DY(I)**2+FDY(I)**2+MASS(I))*GREFFSIP+SBFOT
IF(ABS(FXT(I))) .LE. FORCHT(I) .AND. S(I,I) .LE. 0.0) GO TO 150

GO TO 801

150 CONTINUE
FORSTT(I)=FT(I)
SD1(I,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 STFUT IS POSITIVE
901 IF(SD1(I,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)=ARS(TANH(2.0*SD1(I,I)))
IFR(I)=1

284 IF(S(I,I) .LE. DSTOP .AND. SD1(I,I) .LT. 0.0) GO TO 900
GO TO 902

900 IF(S(I,I) .LE. 0.0) GO TO 903
GO TO 904

903 SD2(I,I)=0.0
SD1(I,I)=0.0
S(I,I)=0.0
DI(I)=0.0
FFORT(I)=0.0
VCUM(I)=0.0
CO(I)=0.0
II=0

C CALL VIRK4(I,N,NT,C1 SPEC,CIMAX,IERR,VAR,CDVAR,DER,ELE1,ELE2,
C 1 ELE2,ELE1,ELE2,ELT,EPPVAL,DERSUB,FSUS,ITEXT)

GO TO 902

904 CONTINUE
IF(I\$\text{STOP}(I)} .NE. 0) GO TO 906

905 DSTOP = S(I, I)
FSTOP = 2.0*MASS(I)*SD1(I, I)**2/DSTOP**2

906 IF(S(I, I) .LE. DSTOP/2.0) GO TO 908

907 FSTOP(I) = FSTOP*(DSTOP - S(I, I))
GO TO 909

908 FSTOP(I) = FSTOP*S(I, I)

909 IFSTOP(I) = 1
IFRI(I) = 0
GO TO 901

902 FSTOP(I) = 0.0

901 IFABS(FT(I)) .LE. FORCHT(I) AND S(I, I) .EQ. 0.0) GO TO 500

IF(SD1(I, I) .LT. 0.0) GO TO 470

470 FFORT(I) = FFORT(I)
CFFOR(I) = CFFOR(I)

471 FORSST(I) = -(PGA1T(I) - PGA2T(I))*(AREA1(I) - APINT(I))

472 FORSST(I) = -(PGA1T(I) - PGA2T(I))*(AREA1(I) - AREA2(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(I, I) .LE. 0.0) GO TO 290, 295

290 IF(IOPCO(I) .EQ. 1) GO TO 295
IF(PGA1T(I) - 1000.0) LT. PGA1T(I) AND
IF(XVALVE(I) - NE. 0.0) GO TO 311
IF(IPASS(I) .EQ. 1) GO TO 296
XVALVE(I) = XVALVE(I) * XVALVE(I) + XBIAS(I)
IPASS(I) = 1
GO TO 294

298 IF(ICS1V(I) .EQ. 1) GO TO 291
IOPCO(I) = 0
IF(XSV(I) .LT. 0.002 AND XSV(I) .GT. -0.002) GO TO 291, 295

291 IF(SD1(I, I) .LE. 0.0 AND ICS1V(I) .EQ. 1) GO TO 311
IF(IOPCO(I) .EQ. 1) GO TO 295
IF(PGA1T(I) .GT. PGA1T(I)) GO TO 292, 293

292 IF(XSVL(I) .EQ. 1) GO TO 294
XVALVE(I) = XVALVE(I) * XSVL*(I) + DDEL*PERCNT(I)
IF(XVALVE(I) .LT. 0.1) GO TO 300, 294
XVALVE(I) = -0.1
IXSVL(I) = 1
GO TO 294

300 XVALVE(I) = 0.1
IXSVL(I) = 1
GO TO 294

293 IF(XSVH(I) .EQ. 1) GO TO 294
XVALVE(I) = XVALVE(I) + XSVH*(I) + DDEL*PERCNT(I)
IF(XVALVE(I) .GE. 0.1) GO TO 302, 294
XVALVE(I) = 0.1
IXSVH(I) = 1

294 CONTINUE
II = 0
CALL VIRK4 (II, NT, CI, SPEC, CIMAX, IERR, VAR, CUVAR, DFR, ELE1, ELE2, 
ELT, ERPVAL, DERSUB, CHSUB, ITEXT)
DLTX1D (II) = 0.0
ICOSV (II) = 1

296 IF (WFORT (I) .GE. 0.0 .AND. S (I, 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) GO TO 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) GO TO 308, 400

308 XVALVE (I) = 0.0
IIXSVL (I) = 1

400 II = 0
CALL VIRK4 (II, NT, CI, SPEC, CIMAX, IERR, VAR, CUVAR, DFR, ELE1, FLF2, 
ELT, ERRVAL, DERSUB, CHSUB, ITEXT)

410 IF (XVALVE (I) .NE. 0.0) GO TO 295
ICOSV (I) = 0
DELTX1 (I) = DELTX (I) * XKF (I)
XMAX (I) = (DF (I) + DELTX1 (I)) * XKA (I)
XMAI1 (I) = XMAX (I)
XSVX (I) = XSV (I) * XMAI1 (I) + YBIAS (I)
CALL LIMITS (XSV (I), XSVDMT (I), XSVMAX (I), XSVMIN (I))
IPASS (I) = 0
IIXSVL (I) = 0
IIXSVH (I) = 0
IIXSVL (I) = 0
IIXSVH (I) = 0
IOPCO (I) = 1
CALL PHLZ2 (PS (I), PR (I), XSV (I), OC (I), XLPSV1 (I), XLPSV3 (I), RCLS (I), 
ES (I), CSV1 (I), CSV3 (I), XMU, QTOLER, NITER, PL (I), GS1 (I), GS3 (I))

C

295 IF (ISTROK (I) .EQ. 1 .AND. S (1, I) .GT. 0.0) IOPCO (I) = 0

C
ENUP (I) = 5 * AMASS * ZG77F1 (I) ** 2
ENUP (I) = ENUP (I) / (NSTPUT - 1)
IF (HAMM (I) .EQ. 1.0) GO TO 130
SA (I) = 0.
IF (WFORT (I) .LT. 0.0) XSTOT (I) = ENUP (I) / ((-WFORT (I)) * COPA)
IF (WFORT (I) .GE. 0.0) XSTOT (I) = 1.0 * 20
C
ZSSC IS A PERCENTAGE OF SP (I) FOR ACTIVATING CONTROL — CDMOC (I) IS USE
ZSSC = 0.6 * CDMOC (I) * SP (I)
IF (XSTOT (I) .LE. (ZSSC - S (I, I)) .OR. RESA (I) .EQ. 1.0) SA (I) = 1.0
PESA (I) = SA (I)
IF (SA (I) .EQ. 0.0 .OR. HAMM (I) .EQ. 1.0) GO TO 130
WLFOR (I) = -WFORT (I)
VELDEC = ((WLFOR (I) + WLFOPR) / 2.0 * (WLFOR (I) - WLFOPR)) / (AMASS * RFDSLP (I))
WRITE(6,121) TIME, WLFOR(I), VELDEC
121 FORMAT(50H ACTIVE CONTROL INITIATED... TIME, WLFOR, VELDEC = ,
  1 3E13.5)
HM(I) = 1.
130 IF(S(1,I) .GT. 0.0) ISET(I) = 0
IF(HMM(I), EQ, 0.) GO TO 451
IF(-WFOR(T(I)) .GT.(WLFOR(I)+EPSIL0(I))) DF(I) = (WLFOR(I)+
  & EPSIL0(I)) - (-WFOR(I))
IF(-WFOR(T(I)) .LT.(WLFOR(I)-EPSIL0(I))) DF(I) = (WLFOR(I)-
  & EPSIL0(I)) - (-WFOR(I))
IF(-WFOR(T(I)) .LE.(WLFOR(I)+EPSIL0(I)).AND.
  & -WFOR(I) .GE.(WLFOR(I)-EPSIL0(I)))
    457,456
457 IF(S(1,I) .LE. 0.0) GO TO 456
453 IF(WFOR(T(I)) .GT. 0.0 .AND. OSVNC(I) .LT. 0.0) 454,455
454 DF(I) = WLFOR(I) - (-WFOR(I))
GO TO 456
455 DF(I) = 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 X(I) = (DF(I) + DELTX1(I)) * XKA(I)
    IF(GNR .EQ. 1. .AND. XMA(I) .GT. 0.) XMA(I) =
    XMA(I) * SORT( (PGAI(I)-PGLAC(I))
    X/(PGAHAC(I)-PGAI(I)))
C
NOTE: SUBROUTINE 'FLOZE2' COMPUTES THE FLOWS FROM THE PRESSURES
IN UNITS OF INCHFS.
    P1(I) = PGAI(I)/144.
C
COMPUTATION OF HIGH PRESSURE ACCUMULATOR NITROGEN VOLUME
    VOLANT(I) = VOLANT(I) + OSVN(I) * DELT - OPUMPS(I) * DELT
    PS(I) = (((PGAHAC(I)+PATH)*VOLANI(I)/VOLANT(I)**GAMAN)-PATH)/144.0
IF(PS(I) .GE. 3000.0) 464,465
464 PS(I) = 3000.0
VOLANT(I) = VOLANI(I)
465 VOLAHT(I) = VOLAC(I) - VOLANT(I)
IF(VOLAHT(I) .LE. 0.0) 466,467
466 WRITE(6,1050) TIME
1050 FORMAT(1HC0.//45H ACCUMULATOR OIL VOLUME INSUFFICIENT AT TIME=,E16.8
1  //)
    CALL LGFAR6
    STOP 500
467 CONTINUE
    CALL FLOZE2(PS(I), PR(I), P1(I), XLPSV1(I), XLPSV3(I), RCLSV(I), DSV(I),
    & XSV(I), OS1(I), QS3(I), CSV1(I), CSV3(I), XMU)
    QS1(I) = QS1(I)/1728.
    QS3(I) = QS3(I)/1728.
CONTINUE

OSV1(I) = 0.0
OSV3(I) = 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) GO TO 461, 462

OSVN(I) = OSV(I)

GO TO 463

OSVN(I) = 0.0

IF(SD1(I, I) .LT. 0.0 .AND. PGA1T(I) .LE. -1600.0) PGA1T(I) = -1600.0

CONTINUE

I1 = I1 + 3
I2 = I2 + 4
I3 = I3 + 5

CALL INTEGR(LA(I1), DP1(I1))
CALL INTEGR(LA(I2), GD1(I2))
CALL INTEGR(LA(I3), OSV(I3))

C*******************************************************************************
C*******************************************************************************

IF(SD1(I, I) .LT. 0.0 .AND. FT(I) .LE. ABS(FORSST(I))) FORSST(I) = FT(I)
AA(I) = (FT(I) + FORSST(I)) / MASS(I)
SD2(I, I) = SR(I) + AA(I) - GZ(I)

HT1 = HT
IF(SD1(I, I)) 76, 77, 78

TTIME = S(I, I) / ABS(SD1(I, I))

HT1 = TTIME
GO TO 77

GO TO 79

77 CONTINUE

IF(S(I1, I) .GT. 0.0) GO TO 50
WRITE(6, 49) I1, S(I1, I)

FORMAT(58X, 4H - ES(I1) = E15.7)

S(I1, I) = 0.5*ES(I)
SD1(I, I) = -1.0*E-10
SD2(I, I) = 1.0*E-10

50 IF(S(I1, I1) .LT. ES(I)) GO TO 51
IF(S(I1, I1) .LT. (SD1(I, I1) - ES(I))) GO TO 55
IF(S(I1, I1) .GT. (SR(I) + ES(I))) GO TO 52
WRITE(6, 53) I1, S(I1, I)

53 FORMAT(58X, 4H EXCEEDED IN ALGEP/
C58X, 2HS(I1, I1) = E15.7)

S(I1, I) = 0.5*ES(I)

52 IF(SD1(I, I1) .GT. 0.0) SD1(I, I1) = 0.
IF(SD2(I, I1) .LT. 0.0) GO TO 55
SD2(I, I1) = 0.
GO TO 55

51 IF(SD2(I, I1) .LT. 0.0) GO TO 30
IF(SD1(I, I1) .LT. 0.0) SD1(I, I1) = 0.
GO TO 55
CONTINUE
I2=2*I+NSTRUT-1
I1=I2+1
CALL INTEG(LA(I2),SD2(I,1))
CALL INTEG(LA(I1),SD1(I,1))
C RE-CHECK SHOCK STRUT FORCE FOR RE-CONDITIONED FULLY EXTENDED STATE
IF(SD1(I,1).EQ.0.0 .AND. FT(I).LT.ABS(FORSSST(I))) FORSSST(I)=-FT(I)
TMP(I)=RZERO(I)-DELT(A(I)
IF(CASK(I).GT.1.E-10)GO TO 200
IF(INDNS. .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)*SORT(FTRY(I)*FTRY(I)+FTRX(I)*FTRX(I))
MA(I)=SIGN(MA(I),-VAYLE(I)-OMET(I,I)*TMP(1))
201 AMA(I)=MA(I)
IF(IB(I).NE.(-1))GO TO 48
OMETO1(I,1)=0.
GO TO 21
48 TMP(I)=0.
IF(OMET(I,1).NE.0.0)TMP(I)=OMET(I,1)/ABS(OMET(I,1))
OMETO1(I,1)=(MA(I)-MP(I)*TMP(1))/(NTIRES(I)*MOMENT(I))
21 CALL INTEG(LA(I),OMETO1(I,1))
C******************************************************************************
65 CONTINUE
C CALCULATION OF FTRA, FTRB, AND FTRC
SFTRY=0.
SFTRX=0.
SFTRZ=0.
DO 70 I=1,NSTRUT
SFTRY=SFTRY+FTRY(I)
SFTRX=SFTRX+FTRX(I)
SFTRZ=SFTRZ+FTRZ(I)
70 FTRA=RL(1,1)*SFTRY+RL(1,2)*SFTRY+RL(1,3)*SFTRZ
FTRB=RL(2,1)*SFTRY+RL(2,2)*SFTRY+RL(2,3)*SFTRZ
FTRC=RL(3,1)*SFTRY+RL(3,2)*SFTRY+RL(3,3)*SFTRZ
C CALCULATION OF MTX, MTY, AND MTZ
SMTRY=0.
SMTRY=0.
SMTRY=0.
DO 75 I=1,NSTRUT
SMTRY=SMTRY+MTRY(I)
SMTRY=SMTRY+MTRY(I)
SMTRY=SMTRY+MTRY(I)
75 MTX=RL(1,1)*SMTRY+RL(1,2)*SMTRY+RL(1,3)*SMTRY
MTY = RL(2,1) * SMTRX * RL(2,2) * SMTRY * RL(2,3) * SMTRZ
MTZ = RL(3,1) * SMTRX * PL(3,2) * SMTRY * PL(3,3) * SMTRZ

C CALCULATION OF FXM, FYM, FZM, LM, MM, AND NM

FXM = 0.
FYM = 0.
FZM = 0.
LM = 0.
MM = 0.
NM = 0.
BFX = 0.
BFY = 0.
BFZ = 0.
BLM = 0.
BNM = 0.
PNM = 0.
DO 82 I = 1, NSTRUT
  DFXM = 0.
  DFYM = 0.
  DFZM = 0.
  DLM = 0.
  DNM = 0.
  DNM = 0.
  DO 14 IL4 = 1, NMODE
    NBB = (IL4 - 1) * NSTRUT + I
    DFXM = DFM - MASS(I) * SXMOD(NBB) * GQD2(IL4)
    DFYM = DFM - MASS(I) * SYMOD(NBB) * GQD2(IL4)
    DFZM = DFM - MASS(I) * SXMOD(NBB) * GQD2(IL4)
    NRH = (I - 1) * NMODF + IL4
    DLM = DLM - MASS(I) * GFORC2(NBH) * GQD2(IL4)
    DNM = DNM - MASS(I) * GFORC3(NBH) * GQD2(IL4)
    14 DNM = DNM - MASS(I) * GFORC4(NBH) * GQD2(IL4)
    BFX = BFX + DFM
    BFY = BFY + DFM
    BFZ = BFZ + DFM
    BLM = BLM + DLM
    BNM = BNM + DNM
    TNM = TM + TMP1(A31(I))
    FM = FM + TMP1(A33(I))
    LM = LM + TMP1(A11(I)) * RX(I)
    MM = MM + TMP1(RCGX(I))
    NM = NM + TMP1(A13(I)) * RX(I)
    FXM = FXM + BFX + FTRA
    FYM = FYM + BFY + FTRB
    FZM = FZM + BFZ + FTRC
    LLM = LLM + LLM + MTX
    MM = MM + BNM + MTY
    NM = NM + BNM + MTZ

C ENTRY SETUP
SUBROUTINE PHLOZ2(PS,PP,X,OC,LAP1,LAP3,RCL,D,COEF1,COEF3,MU,OTOLEP,
& NITER,P1,Q1,Q3)

'PHLOZ2'..................................................R. D. EDSON

THIS SUBROUTINE CALCULATES THE STEADY-STATE CHAMBER PRESSURE (P1) AND FLOW RATES (Q1 & Q3) FOR A TWO-WAY NONSYMMETRICAL SPool VALVE WITH RECTANGULAR WINDOW SLOTS, GIVEN THE STROKE (X) AND THE LOAD FLOW (OC). THE PARAMETERS REQUIRED IN THE 'CALL' STATEMENT ARE THE SAME AS DESCRIBED IN SUBROUTINE 'FLOZ2', WITH THE FOLLOWING ADDITIONAL PARAMETERS:

QC - THE FLOW RATE TO THE LOAD
OTOLEP - THE TOLERANCE ALLOWED IN CALCULATING FLOW RATES, FOR DETERMINING WHETHER OR NOT THE SOLUTION HAS CONVERGED (.0001 IS TYPICAL)
NITER - THE NUMBER OF ITERATIONS REQUIRED TO CONVERGE TO A SOLUTION (INTEGER)
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.

P1A=PR
P1R=PS
CALL FLOZE2(PS,PR,P1A,P1R,P1B,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+OC
GO TO 50
51 P1=(PS+PR)/2.
Q1=0.
Q3=0.
GO TO 400

50 P1=PFN(P1A,P1B,Q3A,Q1A,Q1B,Q3B)
CALL FLOZE2(PS,PR,P1A,P1R,P1B,LAP1,LAP3,RCL,D,X,Q1A,Q3A,COEF1,COEF3,MU)
Q3=Q3+OC
IF(FLAG<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. OTOLER) 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>0.) GO TO 300
GO TO 50
300 CONTINUE
Q3=Q3-OC
400 RETURN
END
SUBROUTINE FLOZE2(PS,PR,P1,LP1,LP3,RL,D,X,Q1,Q3,COEF1,COEF3,MU)

'TFLOZE2'..................................................R. D. EDSON

THIS SUBROUTINE CALCULATES THE STEADY-STATE FLOW RATES (Q1 AND Q3) FOR A TWO-WAY NONSYMMETRICAL SPOOL VALVE WITH RECTANGULAR WINDOW SLOTS, GIVEN THE LOAD CHAMBER PRESSURE (P1) AND STROKE (X). THE PARAMETERS REQUIRED IN THE 'CALL' STATEMENT ARE AS FOLLOWS:

X = VALVE STROKE
P1 = PRESSURE IN CHAMBER 1 (TO LOAD)
Q1 = FLOW RATE FROM SUPPLY LINE TO CHAMBER 1
Q3 = FLOW RATE FROM CHAMBER 1 TO RETURN LINE
PS = SUPPLY PRESSURE
PR = RETURN PRESSURE
LP1 = OVERLAPPED OR UNDERLAPPED LENGTH BETWEEN THE SPOOL AND SLEEVE AT NULL, FOR FLOW Q1. A POSITIVE NUMBER IS USED FOR OVERLAP, A NEGATIVE NUMBER FOR UNDERLAP.
LP3 = OVERLAPPED OR UNDERLAPPED LENGTH BETWEEN THE SPOOL AND SLEEVE AT NULL, FOR FLOW Q3. A POSITIVE NUMBER IS USED FOR OVERLAP, A NEGATIVE NUMBER FOR UNDERLAP.
RL = RADIAL CLEARANCE BETWEEN THE SPOOL AND SLEEVE
D = DIAMETER OF SPOOL
COEF1 = FLOW COEFFICIENT OF ORIFICE 1 (SUPPLY TO CHAMBER 1)
        = CD*W1*SORT(2.*GC/RHO)
COEF3 = FLOW COEFFICIENT OF ORIFICE 3 (CHAMBER 1 TO RETURN)
        = CD*W3*SORT(2.*GC/RHO)
WHERE W1 = TOTAL WINDOW WIDTH OF ORIFICE 1
W3 = TOTAL WINDOW WIDTH OF ORIFICE 3
CD = DISCHARGE COEFFICIENT
GC = GRAVITATIONAL ACCELERATION CONSTANT
RHO = DENSITY OF HYDRAULIC FLUID
MU = VISCOSITY OF HYDRAULIC FLUID, CENTIPOISE

THE METHOD OF SOLUTION UTILIZES THE TURBULENT ORIFICE EQUATION AND THE EQUATION FOR FULLY-DEVELOPED LAMINAR FLOW THROUGH AN ANNULUS, WITH FULL ECCENTRICITY ASSUMED. FOR ORIFICE OPENINGS WHERE SOME OVERLAPPED LENGTH EXISTS, THE PROCEDURE IS TO CALC-
ORIGINAl PAGE IF
OF POOR QUALITY

CULATE THE FLOW RATE BY BOTH EQUATIONS, AND THEN USE THE ONE
THAT GIVES THE SMALLEST ABSOLUTE VALUE AS THE ANSWER. FOR
OPENINGS WHERE NO OVERTAPPED LENGTH EXISTS, ONLY THE TURBULENT
ORIFICE EQUATION APPLIES.

C

IMPLICIT REAL(L,M)
Q12(T1,T2)=SIGN(1.,(T1-T2))*SQR(TABS(T1-T2))
Q34(T3,T4)=4.5E06*(T3-T4)*D*RCL**3/MU

C

*************************************************** CALCULATE Q1 ***************************************************
X2=LAP1-X
X4=SQR(T2*X2**2+RCL**2)
IF(LAP1.LE.0.) GO TO 99

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. APS(Q1)) Q1=Q1L
GO TO 20
65 Q1=X4*COEF1*Q12(PS,P1)
GO TO 20

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. APS(Q1)) Q1=Q1L

C

*************************************************** CALCULATE Q3 ***************************************************
20 X1=LAP3+X
X3=SQR(T2*X1**2+RCL**2)
IF(LAP3 .LE. 0.) GO TO 199
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

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

STATEMENTS FOR THIS SUBROUTINE OBTAINED BY PHONE BY
JOHN R. MCGHEE ON 2/1/77.

C

IF (X .GE. XMAX) GO TO 10
IF (X .LE. XMIN) GO TO 20
GO TO 30
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 ACOBBLK was inserted before the LOGICAL statement and the call to the landing gear routine was changed as follows.

CALL LGFAR2
IF (IABS(INOLG).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, LGEAR3, 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) /5HVL015T,5HVL025T,5HVL035T,5HPG015T,5HPG025T,
* 5HPG035T,5HGA045T,5HC050/
DATA(ACOVAR2(I),I=1,8) /2HQQ,4HVCUM,6HBL050RT,6H05050RT,5HFF0RT,
* 6HF0RCHT,6HF0AHT5T,6HF0NHST/
DATA(ACOVAR3(I), I=1,8) /5HCC050RT,5HF050RT,6HF05050ST,6HXX05050,3HXS00,
* 5HDELT5,6HDELTX1,6HDTX10/
DATA(ACOVAR4(I), I=1,8) /2HDF,3HDP1,2HPS,2H050,3HQS05,4HQS05,4HQS051,
* 4HQS053/
DATA(ACOVAR5(I), I=1,8) /6H02550L0,3H005050,6HDMTA050,5HXT0505,5HIC0500,
* 4HE05UP,5H050RT,5H050050/
DATA(ACOVAR6(I), I=1,8) /5HII0505050,5HII0505050,5HXX0505050,5HXX0505050,
* 5H0505050,5H0505050,
DATA(ACOVAR7(I), I=1,8) /5HVI0505050,6HVL0500RT,6HI0505050,4HZ0505050,
* 4HC0505050,5H0505050,5H0505050,5H0505050/
The actual output is accomplished by the following code which was added three lines below the statement labeled 32.

```fortran
IF(IAABS(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), GAMAI(I), CEOFO(I))
33 CONTINUE
CALL STFL(2,8,ACOVAR2)
DO 34 I = 1, NSTRUT
  CALL STOVAR(8, COD(I), VCUM(I), BLOOPT(I), RFOPT(I), FFOPT(I),
             * FORCHT(I), FOAHST(I), FONHST(I))
34 CONTINUE
CALL STFL(2,8,ACOVAR3)
DO 35 I = 1, NSTRUT
  CALL STOVAR(8, CFFORP(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), OSV(I), OSV1(I),
             * QSV3(I))
36 CONTINUE
CALL STFL(2,8,ACOVAR8)
DO 37 I = 1, NSTRUT
  CALL STOVAR(8, WFOR(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, XSVDD(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)),
             * ENUF(I), WLFOR(I), FLOAT(IOPCD(I)))
39 CONTINUE
CALL STFL(2,8,ACOVAR6)
DO 40 I = 1, NSTRUT
  CALL STOVAR(8, FLOAT(IIXSVH(I)), FLOAT(IIXSVL(I)), FLOAT(IIXSVH(I)),
* FLOAT(IXSVL(I)), FLOAT(IPASS(I)),
* FLOAT(NAC(I)), RESA(I), SA(I))
40 CONTINUE
CALL STFL(2,8,ACOVAR7)
CALL STOVAR(8, VELDFC, WLFGPR, 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(INDG) .NE. 3) GO TO 6
I1 = I+3*NSTRUT
I2 = I+4*NSTRUT
I3 = I+5*NSTRUT
CALL UPDAT(1,LA(I1),PGAI1(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I2),VCUM(I),DU,DU,DU,DU)
CALL UPDAT(1,LA(I3),OSVCI(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),DELY1(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
INXO = 0
JBC = 0

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

DELP = 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 IRAF55), MSG(58), IDATA(646)
EQUIVALENCE (MSG(1), SYM), (MSG(2), OP)
EQUIVALENCE (MSG(3), IRAF1)), (MSG(58), INC)
EQUIVALENCE (DATA(1), IDATA(1))
INTEGER COMMA, POINT, E, BLANK
DATA REMARK, COMMA, BLANK, POINT, E, ENDACT, MINUS, AINT/
1 E, REM, IR, 1R, 1R, 1R, 1R, 1R, 6HENDACT, 1R, 3HINT/
100 CONTINUE
 1 FORMAT(A6, 1X, A3, 1X, 55R1, 1I)
 2 FORMAT(18X, A6, 1X, A3, 1X, 55R1, 1I)
 3 FORMAT(2OHOERRORP, THE SYMBOL **, AC, 1 26H** IS NOT IN THE DIRECTORY/1H )
 4 FORMAT(44HOERRORP, ILLEGAL CHARACTER IN NUMERIC FIELD **, 1R, 2H**/)
READ(5, 1) SYM, DP, 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.COMPA) GO TO 140
IF(IPA(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 170
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) NP = NP + 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(ISIGN.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 DIRECT
COMMON/ACTDIR/NAMEN(71),LOC(71)
DATA NAMEN/6HAMUH, 6HAPINT, 6HAREA1, 6HAREA2, 6HAREAM, 6HAREDO3, 6HETA, 6HBETA, 6HBLMU, 6HBLMU,
6HBUMU, 6HCMOC, 6HCMOKE, 6HCDSV, 6HCD3,
6HCFFOR, 6HDELTA, 6HDIOTA, 6HDSV, 6HEPSILO,
6HEPERSOL, 6HEPSSLP, 6METASV, 6HGMAMH, 6HGMAMH,
6HGNP, 6HIDEACT, 6HKAPT, 6HOMRUN, 6HAPTM,
6HPGAHAC, 6HPGALAC, 6HPGAII, 6HPGA2I, 6HPGA3I,
6HPERCNT, 6HOPUMPS, 6HRCLS3, 6HRHOH, 6HTAUF,
6HTC1, 6HTC2, 6HTC3, 6HTC4, 6HVOLACI,
6HVOLANI, 6HVOLII, 6HVOLI, 6HVOLIII, 6HVOLI,
6HWC1, 6HWSV, 6HWSV1, 6HWSV3, 6HWLFOR,
6HWLFR, 6HXBIAS, 6HXXDMAX, 6HXXDMIN, 6HXKA,
6HXKF, 6HXKSV, 6HXLPSV, 6HXLPSV3, 6HXSCOM,
6HXSTHR, 6HXSVDMN, 6HXSVDMX, 6HXSVMAX, 6HXSVMIN,
6HZETAC1, 6HZETAC2/
DATA LOC/1, 6, 16, 21, 26, 31, 36, 61, 46,
156, 62, 67, 72, 82, 77, 109, 141, 135, 152,
157, 162, 163, 195, 200, 201, 212, 263, 274, 275,
281, 286, 291, 301, 311, 276, 346, 387, 397, 410,
411, 412, 413, 414, 621, 631, 421, 431, 441, 451,
452, 641, 642, 643, 458, 463, 644, 611, 616, 674,
479, 484, 489, 494, 561, 499, 591, 596, 601, 606,
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.

\texttt{DIMENSION \text{TITLE}(20), \text{BUF}(550), \text{NDIL}(28), \text{TBUF}(550)}

\texttt{1 \text{BMM}(2,550), \text{CMMD}(6), \text{DEPVAR}(5), \text{LINE}(7), \text{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

\texttt{DATA \text{ICOSV}, \text{IFSTOP}, \text{II/11*C/}}

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

\texttt{DATA \text{N19/19/}}

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

\texttt{IF(\text{ISUM1.NE.0}) \text{WRITE}(13) \text{N19,ISUM1,DAT3,OP17,ACOVAR8(1),}}

\texttt{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.

\texttt{1 \text{OMET}(1,1), \text{WFOPT}(1), \text{OSVCU}(1), \text{CSV}(1), \text{PGA1T}(1), \text{PGA2T}(1)}}

\texttt{1 \text{OMET}(1,2), \text{WFOPT}(2), \text{OSVCU}(2), \text{CSV}(2), \text{PGA1T}(2), \text{PGA2T}(2)}}

\texttt{1 \text{OMET}(1,3), \text{WFOPT}(3), \text{OSVCU}(3), \text{CSV}(3), \text{PGA1T}(3), \text{PGA2T}(3)}}
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(RAI,INC,PLANK)
```

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 ACOBLOK comdeck, the ACTIVE common block was modified by replacing FOAHST(5) with FWORK(5), replacing PGA1T(5) with PGA1T(5), replacing ZZSSC with ZZSSC(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

```
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),AIYRS)

DATA DSTOP,FSTOPK /0.004,0.0/
DATA ENUP,FSTOP,HMM,QSVN,RESA,SA /30*0.0/
DATA ICOSV,IFRI,FSTOP,IPASS /20*0/
DATA IDPCO,ISET /10*1/
DATA IXSVH,IXSVL,IXSVH,IXSVL /20*0/
```
the definition of OMRUN was changed to,

\[ \text{OMRUN} = \text{ERDEG} \times 0.01745329 \]

and the following definitions were inserted in the 100 loop.

\[
\text{INDINT}(i) = 1 \\
\text{VMASS}(i) = \text{AMASS}
\]

Additionally, the line below statement number 80 was changed to

\[
\text{PGAII}(i) = \text{PGAII}(i)
\]

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

\[
\text{EQUIVALENCE} (\text{DM5}(16), \text{GREFF}) \\
\text{DATA} \text{IFRI}/5*0.0/, \text{FSTOPK}/0.0/, \text{FSTOP}/5*0.0/, \text{DSTOP}/0.004/ \\
\text{DATA} \text{SA}, \text{RESA}, \text{HMM}, \text{IXSVL}, \text{IXSVH}, \text{IXSVL}, \text{IXSVH}, \text{IPASS} \\
1/15*0., 25*0/ \\
\text{DATA} \text{ENUP}/5*0.0/ \\
\text{DATA} \text{ISET}, \text{IOPCO}, \text{OSVN}/10*1, 5*0.0/
\]

and were replaced with

\[
\text{EQUIVALENCE} (\text{DM5}(16), \text{GREFF}), (\text{DM2}(27), \text{AXP7F}), (\text{DM2}(28), \text{AY77F}), \\
* (\text{DM2}(29), \text{AYP7F}), (\text{DM2}(30), \text{AY77F}), (\text{DM2}(31), \text{AZP7F}), (\text{DM2}(33), \text{AZ77F}), \\
* (\text{DM1}(37), \text{AIYY85}) \\
\text{DIMENSION} \text{INDEACT}(5), \text{IPSTOP}(5), \text{AIC}(5), \text{PGAII}(5) \\
\text{DATA} \text{AIC}, \text{INDEACT}, \text{IPSTOP} /5*0.0, 10*0/
\]
The following new code was inserted beneath statement number 760.

\[
CMA\text{NG} = \frac{A\text{IYYBS} + A\text{MASS} \times P\text{X}(1)}{A\text{IYYPS}}
\]

\[
CMA\text{NG} = 7.9677
\]

\[
V\text{MASS}(1) = \frac{A\text{MASS} \times C\text{MA}\text{NG}}{A\text{MASS}}
\]

\[
E\text{NCG} = 0.5 \times A\text{MASS} \times Z\text{G77F1}(1) \times Z\text{G77F1}(1)
\]

\[
ZD\text{ANT} = Z\text{G77F1}(1) \times O177R \times P\text{X}(1)
\]

The following two lines of code

\[
18 \ \text{IF}(\text{IDEACT.EQ.1}) \ \text{GO TO 56}
\]

\[
19 \ \text{IF}(\text{IDEACT.EQ.2}) \ \text{GO TO 80}
\]

were replaced with

\[
18 \ \text{DO 90 } I=1,\text{INSTRUT}
\]

\[
\text{IF}(\text{INDEACT(I).EQ.1}) \ \text{GO TO 56}
\]

\[
\text{IF}(\text{INDEACT(I).EQ.2}) \ \text{GO TO 80}
\]

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

\[
\text{IF}(I,\text{.EQ.1 .AND. ZD\text{ANT} .GE. V\text{LDEC}) GO TO 90)}
\]

\[
\text{IF}(I,\text{.EQ.1 .AND. ZD\text{ANT} .GE. V\text{LDEC}+Z\text{G77F1}(1) \times \text{TAN(OMRUN)})) \ \text{GO TO 90)
\]

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

\[
\text{INDFACT(I)=1}
\]

\[
56 \ \text{IF}(\text{INDINT(I).EQ.0}) \ \text{GO TO 58}
\]

\[
\text{WLFOR(I)=WLFOR(I)-REDSLSP(I)*HT}
\]

\[
\text{EPSILO(I)=EPSILO(I)+EPSSLP*HT}
\]

\[
58 \ \text{CONTINUE}
\]

\[
\text{INDINT(I)=0}
\]
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.

```fortran
C*** CALCULATION OF THE WING-GEAR INTERFACE FORCE (WFORT)
UNSPRNG = 0.0
DO 66 J = 1, NSTRUT
  IF(GMETD1(J) .NE. 0.0) UNSPRNG = UNSPRNG * MASS(J)
66 CONTINUE

DENSITY = (AXP7F * AXP7F + AYP7F * AYP7F + AZP7F * AZP7F + GREFF) * (MASS - UNSPRNG)
WFORT(I) = DENSITY + FORSST(I)
DO 67 I = 2, NSTRUT
  WFORT(I) = WFORT(I) / (NSTRUT - 1)
67 CONTINUE
C
DO 65 I = 1, NSTRUT
C
C***
```

The line below the statement labeled 210 was replaced with

```fortran
PGAIT(I) = P GAIT1(I)
IF(P GAIT1(I) .LE. -1600.0) P GAIT(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 Q0 were replaced with the following.

```fortran
IF(IMODE(I) .EQ. 0 .AND. DDELTA(I) .LE. 0.0) 112, 113
112 Q0(I) = 0.0
GO TO 109

113 Q0(I) = Q0FGO(I) * (AREMC(I) - APINT(I)) * Q1(PG A1T1(I), PGA2T(I))
IF(Q0(I) .LT. 0.0 .AND. VCUM(I) .LE. 0.0) GO TO 102
  AIC(I) = 0.0
  GO TO 103
102 IF(PG A1T1(I) .LT. PGA2T(I)) GO TO 103
  GO TO 111
111 Q0(I) = 0.0
  VCUM(I) = 0.0
  AIC(I) = 1.0
103 IF(Q0(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.
The two-line definition of ENUP(I) was expanded to

\[
\text{IF(I \neq 1) GO TO 119}
\]

\[
\text{ENUP(I) = 0.5 \times AYYBS \times \frac{Q177R + Q177R + (ENCG/CMASNG)}{ZG77F1(I)/\text{ABS}(ZG77F1(I))}}
\]

*GO TO 120*

119 ENUP(I) = ENCG/(NSTRUT-1)

120 CONTINUE

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

\[
\text{IF(WFOR(I) \geq 0.0 \text{ OR } DDDELTA(I) \lt 0.0) XSTOT(I) = 1.E20}
\]

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.

\[
\text{ZSSC(I) = FWORK(I) \times S8(I)}
\]

\[
\text{IF(XSTOT(I) \leq (ZSSC(I) - S(1, I)) \text{ OR } RESA(I) \lt 0.0) SA(I) = 1.0}
\]

\[
\text{VELDEC = ((WLFOR(I) + WLFOPR) / 2.0 \times (WLFOR(I) - WLFOPR)) /}
\]

\[
\epsilon \quad (AMASS*REDSLP(I))
\]

\[
\text{PS(I) = ((PGAHAAC(I) + PAT) \times (VOLANI(I) / VOLANT(I)) \times GAMA - PATM) / 144.0}
\]

The definitions of QSV1(I) and QSV3(I) below statement number 450 were removed and the following new code was inserted in place of the
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. DDDELTA(I),LE.0,0) GO TO 19
IF(S(I,I),NE.0,0 .OR. AIC(I),NE.1.0) GO TO 20
PGA1T1(I)=PGA1T(I)
PGA1T(I)=PGA1T(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),XSVD(I))
CALL INTEG(LA(I8),XSVDCT(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, 6HEPSLPL, 6HETASV, 6HFWORK, 6HGMAMH,
5 6HGNNR, 6HAPP, 6HOMRUN, 6HPATH,
5 6HETAC1, 6HETAC2, 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 preceding 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 WFOR. 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.

\[ CMASNG = 1+\frac{AMASS*RX(1)*RX(1)}{AIYYPS} \]

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.

\[
\begin{align*}
\text{IF}(J \text{ EQ } 1 \text{ AND } O\text{MFTD1}(1) \text{ NE } 0.0) \text{ UNSPR } &= \text{ MASS}(1) \\
\text{IF}(J \text{ GT } 1 \text{ AND } O\text{MFTD1}(J) \text{ NE } 0.0) \text{ UNSPRNG } &= \text{ UNSPRNG+MASS}(J)
\end{align*}
\]

Below this, the definition of WFOR(1) was changed as follows.
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 CSCMODS, 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),OMED1(1,1),
    *OMFT(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),OMED1(1,2),
    *OMFT(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),OMED1(1,3),
    *OMFT(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),OMED1(1,4),
    *OMFT(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),OMED1(1,5),
    *OMFT(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

**FATAL VARIABLES**

```
COMMON/DIRCOM/DM1(115), ALPHD, DM2(20), AMASS, DM2(147), DCL1, DCM1,
C DCN1, DCL2, DCM2, DCM2, DCL3, DCM3, DCM3, DM5(17), GB7F, DM6(8), GB7F,
C DM7(218), INDSTE(48), PHPID, INDSTE1(23), PSSIPD, INDSTE2(156), THTPD,
C INDSTE3(5), TIME, DM8(287), P177P(2), P177R(2), P177R(2), DM9(4),
C 177P(2), 177R1(2), DM10(4), R177R(2), R177R(2), DM11(48),
C XG77F(2), YG77F1(12), YG77F2(2), YG77F2(12),
C ZG77F(2), ZG77F1(2), ZG77F1(2),
C NSTRUT, MASS(5), RX(5), RY(5), RZ(5), THETA(5), EPDEG, PGR,
C NTIRES(5), ZZERO(5), W(5), DELTA(5), MOMENT(5),
C A2(5), IL, S2L(5), C2L(5), M2S(2), MUS(5),
C CC(5), CE(5), C2C(5), C2E(5), NVGPT, NPP, MB(5), PLT, NDELA,
C ES(5), SB(5), SD21(2), SD22(2), SD23(2), SD24(2), SD25(2),
```

```
COMMON/DIPCOM/
C SD11(2), SD12(2), SD13(2), SD14(2), SD15(2),
C S11(2), S12(2), S13(2), S14(2), S15(2),
C SD21(2), SD22(2), SD23(2), SD24(2), SD25(2),
C SD11(2), SD12(2), SD13(2), SD14(2), SD15(2),
C S11(2), S12(2), S13(2), S14(2), S15(2),
C SD21(2), SD22(2), SD23(2), SD24(2), SD25(2),
C SD21(2), SD22(2), SD23(2), SD24(2), SD25(2),
C OMT011(2), OMT012(2), OMT013(2), OMT014(2), OMT015(2),
C OMT01(2), OMT02(2), OMT03(2), OMT04(2), OMT05(2),
C A1(5), PL(5), DELTA1, DELTA2, DELTA3, DELTA4, DELTA5,
C DDELTA1, DDELTA2, DDELTA3, DDELTA4, DDELTA5, ISTAGF,
C PRSMIN, IPLT, ISDF, ISTPL1, ISTPL2, ISTPL3, ISTPL4, ISTPL5,
C DM14(22), IB(5), DM15(127), INDLC, DM16(107), CASK(44), INDLY,
```
* NMODE, DM0 (10), SXMOD (100), SYMOD (100), SZMOD (100), DM21 (1686),
* GOD2 (20), DDM22 (20), DM21 (20), SLEN1 (5), SLEN2 (5),
* GAM, DDM15 (12), IONWS, 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 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), FDELT (5),
C FTRZ (5), RXD (5), RDXG (5), RXG (5), RDX (5), RDXG (5), RDX (5),
C VTX (5), VTY (5), VTX (5), RZ (5),
C SR (5), SF (5), PSK (5), MVUP (5), MTRX (5), MTRY (5),
C MTRZ (5), MA (5), RG11 (5), RG13 (5), RG31 (5), RG33 (5), IPRT,
C MTRX, MTRY, MTRZ, SFTRX, SFTRY, SFTRZ, FTRA,
C FTRB, FTRC, SMTRX, SMTRY, SMTRZ

C COMMON/HTCOM/HT, HT1, HT2, INDEACT (5)

* CALL ACOBLOK

C REAL MASS, MOMENT, MASS2, MUS, NTIRES, MB
C REAL MVUP, MTRX, MTRY, MTRZ, MA, MTX, MTY, MTL

C DIMENSION IPSTOP (5), AIC (5), PGAIT (5)
C DIMENSION DELTA (5), DDELTA (5), DLGDE (47),
C
* SD2 (2, 5), SD1 (2, 5), S (2, 5)
C
C EQUIVALENCE (DLGDE (1), LA (1))
C EQUIVALENCE (DELTA (1), DELTA (1), DDELTA (1), DDELTA (1))
C EQUIVALENCE (SD2 (1), SD1 (1)), SD1 (1))
C EQUIVALENCE (DM5 (1), ITO)
C EQUIVALENCE (DM5 (1), GPEFF),
C
* (DM1 (37), A1YRS)

C DATA AIC, IPSTOP / 5 * 0.0, 0.5 *

C 01 (T1, T2) = SIGN (1., (T1 - T2)) * SORT (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 (DMRUN, GT, 0.0) GO TO 25
IF (ZDANT, GE, VELDEC) 90, 40
25 IF (ZDANT, GE, VELDEC * XG77F1 (1) * TAN (DMRUN)) GO TO 90
40 WRITE (6, 1014) TIME
1014 FORMAT (1H036H REDUCE CONTROL LIMIT FORCE AT TIME= , E16.8)
INDEACT (I) = 1
C
56 IF (INDINT (I), EQ, 0) GO TO 58
WLFOR (I) = WLFOR (I) - FDSDLPI (I) * HT
EPSILON (I) = EPSILON (I) * EPSSSL * HT
CONTINUE

INDINT(I) = 0
IF(WLFOR(I) .GT. WLFORR) GO TO 90
WRITE(*,1015)TIME

1015 FORMAT (1H027H CONTROL AT WLFORR AT TIME=,F16.8)
INDEACT(I) = 2
80 WLFOR(I) = WLFORR
EPSILON(I) = EPSRLOR(I)
90 CONTINUE

C***********************************************************************C
IF(KAPT(I) .NE. 0) GO TO 210
APINT(I) = 0.0
210 CONTINUE
PGAIT(I) = PGA1T(I)
IF(PGA1T(I) .LE. -1600.0) PGA1T(I) = -1600.0
VOLT(I) = VOLT(I) - (AREA(I) - APINT(I) * S(I, I))
VOLT2(I) = VOLT3(I) + AREA3(I) * S(I, I)
VOLT3(I) = VOLT2(I) - (AREA3(I) - AREA1(I) + APINT(I)) * S(I, I) + (VOLT3T(I) * X - VOLT3(I)) * VCUM(I)
PGA2T(I) = AP2T0(I) * ((VOLT2(I) / VOLT2T(I)) ** GAMMA) * PATH
IF(SD1(I, I) .EQ. 0.0) GO TO 104
PGA3T(I) = ((COEF3(I) * AREA3(I)) ** 2 * PGA2T(I) - SD1(I, I) / ABS(SD1(I, I))) * (SD1(I, I) * AREA3(I)) ** 2
E/((COEF3(I) * AREA3(I)) ** 2)
GO TO 105
104 PGA3T(I) = PGA2T(I)
105 IF(PGA1T(I) .GT. PGA2T(I)) GO TO 106
GO TO 107
106 GAMAH = RHOH * GREFF * (1.0 + (PGA1T(I) * 3.04E-08) - (PGA1T(I) ** 2 * 2.72E-15))
GO TO 108
107 GAMAH = RHOH * GREFF * (1.0 + (PGA2T(I) * 3.04E-08) - (PGA2T(I) ** 2 * 2.72E-15))
108 IF(PGA1T(I) .GT. PGA2T(I)) COEFO(I) =
* COMDC(I) * SORT(Abs(Z * GREFF/GAMAH(I)))
IF(PGA2T(I) .GT. PGA1T(I)) COEFO(I) =
* COMDC(I) * SORT(Abs(Z * GREFF/GAMAH(I)))
IF(IMODE(I) .NE. 0 .OR. DR. DDELT(A(I).GT.0.0) GO TO 113
QD(I) = 0.0
GO TO 109
C
113 QD(I) = COEFO(I) * (AREM0(I) - APINT(I)) * Q1(PGA1T(I), PGA2T(I))
IF(QD(I) .LT. 0.0 .AND. VCUM(I) .LE. 0.0) GO TO 102
AIC(I) = 0.0
GO TO 103
102 IF(PGA1T(I) .LT. PGA2T(I)) GO TO 103
QD(I) = 0.0
VCUM(I) = 0.0
AIC(I) = 1.0
103 IF(QD(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(DELT(A(I) .LE. 0.0 .AND. TIME .GT. DELT) GO TO 101
GO TO 110
101 FFOFT(I)=0.0
GO TO 140
110 CONTINUE
C COMPUTE STRUT AXIAL PENDING FRICTION FORCE
BLFORT(I)=FONHST(I)*(SLEN2(I)-S(1,I))/(SLEN1(I)+Z(S1,I)+1.0)
BUFORT(I)=FONHST(I)*(SLEN2(I)-S(1,I))/(SLEN1(I)+Z(S1,I))
FFORT(I)=B mixture*(ABS(BIFORT(I))+BLMU(I)*ABS(RLafort(I)))
140 CONTINUE
C COMPUTE SHOCK STRUT CHARGING FORCE
IF(S1(I,I) .GT. 0.0) GO TO 142
FORCHT(I)=SFALT(I)*AREA1(I)+PGA2T(I)*(AREA2(I)-AREA1(I))-PGA3T(I)
* AREA3(I)+FFORT(I)+CFFORT(I)
142 CONTINUE
C COMPUTE NORMAL AND AXIAL HUB TO SHOCK STRUT FORCES AT HUB
FONHST(I)=SORTX((1))**2+FDY(I)**2)-MASS(I)*GFF*SIAP+SFBOT
IF(ABS(FI(T(I))) .LE. FORCHT(I) .AND. S(1,I) .EQ. 0.0) GO TO 150
GO TO 801
150 CONTINUE
C COMPUTE NORMAL AND AXIAL HUB TO SHOCK STRUT FORCES AT HUB
FONHST(I)=SORTX((1))**2+FDY(I)**2)-MASS(I)*GFF*SIAP+SFBOT
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
S1(I,I)=0.0
IPSTOP(I)=0
GO TO 902
904 CONTINUE
C COMPUTE NORMAL AND AXIAL HUB TO SHOCK STRUT FORCES AT HUB
IF(IFSTOP(I) .NE. 0) GO TO 906
DSTOP=S(1,I)
FSTOPK=2.0*MASS(I)*SD1(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
IF(PGA1T(I) .LE. (PGA1T(I)+500.0) .AND. 
PGA1T(I) .GT. (PGA1T(I)-500.0)) GO TO 158
GO TO 159
158 IF(ABS(FT(I)).LE.FORCHT(I) .AND. S(I,I).EQ.0.0) GO TO 500
159 IF(S(I,I).GT.0.0) GO TO 479
GO TO 471
470 FFORT(I)=FFFOR(I)
CFFOR(I)=CFFOR(I)
471 FORSST(I)=-(PGA1T(I)-PGA2T(I))*(AREA1(I)-APINT(I))
+PGA2T(I)*AREA2(I)
+PGA3T(I)*AREA3(I)+(FFORT(I)
+CFFOR(I))*DMTANH(I)+FSTOPI(I))
500 IF(INDFLX.GE.1)GO TO 295
IF(IMODE(I).EQ.0) GO TO 297
ISTROK(I)=1
289 IF(S(I,I).GT.0.0) GO TO 295
IF(IPD(I).EQ.1) GO TO 295
IF((PGA1(I)-1000.0).LT.PGA1T(I) .AND. 
PGAI(I).LT.(PGA1(I)+1000.0)) GO TO 299, 299
299 IF(XVAlVF(I) .NE. 0.0) GO TO 311
IF(IPASS(I).EQ.1) GO TO 296
XVALVE(I)=XSV(I)+XMA11(I)+XBIAS(I)
IPASS(I)=1
GO TO 294
298 IF(ICOSV(I).EQ.1) GO TO 291
IPOC(I)=0
IF(XSV(I).LT.0.002 .AND. XSV(I) .GT. -0.002) GO TO 291, 295
291 IF(SD2(I,I).LE.0.0 .AND. ICOSV(I) .EQ. 1) GO TO 311
IF(IPOC(I).EQ.1) GO TO 295
IF(SD2(I,I).LE.0.0 .AND. ICOSV(I).EQ.1) GO TO 295
IF(IXSVL(I).EQ.1) GO TO 294
XVALVE(I)=XVALVE(I)+XSSV(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)+XSSV(I)*DELT*PERCNT(I)
IF(XVALVE(I) .LT. 0.1) GO TO 294
XVALVE(I)=0.1
IXSVH(I)=1
GO TO 294
294 CONTINUE
DLYX10(I)=0.0
ICOSV(I)=1
296 IF(WFORT(I).GT.0.0 .AND. S(I,I) .LE. 0.0) GO TO 410
311 IF(NAC(I) .EQ. 1) GO TO 307
IF(IXSVH(I).EQ.1) GO TO 305
XVALVE(I)=XVALVE(I)+XSSV(I)*DELT*PERCNT(I)
IF(XVALVE(I) .LE. 0.0) GO TO 305
XVALVE(I)=0.0
IXSVH(I)=1
GO TO 400
307 IF((IYSVL(I) .EQ. 1) GO TO 308
XVALVE(I)=XVALVE(I)+XSVDMX(I)*DELT*PEPCNT(I)
IF(XVALVE(I) .GE. 0.0) GO TO 308
XVALVE(I)=0.0
IXSVL(I)=1
CONTINUE
400 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)
XMAII(I)=XMA(I)
XSV(I)=XSV(I)*XMAII(I)+XBIAS(I)
CALL LIMITS(XSV(I),XVDOT(I),YSYM(I),YSVMIN(I))
IPASS(I)=0
IXSVL(I)=0
IXSVH(I)=0
IXSVL(I)=0
IOPCO(I)=1
CALL PHLOZ2(PS(I),PR(I),XSV(I),QC(I),XLPSV1(I),XLPSV3(I),PCLS(I),"CVSV(I),CSV3(I),XMSV3(I),XMU,OCLER,NITER,P1(I),PS1(I),Q3(I))
IF(IYSTRK(I) .EQ. 1 .AND. S(I,I) .GT. 0.0) IOPCO(I)=0
295 IF(IYSTRK(I) .EQ. 1) GO TO 295
I2 = 2*I+INSTRUT-1
CALL INTEG(LA(I2),SD2(I,I))
CALL INTEG(LA(I1),SD1(I,I))
IF(IMODE(I) .EQ. 0) GO TO 450
C
FNUP(I) = 0.5*AIYYBS*O177P(I)*QI77P(I)+(ENC/GMNASG)*
(ZG77F1(I)+ABS(ZG77F1(I))
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. DELT(I).LT.0.0) XSTOT(I)=1.020
ZSSC IS A PERCENTAGE OF SB(I) FOR ACTIVATING CONTROL-CMOC(I) IS U
ZSSC(I)=FWORK(I)*SB(I)
IF(XSTOT(I) .LE. (ZSSC(I)-S(I,I)) .OR. PESA(I).EQ.1.0) SA(I)=1.0
RESA(I)=SA(I)
IF(S(I,I).EQ.0.0 .OR. HMM(I).EQ.1.) GO TO 130
WFOR(I)=-WFORT(I)
C
VELDEC=((-WFORT(I)+WFOPR)/2.*WFORT(I)-WFOPR))/
& (VMASS(1)*REDSLP(I))
WRITE(6,121)TIME, WLFOR(I), VELDEC
121 FORMAT(50H ACTIVE CONTROL INITIATED...TIME, WLFOR, VELDEC = ,
1 3F13.5)
HMM(I)=1.
130 IF(S(I,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(I,I) .LE. 0.0) GO TO 456
IF(WFORT(I) .GT. 0.0 .AND. QSVCU(I) .LT. 0.0)454,455
454 DF(I)=WLFOP(I)-(WFORT(I))
GO TO 456
455 DF(I)=0.0
456 DELTY(I)=S(I,I)-YSCOM(I)
IF(S(I,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. 1.0 .AND. XMA(I) .GT. 0.) XMA(I)=
1 YMA(I)*SORT((PGA1T(I)-Pgalc(I))
& (PGA1T(I)-Pgalc(I))}
C NOTE: SUBROUTINE 'FLOZ2' COMPUTES THE FLOWS FROM THE PRESSURES
C IN UNITS OF INCHES.
E P1(I)=PGA1T(I)/144.
C COMPUTATION OF HIGH PRESSURE ACCUMULATOR NITROGEN VOLUME
C AND ACCUMULATOR PRESSURE
VOLANT(I)=VOLANT(I)+OSVNI(I)*DELT-OPUMPS(I)*DELT
PS(I)=(((PGA1HAC(I)+PAM)*VOLANI(I)/VOLANT(I))*GAMA)-PAM/144.0
IF(PS(I) .GE. 3000.0)1464,1465
1464 PS(I)=3000.0
VOLANT(I)=VOLANI(I)
1465 VOLANT(I)=VOLACI(I)-VOLANT(I)
IF(VOLANT(I) .LE. 0.0)1466,1467
1466 WRITE(6,1050)TIME
1050 FDPMAT(1HC/45H ACCUMULATOR OIL VOLUME INSUFFICIENT AT TIME=,E16.8
1 //
CALL LGEAR
STOP 500
1467 CONTINUE
CALL FLOZ2(PS(I),PR(I),P1(I),YLSV1(I),YLSV3(I),RCLS(I),OSV(I),
6 YSV(I),OS1(I),OS3(I),CSV1(I),CSV3(I),XMU)
OSV1(I)=OS1(I)/1728.
CONTINUE
OSV(I) = OSV(1) - OSV(3(I)
IF(OSV(I) .LT. 0.0) NAC(I) = 1
IF(OSV(I) .GT. 0.0) NAC(I) = 2
IF(NAC(I) .NE. 2) GO TO 462
OSV(N(I) = OSV(I)
GO TO 463

OSV(N(I) = 0.0
463 IF(SDI(I,I) .LT. 0.0 .AND. PGA(I) .LE. -1600.0) PGA(I) = -1600.0
RETURN
END

In the comdeck AC0BLK, 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)
EQUVALENCE (S(1,1),S(1,1))
EQUVALENCE (DMP15(I),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).

\begin{verbatim}
INDFACT(I) = 0
IF(ITO.EQ.1) INDFACT(I) = 2
\end{verbatim}

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.

\begin{verbatim}
EQUIVALENCE (DM15(I),IT0)
\end{verbatim}

The call to LGEA3C was replaced with the following code,

\begin{verbatim}
IF(INITSW.EQ.1) GO TO 16
\end{verbatim}

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

\begin{verbatim}
CALL LGEA3C
\end{verbatim}

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

\begin{verbatim}
18 CALL ACTNG
   DO 274 I=2,NSTRUT
\end{verbatim}

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.

\[ \text{& VMASS(I)*REDSLPI(I))} \]

The following code was inserted below statement number 274,

\[ \text{DO 65 I=1,NSTRUT} \]

the SETUP entry point was replaced with the following,

\[ \text{16 CONTINUE} \]
\[ \text{INITSW = 0} \]

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,

\[ \text{DATA N20 /20/} \]
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),ETADIS,
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, WFOR /25*0.0/
DATA QC, OSV1, OSV3, QSVCU, QTOLEP /25*0., 0.0001/
DATA DCON, MTANH, DF, DPI, DELTX, DELTX1, DLTX1D /6*1.0, 25*0.0/
DATA XSVDOT, XSVDD, XSVDDD /15*0.0/
DATA REDSLP, SBFOT, VCVU, VELDEC /5*100000., 0.0, 10*0.0/
DATA XMA1, XMA2, XMA3, XMA4, XMA5, XMA7, XMA9, XMA10 /40*0.0/
DATA XMA, XMA5, XMA8, XMA11 /20*0.0/
```

Assignment statements for the variables DCON, QTOLE, 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), DPI(I), COEF, CSV1(I), and CSV3(I) were removed. The following code was added after the definition of COEF3(I),

```
COEF1(I) = CSV(I) * SORT(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.

```
CSV1(I) = 0.0
CSV3(I) = 0.0
IF (IT0.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

60 CONTINUE
HM(l) = 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))*CAPA)-PAttm
PGAII(I) = PGA2I(I)
PGA3I(I) = PGA2I(I)
FORSSST(I) = -(PGA2I(I)*AREA2(I)-PGA3I(I)*AREA3(I)
+ DMTHN(I)*(FORT(I)+CFORT(I))*FSTOP(I))
WFORT(I) = FORSSST(I)

80 CONTINUE

Assignment statements for the variables QO(I), VCUM(I), FFORT(I), FORSSST(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*C/
The definition of CMASNG was changed to the following,

\[ PXCG1 = PX(1) - XCGRF \]
\[ CMASNG = 1 + (AMASS*PXCG1*PXCG1)/AIYYBS \]

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.

\[ UNSPRNG = 0.0 \]
\[ DO 22 I=1,NSTRUT \]
\[ IF(IT0.EQ.1 .AND. TIME.GE.2.0) IMODE(I) = LMODE(I) \]
\[ IF(OMETD1(I).NE.0.0) UNSPRNG = UNSPRNG+MASS(I) \]
\[ 22 CONTINUE \]
\[ DWFOR(I) = (-SQRT(AXP7F*AXP7F+AYP7F*AYP7F+AZP7F*AZP7F)+GREFF)* (AMASS-UNSPRNG) \]

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

\[ WFORT(I) = DWFOR(I)/NSTRUT-1 \]

The four occurrences of VELDEC were changed to VELDEC(I), and the two definitions of I, 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.

\[ IF(IMODDF(I).NE.0.0) GO TO 119 \]
\[ QSV1(I) = 0.0 \]
\[ QSV3(I) = 0.0 \]
\[ GO TO 450 \]
The continuation line in the definition of VELDEC(I) was corrected as follows.

\[(VMASS(I) \times REDSLP(I))\]

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.

\[
UNSPRNG = 0.0
\]
\[
IF(OMETD1(I) \neq 0.0) UNSPRNG = MASS(I)
\]
\[
WFORT(I) = SR(I) \times (VMASS(I) - UNSPRNG)
\]

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.

\[
IF(IMODE(I) \neq 0.0) \text{ GO TO 119}
\]
\[
OSV1(I) = 0.0
\]
\[
CSV3(I) = 0.0
\]
\[
\text{GO TO 450}
\]
\[
119 \text{ CONTINUE}
\]

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 ACOBLOK, 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) .LT. STRON(J+1)) GO TO 20
  10 CONTINUE
  J = J - 1
  15 APINT(I) = PINN(J)
  GO TO 50
  20 APINT(I) = PINM(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) .LT. STROM(J+1)) GO TO 40
  30 CONTINUE
  J = J - 1
  35 APINT(I) = PINM(J)
  40 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 6H7ETAC1, 6H7ETAC2, 6HIMODE, 6HPINN, 6HPINM,
6 6HSTRON, 6HSTROP /

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.

\[
\begin{align*}
S2(1, I) &= -0.5 \times ES2(I) \\
S2D1(1, I) &= -1. \times 10^{-10} \\
S2D2(1, I) &= -1. \times 10^{-10}
\end{align*}
\]

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

\[
\begin{align*}
61 & \quad \text{IF}(S2D2(1, I) \times LT.0.) \text{ GO TO 140} \\
& \quad \text{IF}(S2D1(1, I) \times LT.0.) S2D1(1, I) = 0.
\end{align*}
\]

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).

\[
\begin{align*}
\text{IF}(\text{TMP}(1) \times LT.0.) & \text{ WRITE}(6, 1234) I, S2(1, I), S2D1(1, I), \\
& \text{IF}(\text{TMP}(1) \times LT.0.) \text{ GO TO 31}
\end{align*}
\]

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 $S_2(1,I)$ for $S(1,I)$.

The correction set CSCMOD6 was intended to introduce a new variable, $HT_3$, 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/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(NDEO,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.STROK(J+1)) GO TO 720
      CONTINUE
      J = J - 1
      715  
      720  APINT(I) = PINN(J)
      GO TO 750
      725  STROK = -1.
      DO 730 J=1,29
         IF(STROK(J).LT.STROK) GO TO 735
         STROK = STROK(J)
      IF(S(1,I).GE.STROK(J).AND.S(1,I).LE.STROK(J+1)) GO TO 740
      CONTINUE
      735  J = J - 1
      740  APINT(I) = PINM(J)
      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/DlRCOM/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/FLXDP/DM10(608)
COMMON/AUTSAC/DM11(6)
COMMON/HTC0M/DM12(10)
COMMON/CONTROL/DM13(4)
COMMON/UPDCAL/DM14(181)
COMMON/LGDE/DM15(72)
COMMON/STGT/DM16(10)
COMMON/TABCOM/DM17(230)
COMMON/CLEANP/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/XLGEP1/DM31(21)
```
COMMON/XLGF3C/DM32(12),IFRI(5),DM33(19)
COMMON/XALGEX/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/XAUTES/DM38(8)
COMMON/XFLARE/DM39(20)
COMMON/XAUTPR/DM40(4)
COMMON/RESTRT/IPST

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,DM34Z,DM34A/5*0.0,5*31000.,65*0./
DATA AICX,IPSTOX/5*0.0,5*0/

The following read sequence was placed immediately before the call

READ(5,2) IPST
2 FORMAT(I1)

In subroutine EXE, the following declaration statements were added,

COMMON/XEXE/END,SWT2,SWT3,BLANK,STCOM2,STCOM3,
1 MIM,NEFSTIMEA,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 statement 416.

```
IF(IRST.EQ.1 .OR. IPST.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, YF, XK, YO,
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, ETADE, HG, IGO1, IGO2, IGO3,
1 TC, TMP, CN
COMMON/XOPT1/ALPHD1, P, BETAD1, C, D, DRAGC, E, F, FDC,
1 G, H, INDER, J, K, LA, LTTHRP, M, MDX, MDY, MDZ, P, THTRP,
2 THTRR1, TMP
```
In subroutine LGEAR1, local variables were moved to XLGER1,

```
COMMON/XLGER1/BFX, BFY, BFZ, BLM, BMM, BNM, DFXM, DFYM,
1  DFM, DLM, DMM, ERR, IL4, INFLAG, II, 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/CNPWR, CRNP, DELMN, DFTRX, DFTRXM,
1  DFTRY, DFTRYM, ETAVE, ETAVEM, FGPYM, FI, HYPTAN, IFRI,
2  ILP, IL2, IL3, ND, NDO, NOD, SIDMU, SIDUM, 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, DFWORT, 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, CDPMAS, CTMP1, DIFF, FDC, GF,
1  GFORC1, GMASS, GSMOD, GTF, HDR, IG, II, LA, OMXD1M,
2  OMYD1M, OM7D1M, PNT, Q5, Q51, RKSY, RVAR, RXA13, RZA11,
3  SDQ, SD1, SD2, SMAS, SUM1, SUM3, TITL, VARY1
COMMON/XDRLG/ACOVAR1, ACOVAR2, ACOVAR3, ACOVAR4,
1  ACOVAR5, ACOVAR6, ACOVAR7, ACOVAR8, ACOVAR9, DAT1,
```
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 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)-REDSL(P(I))*DELT
EPSILO(I)=EPSILO(I)+EPSFSLP*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
W*FOR(I) = W*FOR(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,COEFF(5) ,LMOOE(5) ,PINN(30) ,PINM(30) ,STRON(30) ,STROM(30)
2,XMA1DT(5) ,XMA2DT(5) ,XMA3DT(5) ,XMA4DT(5) ,XMA5DT(5) ,XMA6DT(5)
3,XMA7DT(5) ,XMA8DT(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) NDEO=10*NSTRUT
```

and in ACTINIT the following DATA statement was added.

```
DATA XMA1DT,XMA2DT,XMA3DT,XMA4DT /20*0.0/,
     XMA5DT,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),XMA10DT(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 servo valve 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 ALGEAR 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, MIN 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 ALGEB1 and ACTNG.

In subroutine ALGEB1, the following code was inserted as the first executable statements,

\[
\begin{align*}
\text{IF}(\text{NIN.EQ.0)} & \quad \text{TIMEL} = \text{TIME} \\
\text{NIN} & \quad = 1 \\
\text{DELT} & \quad = \text{TIME} - \text{TIMEL} \\
\text{TIMEL} & \quad = \text{TIME}
\end{align*}
\]

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 XALGEB 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 ALGEB1 and ACTNG, the three statements beginning with statement 159 were replaced with the following.

\[159 \text{IF}(\text{SD1}(1,1) \lt \text{0.}) \quad \text{GO TO 471}\]
The MOD296 correction set changed the active code. In subroutine ALGEAR1, the statement above statement 112 was changed to the following,

\[ \text{IF(\text{IMODE(I).EQ.0.AND.S(1,I).EQ.0.) 112,113}} \]

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

\[ \text{IF(S(1,I).GT.0..OR.IMODE(I).EQ.1) GO TO 113} \]

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

\[
\text{IF(IMODE(I).EQ.0) GO TO 28}
\]

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

\[
\text{IF(IMODE(I).EQ.0) GO TO 6}
\]

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

\[
\text{PGA1T1(I) = PGA1I(I)}
\]
\[
\text{PGA1T(I) = PGA1I(I)}
\]

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.

```
DELQ1 = 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(I,I)+1.E-4.GE.S2D1(I,I)) 84,85
84 SD1(I,I) = SD1(I,I)
SD2(I,I) = SD2(I,I)
GO TO 60
85 CONTINUE

A method for slowly and continuously varying the aerodynamic coefficients $C_A$ and $C_N$ 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/LOCSS(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/0.0,0/
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
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
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
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 AERODIN(SYM,RA)
GO TO 100

The following new subroutine was added after subroutine READ.

SUBROUTINE AERODIN(SYM,RA)
DIMENSION RA(55)
COMMON/AEROCO/DATAX(8)
DATA DATAX/8*0.1/
CALL LINES(I)
WRITE(6,1) SYM,RA
1 FORMAT(18X,A6,5X,55A1)
2 FORMAT(*ERROR. ILLEGAL CHARACTER IN NUMERIC FIELD, 1R1,2H**/)
IF(SYM.EQ.6HRTAB10) INDEX = 1
IF(SYM.EQ.6HRTAB80) INDEX = 3
IF(SYM.EQ.6HRTAB10) INDEX = 5
IF(SYM.EQ.6HRTAB80) INDEX = 7
NUMEXP = 0
NEXP = 0
N = 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(JSIGN.EQ.1) X = -X
150 DATA(XINDEX) = 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(JSIGN.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 incorporated 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.

```
287 IF(S(I,I).GT.ES(I)) ISTROK(I)=1
   ICU(I)=0
   IQCU(I)=0
   IXS(I)=0

C THE FOLLOWING LOGIC RETURNS THE GEARS, DURING REBOUND, TO INITIAL CONDITIONS IN THE EVENT THE GEAR CONTACTS THE SURFACE BEFORE 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) GO TO 227,297
227 IF(IGO(I).EQ.0) GO TO 226
   IF(Delta(I).GT.0.0.AND.WLFOR(I).EQ.0.0) GO TO 220,226
220 IF(VCUM(I).GT.0.00001.OR.VCUM(I).LT.-0.00001) GO TO 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) GO TO 290,297
```
The code between statements 299 and 400 was replaced with the following:

312 IF(ICSV(I).NE.1) GO TO 313, 314
314 IF(XVALVE(I).NE.XBIAS(I)) GO TO 311
298 IF(ICSV(I).EQ.1) GO TO 311
313 IF(PGA2T(I).GT.PGA1(I)+2000.) GO TO 311
292 IF(IIXSVL(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) GO TO 303
300 XSV(I)=XVALVE(I)
GO TO 297
303 XSV(I)=XVALVE(I)
GO TO 297
302 XVALVE(I)=0.13
IIXSVL(I)=1
GO TO 294
293 IF(PGA2T(I).LT.PGA1(I)-2000.) GO TO 295, 294
295 IF(IIXSVH(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) GO TO 304
304 XSV(I)=XVALVE(I)
GO TO 297
302 XVALVE(I)=0.13
IIXSVH(I)=1
294 CONTINUE
XSVDOT(I)=0.0
ICSV(I)=1
311 IF(PGA2T(I).GT.PGA1(I)+4000.0) GO TO 316, 315
316 IF(QS(I).LT.QD(I).AND.ICSV(I).EQ.1) GO TO 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) + XSVDNM(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) = XSVDXM(I)
    IPASS(I) = 4
    XVALVE(I) = XVALVE(I) + XSVDXM(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.

DETX1(I) = 0.0
ICOSV(I) = 0
QD(I) = 0.0
QSVCU(I) = 0.0
QSV1(I) = 0.0
QSV3(I) = 0.0
VCUM(I) = 0.0
VOLT1(I) = VOLT1(I)
VOLT2(I) = VOLT2(I)
VOLT3(I) = VOLT3(I)
ITRIP(I) = 1
PGA1T1(I) = PGA1T1(I)
PGA1T(I) = PGA1T1(I)
PGA2T(I) = PGA2T(I)
PGA3T(I) = PGA3T(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(VSUM(I).LT.0.00001.OR.VCUM(I).LT.-0.00001)GO TO 431
   QQ(I)=0.0
   ICU(I)=1
   GO TO 440
431 IF(VSUM(I).LT.-0.000001)GO TO 432
   QQ(I)=-VCUM(I)/DSTOP
   GO TO 440
432 QQ(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
   IKS(I)=1
600 IF(XMA1(I).GT.0.00001.OR.XMA1(I).LT.-0.00001)GO TO 601
   XMA1DT(I)=0.0
   IMA1(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
   IMA2(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
   IMA3(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
   IMA4(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
   IMA6(I)=1
   GO TO 610
609 XMA6DT(I)=-XMA6(I)/DSTOP
The test on HMM(I) below the definition of VELDEC(I) was replaced with the following code.

```plaintext
IF(IGE(I).EQ.0)GO TO 131
WLFOR(I)=WLFORR
INDEACT(I)=2
EPSILO(I)=EPSROL(I)
GO TO 451

131 IF(S(I,I).LE.ES(I))GO TO 451
IF(HMM(I).EQ.0)GO TO 451
```
The following test was inserted before statement 452,

\[ \text{IF}(S(1, I) \leq E_S(I) \text{ AND } ITRIP(I) \text{ EQ } 1) \text{ GO TO 458} \]

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

\[ \text{P1}(I) = \frac{P_{GA1T}(I)}{144}. \]

\[ \text{VOLANT}(I) = \text{VOLANT}(I) + \text{QSVN}(I) \times \text{DELT} - \text{QPUMPS}(I) \times \text{DELT}. \]

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

\[ \text{IF}(S(1, I) \leq E_S(I) \text{ AND } ITRIP(I) \text{ EQ } 1) \text{ GO TO 462, 410} \]

\[ \text{CALL LIMITS}(XSV(I), XSVDOT(I), XSVMAX(I), XSVMIN(I)) \]

\[ \text{CALL FLOZE2}(PS(I), PR(I), P1(I), XLPSV1(I), XLPSV3(I), RCLSV(I), DSV(I)), \]

The following initializations were added below statement 50,

\[ \text{IGE}(I) = 0 \]

\[ \text{ITRIP}(I) = 0 \]

and the following tests were inserted above statement 55.

\[ \text{IF}(I \text{ MODE}(I) \text{ EQ } 0) \text{ GO TO 55} \]

\[ \text{IF}(I \text{ ISTROK}(I) \text{ EQ } 1) \text{ IGE}(I) = 1 \]

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

\[ \text{CALL LIMITS}(XSV(I), XSVDOT(I), XSVMAX(I), XSVMIN(I)) \]

\[ \text{CALL LIMITS}(XSVDOT(I), XSVDD(I), XSVDMX(I), XSVDMIN(I)) \]

\[ \text{CALL LIMITS}(XSVDD(I), XSVDD(I), XDDMAX(I), XDDMIN(I)) \]

Similar modifications were required in the ACTING subroutine.
The MOD2235 correction set corrected an error in the MOD2203 logic by replacing the definition of IG0(I) below statement 222 with the following.

\[ \text{IF} (\text{DELTA}(I) \gt 0.) \text{ IG0}(I) = 0 \]
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.

The allowed values of the integer restart flag are presented below:

<table>
<thead>
<tr>
<th>Value</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reject the restart option - This run will read all input data from cards and a subsequent restart will not be possible.</td>
</tr>
<tr>
<td>1</td>
<td>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.</td>
</tr>
<tr>
<td>2</td>
<td>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.</td>
</tr>
</tbody>
</table>
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
              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)
Figure 3-a. Full Input Deck
ORIGINAL PAGE IS OF POOR QUALITY

PRO C5 VEHICLE PHYSICAL PPOLP. DATA

TNDVPC 1
YCRRF 0.

VTAR01 2.0.0.5000.0.
VTAR02 2.0.24873.1200.24873.
VTAR03 2.0.144516.1200.144516.
VTAR04 2.0.154897.1200.154897.

VTAR06 2.0.3361.1200.3361.

PRO C4 AERODYNAMIC INPUT DATA

TNDAFP 1
ARFF 547.87
A1PFF 16.042
A2PFF 3.667

INPA01 1
ATAR01 -0.000709, -0.000709
TNDAP2 1
ATAR02 0.000423, 0.000423
TNDAP3 1
ATAR05 -0.001027, -0.001027
TNDAP4 1
ATAR06 0.000136, 0.000136
TNDAP0 0
ATAR08 0.00.
TNDAP10 1
ATAR10 .379202, .379202
TNDAP11 1
ATAR11 .063627, .063627
TNDAP12 1
ATAR12 -0.0018, -0.0018
TNDAP15 1
ATAR16 .0016, .0016
TNDAP16 1
ATAR16 -0.0002, -0.0002
TNDAP51 1
ATAR51 -1.0002, -0.0002
TNDAP52 1
ATAR52 -.005946, -.005946
TNDAP53 1

Figure 3-a. Cont'd
| ATAR53 | -0.000216, -0.000216 |
| ATAR56 | 1 |
| ATAR56 | -0.010317, -0.010317 |
| ATAR57 | 1 |
| ATAR57 | -0.000005, -0.000005 |
| ATAR58 | 0 |
| ATAR59 | 0, 0 |
| ATAR60 | 1 |
| ATAR60 | 0.141245, 0.141245 |
| REM RCD | ENGINE THRUST DATA |
| TNO F | 3 |
| TNO TSO | 1 |
| TTOY | 6 |
| TTOY | 5 |
| TTAR10 | -2.0, -1.0, 1.0, 2.0 |
| TTAR10 | 0.0, 0.05, 1.0, 15.0, 2.25 |
| TTAR10 | 0.0, 0.0, 0.470, 0.17600 |
| TTAR10 | 0.0, 0.0, 0.300, 0.17300 |
| TTAR10 | 0.0, 0.0, 0.220, 0.17200 |
| TTAR10 | 0.0, 0.0, 0.160, 0.17080 |
| TTAR10 | 0.0, 0.0, 0.0, 0.17080 |
| TTAR10 | 0.0, 0.0, 0.0, 0.17100 |
| REM RCD | 3LANDING GEAR DATA |
| A M | 0.0018 |
| STRON | 0.0, 0.1, 1.0333, 2.6667, 3.35, 4.3333, 5.1667, 6.0, 6.8333, 7.6667 |
| STRON | 0.85, 0.93333, 1.01667, 1.1, 1.18333, 1.26667, 1.35, 1.43333 |
| STRON | 1.51667, 1.6, 1.68333, 1.76667, 1.85, 1.93333, 2.0 |
| DTNN | 0.0231, 0.0231, 0.0231, 0.0204, 0.0242, 0.03, 0.0339, 0.0352 |
| DTNN | 0.0396, 0.0435, 0.0454, 0.0473, 0.0492, 0.0511, 0.0511, 0.0511 |
| DTNN | 0.0531, 0.0531, 0.055, 0.0569, 0.0601, 0.0627, 0.0646, 0.0665 |
| DTNN | 0.0665 |
| STRON | 0.0, 0.06604, 0.10771, 0.14938, 0.19104, 0.23271, 0.27438, 0.31604 |
| STRON | 0.35771, 0.39938, 0.44104, 0.48271, 0.52438, 0.56604, 0.60771, 0.64938 |
| STRON | 0.69104, 0.73271, 0.77438, 0.81604, 0.85771, 0.89938, 0.94104, 0.9827117 |
| STRON | 1.02438, 1.06604, 1.10771, 1.14938, 1.19104 |
| DTNN | 0.0603, 0.0595, 0.0572, 0.055, 0.0518, 0.0492, 0.0458, 0.0424 |
| DTNN | 0.039, 0.0373, 0.0356, 0.0336, 0.0316, 0.0296, 0.0276, 0.0256 |
| DTNN | 0.0486, 0.0511, 0.0528, 0.0554, 0.0571, 0.0588, 0.0614, 0.063117 |
| DTNN | 0.0648, 0.0665, 0.0674, 0.0682, 0.0682 |
| A M | 0.0018 |
| AREA1 | 0.0705, 0.1134, 0.134 |
| AREA2 | 0.12962, 0.16082, 0.16082 |
| AREA3 | 0.01651, 0.00819, 0.00819 |
| AREOQ | 0.00684, 0.00684, 0.00684 |
| APFO3 | 0.000217, 0.000668, 0.000668 |
| ETA | 1.4400000 |
| RLH | 15, 15, 15 |
| RIMU | 15, 15, 15 |
| CDNOC | 9, 9, 9 |

Figure 3-a. Cont'd
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMHDF</td>
<td>0.9, 0.9, 0.9</td>
</tr>
<tr>
<td>CMV</td>
<td>0.62, 0.62, 0.62</td>
</tr>
<tr>
<td>CN3</td>
<td>0.6, 0.6, 0.6</td>
</tr>
<tr>
<td>CFFDR</td>
<td>50, 50, 50</td>
</tr>
<tr>
<td>DELT</td>
<td>0.0001</td>
</tr>
<tr>
<td>DTOTA</td>
<td>1</td>
</tr>
<tr>
<td>DSV</td>
<td>1.1875, 1.1875, 1.1875</td>
</tr>
<tr>
<td>EPSILIP</td>
<td>200, 200, 200</td>
</tr>
<tr>
<td>EPSRDL</td>
<td>2000, 2000, 2000</td>
</tr>
<tr>
<td>ESSLP</td>
<td>0</td>
</tr>
<tr>
<td>ETASV</td>
<td>0.436</td>
</tr>
<tr>
<td>FUPFR</td>
<td>0.9, 0.9, 0.9</td>
</tr>
<tr>
<td>GAMAH</td>
<td>52.36, 52.36, 52.36</td>
</tr>
<tr>
<td>GNP</td>
<td>0</td>
</tr>
<tr>
<td>KAPT</td>
<td>2, 2, 2</td>
</tr>
<tr>
<td>PMUN</td>
<td>0</td>
</tr>
<tr>
<td>PATM</td>
<td>2116.8</td>
</tr>
<tr>
<td>PCHAMAC</td>
<td>432000, 432000, 432000</td>
</tr>
<tr>
<td>PCALAC</td>
<td>0, 0, 0</td>
</tr>
<tr>
<td>PCA11T</td>
<td>16272, 28800, 28800</td>
</tr>
<tr>
<td>PCA2T</td>
<td>16272, 28800, 28800</td>
</tr>
<tr>
<td>PCD3T</td>
<td>16272, 28800, 28800</td>
</tr>
<tr>
<td>PERCNT</td>
<td>1.66115, 1.66115, 1.66115</td>
</tr>
<tr>
<td>POMP5</td>
<td>0.05793, 0.05793, 0.05793</td>
</tr>
<tr>
<td>RQLSV</td>
<td>4.75E-5, 4.75E-5, 4.75E-5</td>
</tr>
<tr>
<td>RDNH</td>
<td>1.626</td>
</tr>
<tr>
<td>TAUUF</td>
<td>0.1</td>
</tr>
<tr>
<td>TC1</td>
<td>2.91</td>
</tr>
<tr>
<td>TC2</td>
<td>1.41</td>
</tr>
<tr>
<td>TC3</td>
<td>0.001</td>
</tr>
<tr>
<td>TC4</td>
<td>0.0001</td>
</tr>
<tr>
<td>VOLAC1</td>
<td>1.33333, 1.33333, 1.33333</td>
</tr>
<tr>
<td>VOLANT</td>
<td>26667, 26667, 26667</td>
</tr>
<tr>
<td>VOL1T</td>
<td>11733, 11507, 11507</td>
</tr>
<tr>
<td>VOL2T</td>
<td>26923, 26239, 26239</td>
</tr>
<tr>
<td>VOL3T</td>
<td>0, 0, 0</td>
</tr>
<tr>
<td>WC</td>
<td>1263</td>
</tr>
<tr>
<td>WCI</td>
<td>251.3</td>
</tr>
<tr>
<td>WSV</td>
<td>655.5</td>
</tr>
<tr>
<td>WSV1</td>
<td>3.48</td>
</tr>
<tr>
<td>WSV3</td>
<td>3.48</td>
</tr>
<tr>
<td>WLFOR</td>
<td>500, 500, 500</td>
</tr>
<tr>
<td>WLFORD</td>
<td>0, 0, 0</td>
</tr>
<tr>
<td>YRTAS</td>
<td>-0.00227, -0.000171, -0.000171</td>
</tr>
<tr>
<td>YDONH</td>
<td>1.0E20, 1.0E20, 1.0E20</td>
</tr>
<tr>
<td>YDONMIN</td>
<td>-1.0E20, -1.0E20, -1.0E20</td>
</tr>
<tr>
<td>YKA</td>
<td>0.04, 0.04, 0.04</td>
</tr>
<tr>
<td>YKF</td>
<td>60, 60, 60</td>
</tr>
<tr>
<td>YKSU</td>
<td>0.0025, 0.0025, 0.0025</td>
</tr>
</tbody>
</table>

Figure 3-a. Cont'd
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>YLPSV1</td>
<td>0.0, 0.0</td>
</tr>
<tr>
<td>YLPSV2</td>
<td>0.0, 0.0</td>
</tr>
<tr>
<td>YSCOM</td>
<td>1.54667, 1.15667, 1.15667</td>
</tr>
<tr>
<td>YSTHR</td>
<td>0.02083, 0.02083, 0.02083</td>
</tr>
<tr>
<td>XSUMMN</td>
<td>-30.1, -30.1, -30.1</td>
</tr>
<tr>
<td>YSUMMY</td>
<td>30.1, 30.1, 30.1</td>
</tr>
<tr>
<td>YSUMAY</td>
<td>1.0, 1.0, 1.0</td>
</tr>
<tr>
<td>XSUMIN</td>
<td>-1.0, -1.0, -1.0</td>
</tr>
<tr>
<td>7FTAC1</td>
<td>5.1</td>
</tr>
<tr>
<td>7FTAC2</td>
<td>1.0</td>
</tr>
<tr>
<td>ITMDE INT</td>
<td>0.1, 1.1</td>
</tr>
<tr>
<td>ENDFACT</td>
<td></td>
</tr>
<tr>
<td>NSTPUT</td>
<td>3</td>
</tr>
<tr>
<td>MASS</td>
<td>4.26971, 13.98819, 13.98819</td>
</tr>
<tr>
<td>RX</td>
<td>19.57503, -4.34308, -4.34308</td>
</tr>
<tr>
<td>DY</td>
<td>0.0, -8.95333, 8.95333</td>
</tr>
<tr>
<td>DT</td>
<td>1.40617, 1.41425, 1.41425</td>
</tr>
<tr>
<td>THEFTAN</td>
<td>0.0, 4.29137, 4.29137</td>
</tr>
<tr>
<td>ERDEC</td>
<td>0.0</td>
</tr>
<tr>
<td>CONEM</td>
<td>23228.5</td>
</tr>
<tr>
<td>CTHWDM</td>
<td>69983.8</td>
</tr>
<tr>
<td>CTHRFM</td>
<td>3180.1</td>
</tr>
<tr>
<td>CENQDM</td>
<td>6581.4</td>
</tr>
<tr>
<td>CONFRM</td>
<td>1683</td>
</tr>
<tr>
<td>DELPPM</td>
<td>4173</td>
</tr>
<tr>
<td>UPQPODM</td>
<td>0.0</td>
</tr>
<tr>
<td>AH</td>
<td>0.08705, 1.1020, 1.1020</td>
</tr>
<tr>
<td>RAR</td>
<td>0.0</td>
</tr>
<tr>
<td>NTYRES</td>
<td>2.0, 1.0, 1.0</td>
</tr>
<tr>
<td>RZERN</td>
<td>73333, 1.21875, 1.21875</td>
</tr>
<tr>
<td>W</td>
<td>46042, 9375, 9375</td>
</tr>
<tr>
<td>DELTAM</td>
<td>21666, 36875, 36875</td>
</tr>
<tr>
<td>PLT</td>
<td>1.5F4+4</td>
</tr>
<tr>
<td>TFD</td>
<td>1</td>
</tr>
<tr>
<td>AT</td>
<td>1.06148E+5, 1.94542E+5, 1.94542E+5</td>
</tr>
<tr>
<td>RT</td>
<td>1.3, 1.2, 1.2</td>
</tr>
<tr>
<td>SLEN1</td>
<td>1.04037, 7.8723, 7.8723</td>
</tr>
<tr>
<td>SLEN2</td>
<td>3.0122, 3.095, 3.095</td>
</tr>
<tr>
<td>INNAMT</td>
<td>1</td>
</tr>
<tr>
<td>CAMA</td>
<td>1.06</td>
</tr>
<tr>
<td>CONE</td>
<td>6483.9</td>
</tr>
<tr>
<td>CTWO</td>
<td>32406.4</td>
</tr>
<tr>
<td>CTHREE</td>
<td>534.12</td>
</tr>
<tr>
<td>CFOUR</td>
<td>1837.1</td>
</tr>
<tr>
<td>CONFR1</td>
<td>1683</td>
</tr>
<tr>
<td>DELPVR</td>
<td>4173</td>
</tr>
<tr>
<td>DTIRE</td>
<td>1.95208</td>
</tr>
<tr>
<td>ETAMAX</td>
<td>25.0</td>
</tr>
<tr>
<td>ETAMIN</td>
<td>-25.0</td>
</tr>
</tbody>
</table>

Figure 3-a. Cont'd
<table>
<thead>
<tr>
<th>FA</th>
<th>30.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETANOS</td>
<td>0.0</td>
</tr>
<tr>
<td>FTPR1</td>
<td>20.0</td>
</tr>
<tr>
<td>TNMWH</td>
<td>0.0</td>
</tr>
<tr>
<td>KANUS</td>
<td>0.0</td>
</tr>
<tr>
<td>KPNUS</td>
<td>0.0</td>
</tr>
<tr>
<td>JSTER</td>
<td>0.0</td>
</tr>
<tr>
<td>TRUDD</td>
<td>0.0</td>
</tr>
<tr>
<td>TNWPUP</td>
<td>0.0</td>
</tr>
<tr>
<td>PCTETA</td>
<td>1.00</td>
</tr>
<tr>
<td>PSIDES</td>
<td>0.00</td>
</tr>
<tr>
<td>RPST</td>
<td>0.00</td>
</tr>
<tr>
<td>RYP</td>
<td>10.0</td>
</tr>
<tr>
<td>VDPWP</td>
<td>0.00</td>
</tr>
<tr>
<td>YRDW</td>
<td>0.00</td>
</tr>
<tr>
<td>FTARO2</td>
<td>2.0, 0.0, 1.5E+5, 0.0</td>
</tr>
<tr>
<td>FTARO3</td>
<td>11.0, 0.0005, 0.03, 0.15, 0.06, 0.22, 0.12, 0.64, 0.15, 0.8</td>
</tr>
<tr>
<td>MMNT</td>
<td>53709.8, 0.19, 0.8, 0.19</td>
</tr>
<tr>
<td>MR</td>
<td>0.0, 0.0</td>
</tr>
<tr>
<td>OF</td>
<td>6.0108, 5.1525, 5.1525</td>
</tr>
<tr>
<td>VT</td>
<td>0.0</td>
</tr>
<tr>
<td>P7ER0</td>
<td>16272.2, 28800.0, 28800.0</td>
</tr>
<tr>
<td>V7ER0</td>
<td>0.09343, 22793.2, 22793.2</td>
</tr>
<tr>
<td>A</td>
<td>0.08705, 11020.0, 11020.0</td>
</tr>
<tr>
<td>TL</td>
<td>1.0</td>
</tr>
<tr>
<td>TNDC01</td>
<td>1.0</td>
</tr>
<tr>
<td>NSMAINT</td>
<td>51.0</td>
</tr>
</tbody>
</table>

**Figure 3-a. Cont’d**
Figure 3-a. Cont'd
Figure 3-a. Cont'd
1. RCD 5H. LANDING ROLL MANEUVER DATA

2. RCD 5I. ENGINE FAILURE STAGE DATA

3. RCD 5J. BRAKE COND. STAGE DATA

4. RCD 4K. PITCH Autopilot Data

5. RCD 4L. YAW Autopilot Data

Figure 3-a. Cont'd
Figure 3-a. Cont'd
Figure 3-a. Cont'd
<table>
<thead>
<tr>
<th>Scenario</th>
<th>ATARPO 1</th>
<th>ATARPO 2</th>
<th>ATNCPS RCD 1</th>
<th>TIME 1</th>
<th>TEST 1</th>
<th>TPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario 1</td>
<td>0.17, 0.17</td>
<td>0.18, 0.18</td>
<td>1.00</td>
<td>90</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenario 2</td>
<td>0.25, 0.25</td>
<td>0.25, 0.25</td>
<td>1.10</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenario 3</td>
<td>0.20, 0.20</td>
<td>0.21, 0.21</td>
<td>1.20</td>
<td>110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenario 4</td>
<td>0.22, 0.22</td>
<td>0.23, 0.23</td>
<td>1.30</td>
<td>120</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenario 5</td>
<td>0.24, 0.24</td>
<td>0.25, 0.25</td>
<td>1.40</td>
<td>130</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenario 6</td>
<td>0.00005</td>
<td>0.001</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3-a. Cont'd
Figure 3-b. Short Input Staging Deck for a restart at 0.60 second.
<table>
<thead>
<tr>
<th>STCASE</th>
<th>ATARPO</th>
<th>AinCps RCD 1TIME</th>
<th>STFST</th>
<th>TPA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>.21, .21</td>
<td></td>
<td>1.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>.22, .22</td>
<td></td>
<td>1.4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>.23, .23</td>
<td></td>
<td>1.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>.24, .24</td>
<td></td>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>.25, .25</td>
<td></td>
<td>4.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>.00005</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>.001</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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.
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 CA0 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,LRCCOSF/UN=LIBRARY.
REWIND,TAPE13.
COPYEI,TAPE13,TAPE3.
RETURN,TAPE13.
REWIND,TAPE3.
LDSET,LIB=LRCCOSF.
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,OLDP=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.
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.

---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


Appendix A
Program Modifications

*IDENT 0B1877
*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
  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(315)
*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 (DG18(72), PSIPD)
*INSERT LGEA3C.61
  DATA RADDEG, DEGRAD /57.2957795, 0.1745329 /
*DELETE LGEA3C.143, LGEA3C.143
  IF(I.EQ.1 .AND. INDNWS.EQ.0) GO TO 200
  GO TO 7
  200 VTX(I)=VTX(I)-TMP(2)
    VTY(I)=RDYG(I)
    7 VTZ(I)=RG31*RDYG(I)+RG33*RDZG(I)
*INSERT LGEA3C.167
  IF(DELAT1 .EQ. 0.0) FGPY=0.0
  IF(DELAT1 .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
120 IF((DELTAL/DTIRE)*ETG.0.0875)GO TO 130
CRNPWR=CONE*DELTAL-CTWO*DELTAL**2.
GO TO 170
130 CRNPWR=CTHREE-CFOUR*DELTAL
170 YAWPRM=ABS(CRNPWR*DELNL/(SIDEMU*FTRZ(I)))
   IF(YAWPRM.LE.1.5) GO TO 150
   FGPY=SIDEMU*FTRZ(I)*COS(DELNL*DEGRAD)
   GO TO 160
150 FGPY=IF(YAWPRM-(4.0/27.0)*YAWPRM**3.0)*SIDEMU*FTRZ(I))
   +**COS(DELNL*DEGRAD)
160 FGPY=PCTETA*FGPY
   IF(DELNL.GT.0.0) FGPY=ABS(FGPY)
   IF(DELNL.LT.0.0) FGPY=-ABS(FGPY)
   DFTRX=FGPY*SINCET*VE)
   DFTRY=FGPY*COS(ETAVE)
   FTRX(I)=FTRX(I)+DFTRX
   FTRY(I)=FTRY(I)+DFTRY
   IF(OElN.GT.0.0) FGPY=ABS(FGPY)
   IF(OElN.LT.0.0) FGPY=-ABS(FGPY)
*INSERT LGEA3C.170
   *INSERT FGPY = 0.0
   *INSERT SDFLGP.34
   COMMON/NWSTER/CFOUR,CONE,CONFRI,CTHREE,CTWO,DELPWR,DTIRE
   EA,ETAMAX,ETAMIN,ETANOS,ETART1,INDNWS,KANWS,KPNWS,PCTETA
   PSIDES,RPSI,RYP,VPPOWR,VRDES
   ,DELN,ETADES,FGPY
   *(OP19(I),I=1,7) /3HFP2P,2HLM,2HNM,4HFGPY,4HDELN,6HETADES/,
   *DELETE SDFLGP.67,SDFLGP.67
   DATA DAT1/4HTIME/,
   *(OP19(I),I=1,18) /2HLM,2HMM,2HNM,5HQP177R,
   *DELETE SDFLGP.75,SDFLGP.75
   *5HYN77F,5HYG77F,5HAX77F,6HXY77F,4HFGPY,4HDELN,6HETADES/,
   *INSERT SDFLGP.80
   DATA N1/I/N15/N14/15/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(19,7,19)
   CALL STOVAR(7,FM,LM,MM,NM,FGPY,DELN,ETADES,DU)
   *INSERT AUTS.74
   COMMON/NWSTER/CFOUR,CONE,CONFRI,CTHREE,CTWO,DELPWR,DTIRE
   EA,ETAMAX,ETAMIN,ETANOS,ETART1,INDNWS,KANWS,KPNWS,PCTETA
   PSIDES,RPSI,RYP,VPPOWR,VRDES
   DE LN,ETADES,FGPY
   *INSERT AUTS.81
DATA ILIM, INOSE/0.,0/

*INSERT AUS.102

IF(DELTAl .EQ. 0.,0.) GO TO 8
IF(INDWS.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*YRDI

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

35 CONTINUE

*INSERT EXE.68

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
1*DM8(100)
*INSERT LGDET.8
C*DM9(100)
*DELETE LGDET.33, LGDET.33
49 FORMAT(58X,4H-ESf,Y1,19H) 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
55 IF(Y(JJ),LT,0.)Y(JJ)=0.
GO TO 55
30 P(J)=0.
P(JJ)=0.
*INSERT STGTSI.7
1*DM8(100)
*INSERT STGTSI.6
1*DM8(100)
*INSERT DEF.5
1*DM5(100)
*INSERT LINES.4
C*DM4(100)
*INSERT ERROR.4
C*DM3(100)
*INSERT EXERR.3
C*DM3(100)
*DELETE ATMS.10
1*DM8(100)
*INSERT TFFS1.16
1*DM16(100)
*INSERT TFFS8.16
1*DM16(100)
*DELETE VPCS1.23, VPCS1.23
9*XCGBF , XCGRF , DM19(930), DM20(2068), DM21(100)
*INSERT SACSI.63
*DDM78(100)
*INSERT OPT1.72
*DDM158(100)
*INSERT LGEAR1.30
*DDM21160), 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(1,1)*
*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(1,1),EQ,0.0) AND FT(I),LE,ABS(SF(I)) SF(I)=-FT(I)
*DELETE LGEAR1.222,LGEAR1.222
49 FORMAT(58X,4H-EXF,Il,ZOH)
  IN LGEAP1
  INSERT LGEAR1.Z23
$ f1,J)=-0.5*ES(I)
S1(1,1)=1.0E-10
S02(1,1)=1.0E-10
*INSERT LGEAP1.229
S(1,1)=-0.5*ES(I)
*DELETE LGEAP1.234,LGEAP1.235
51 IF(SD2(1,1),LT,0.)GO TO 30
  IF(SD1(1,1),LT,0.)SD1(1,1)=0.
  GO TO 55
30 SD2(1,1)=0.
  SD1(1,1)=0.
*INSERT LGEAR1.240
C RE-CHECK SHOCK STUT FOR FOR RE-CONDITIONED FULLY EXTENDED STATE
 IF(SD1(1,1),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),VPWR,
  *DDM34,FGPY,DELM,ETADES,DDM35(45)
*DELETE LGEA3C.32,LGEA3C.33
*DELETE C82477.14,C82477.17
*INSERT LGEA3C.59
  EQUVALENCE (DM18(31),PA77P)
*DELETE LGEA3C.105,LGEA3C.106
  P(I) = (PZERO(I)+PA77P)*(VZERO(I)/(VZERO(I)+A2(I)*S2(1,1))-
  *S(1,1)*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(1,1),TMP(1))*COS((PSIPD+ETADES)*DGRAD)*RI(2,2,1))
  1 GE, (RG11*RDXG(I)+RG13*RDZG(I))GO TO 201
  VTX(I)=VTX(I)-TMP(2)*(OMET(1,1)*TMP(1))*COS((PSIPD+ETADES)*
  1 DGRAD)*RI(2,2,1)
  VTY(I)=RDYG(I)-(OMET(1,1)*TMP(1))*SIN((PSIPD+ETADES)*DGRAD)*
  1 RI(2,2,1)
*INSERT C82477.24
  GO TO 203
201 VTX(I)=1.E-10
  OMET(1,1)=-(RG11*RDXG(I)+RG13*RDZG(I))/(RZERO(I)-DELT(A(I)))
\[ VTY(I) = RDYG(I) - (OMET(1,I) * TMP(1)) * \sin((PSIPD + ETADES) * DEGRAD)* \\
1 \text{RI}(2,1,I) \]

\[ VTZ(I) = RG31 * RDYG(I) + RG33 * RDZG(I) \]

*DELETE LGEA3C.145, LGEA3C.145

\[ 203 \text{TMP}(1) = RG11 * RDYG(I) + RDZG(I) * RG13 \]

*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/FNAME(1000), LOC(1000), NCOUNT

*DELETE DIRODA.3, DIRODA.3

COMMON/FIXDIR/FNAME(1000), LOC(1000), NCOUNT

*DELETE DIRIDA.3, DIRIDA.3

COMMON/FIXDIR/FNAME(1000), LOC(1000), NCOUNT

*DELETE DIR2DA.3, DIR2DA.4

COMMON/FIXDIR/FNAME(1000), LOC(1000), NCOUNT

DATA NCOUNT/929/

*DELETE DIR3DA.3, DIR3DA.4

COMMON/FIXDIR/FNAME(1000), LOC(1000), NCOUNT

DATA (FNAME(K3), K3 = 876, 929)/

*DELETE DIR3DA.11, DIR3DA.11

* 6HPF , 6HGQ, 6HGD1, 6HGD2, 6HFLX, 6HGDAMP, 6HSLEN1, 
* 6HSLEN2, 6HGAMA, 6HCFOUR, 6HCONE, 6HCONFRI, 6HCTREE, 6HCTWO, 
* 6HDELPWR, 6HDTIRE, 6HEA, 6HETAMAX, 6HETAMIN, 6HETANOS, 6HETART1, 
* 6HINDNWS, 6HKANWS, 6HPKPNWS, 6HCPTE, 6HPSIDES, 6HPSIDES, 6HRPSI, 6HHRY, 
* 6HVPDPWR, 6HYRDES, 6HFGBP, 6HDELN, 6HETADES, 6HILIM, 6HINCOSE, 
* 6HISTER, 6HIRODD, 6HINWRUD, 6HAIN, 6HHPH

DATA (LOC(K4), K4 = 876, 929)/

*DELETE DIR3A.7, DIR3A.8

* 3760 , 3880, 3900, 3920, 3940, 3960, 3980, 
* 3985, 3990, 3991, 3992, 3993, 3994, 3995, 
* 3996, 3997, 3998, 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, INSEOE,
*ISTEIRUDDINAWRUDDOMM45(40)
*DELETEC82477.67,C82477.70
*DELETEC82477.71,C82477.71
*DELETEC82477.72,C82477.72
IF(DTAT1.EQ.0.0)GO TO 130
*INSEPTC82477.77
130 IF(ISTEIRUDD.EQ.1)GO TO 29
IF(INWRUDD.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/DELRR)
*DELETEC82477.78,C82477.78
*INSEPTC82477.79
29 IF(ITR.LE.TST)GO TO 35
ETADES = ETAMOS + ETART1*(TR-TST) 
GO TO 31
35 ETADES = ETAMOS 
GO TO 31
*DELETEC82477.93,C82477.95
GO TO 31
*INSERTC82477.103
31 IF(ETADES.GT.ETAMAX) ETADES = ETAMAX
IF(ETADES.LT.ETAMIN) ETADES = ETAMIN
*DELETEAUTS.447,AUTS.448
34 IF(IAP.GT.2)GO TO 500
DELRR = DELRNN
*DELETEAUTS.454,AUTS.454
500 IF(IRUDD.EQ.1)GO TO 5 
GO TO 150
5 IF(INWRUDD.EQ.1)GO TO 6
IF(ITR.LE.TST)GO TO 27
DELRDE = DELRDI + DELRDRD*(TR-TST) 
GO TO 101
27 DELRDI = DELRD 
GO TO 101
150 DELRN = DELRNN 
PSIE = PSIPD
*INSERTAUTS.457
GO TO 101
6 IF(ITR.LE.TST)GO TO 110
DELRDE = DELRDI + DELRDRD*(TR-TST) 
GO TO 101
110 IF(ABS(DELRDE).GT.0.0)GO TO 120
DELRD = DELRD 
GO TO 101
120 DELRDI = DELRDE 
*INSERTAUTS.460
CONTINUE

*DELETE AUTS.582,AUTS.583
If(IRUDD.EQ.1)GO TO 49
If(DLRDE.LT.DLRD)DELRD1=-DLRRD
49 DELPD1=DELA

*DELETE AUTS.607,AUTS.607
61 If(IAP.EQ.4)GO TO 62
If(Abs(DELRD1*DELS)*.GE.Abs(DELRDE-DLRD))GO TO 62

*INSERT FLARE1.38
  *DM21(100)
  *INSERT AUTPR1.35
  *DM41(100)
  *INSERT THAUTC.32
  *DDM9(100)
  *INSERT ENGFL.29
  *DM8(100)
  *INSERT CENGL.9
  *DM51(100)

*IDENT SDDMODS

*DELETE C11778.2,C11778.2
34 DD 35 IF=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,.017453291/

*INSERT LGEAR1.242
If(INDNWS.EQ.1.AND.I.EQ.1)GO TO 301

*INSERT LGEAR1.245
301 MA(I)=FTRY(I)*TMP(I)*SIN((PSIPD+ETADES)*DEGRAD)+
     1 FTRX(I)*TMP(I)*COS((PSIPD+ETADES)*DEGRAD)
     GO TO 201

*DELETE C11778.54,C11778.54
*DDM34,FGPY,DELN,ETADES,DDM35(15),CONFMR,DELPMM,VPDWRM,CONEM,CTWOM,
 *CTHRREM,CFOUPM,DDM36(23)

*INSERT LGEA3C.142

*DELETE CB2377.21,CB2377.21

*DELETE C11778.49,C11778.49

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)

*DELETE C11778.67,C11778.69
OMET(I,1)=(PG11*RDYG(I)+PG13*RDZG(I))/(COS((PSIPD+TMPETA)*
     1 DEGRAD))/[(PG20(I)-DELT(I))]

*DELETE C11778.74,C11778.74
IF(DELTA(I) .EQ. 0.0) GO TO 48
*DELETE C82477.32,C82477.32
SIDEMU = CONFRI*(ABS(DELNM))**DELPWR*VAXLE(I)**VPDWR
*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))
DELM = PSIPE-ETA VEM*RADDEG
IF(DELTA(I) .EQ. 0.0) FGPY = 0.0
IF(DELTA(I) .EQ. 0.0) GO TO 48
IF(DELNM .EQ. 0.0.O)DP.XG77FL.EQ.0.0) GO TO 1100
SIDUM = CONFRM*(ABS(DELNM))**DELPRM*VAXLE(I)**VPDWRM
GO TO 1800
1100 SIDUM = 1.0F-6
1800 IF((DELTA(I)/2.0) .GT. 0.0875) GO TO 1300
CRNP RM = CONEM*DELTA(I)-CTWOM*DELTA(I)**2.
GO TO 1700
.1300 CRNP RM = CTHREM-CFOURM*DELTA(I)
1700 YAWPMM = ABS(CRNP RM*DELM/(SIDUM*FTRZ(I))))
IF(YAWPMM .LE. 1.5) GO TO 1500
FGPY = SIDUM*FTRZ(I)*COS(DELM*DEG RAD)
GO TO 1600
1500 FGP Y = ((YAWPMM-(4.0/27.0)*YAWPMM**3.0)*SIDUM*FTRZ(I))
1 *COS(DELM*DEG RAD)
1600 IF(DELM .GT. 0.0) FGPY = ABS(FGP Y)
IF(DELM .LT. 0.0) FGPY = ABS(FGP Y)
DFTRXM = FGPY*SIN(ETA VEM)
DFTRYM = FGPY*COS(ETA VEM)
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, 6HCONF RM, 6HDELP RM, 6HV PDWRM, 6HC O NEM, 6HT WOM, 6HTHREM, 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 ACBLK
*INSERT TOLA.27
*COMPLK ACBLK

C********************************************************** ACTIVE VARIABLES **********************************************************
COMMON /ACTIVE/ AMUH ,ACON ,BCON ,CCON ,DCON,
 1,APINT(5) ,AP2TO(5) ,AREA1(5) ,AREA2(5) ,AREA3(5)
 2,AREM0(5) ,ARED3(5) ,RLFORT(5) ,RLM(5) ,RUMU(5)
 3,BET A ,CDOC(5) ,CDMOE(5) ,CDSV(5) ,CKFFOR(5) ,CD3(5)
 4,COEF ,COEF0(5) ,COEF3(5) ,COPA,
 5,COSVI(5) ,CSV3(5) ,DELT ,DELTX(5) ,DELTX1(5) ,D LTX10(5)
 6,DF(5) ,DP1(5) ,DSV(5) ,DSTOP ,DIOTA ,DMTANH(5)
 7,ENUP(5) ,EPSL0(5) ,EPSSL(5) ,ETASV ,FSTOPI ,FST0P
 8,FFORT(5) ,FOAHT(5) ,FONST(5) ,FORCHT(5) ,FORSTT(5) ,FST0P(5)
 9,GAHAI(5) ,GANAI ,GMR ,HM(5) ,ICOSV(5) ,IDEACT
 10,IFRI(5) ,IXSVH(5) ,IXSVL(5) ,IOPCO(5) ,IPASS(5) ,ISET(5)
 11,IXSVH(5) ,IXSVL(5) ,IFSTOP(5) ,ISTROK(5) ,KAPT(5)
 12,NAC(5) ,NTER ,OMRUN ,PATH
 13,PERCNT(5) ,PGAHAC(5) ,PGALAC(5) ,PGA1I(5) ,PGA1T(5)
 14,PGAII(5) ,PGA2T(5) ,PGA3(5) ,PGA3T(5)
 15,PI(5) ,PO(5) ,QC(5) ,OO(5) ,QPMJ(5) ,QPMJS(5)
 16,OSV(5) ,OSVCU(5) ,OSV(5) ,OSV1(5) ,OSV3(5) ,OS1(5)
 17,OTS(5) ,OTCLER ,RCLSV(5) ,RDSL(5) ,RH0H ,SBOT
 18,SIPA ,SA(5) ,RESAI(5) ,TAFU ,TC1 ,TC2

COMMON /ACTIVE/ TC3,TC4,TC5
 1,VBOL1(5) ,VOL2(5) ,VOL2T(5) ,VOL3(5) ,VOL3T(5) ,VC
 2,W1 ,WF1(5) ,W1F0R(5) ,W1FOR(5) ,WBIAS(5) ,XVALVE(5)
 3,XKA(5) ,XKF(5) ,XKSV(5) ,XLPSV(5) ,XLPSV3(5) ,XSTHR
 4,XMA(5) ,XMA1(5) ,XMA2(5) ,XMA3(5) ,XMA4(5) ,XMA5(5)
 5,XMA6(5) ,XMA7(5) ,XMA(5) ,XMA9(5) ,XMA10(5) ,XMA11(5)
 6,XMU ,XSCOM(5) ,XSTOT(5) ,XSV(5) ,XSVDOT(5) ,XSVDD(5)
 7,XSVDD(5) ,XSVDM(5) ,XSVDMX(5) ,XSVMAX(5) ,XSVMIN(5) ,XDMAX(5)
 8,X/DDMN(5) ,VOLACI(5) ,VOLAT(5) ,VOLAN(5) ,VOLANT(5)
 9,W3 ,WS1 ,WSV3 ,Z1TAC1 ,ZETAC2 ,ZSSC

********************************************************** CSCHMODS **********************************************************
IDENT CSCHMODS
*INSERT EXE.42
*CALL ACBLK
*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 = NNUM + 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(II,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(II,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.93
      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, 5HGAHAN5, 5HCOEFO
   DATA(ACOVAR2(I), I-1,8 )/2HQQ,4HVCUM,6HBLFORT,6HBUFFORT,5HFSIZE,
   * 6HFORCHT, 6HFOAHST, 6HFOHHST/
   DATA(ACOVAR3(I), I-1,8 )/5HCF0R, 5HFSTOP, 6HFORSST, 6HXVALVE, 3HXXSV,
   * 5HDSTEM, 6HDTLTX1, 6HDLTX10/
   DATA(ACOVAR4(I), I-1,8 )/2HDF, 3HDP1, 2HPS, 2HP1, 3HHSV, 4HOSV, 4HOSV1,
   * 4HO5V3/
   DATA(ACOVAR5(I), I-1,8 )/6HEPSIL0, 3HHMM, 6HDMTANH, 5HXSTOT, 5HICOSV,
* 4HENUP,5HWFOR,5HIOPCD/
DATA(ACOVAR6(I),I=1,8) /6HIIXSVH,6HIIXSVL,5HIXSVH,5HIxSVL,5HIPASS
*,3MCAC,4HRESA,2HSA/
DATA(ACOVAR7(I),I=1,8) /6HVELDEC,6HWFORR,6HIDEACT,4HZSSC,4HCOPA,
* 4HSIPA,5HDSOTP,6HFSOPK/
DATA(ACOVAR8(I),I=1,8) /5HWFORT,6HVOLAHT,6HVOLANT,5HOSVCU,3HXA,
* 4HXMLA5,4HXMLA8,5HXMLA11/
DATA(ACOVAR9(I),I=1,8) /5HXSVD,5HXSVD,5HXSVD,2HPR,4HIFRI,
* 6HIFSTOP,6HISTROK,4HISET/
*INSEPT
CR2477.66
IF(ABS(INDLG),NE.3) GO TO 50
CALL STFL(2,8,ACOVAR1)
DO 33 I=1,NSTRUT
CALL STOVAR8(VOL1(I),VOL2(I),VOL3(I),PGA1(I),PGA2(I),
* PGA3(I),GAMAH(I),CEOFO(I))
33 CONTINUE
CALL STFL(2,8,ACOVAR2)
DO 34 I=1,NSTRUT
CALL STOVAR8(GO(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 STOVAR8(DF(I),DP1(I),PS(I),P1(I),QSV(I),QSVN(I),QSV1(I),
* QSV3(I))
35 CONTINUE
CALL STFL(2,8,ACOVAR4)
DO 36 I=1,NSTRUT
CALL STOVAR8(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 STOVAR8(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 STOVAR8(XSVDOG(I),XSVDOG(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 STOVAR8(EPSILO(I),HMM(I),DMTANH(I),XSTOT(I),FLOAT(ICOSV(I)),
* ENUP(I),WLFOR(I),FLOAT(IOPCD(I)))
39 CONTINUE
CALL STFL(2,8,ACOVAR6)
DO 40 I=1,NSTRUT
CALL STOVAR8(FLOAT(IIXSVH(I)),FLOAT(IIXSVL(I)),FLOAT(IIXSVH(I)),
* FLOAT(IIXSVL(I)),FLOAT(IIXSVH(I)))
* FLOAT(ISVL(I)), FLOAT(IPASS(I)),
* FLOAT(NAC(I)), RESA(I), SA(I))
40 CONTINUE
   CALL STFL(I,8,ACOVAR7)
   CALL STOVAR(8, VELDEC, WLDPR, FLOAT(I), ZSSC, COPA, SIPA, DSTOP,
* FSTOPK)
50 CONTINUE
* INSERT SDFLG, 103
   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), PGA(I1), 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)
* 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,
DCN1, DCL2, DCM2, DCN2, DCL3, DCM3, DCN3, DCM4(99), FXB7P,
CDM4(3), FYB7P(4), FZB7P, DM5(17), GXB7F, DM6(9), GZB7F,
DM7(218), INDSTE(48), PHIPD, INDSTEARM, PI77P, INDSTE2(156), THTPD,
CINDSTEARM, TIMQ, MB8(287), PI77F, PI77R(2), PI77R1(2), DM9(4),
CI77F(2), CI77R1(2), DMI0(4), RI77R(2), RI77R1(2), DMI1(48),
CXG77F(2), CXG77F1(12), YG77F(2), YG77F1(12),
CZG77F(2), CZG77F1(2), DUM13(52),
CNSTRUT, MASS(5), RX(5), RY(5), RZ(5), THETA(5), ERDFG, GR,
CNTIRES(5), RZERO(5), W(5), DELTAP(5), MOMENT(5),
CCE(5), CE(5), C2C(5), C2E(5), NVGPT, NPP, MB(5), RLT, INDELA,
CES(5), SR(5), S021(2), S022(2), S023(2), S024(2), S025(2)
COMM/DIRCOM/
CSD11(2), S012(2), S013(2), S014(2), S015(2),
CALL ACNBLK
DIMENSION CD3(5), SD1(2,5)
EQUIVALENCE(SD11(1),SD1(1,1)), (DM5(16),GREFF)

DATA STATEMENTS TO SET VALUES OF INPUT VARIABLES

DATA AMUH/0.00018/, BETA/14400000.0/
DATA BLMU,BUMU/10*0.15/
DATA AREA1,AREA2,AREA3/5*0.01139,5*0.32167,5*0.00698/
DATA AREM0,AREOM/5*0.0066,5*0.19635,5*0.0/
DATA CDMJ,CDM0,CD3/5*0.9,5*0.14197,5*0.61/, CFFOR/5*50.0/
DATA CDSV,DELTA,DIOTA/5*0.62,0.0001,0.0/
DATA EPSLO,EPSSL,EPSR0L,ETASV/5*200.0,5*0.0000475,2*3.4e-04/
DATA GMAM,GAHAH,GNR/1.06,5*2.36,5*0.01/
DATA IC0S1,IFSTOP/10*0.1/
DATA IOEACT,II,IOAPT,IOBUN,IOPCNT/7*0.07575,5*0.07575,5*0.0/
DATA IOAT,IOVAl,IOVAl,IOVAl/5*0.36379,5*0.47164,5*0.0/
DATA I0H0,0LFD0,0LFD0/1.626,5*500.0,0.0/
DATA DS,V,RCLS,WSV1,WSV3/5*1.1875,5*0.000475,2*3.480/
DATA WC,WCI,WSV,XSV1,XSV3/1263.0,5*251.3,5*0.0250,5*0.026675/
DATA XK1,KF,XPV3,XPV5/5*0.04,5*60.0,5*0.0,5*0.0/
DATA XBIAS,XSCOM,XSVDM/XSVDM/5*-0.0014,5*1.206,5*-30.1,5*30.1/
DATA XDMMN,XDEMN/5*1.0,5*1.0,5*1.0/
DATA XSVMAX,XSVMIN,ZETAC1,ZETAC2/5*0.100,5*0.100,5*0.100/
DATA VOLAC1,VOLANI,QUOMPS/5*1.3333,5*0.26675,5*0.0005/

IF (TIME .GT. DELT) RETURN.

ACON=1./(WSV*WSV*WC)
BCON=1./(WSV*WSV)+2*ETASV/(WSV*WC)
C
C
C

CCON=2.*ETASV/WSV+1./WC
DCON=1.
XMU=AMUH/0.0000209
OTOLER=0.0001
SBFOT=0.0
VELDEC=0.
OMRUN=OMRUN*0.01745329
II = 0
C
DO 100 I=1,NSTRUT
REDSLIP(I) = 100000.0
ISTPOK(I)=0
NAC(I)=0
VOLANT(I) = VOLANI(I)
VOL1(I)=VOL1(I)
VOL2(I)=VOL2(I)
VOL3(I)=VOL3(I)
OSVCU(I)=0.0
DF(I)=0.
DEXTX(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.
XSVD0D(I)=0.
XSVD0D(I)=0.
XSVD0T=0.0
DP1(I)=0.
COEF3(I)=CD3(I)*SORT(2.*GREFF/GAMAH(I))
C
C
NOTE: SUBROUTINE 'PHLOZ2' COMPUTES INITIAL PRESSURES AND FLOWS IN
UNITS OF INCHES.
COEF=CDSV1(I)*SORT(2.*GREFF/GAMAH(I))*144.
CSV1(I)=COEF*WSV1
CSV3(I)=COEF*WSV3
PS(I)=PGAHAC(I)/144.
PR(I)=PGALAC(I)/144.
OC(I)=0.
CALL PHLOZ2(PS(I),PR(I),XSV(I),QC(I),XLPSV1(I),XLPSV3(I),RCLSV,DSV & ,CSV1(I),CSV3(I),XMU,OTOLER,NITFR,P1(I),Q1(I),Q2(I),Q3(I))
PGA11(I)=P1(I)*144.
PGA21(I)=PGA11(I)
PGA31(I)=PGA21(I)
QSV1(I)=Q1(I)/1728.
QSV3(I)=Q2(I)/1728.

C

PGA1T(I)=PGA11(I)
PGA2T(I)=PGA21(I)
PGA3T(I)=PGA31(I)
QD(I)=0.
QSV(I)=QSV1(I)-QSV3(I)
VCUM(I)=0.

C

AP2TO(I)=PGA21(I)+PATM
FFORT(I)=0.0
FORST(I)=0.0
WFORT(I)=0.0
FONHST(I)=0.0
FORCHT(I)=0.0
XVALUE(I)=XSV(I)

100 CONTINUE
C

CALL SETUP
RETURN
END

*IDENT ALGEAR
*INSERT ACTINIT.145
*DECK ALGEAR
SUBROUTINE ALGEAR1

*************** FATCLA VARIABLES ***********************
COMMON/DIRCOM/DM1(115),ALPHAD,DM1A(20),AMASS,DM2(147),DCL1,DCM1,
DCN1,DCL2,DCN2,DCL3,DCN3,DCN4,FXB7P,
CDUM4(3),FYB7P(4),FXB7P,DM5(17),GXB7F,DM6(8),GBZ7F,
CDM7(218),INDSTE(48),PHIPD,INDSTE1(23),PSIPD,INDSTE2(156),THTPD,
CINDSTE3(5),TIME,DM8(287),PI77R(12),PI77R(2),DM9(4),
CO177R(2),DI77R(2),DI10(4),RI77R(1),RI77R(2),DI11(48),
CXG77F(2),XG77F1(12),YG77F(2),YG77F1(12),
CZG77F(2),ZG77F1(2),DM13(52),
CNSTRUT,MASS(5),RX(5),RY(5),RZ(5),THETAD(5),EPDEG,GR,
CNIRES(5),ZERO(5),W(5),DELTAM(5),MOMENT(5),
CA2(5),IL,STT(5),ES2(5),S2L(5),MASS2(5),MUS(5),
CCC(5),CE(5),CC(5),C2E(5),CVGPT,NPP,MB(5),RLT,NDelta,
CES(5),SR(5),SD21(2),SD22(2),SD23(2),SD24(2),SD25(2)
COMMON/DIRCOM/
CSD11(2),SD12(2),SD13(2),SD14(2),SD15(2),
C****************************************ACTIV** CODE*****************************************
EQUIVALENC* (DM5(16), GREFF)
DATA IFRI/5*0, FSTO*PK/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, IOPCD, OSVN/10*1, 5*0, 0/
Q1(T1, T2)=SIGN(1., (T1-T2))*SORT(ABS(T1-T2))
C****************************************ACTIV** CODE*****************************************
C
C MAIN COMPUTATIONAL AREA
C
C R 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****************************************ACTIV** CODE*****************************************
C
C START ACTIVE GEAR CALCULATIONS
C
C COPA = COS((THTPD+DIOTA)*DEGRAD)
C SIPA = SIN((THTPD+DIOTA)*DEGRAD)
C 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(HM(1) .EQ. 0.) GO TO 90
IF(QMRUN .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(QMRUN)) 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)+EPSLSLP*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
C*********** EPSILON(I)=EPSROL(I)************
90 CONTINUE

C**********************************************************************************************
DO 65 I=1,NSTRUT
WFORT(I) = (SQRT(FXB7P*FXB7P+FYB7P*FYB7P+FZB7P*FZB7P))/(NSTRUT-1)
& +((FPSST(I)*COPA)
65 CONTINUE
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(I,I)
VOL3T(I)=VOL3I(I)+AREA3(I)*S(I,I)
VOL2T(I)=VOL2I(I)-(AREA2(I)-AREA1(I)+APINT(I))*S(I,I)+(VOL3T(I)
X -VOL3I(I))-VCUM(I)
PGA2T(I)=AP2TO(I)*((VOL2I(I)/VOL2T(I))*GAMAN)-PATM
IF(SD1(I,I) .EQ. 0.0)GO TO 104
PGA3T(I)=((COEF3(I)*AREO3(I))**2*PGA2T(I)-SD1(I,I)/ABS(SD1(I,I))
X *((SD1(I,I)*AREA3(I))**2)
6/((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)*SORT(Abs(2.*GREFF/GAMAH(I)))
IF(PGA2T(I) .GE. PGA1T(I))COEFO(I)=
* COMOE(I)*SORT(Abs(2.*GREFF/GAMAH(I)))
IF(SD1(I,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(DELT(A(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(I,I))/(SLEN1(I)+S(I,I)))*1.0
BUFORT(I)=FONHST(I)*((SLEN2I)-S(I,I))/(SLEN1(I)+S(I,I))
FFORT(I)=BUMU(I)*ABS(BUFORT(I))+BLMU(I)*ABS(BLFORT(I))
140 CONTINUE
C COMPUTE SHOCK STPUL Charging Force
IF(S(I,I) .GT. 0.0)GO TO 142
FORCHT(I) = PGA1T(I) * AREA1(I) + PGA2T(I) * (AREA2(I) - AREA1(I)) - PGA3T(I) * AREA3(I) + FFORT(I) + CFFOR(I)

C COMPUTE NORMAL AND AXIAL HUB TO SHOCK STRUT FORCES AT HUB

142 FORNSST(I) = SORT(DFX(I)**2 + DFX(I)**2) ** 0.5 * MESS(I) * GREEF * SIFA + SBFOT

IF(ABS(FT(I)) * LE. FORCHT(I) * AND. S(I,I) * EQ. 0,0) GO TO 150

GO TO 801

150 CONTINUE

FORSST(I) = FT(I)

SD1(I,I) = 0.0

IF(I,E Q. 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(I,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(I,I)))

IFRI(I) = 1

284 IF(S(1,I) * LE. DSTOP * AND. SD1(I,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(I,I) = 0.0

SD1(I,I) = 0.0

S(I,I) = 0.0

DP1(I) = 0.0

FFORT(I) = 0.0

VCUM(I) = 0.0

OD(I) = 0.0

II = 0

C CALL VIPK4(I1,NT,C1 SPEC,C1MAX,IERR,VAR,CUVAR,DER,ELE1,ELE2,

C ELT,ERRVAL,DERSUB,CHSUB,ITEXT)

GO TO 902

904 CONTINUE

IF(IFSTOP(I) * NE. 0) GO TO 906

905 DSTOP = S(I,I)

FSTOPK = 2.0 * MESS(I) * SD1(I,I)**2 / DSTOP**2

906 IF(S(1,I) * LE. DSTOP/2.0) GO TO 908

907 FSTOP(I) = - FSTOPK * (DSTOP - S(I,I))

GO TO 909

908 FSTOP(I) = - FSTOPK * S(I,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(I,I) * EQ. 0,0) GO TO 500

IF(SD1(I,I) * LT. 0.0) GO TO 470
GO TO 471

470 FFOR(I)=-FFORT(I)
CFFOR(I)=-CFFOR(I)

471 FORSST(I)=-(PGA1T(I)-PGA2T(I))*(AREA1(I)-APINT(I))
X=PGA2T(I)*AREA2(I)
1+CFFOR(I)*DMTANG(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(I,I) .LE. 0.0)290,295
290 IF(IOPCO(I) .EQ. 1)GO TO 295
IF((PGA1I(I)-1000.0)*LT.PGA1T(I),AND.
X-PIG(A1T(I)+1000.0))299,298

299 IF(I*XVALUE(I) .NE. 0.0)GO TO 311
IF(IPASS(I) .EQ. 1)GO TO 296
XVALUE(I)=XSV(I)*XMA11(I)+XBIAS(I)
IPASS(I)=1
GO TO 294

298 IF(ICSOTT(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(S02(I,I) .LE. 0.0 .AND. ICSOTT(I) .EQ. 1)GO TO 311
IF(IOPCO(I) .EQ. 1)GO TO 295
IF(PGA1T(I) .LT. PGA1I(I))292,293

292 IF(I*XSV(L(I)) .EQ. 1)GO TO 294
XVALUE(I)=XVALUE(I)+XSVDMX(I)*DEL*PERCNT(I)
IF(XVALUE(I) .LE. -0.1)300,294

300 XVALUE(I)=-0.1
IXSVL(I)=1
GO TO 294

293 IF(I*XSVHI(I) .EQ. 1)GO TO 294
XVALUE(I)=XVALUE(I)+XSVDMN(I)*DEL*PERCNT(I)
IF(XVALUE(I) .GE. 0.1)302,294

302 XVALUE(I)=0.1
IXSVH(I)=1

294 CONTINUE
II=0

C CALL VIRK4(I,I,N,NT,CI,SPEC,CIMAX,IER,VAR,CUVAR,DER,ELE1,ELE2,
C 1 ELT,ERRVAL,DERSUB,CHSUB,ITEXT)

DLTXD0(I)=0.0
ICOSVI(I)=1

296 IF(NFOR(I) .GT. 0.0 .AND. S(I,I) .LE. 0.0)GO TO 410
311 IF(NAC(I) .EQ. 1)GO TO 307
IF(I*XSVH(I) .EQ. 1)GO TO 305
XVALUE(I)=XVALUE(I)+XSVDMN(I)*DEL*PERCNT(I)
IF(XVALUE(I) .LE. 0.0)305,400
305  XVALVE(I)=0.0
IIXSVL(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)308,300,400
308  XVALVE(I)=0.0
IIXSVL(I)=1
400  II=0
CALL VIRK4II,N,T,C,SPC,CMAX,IERR,VAR,CMUAR,DER,ELE1,ELE2,
C 1 ELT,ERRV,DERSUB,CHSUB,ITEXT)
410  IF(XVALVE(I) .NE. 0.0)GO TO 295
ICSV(I)=0
DELTX(I)=DELTX(I)*XKF(I)
XMA(I)=(DF(I)+DELTX(I))*XMA(I)
XMAII(I)=XMA(I)
XSV(I)=XSV(I)*XMAII(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
IOPCO(I)=1
CALL PHLOZ2(PS(I),PR(I),XSV(I),QC(I),XLPSV1(I),XLPSV3(I),RCLSV(I),
& DSV(I),CSV1(I),CSV3(Y),XUM,OTOLER,NITER,P1(I),QS1(I),QS3(I))
C
295  IF(ISTROK(I) .EQ. 1 *AND. S(I,I) .GT. 0.0)IOPCO(I)=0
C
ENUP(I)=.5*AMASS*ZG77FL(I)**2
ENUP(I)=ENUP(I)/(NSTRUT-1)
IF(HMM(I) .EQ. 1.0)GO TO 130
SA(I)=0.
IF(WFORT(I) .LT. 0.0) XSTOT(I)=ENUP(I)/((-WFORT(I))*COPA)
IF(WFORT(I) .GE. 0.0) XSTOT(I)=1.0E20
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(I,I)) .OR. RESA(I) .EQ. 1.0)SA(I)=1.0
RESA(I)=SA(I)
IF(SA(I) .EQ. 0.0 .OR. HMM(I) .EQ. 1.) GO TO 130
WLFOR(I)=-WFORT(I)
VEDEC=((WLFOR(I)+WLFORR)/2.0*(WLFOR(I)-WLFORR))/(AMASS*REDSLP(I))
WRITE(6,121)TIME,WLFOR(I),VEDEC
121  FORMAT(50H ACTIVE CONTROL INITIATED...TIME, WLFOR, VEDEC = ,
& 1 3E13.5)
HMM(I)=1.
130  IF(S(I,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))
C EPSILO(I)=-(-WFORT(I))
IF(-WFORT(I)).LT.(WLFOR(I)-EPSILO(I)) DF(I)=(WLFOR(I)-
C EPSILO(I))=-(-WFORT(I))
IF(-WFORT(I)).LE.(WLFOR(I)+EPSILO(I)).AND.
C -WFORT(I).GE.(WLFOR(I)-EPSILO(I)))
1 457,456
457 IF(S(I,J) .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(I,J) - XSCOM(I)
IF(S(I,J) .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)+DELTX(I))*XKA(I)
IF(GNR.EQ.1. .AND. XMA(I) .GT. 0.) XMA(I) =
1 XMA(I)*SORT((PGAHT(I)-PGALAC(I)))
X /{PGAHT(I)-PGALAC(I))}
C NOTE: SUBROUTINE 'FLOZ2' COMPUTES THE FLOWS FROM THE PRESSURES
C IN UNITS OF INCHES.
PI(I) = PGAHT(I)/144.
C COMPUTATION OF HIGH PRESSURE ACCUMULATOR NITROGEN VOLUME
C AND ACCUMULATOR PPRESSURE
VOLANT(I) = VOLANT(I)+OSVN(I)*DELT-OPUMPS(I)*DELT
PS(I) = (((PGAHT(I)+PAM)*(VOLANI(I)/VOLANT(I))**GAMAN)-PAM)/144.0
IF(PS(I) .GE. 3000.0) 464,465
464 PS(I) = 3000.0
VOLANT(I) = VOLAN(T(I))
465 VOLANT(I) = VOLACT(I)-VOLANT(I)
IF(VOLANT(I) .LE. 0.0) 466,467
466 WRITE(6,1050)TIME
1050 FORMAT(1HO//45H ACCUMULATOR OIL VOLUME INSUFFICIENT AT TIME=E16.8
1 //)
CALL LGEAR6
STOP 500
467 CONTINUE
CALL FLOZ2(PS(I),PR(I),PI(I),XLPSV1(I),XLPSV3(I),RCLS(I),DSV(I),
C XSV(I),OS1(I),QS3(I),CSV1(I),CSV3(I),XMU)
QSV1(I) = QSV(I)/1728.
QSV3(I) = QSV3(I)/1728.
C 450 CONTINUE
QSV1(I) = 0.0
QSV3(I) = 0.0
QSV1(I) = QSV1(I)-QSV3(I)
IF(QSV(I) .LT. 0.0) NAC(I)=1
IF(OSV(I) .GT. 0.0) NAC(I)=2
IF(NAC(I) .EQ. 2) GO TO 462
OSV(N)=OSV(I)
GO TO 463
OSV(N)=0.0

IF(SD1(I,I).LT. 0.0 .AND. PGA1T(I).LE. -1600.0) PGA1T(I)=-1600.0

CONTINUE
I1 = I+3*NSTRUT
I2 = I+4*NSTRUT
I3 = I+5*NSTRUT
CALL INTEG(LA(I1),DP1(I))
CALL INTEG(LA(I2),QO(I))
CALL INTEG(LA(I3),OSV(I))

IF(SD1(I,I).EQ.0.0 .AND. FT(I).LE.ABS(FORSS(I))) FORSS(I)=-FT(I)
AA(I) = (FT(I)+FORSS(I))/MASS(I)
SD2(I,I) = SR(I)+AA(I)-GZ(I)
HT1=HT

IF(SD1(I,I).LT.76,77,78
76 TTIME=S(I,I)/ABS(SD1(I,I))
79 IF(TTIME.GE.HT) GO TO 77
HT1=TTIME
GO TO 77
78 TTIME=(SB(I)-S(I,I))/SD1(I,I)
GO TO 79
77 CONTINUE
IF(S(I,I).GT.(-ES(I))) GO TO 50
WRITE(6,49) I, S(I,I)

FORMAT(58X,4H-ES(I),12.4H) EXCEEDED IN ALGFA/
C58X,2HS(I,I,14H) = E15.7)
S(I,I)=0.5*ES(I)
SD1(I,I)=-1.0E-10
SD2(I,I)=-1.0E-10

50 IF(S(I,I).LE.ES(I)) GO TO 51
51 IF(S(I,I).LE.(SB(I)-ES(I))) GO TO 55
52 IF(S(I,I).LE.(SB(I)+ES(I)) ) GO TO 52
WRITE(6,53) I, S(I,I)
53 FORMAT(58X,4H ES(I),12.4H) EXCEEDED IN ALGFA/
C58X,2HS(I,I,14H) = E15.7)
S(I,I)=0.5*ES(I)

52 IF(SD1(I,I).GT.0.) SD1(I,I)=0.
IF(SD2(I,I).LT.0.) GO TO 55
SD2(I,I)=0.
GO TO 55
51 IF(SD2(I,I).LT.0.) GO TO 30
50 IF(SD1(I,I).LT.0.) SD1(I,I)=0.
GO TO 55
30 SD2(I,I)=0.
SD1(I,I)=0.
CONTINUE
I2=2*I+NSTRUT-1
I1=I2+1
CALL INTEG(LA(I2),SD2(1,I))
CALL INTEG(LA(I1),SD1(1,I))

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(I)=ZERO(I)-DELT(A(I))
IF(CASK(I).GT.1.E-10)GO TO 200
IF(INDNS.W.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 FTRY(I)*TMP(1)*COS((PSIPD+ETADES)*DEGRAD)
GO TO 201
200 MA(I)=TMP(1)*SORT(FTRY(I)*FTRY(I)+FTRX(I)*FTRX(I))
MA(I)=SIGN(MA(I),-VAXLE(I)-OMET(I,I)*TMP(1))
201 AMAI(I)=MA(I)
IF(IB(I).NE.(-1))GO TO 48
OMETD1(I,I)=0.
OMET(1,I)=0.
GO TO 21
48 TMP(I)=0.
IF(OMET(I,I).NE.0.)TMP(1)=OMET(I,I)/ABS(OMET(1,I))
OMETD1(I,I)=(MA(I)-MP(I)*TMP(1))/(NTIRES(I)*MOMENT(I))
21 CALL INTEG(LA(I),OMETD1(I,I))

C ****************************************************************************************************
CONTINUE
C CALCULATION OF FTRA, FTRP, AND FTRC
SFTRY=0.
SFTRY=0.
SFTRY=0.
DO 70 I=1,NSTRUT
SFTRY=SFTRY+FTRY(I)
SFTRY=SFTRY+FTRY(I)
SFTRY=SFTRY+FTRY(I)
70 FTRA=RL(1,1)*SFTRY+RL(1,2)*SFTRY+RL(1,3)*SFTRZ
FTRP=RL(2,1)*SFTRY+RL(2,2)*SFTRY+RL(2,3)*SFTRZ
FTRC=RL(3,1)*SFTRY+RL(3,2)*SFTRY+RL(3,3)*SFTRZ
C CALCULATION OF MTX, MTY, AND MTZ
SMTRY=0.
SMTRY=0.
SMTRY=0.
DO 75 I=1,NSTRUT
SMTRY=SMTRY+MTPR(I)
SMTRY=SMTRY+MTPR(I)
75 SMTRY=SMTRY+MTPR(I)
MTX=RL(1,1)*SMTRY+RL(1,2)*SMTRY+RL(1,3)*SMTRY
MTY = RL(2,1) * SMRX + RL(2,2) * SMTPY + RL(2,3) * SMTRZ
MTZ = RL(3,1) * SMRX + RL(3,2) * SMTPY + RL(3,3) * SMTRZ

CALCULATION OF FXM, FYM, FXM, LM, MM, AND NM

FXM = 0.
FYM = 0.

DO 82 I = 1, NSTRUT
     DFXM = 0.
     DFYX = 0.
     DFM = 0.
     DMM = 0.
     DNM = 0.
     BFX = 0.
     BFY = 0.
     BFZ = 0.
     BLM = 0.
     BMM = 0.
     BNM = 0.
     NRP = (IL4-1) * NSTRUT + I
     DFXM = DFXM - MASS(I) * SXMOD(NBB) * GQD2(IL4)
     DFYX = DFYX - MASS(I) * SYMOD(NBB) * GQD2(IL4)
     DFM = DFM - MASS(I) * SZMOD(NBB) * GQD2(IL4)
     DMM = DMM - MASS(I) * GFORC2(NBB) * GQD2(IL4)
     DNM = DNM - MASS(I) * GFORC4(NBB) * GQD2(IL4)
     BFX = BFX + DFXM
     BFY = BFY + DFYX
     BFZ = BFZ + DFM
     BLM = BLM + DMM
     BMM = BMM + DNM
     TMP(1) = MASS(I) * SD2(1, I)
     FXM = FXM + TMP(1) * A32(I)
     FYX = FYX
     FZX = FZX + TMP(1) * A33(I)
     LM = LM + TMP(1) * A11(I) * RY(I)
     MM = MM - TMP(1) * RRCGY(I)
     BFX = BFX + TMP(1) * A13(I) * RY(I)
     FXM = FXM + BFX + FTRA
     FYX = FYX + BFY + FTRB
     FZX = FZX + BFZ + FTRC
     LM = LM + BL + MTX
     MM = MM + BMM + MTY

82
SUBROUTINE PHLOZ2(PS,PR,X,OC,LAP1,LAP3,RCLD,COEF1,COEF3,MU,QTDSR
   6,NITER,P1,Q1,Q3)
   C  *PHLOZ2* ......................................................... R. D. EDSON
   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
FLOW (QC). THE PARAMETERS REQUIRED IN THE 'CALL' STATEMENT ARE
THE SAME AS DESCRIBED IN SUBROUTINE 'FLOZE2', WITH THE FOLLOWING
ADDITIONAL PARAMETERS:
QC = THE FLOW RATE TO THE LOAD
QTOler = THE TOLERANCE ALLOWED IN CALCULATING FLOW RATES, FOR
DETERMINING WHETHER OR NOT THE SOLUTION HAS CONVERGED
 (.0001 IS TYPICAL)
NITER = THE NUMBER OF ITERATIONS REQUIRED TO CONVERGE TO A
SOLUTION (INTEGER)

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.
PI = PI
P1B=PS
CALL FLOZE2(PS,PS,P1A,LAP1,LAP3,RCL,D,X,Q1A,Q3A,COEF1,COEF3,MU)
CALL FLOZE2(PS,PS,P1B,LAP1,LAP3,RCL,D,X,Q1B,Q3B,COEF1,COEF3,MU)
IF(Q1A.EQ.0. .AND. Q3B.EQ.0.) GO TO 51
Q3A=Q3A+OC
Q3B=Q3B+OC
GO TO 50
51 PI=(PS+PS)/2.
Q1=0.
Q3=0.
GO TO 400
50 PI=PFN(P1A,P1B,Q3A,Q1A,Q1B,Q3B)
CALL FLOZE2(PS,PS,P1A,LAP1,LAP3,RCL,D,X,Q1,Q3,COEF1,COEF3,MU)
Q3=Q3+QC
IF(FLAG.LT.0.) GO TO 55
IF(P1.EQ.P1) GO TO 100
55 P1=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
SUBROUTINE FLOZE2(PS, PR, P1, LAP1, LAP3, RCL, D, X, Q1, Q3, COEF1, COEF3, MU)
     *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
     THE METHOD OF SOLUTION UTILIZES THE TURBULENT ORIFICE EQUATION
     AND THE EQUATION FOR FULLY-DEVELOPED LAMINAR FLOW THROUGH AN
     ANNULUS, WITH FULL ECCENTRICITY ASSUMED. FOR ORIFICE OPENINGS
     WHERE SOME OVERLAPPED LENGTH EXISTS, THE PROCEDURE IS TO CALC-
ULATE THE FLOW RATE BY BOTH EQUATIONS, AND THEN USE THE ONE
THAT GIVES THE SMALLEST ABSOLUTE VALUE AS THE ANSWER. FOR
OPENINGS WHERE NO OVERLAPPED LENGTH EXISTS, ONLY THE TURBULENT
ORIFICE EQUATION APPLIES.

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

*************** CALCULATE Q1 ***************
X2=LAP1-X
X4=SQRT(X2**2+RCL**2)
IF(LAP1 .LE. 0.) GO TO 99

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

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

*************** CALCULATE Q3 ***************
20 X1=LAP3+X
X3=SQRT(X1**2+RCL**2)
IF(LAP3 .LE. 0.) GO TO 199

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

NEGATIVE LAPS:
199 IF(X .GT. -LAP3) GO TO 110
Q3=X3*COEF3*Q12(P1,PR)
GO TO 120
ORGINAL PAGE IS
OF POOR QUALITY

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
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  DB1877.1, DB1877.1
     DIMENSION TITLE(20), BUF(550), NDIL(28), TBUF(550)
*DELETE  PLTDAT.6, PLTDAT.6
     1 BMM(2,550), CMGDS(6), DEPVAR(5), LINE(7), NDVA(5)
*IDENT  JMMODS
*DELETE  ACTINIT.40, ACTINIT.65
     DATA ICOSV, IFSTOP, II/II*O/
*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/XNAME(71),LOC(71)
DATA NAME/ 6HAMSU, 6HAPINT, 6HAREA1, 6HAREA2,
1 6HAREA3, 6HAREMO, 6HAREO3, 6HBETA, 6HBLMU,
2 6HRMU, 6HCOMDC, 6HCOMDE, 6HCDSV, 6HCD3,
3 6HCCFOR, 6HDELT, 6HIOTA, 6HDSV, 6HEPSIL0,
4 6HEPSROL, 6HEPSSSL, 6HEATSV, 6HGAMAH, 6HGAMAN,
5 6HGNR, 6HIDEACT, 6HKAPT, 6HDMRUN, 6HPATM,
6 6HPGAHAC, 6HPGALAC, 6HPGA1I, 6HPGAI2, 6HPGA3I,
7 6HPERCNT, 6HOPUMPS, 6HRCLSV, 6HRH0H, 6HTAUFL,
8 6HTC1, 6HTC2, 6HTC3, 6HTC4, 6HVOLACI,
9 6HVOLANI, 6HVOL1I, 6HVOL2I, 6HVOL3I, 6HWC,
1 6HWC1, 6HWSV, 6HWSV1, 6HWSV3, 6HWSVL,
2 6HLFORR, 6HXBIAI, 6HXDDMAX, 6HXDDMIN, 6HXKA,
3 6HXKFC, 6HXKSF, 6HXLSV1, 6HXLSV3, 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, 81, 77, 109, 141, 152,
2 175, 162, 163, 193, 200, 201, 212, 263, 274, 275,
3 321, 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, 598, 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/
 3HREM,1R,,IR,,1R,,IRE,6HENDACT,1R,,3HINT/
100 CONTINUE
 1 FORMAT(A6,1X,A3,1X,55R1,I1)
 2 FORMAT(A6,1X,A3,1X,55R1,I6)
 3 FORMAT(20HERROR.THE SYMBOL **,A6,
 1 26H** IS NOT IN THE DIRECTORY/1H)
 4 FORMAT(44HERROR.ILLEGAL CHARACTER IN NUMERIC FIELD **,1R1,2H**/)
READ(S,1) SYM,DP,IRA,INC
CALL LINES(I)
  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

*IDENT CSCMOD2

*DELETE ACOBK.11,ACOBK.11

*DELETE HHAC(5),FONHST(5),FORCHT(5),FORSST(5),FSTOP(5)

*DELETE ACOBK.16,ACOBK.16

*DELETE PERCN(5),PGAHR(5),PGALAC(5),PGAII(5),PGAII(5)

*DELETE ACOBK.31,ACOBK.31

*DELETE IMODE(5),CMASNG,VMASS(5)

*DELETE EXE.41,EXE.41

COMMON /HTCOM/ HT, HT1, HT2, INTI(5)

*INSERT EXE.280

STOP "FLIGHT TIME LIMIT"

*DELETE MIMIN.8,MIMIN.8

COMMON /HTCOM/ HT, HT1, HT2, INTI(5)

*INSERT MIMIN.10

C

DO 10 I=1,5

10 INDINT(I) = 1

*DELETE LGDET.9, LGDET.9

COMMON /HTCOM/ HT, HT1, HT2, INTI(5)

*DELETE LGEOAI.50, LGEOAI.50

COMMON /HTCOM/ HT, HT1, HT2, INTI(5)

*INSERT LGEOAI.81

DO 6 I=1,5

6 P1(I) = 0.0

P2(I) = 0.0

*INSERT ACTINIT.35

COMMON /HTCOM/ HT, HT1, HT2, INTI(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 /300.00/

DATA ICOSV, IPRI, IFSTOP, IPASS /2000/

DATA IOPCO, ISET /1001/

DATA IXSVH, IXSVL,IXSVH, IXSVL /2000/

C

*INSERT ACTINIT.75

OMRUN=ERDEG*0.01745329

*DELETE ACTINIT.76,ACTINIT.77

*INSERT ACTINIT.80

INDINT(I) = 1

*DELETE ACTINIT.127, ACTINIT.127

PGAII(1)=PGAII(1)

*INSERT ACTINIT.140
**ORIGINAL PAGE IS OF POOR QUALITY**

VMAS\(\text{S}(I)\) = \(\text{MASS}\)

*DELETE ALG\(\text{E}A\)R.32, ALG\(\text{E}A\)R.33

*G\(\text{D}\)2(20), D\(\text{M}\)20(20), D\(\text{M}\)21(20), S\(\text{LEN}\)1(5), S\(\text{LEN}\)2(5),
*G\(\text{AMA}\), \(\text{DUM}\)15(12), \(\text{INDN}\)W5, \(\text{DMD}\)22(10), ETADE\(\text{S}\),

*DELETE ALG\(\text{E}A\)R.54, ALG\(\text{E}A\)R.54

COMMON /HT\(\text{COM}/, HT, HT1, HT2, IND\(\text{INT}\)(5)

*DELETE ALG\(\text{E}A\)R.71, ALG\(\text{E}A\)R.76

EQUIVALENCE (D\(\text{M}\)5(16), GREFF), (D\(\text{M}\)2(27), AXP7F), (D\(\text{M}\)2(28), AX77F),
*(D\(\text{M}\)2(29), AYP7F), (D\(\text{M}\)2(30), AY77F), (D\(\text{M}\)2(31), AZP7F), (D\(\text{M}\)2(33), AZ77F),
*(D\(\text{M}\)1(37), AIYYBS)

DIMENSION INDE\(\text{ACT}\)(5), IP\(\text{STOP}\)(5), A\(\text{I}\)C(5), PGA1T(5)

*INSERT ALG\(\text{E}A\)R.96

C C\(\text{MAS}N\)G = AIYYBS/AMASS+AMASS*RX(1)*RX(1)/AIYYBS

C\(\text{MAS}N\)G = 7.9677

V\(\text{MASS}(1)\) = AMASS/C\(\text{MAS}N\)G

EN\(\text{G}\) = 0.5*AMASS*ZG77F1(1)*ZG77F1(1)

Z\(\text{DANT}\) = ZG77F1(1)-Q\(\text{I}\)77P*RX(1)

*DELETE ALG\(\text{E}A\)R.102, ALG\(\text{E}A\)R.103

C

18 DD 90 I=1,NSTRUT

IF(INDE\(\text{ACT}(I).EQ.1) GO TO 56

IF(INDE\(\text{ACT}(I).EQ.2 GO TO 80

*INSERT ALG\(\text{E}A\)R.105

IF(I.EQ.1 AND Z\(\text{DANT}\).GE.VE\(\text{L}\)DEC) GO TO 90

*INSERT ALG\(\text{E}A\)R.108

IF(I.EQ.1 AND Z\(\text{DANT}\).GE.VE\(\text{L}\)DEC+XG77F1(1)*T\(\text{AN}\)COMRUN)) GO TO 90

*DELETE ALG\(\text{E}A\)R.111, ALG\(\text{E}A\)R.113

INDE\(\text{ACT}(I)-1

56 IF(IND\(\text{INT}(I).EQ.0) GO TO 58

W\(\text{L}\)FOR(I)=W\(\text{L}\)FOR(I)-RED\(\text{S}L\)P(I)*HT

EPSILO\(\text{L}(I)=EPSILO\(\text{L}(I)+EP\(\text{SSL}\)P*HT

58 CONTINUE

IND\(\text{INT}(I)=0

*DELETE ALG\(\text{E}A\)R.117, ALG\(\text{E}A\)R.117

INDE\(\text{ACT}(I)=2

*DELETE ALG\(\text{E}A\)R.122, ALG\(\text{E}A\)R.125

C*** CALCULATION OF THE WING-GEAR INTERFACE FORCE (WF\(\text{ORT}\))

UN\(\text{SP}R\)NG = 0.0

DD 66 J=1,N\(\text{ST}R\)UT

IF(D\(\text{METD1}\)(J).NE.0.0) UN\(\text{SP}R\)NG = UN\(\text{SP}R\)NG+MA\(\text{S}S(J)

66 CONTINUE

D\(\text{W}F\(\text{ORT} = (-SQRT(AXP7F*AXP7F+AYP7F*AYP7F+

AZP7F*AZP7F)+GREFF)*(AMASS-UN\(\text{SP}R\)NG)

WF\(\text{ORT}(1) = D\(\text{W}F\(\text{ORT}+ARS\(\text{ST}(1)

DO 67 I=2,N\(\text{ST}R\)UT

WF\(\text{ORT}(I) = D\(\text{W}F\(\text{ORT}(/N\(\text{ST}R\)UT-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)*((VOLZ(I)/VOL2T(I))**GAMA)-PATM
*DELETE ALGEAR.152, ALGEAR.153
IF(IMODE(I).EQ.0 .AND. DDELTA(I).LE.0.0) 112, 113
112 QQ(I) = 0.0
GO TO 109
C
113 QQ(I)=COEQO(I)*((AREMO(I)-APINT(I))*Q1(PGA1T1(I),PGA2T(I))
IF(QQ(I).LE.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 QQ(I) = 0.0
VCUM(I)=0.0
AIC(I)=1.0
103 IF(QQ(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).LE.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.
1 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),SO2(I,I))
CALL INTEG(LA(I1),S01(I1,I))
C
IF(IMODE(I).EQ.0) GO TO 450
*DELETE ALGEAR.295,ALGEAR.296
IF(I.NE.1) GO TO 119
ENUP(I) = 0.5*AIYYBS*QI77R+QI77R+(ENC/GCMASNG)*
* (ZG77F1(I))/ABS(ZG77F1(I))
GO TO 120
119 ENUP(I) = ENCG/(NSTRUT-1)
120 CONTINUE
*DELETE ALGEAR.300,ALGEAR.300
IF(WFOR(I).GE.0.0 .OR. DDELTA(I).LT.0.0) XSTOT(I)=1.E20
*DELETE ALGEAR.302,ALGEAR.303
ZSSC(I)=WFOR(I)*SBI(I)
IF(XSTOT(I).LE.(ZSSC(I)-S(I,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),SD0(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(I,I).NE.0.0 .OR. AIC(I).NE.1.0) GO TO 20
PGAIT(I)=PGAIL(I)
PGAIL(I) =PGAIL(I)
19 DPI(I)=0.0
*DELETE ALGEAR.539,ALGEAR.546
I3 = I+3*NSTRUT
CALL INTEG(LA(I3),DPI(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),XSVD0T(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,6HPGA1T,6HPGA2T,
*DELETE CSCMODS.35,CSCMODS.35
*6HFORCHT,5HVMASS,6HFONHST/
*DELETE JMMODS.5,JMMODS.5
  1 OMET(1,1),WFOR(1),OSVCU(1),OSV(1),PGA1T1(1),PGA2T(1)
*DELETE JMMODS.6,JMMODS.6
  1 OMET(1,2),WFOR(2),OSVCU(2),OSV(2),PGA1T2(2),PGA2T(2)
*DELETE JMMODS.7,JMMODS.7
  1 OMET(1,3),WFOR(3),OSVCU(3),OSV(3),PGA1T3(3),PGA2T(3)
*DELETE JMMODS.8,JMMODS.8
  1 OMET(1,4),WFOR(4),OSVCU(4),OSV(4),PGA1T4(4),PGA2T(4)
*DELETE JMMODS.9,JMMODS.9
  1 OMET(1,5),WFOR(5),OSVCU(5),OSV(5),PGA1T5(5),PGA2T(5)
*DELETE CSCMODS.53,CSCMODS.53
CALL STOVAR(8,VOL1T(I),VOL2T(I),VOL3T(I),PGA1T(I),PGA2T(I),
*DELETE CSCMODS.59,CSCMODS.59
*FORCHT(I),VMASS(I),FONHST(I))
*DELETE CSCMODS.100,CSCMODS.100
CALL UPDAT(I,LAI(I),PGA1T(I),DU,DU,DU,DU)
*DELETE ACTIN.4,ACTIN.5
DIMENSION IPA(55),MSG(58),IDATA(656)
*DELETE PACK.3,PACK.3
DIMENSION IL(6)
*DELETE DIRACT.8,DIRACT.9
  4 6HEPSROl, 6HEPSLPL, 6HETASV, 6HFWORK, 6HGAMAH ,
  5 6HGFR, 6HKAPT, 6HOMRUN, 6HPATM ,
*DELETE DIRACT.10,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
*EOR
*IDENT CSCMOD3
*DELETE CSCMOD2.4,CSCMOD2.4
*,IMODE(5),CMASNG,VMASS(5),ZDANT
*INSERT EXE.42
COMMON/TABDIR/TABLE(800)
COMMON/READI/DUM1(64),JBC,INXO
COMMON/UPDCAL/ NUM, P(90), Y(90)
COMMON/LGDE/LA(50), FC2(5), PZ(5), PRES(5), C(5), IPPT, LTPT
COMMON/STG/ICOUNT,KCOUNT,LOCAIN(4),LOCADCA(4)
COMMON/TABCOM/LOCs(115),STABLE(115)
COMMON/CLEAU/P/II,CLEAN,INTEG
COMMON/STORA/ARG(48), ALIST(8), GETARG(8), NENT, LENT, K
CPL(3,3), R1(3,3,5), RAY(5), RAY(5), RAY(5), RAY(5), RAY(5), TEMP(3), ZERO(5),
CXR(5), YR(5), ESPLO(5), PA(5), DDELTA(5),
CFTRZ(5), RDY(5), RDY(5), RDZ(5), RDY(5), RDZ(5),
CVTX(5), VTY(5), VTV(5), GZ(5),
CMR(5), SF(5), CMSK(5), MUXP(5), MTRX(5), MTR(5),
CMTRZ(5), MA(5), RG11, RG11, RG31, RG33, IPRT,
CMTX, MTR, MTR, MTR, MTR, MTR, MTR,
CFTRB, FTRC, SMTRX, SMTR, SMTRZ
COMMON/ACTDIR/NAME(71), LOC(71)

*INSERT  C11778.22

DATA NCASE /10H /

*INSERT SDDMOS.4
DATA PA11T1, PA2T, OSV, OSVC, WFOR /25*0.0/

*INSERT LGEA3C.47
*CALL ACOBLK
*DELETE CSCMOD2.44, CSCMOD2.45
CMASNG = 1+(AMASS*X1*X1)/AIYBS

*INSERT CSCMOD2.62
UNSPR = 0.0

*DELETE CSCMOD2.65, CSCMOD2.65

IF(J.EQ.1 .AND. OMGETD1(1).NE.0.0) UNSPR = MASS(1)

IF(J.GT.1 .AND. OMGETD1(J).NE.0.0) UNSPRNG = UNSPRNG+MASS(J)

*DELETE CSCMOD2.69, CSCMOD2.69
WFOR(1) = SR(1)*(VMASS(1)-UNSPR)

*DELETE CSCMOD2.125, CSCMOD2.125
6 (VMASS(1)*REOSL(1))

*DELETE CSCMODS.44, CSCMODS.44
DATA(AAMASS(1), I=1,8) /6HVELDEC, 6HWLFORR, 5HMAEST, 4HSAFEC, 4HSCOPA,

*DELETE SDFLG.80, SDFLG.80

*INSERT C82477.63

IF(IABS(INDSLG).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(INDSLG).NE.3) 210, 220

210 CONTINUE

IF(ISTP1. 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), OMGETD1(1,1),
*OMGET(1,1)

IF(ISTP2. 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), OMGETD1(1,2),
*OMGET(1,2)

IF(ISTP3. 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,VELOEC,WLFORR,IZANT,ZSSC,COPA,SIPA,DSTOP,
*IDENT CSCMOD4
*DELETE ACOBBLK.12,ACOBBLK.12
9,GMAM(5),GAMAN,GNR,HMM(5),ICOSV(5),INITSW
*DELETE ACOBBLK.13,ACOBBLK.13
*IFR(5),IIXSVM(5),IIXSVL(5),TOPCO(5),IPASS(5),ISET(5)
*DELETE ACOBBL2.22,ACOBBLK.22
COMMON /ACTIVE/ TC3,TC4,VCUM(5),ENC,ENCG,VCUM(5),VOLII(5)
*DELETE CSCMOD3.1,CSCMOD3.1
*IMODB(5),CMASNG,VMASS(5),ZDANT,INDEACT(5),VELDEC
*DELETE EXE.175,EXE.175
*DELETE EXE.286,EXE.286
STOP "EXECUTIVE ROUTINE"
*INSERT JMODOS.1
DIMENSION S(2,5)
EQUIVALENCE (S(1,1),S(1,1))
EQUIVALENCE (DM15(1),ITD)
*DELETE CSCMOD2.21,CSCMOD2.21
DATA ICOSV,IFR,IFSTOP,IPASS /20*0/
*INSERT CSCMOD2.25
INITSW = 1
*INSERT CSCMOD2.25
INDEACT(I) = 0
IF(ITD.EQ.1) INDEACT(I) = 2
*DELETE ACTINIT.143,ACTINIT.143
CALL ALGEAR
*DELETE ALGEAR.70,ALGEAR.70
EQUIVALENCE (DM15(1),ITD)
*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(I,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
*DELETE VMASS(I)*REDSP(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) /6HXSVD3,5HXSVD5,6HXSDOT,2HPR,3HIFR,
*DELETE JMMODS.2,JMMODS.2
DATA N20 /20/
*DELETE JMMODS.3,JMMODS.3
IF(ISUM1.NE.0) WRITE(13) N20,ISUM1,DATA3,OP17,ACOVAR8(I),
*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,XSVD3(I),XSVD5(I),XSVDOT(I),PP(I),FLOAT(IFR(I)));
*IDENT ACTNG
*INSERT ALGEAR.549
*DECK ACTNG

SUBROUTINE ACTNG

*************** FATOLA VARIABLES ***************

COMMON/DIRCOM/DM1(115),ALPHD,DM1A(20),AMASS,DM2(147),DCL1,DCM1,
DCM1,DCL2,DCM2,DCN2,DCL3,DCM3,DCN3,DM3(99),FXB7P,
DUM4(3),FYB7P(4),FXB7P,DM5(17),GX87F,DM6(8),GZB7F,
DM7(218),INDSTE(48),PHIPD,INDSTE1(23),P5IPD,INDSTE2(156),THPDP,
INDSTE3(5),TIME,DM8(287),P177R(2),P177R1(2),OM9(4),
QI77R(2),QI77R1(2),DM10(4),R177R(2),R177R1(2),DM11(48),
XG77F(2),XG77F1(12),YG77F(2),YG77F1(12),
ZG77F(2),ZG77F1(12),DUM13(52),
MNRTRUT,MASS(5),RX(5),RY(5),RZ(5),THETAD(5),ERDEG,RGR,
NTIRES(5),ZER0(5),W(5),DELTAM(5),MOMENT(5),
RF(3),VZ,FDOPZER0(5),VZERO(5),A(5),P20(5),V20(5),
CC(5),CE(5),C2C(5),CZ(5),NVGPT,NMPB(5),RLT,NDELTAL,
ES(5),SB(5),SD12(2),SD22(2),SD23(2),SD24(2),SD25(2),

COMMON/DIRCOM/
SD11(2),SD12(2),SD13(2),SD14(2),SD15(2),
S1(2),SS2(2),S3(2),S4(2),S5(2),
S2D11(2),S2D12(2),S2D13(2),S2D14(2),S2D15(2),
S2D11(2),S2D12(2),S2D13(2),S2D14(2),S2D15(2),
S2L1(2),S2L2(2),S2L3(2),S2L4(2),S2L5(2),
OMT011(2),OMT012(2),OMT013(2),OMT014(2),OMT015(2),
OMT1(2),OMT2(2),OMT3(2),OMT4(2),OMT5(2),
AI(5),BI(5),DELT1,DELT2,DELT3,DELT4,DELT5,ISTAGE,
PRTMIN,IPLT,ISDF,ISTPL1,ISTPL2,ISTPL3,ISTPL4,ISTPL5,
DM14(22),TB(5),DM15(127),INDLG,DM16(107),CASK(44),INFLX,
NMODE,DM18(40),SMOD(100),SYMOD(100),SZMOD(100),DM19(1686),
GQDZ(20),DMM20(20),DMM21(20),SMN1(5),SMN2(5),
GAMA,DUM15(12),INDNWS,DDM22(10),ETADE5,
* DDM23(5),AH(5),PH(5),DDM24(30),

COMMON/LGDE/LA(50),FC2(5),P2(5),PRES(5),C(5),IPPT,LPT,
PL(3),RT(3),R(3),R(5),RAY(5),RAZ(5),TMP(3),ZER0(5),
XR(5),YR(5),EPSLON(5),PA(5),FDELT(5),
FTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
VTRX(5),VTRY(5),VTZ(5),GZ(5),
FTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
VTRX(5),VTRY(5),VTZ(5),GZ(5),
FTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
VTRX(5),VTRY(5),VTZ(5),GZ(5),
FTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
VTRX(5),VTRY(5),VTZ(5),GZ(5),
FTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
VTRX(5),VTRY(5),VTZ(5),GZ(5),
FTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
VTRX(5),VTRY(5),VTZ(5),GZ(5),
FTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
VTRX(5),VTRY(5),VTZ(5),GZ(5),
FTRZ(5),RDX(5),RDY(5),RDZ(5),RDXG(5),RDYG(5),RDZG(5),
VTRX(5),VTRY(5),VTZ(5),GZ(5),
C FTRB, FTRC, SMTRX, SMTRY, SMTRZ
C COMMON /HTCOM, HT, HT1, HT2, INDINT(5)
C *CALL ACOBLK
C
REAL MASS, MOMENT, MASS2, MUS, NTIES, MB
REAL MUVP, MTRY, MTRZ, MA, MUX, MUTY, 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), DDELTA1)
EQUIVALENCE (SD21(1), SD2(1,1)), (SD11(1), SD1(1,1)), (S1(1), S(1,1))
EQUIVALENCE (DM15(1), ITD)
EQUIVALENCE (DM5(16), GREFF),
* (DM1(37), AIYYBS)
C
DATA AIC, IPSTOP /5*0.0, 5*0/
C
Q1(T1, T2)=SIGN(1., (T1-T2))*SORT(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(H&H(I) .EQ. 0.) GO TO 90
IF(OMRUN .GT. 0.0) GO TO 25
IF(IZDANT .GE. VELDEC) GO TO 90
25 IF(IZDANT .GE. VELDEC+X677F1(1)*TAN(OMRUN)) GO TO 90
40 WRITE(6,1014) TIME
1014 FORMAT (1H036H REDUCE CONTROL LIMIT FORCE AT TIME=E16.8)
INDACT(I)=1
56 IF(INDINT(I).EQ.0) GO TO 58
WLFOR(I)=WLFOR(I)-REDSL(I)*HT
EPSILO(I)=EPSILO(I)+EPSLPSHT
58 CONTINUE
INDINT(I)=0
IF(WLFOR(I) .GT. WLFORR) GO TO 90
WRITE(6,1015) TIME
1015 FORMAT (1H027H CONTROL AT WLFOR AT TIME=E16.8)
INDACT(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)=PGA1T(I)
   IF(PGA1T(I).LE.-1600.0) PGA1T(I)=-1600.0
   VOL1T(I)=VOL1T(I)-(AREA1(I)-APINT(I))*S(I,I)
   VOL3T(I)=VOL3T(I)+AREA3(I)*S(I,I)
   VOL2T(I)=VOL2T(I)-(AREA2(I)-AREA1(I)+APINT(I))*S(I,I)+(VOL3T(I)
   X-VOL3T(I)-VCUM(I)
   PGA2T(I)=AP2T(I)*((VOL2T(I)/VOL2T(I))*GAMA)-PATM
   IF(SD1(I,I).GE.0.0)GO TO 104
   PGA3T(I)=((COEF3(I)*AREO3(I))**2*PGA2T(I)-SD1(I,I)/ABS(SD1(I,I))
   X*(SD1(I,I)*AREA3(I))**2)
   GO TO 105
104 PGA3T(I)=PGA2T(I)
105 IF(PGA1T(I).GE. PGA2T(I))GO TO 106
   GO TO 107
106 GAMAHI=PHOH*GREFF*(1.0+(PGA1T(I)*3.04E-08)-
   * (PGA1T(I)**2*2.72E-15))
   GO TO 108
107 GAMAHI=PHOH*GREFF*(1.0+(PGA2T(I)*3.04E-08)-
   * (PGA2T(I)**2*2.72E-15))
108 IF(PGA1T(I).GE. PGA2T(I))COEFO(I)=
   * CDMAO(I)*SORT(ABS(2.0*GREFF/GAMAHI(I))
   IF(PGA2T(I).GE. PGA1T(I))COEFO(I)=
   * CDMAO(I)*SORT(ABS(2.0*GREFF/GAMAHI(I))
   IF(IMODE(I).NE.0 .OR. DDELTA(I).GT.0.0) GO TO 113
   QQI(I)=0.0
   GO TO 109
C
113 QQI(I)=COEFO(I)*(AREO2(I)-APINT(I))*Q1(PGA1T(I),PGA2T(I))
   IF(QQI(I).LT.0.0 .AND. VCUM(I).LE.0.0) GO TO 102
   AIC(I)=0.0
   GO TO 103
102 IF(PGA1T(I).LT.PGA2T(I)) GO TO 103
   QQI(I)=0.0
   VCUM(I)=0.0
   AIC(I)=1.0
103 IF(QQI(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(DDELTA(I).LE.0.0 .AND. TIME.GT.DELT)GO TO 101
   GO TO 110
101 FFORTU=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) = BUFIN(U(I) * ABS(BUFIN(I)) + BLFIN(U(I) * ABS(BLFIN(I))

CONTINUE

C COMPUTE SHOCK STRUT CHARGING FORCE
IF(S(1,I) .GT. 0.0) GO TO 142
FORDER(I) = PGA1T(I) * AREA1(I) + PGA2T(I) * (AREA2(I) - AREA1(I)) - PGA3T(I) * AREA3(I) + FFORT(I) + CFFOPT(I)

CONTINUE

C COMPUTE NORMAL AND AXIAL HUB TO SHOCK STRUT FORCES AT HUB
FONHST(I) = SORT(FDX(I)**2 + FLY(I)**2) - MASS(I) * GREEF * SIPA + SBFOT
IF(ABS(FORM(I)) .LE. FORCHT(I) .AND. S(I,I) .EQ. 0.0) GO TO 150
GO TO 801

CONTINUE

FORDERST(I) = FT(I)
SD1(I,I) = 0.0
IF(JMODE(I) .EQ. 0) GO TO 297
ISTROK(I) = 1
GO TO 289

C COMPRESSION VELOCITY OF SHOCK STRUT IS POSITIVE
801 IF(SD1(I,I) .LE. 0.08 .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(I,I)))
IFR(I) = 1
284 IF(S(I,I) .LT. 0.0) GO TO 160
GO TO 161
160 IPSTOP(I) = 1
161 IF(S(I,I) .LE. DSTOP .AND. IPSTOP(I) .EQ. 1) GO TO 900
GO TO 902
900 IF(S(I,I) .LE. 0.005) GO TO 903
GO TO 904
903 SD2(I,I) = 0.0
SD1(I,I) = 0.0
S(I,I) = 0.0
IPSTOP(I) = 0
GO TO 902
904 CONTINUE
IF(IPSTOP(I) .NE. 0) GO TO 906
DSTOP = S(I,I)
FSTOPK = 2.0 * MASS(I) * SD1(I,I)**2 / DSTOP**2
906 IF(S(I,I) .LE. DSTOP/2.0) GO TO 908
FSTOP(I) = -FSTOPK * (DSTOP - S(I,I))
GO TO 909
908 FSTOP(I) = -FSTOPK * S(I,I)
909 IPSTOP(I) = 1
IFR(I) = 0
GO TO 901
902 FSTOP(I) = 0.0
901 IF(PGA1T(I)<=PGA1T(I)+500.0) .AND.
               PGA1T(I)GT(PGA1T(I)-500.0)) GO TO 158
               GO TO 159
158 IF(ABS(FRT(I))LE.FORCHT(I) .AND. S(I,I).EQ.0.0) GO TO 500
159 IF(S(I,I).GE.0.0) GO TO 470
           IF(SD1(I,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)
               +CFFOR(I))*DHTANH(I)*FSTOP(I)
500 IF(INDFLXGE.1) GO TO 295
           IF(IMODE(I).EQ.0) GO TO 297
           ISTROK(I)=1
289 IF(S(I,I).GT.0.0) GO TO 295
           IF(IOPC0(I).EQ.1) GO TO 295
           IF(PGA1T(I)-1000.0).LT.PGA1T(I) .AND.
               PGA1T(I).LT.(PGA1T(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(I,I).LE.0.0 .AND. ICOSV(I).EQ.1) GO TO 311
           IF(IOPCO(I).EQ.1) GO TO 295
           IF(PGA1T(I).LE.PGA1T(I)) GO TO 293
           IF(IIXSVL(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
           IXXSVL(I)=1
           GO TO 294
293 IF(IIXSVH(I).EQ.1) GO TO 294
           XVALVE(I)=XVALVE(I)+XSVDX(I)*DELT*PERCNT(I)
           IF(XVALVE(I).LT.0.1) GO TO 294
           XVALVE(I)=0.1
           IXXSVH(I)=1
294 CONTINUE
           DLTX1O(I)=0.0
           ICOSV(I)=1
296 IF(NFORT(I).GT.0.0 .AND. S(I,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) GO TO 305, 400
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) GO TO 308, 400
XVALVE(I) = 0.0
IIXSVL(I) = 1
CONTINUE

400 IF(XVALVE(I) .NE. 0.0) GO TO 295
ICOSV(I) = 0
DELTX1(I) = DELTX(I)*XKF(I)
XMA(I) = (DF(I) + DELTX1(I))*XMA(I)
XMA11(I) = XMA(I)
XSV(I) = XSV(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

295 IF(ISTROK(I) .EQ. 1 .AND. S(I,I) .GT. 0.0) IOPCO(I) = 0
I2 = 2*I+NSTRUT-1
I1 = I2+1
CALL INTEG(LA(I2), SD2(1, I))
CALL INTEG(LA(I1), SD1(1, I))

IF(IMODE(I) .EQ. 0) GO TO 450

C

ENUP(I) = 0.5*AIYYBS*Q177R(I)*QI77R(I) + (ENC/CMASNG)*
(ZG77F1(I)/ABS(ZG77F1(I)))
IF(HMM(I) .EQ. 1.0) GO TO 130
SA(I) = 0.
IF(WFORT(I) .LT. 0.0) XSTOT(I) = ENUP(I) / ((-WFORT(I))*COPA)
IF(WFORT(I) .GE. 0.0 OR DDELTA(I) .LT. 0.0) XSTOT(I) = 1.0

C

ZSSC IS A PERCENTAGE OF SB(I) FOR ACTIVATING CONTROL-

C

ZSSC(I) = FWORK(I)*SB(I)
IF(XSTOT(I) .LE. (ZSSC(I) - S(I,I)) .OR. RESA(I) .EQ. 1.0) SA(I) = 1.0
RESA(I) = SA(I)
IF(SA(I) .EQ. 0.0 OR HMM(I) .EQ. 1.0) GO TO 130
WLFOR(I) = -WFORT(I)

C

VELDEC = ((WLFOR(I) + WLFORR) / 2.0) * (WLFOR(I) - WLFORR) / 
6.0
(VMASS(I)*REDSLP(I))
WRITE(6,121)TIME,WFORT(I),VELDEC
121 FORMAT(5OH ACTIVE CONTROL INITIATED...TIME, WFORT, VELDEC = ,
130 IF(S(I,I) .GT. 0.0)ISET(I)=0
131 IF(HMM(I) .EQ. 0.0) GO TO 451
132 IF(-WFORT(I) .GT. (WFORT(I)+EPSILO(I))) DF(I)=(WFORT(I)+
133 EPSILO(I))-(WFORT(I))
134 IF(-WFORT(I) .LT. (WFORT(I)-EPSILO(I))) DF(I)=(WFORT(I)-
135 EPSILO(I))-(WFORT(I))
136 IF(-WFORT(I) .LE. (WFORT(I)+EPSILO(I)) .AND. 
137 EPSILO(I))=-WFORT(I)
138 IF(S(I,I) .LE. 0.0)GO TO 456
139 IF(WFORT(I) .GT. 0.0 .AND. QSVCU(I) .LT. 0.0)454,455
140 DF(I)=WFORT(I)-(-WFORT(I))
141 GO TO 456
142 DF(I)=0.0
143 DELTX(I)=S(I,I)-XSOCM(I)
144 IF(S(I,I) .LE. 0.0 .AND. ISET(I) .EQ. 1)GO TO 451
145 GO TO 452
146 DF(I)=0.0
147 DELTX(I)=0.0
148 XMA(I)=(DF(I)+DELTX(I))*XKA(I)
149 IF(GNR.EQ.1 .AND. XMA(I) .GT. 0.0) XMA(I)=
150 XMA(I)*SQR((PGA1T(I)-PGALC(I))
151 X/(PGA1T(I)-PGA1T(I)))
C
NOTE: SUBROUTINE 'FLOZE2' COMPUTES THE FLOWS FROM THE PRESSURES
C IN UNITS OF INCHES.
C
PL(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)=(((PGA1C(I)+PATM)*VOLANI(I)/VOLANT(I))**GAMA)-PATM)/144.0
C
IF(PS(I) .GE. 3000.0)464,465
C
464 PS(I)=3000.0
465 VOLANT(I)=VOLANI(I)
466 VOLAHT(I)=VOLACI(I)-VOLANT(I)
C
IF(VOLAHT(I) .LE. 0.0)466,467
467 WRITE(6,1050)TIME
C
1050 FORMAT(1HO//45H ACCUMULATOR OIL VOLUME INSUFFICIENT AT TIME=*E16.8
C
1 //
C
CALL LGEAR6
C
STOP 500
C
CONTINUE
C
CALL FLOZE2(PS(I),PR(I),PL(I),XLPSV1(I),XLPSV3(I),RCLSV(I),DSV(I),
C XSV(I),Q51(I),Q53(I),CSV1(I),CSV3(I),XMU)
C
Q51(I)=Q51(I)/1728.
CONTINUE

QSV(I) = QSV(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(I) .LT. 0.0 .AND. PGA1T(I) .LE. -1600.0) PGA1T(I) = -1600.0
RETURN
END
CSV1(I) = COEF1(I)*WSV1
CSV3(I) = COEF1(I)*WSV3
*DELETE ACTINIT.118,ACTINIT.118
CSV1(I) = 0.0
CSV3(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) = VOL1T(I)-AREAI(I)-APINT(I)*S(1,I)
VOL3T(I) = VOL3T(I)+AREAI(I)*S(1,I)
VOL2T(I) = VOL2T(I)-(AREA2(I)-AREA1(I)+APINT(I))*S(1,I)
   +(VOL3T(I)-VOL3T(I))**GAMA-PATM
PGA2I(I) = AP2TO(I)*((VOL2T(I)/VOL2T(I))*PATM
PGA2I(I) = PGA2I(I)
PGA3I(I) = PGA2I(I)
FORSST(I) = -(PGA2I(I)*AREA2(I)-PGA3I(I)*AREA3(I)
   +DMTANH(I)*(FFORT(I)+CFOR(I))+FSTOP(I))
WFOR(I) = FORSST(I)
C
80 CONTINUE
*DELETE ACTINIT.130,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 = ZGCF1(1)-QGTR(1)*RX(1)
*DELETE CSCMOD4.21,CSCMOD4.21
18 CONTINUE
*INSERT CSCMOD4.22
UNSPRNG = 0.0
DO 22 I=1,NSTRUT
   IF(TEM.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 = (-SORT(AXP7F*AXP7F+AYP7F*AYP7F+AZP7F+AZP7F)*Greff)*
                  (AMASS-UNSPRNG)
C
   CALL ACTNG
C
*INSERT    CSCMOD4.23
C
   WFOR(I) = DWFORT/(NSTRUT-1)
C
*DELETE    ALGEAR.106,ALGEAR.106
   IF(ZG77F1(I).GE.VELDEC(I)) GO TO 90
*DELETE    ALGEAR.108,ALGEAR.108
   25 IF(ZG77F1(I).GE.VELDEC(I)*XG77F1(I)*TAN(OMRUN)) GO TO 90
*DELETE    ALGEAR.256,ALGEAR.256
*DELETE    ALGEAR.261,ALGEAR.261
   296 IF(WFOR(I).GT.0.0 .AND. S(I,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
   QSV1(I) = 0.0
   QSV3(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)*RED5LP(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 (DLGEDE(1),LA(1)), (OMTD11(1),OMTD1(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(I)*TAN(OMRUN)) GO TO 90
*INSERT    ACTNG.97
   UNSPRNG = 0.0
   IF(OMETD1(I).NE.0.0) UNSPRNG = MASS(I)
   WFOR(I) = SR(I)*(VMASS(I)-UNSPRNG)
*DELETE    ACTNG.241,ACTNG.241
296 IF(WFOR(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
QSV1(I) = 0.0
QSV3(I) = 0.0
GO TO 450
119 CONTINUE
*DELETE ACTNG.292,ACTNG.292
VELDEC(I) = ((WFOR(I)+WFORR)/2.*(WFOR(I)-WFORR))
*DELETE ACTNG.294,ACTNG.294
WRITE(6,121) TIME,WFOR(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 END ACT IS ENCOUNTERED
C-----------------------------------------------------------------
C
*INSERT ACTIN.17
C
*INSERT READ A DATA CARD ----------------------------------------
C
*INSERT ACTIN.22
C
*INSERT IGNORE REMARKS -------------------------------------------
C
*INSERT ACTIN.23
C
*INSERT TEST FOR END OF ACTIVE DATA -------------------------------
C
*INSERT ACTIN.24
C
*INSERT LOOK FOR A MATCHING MNEUMONIC -----------------------------
C
*INSERT ACTIN.28
C
*INSERT NO MATCH FOUND --------------------------------------------
C
*INSERT ACTIN.30
C
*INSERT FOUND A MATCH ---------------------------------------------
C
*INSERT ACTIN.31
C
*INSERT SET INDEX TO ADDRESS IN ACTIVE AND CORRECT FOR OFFSET
C (INC ON DATA CARD)
C
*INSERT ACTIN.34

C - 3
C---------- INITIALIZE ALL FLAGS AND COUNTERS

*INSERT ACTIN.44

C

C ----- THE 210 LOOP SCANS DATA COLUMNS ON A SINGLE CARD

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 A COMMA OR END OF DATA CARD SIGNALS END OF NUMBER

C

*INSERT ACTIN.66

C

C ----- COMBINE LEFT AND RIGHT PORTIONS OF REAL NUMBER

*INSERT ACTIN.70
*INSERT ACTIN.71
C
*INSERT ACTIN.83
C
*INSERT ACTIN.86
C
*INSERT ACTIN.89
C
*INSERT ACTIN.90
C
*INSERT ACTIN.93
C
*INSERT ACTIN.96
C
*INSERT ACTIN.99
C
*INSERT ACTIN.103
C
*INSERT ACTIN.104
C
*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),LMODF(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

A-55
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(I,I).GE.STRON(J).AND.S(I,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(I,I).GE.STROM(J).AND.S(I,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)
*INSEPT 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,IX,A3,I9.5R1,I2)
*DELETE ACTIN.25,ACTIN.25
DO 110 I=1,75
*DELETE DIRACT.3,DIRACT.3
COMMON/ACTDIR/XNAME(75),LOC(75)
*DELETE CSCMOD2.31,CSCMOD2.31
5 6HZETAC1, 6HZETAC2, 6HIMODE, 6HPINN, 6HPINM,
6 6HSTROM, 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(IX,7H--.--.--.--,I5,3F16.8)
   IF(TMP(1).LT.0.) GO TO 91
*IDENT TABFIX
*DELETE LGEA3C.205,LGEA3C.205
   CALL HIHO(3,LOC(7),NS2NDY,NSTRUT,DU,DU,S2(1,I),FI,DU,DU,C2(I))
   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 GENFX
*DELETE REST.32,REST.32
   COMMON/XACTNG/AICX(5),IPSTOX(5),DN35(3)
*DELETE EXE.24,EXE.24
* DM43  , INDVPC  , DM44  ( 7) , NCASE  , DM45  

*DELETE SDDMODS.1 , SDDMODS.1  
34 DO 35 I=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 DELTFIX  
*INSERT CSCMOD5.48  
DATA TIMEL,NIN/O.,0/  
*INSERT ALGEO.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/O.,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),DM34x(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, RXC1, TTIM, UNSPRNG,NIN,TIMEL  
*DELETE CSCMOD2.106 , ALGEO.217  
159 IF(SD1(I,I).LT.0.) GO TO 471  
*DELETE ACTNG.200 , ACTNG.202  
159 IF(SD1(I,I).LT.0.) GO TO 471  
*IDENT MOD296  
*DELETE CSCMOD2.80 , CSCMOD2.80  
IF(IMODE(I).EQ.0.AND.S(I,I).EQ.0.) 112,113  
*DELETE CSCMOD2.133 , CSCMOD2.133  
*DELETE ACTNG.126 , ACTNG.126  
IF(S(I,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  MINN.17
H = AMIN1(H1,H2,H3)
*DELETE  HTTRY.2,HTTRY.2
*DELETE  CSCMOD2.131,CSCMOD2.132
*INSERT  ALGEO.439
DP1(I) = (-QQ1(I))\*OSV1(I)-OSV3(I)+(AREAI(I) - APINT(I))
1 *SD1(I)\*BETA/VOLI(I)
IF(S1(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  ALGEC.515
IF(IMODE(I).EQ.0) GO TO 28
*DELETE  ALGEC.517,ALGEC.518
*DELETE  CSCMOD2.138,CSCMOD2.139
I9 = I +9\*NSTRUT
CALL INTEG(LA(I9),DLTX(I))
*INSERT  CSCMODS.102
IF(IMODE(I).EQ.0) GO TO 6
*IDENT  MOD351
*INSERT  MOD329.5
PGA1T1(I) = PGA11(I)
PGA1T1(I) = PGA11(I)
*IDENT  MOD1029
*DELETE  REST.14,REST.14
COMMON/UPDCAL/DM14(201)
*DELETE  REST.24,REST.24
COMMON/XMINIM/DM26(1008)
*DELETE  CSCMOD3.4,CSCMOD3.4
COMMON/UPDCAL/NUP,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/NUP,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), Y0(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 MOD1M
*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(I,I).GE.0.0) GO TO 59
   IF(S(I,I).GE.SB(I)-(S2T(I)-S2(I,I))) 83, 85
C SECONDARY PISTON IN CONTACT WITH ORIFICE TUBE
  83 IF(SD1(I,I)+1.0E-4.GE.S2D1(I,I)) 84, 85
  84 S2D1(I,I) = SD1(I,I)
  85 CONTINUE
*IDENT AERAT
*INSERT REST.37
  COMMON/AEROC/DI41(8)
  COMMON/AD4E/DM42(2)
*DELETE MOD1029.4, MOD1029.4
  READ(7)(DM1(I,J),I,J=1,11214)
*DELETE MOD1029.5, MOD1029.5
  WRITE(7)(DM1(I,J),I,J=1,11214)
SUBROUTINE AER04
COMMON/DIRCOM/DM1(2),X
COMMON/TABCOM/LOC5(115),ST(115)
COMMON/TABDIR/TABLE(800)
COMMON/AEROCM/RTAB10(2),RTAB80(2),LTAB10(2),LTAB80(2)
COMMON/XAERO/NIN,TIME
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
TINE = 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(INO2+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/DATA(8)
DATA DATA/X.0.1
CALL LINES(1)
WRITE(6,1) SYM,RA
1 FORMAT(18X,A6,5X,55A1)
2 FORMAT(*ERROR.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.000000000000000000778
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 DATA(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*0.1
*DELETE  CSCMOD2.17,CSCMOD2.17
EQUVALENCE (DM5(16),GREFF),(DM1(37),AIYYBS)
*DELETE  AERAT.64,AERAT.64
*DELETE  PINARY.26,PINAPY.26
ORIGINAL PAGE IS OF POOR QUALITY

```
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, 6HPEDSLP, 6HDSTOP /
*DELETE PINARY.40,PINARY.40
7 644, 645, 651, 683, 713, 743, 773, 392, 140 /
*IDENT MOD2075
*DELETE MOD282.5,ALGGEAR.219
  159 SIGX = 0.
  IF(SD1(I,I).EQ.0.) GO TO 471
  SIGX = SD1(I,I)/ABS(SD1(I,I))
*DELETE ALGGEAR.223,ALGGEAR.223
  1 + CFFOR(I)*SIGX*DMTANH(I) + FSTOP(I)
*DELETE MOD282.6,ACTNG.204
  159 SIGX = 0.
  IF(SD1(I,I).EQ.0.) GO TO 471
  SIGX = SD1(I,I)/ABS(SD1(I,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/XALGGEAR/AIC(5),DM34(14),IPSTOP(5),DM34X(12),DM34Y(5),
  1 DM34Z(5),DM34A(65)
  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,IOCU,IXS,IA1,IA2,IA3,IA4,
  4 IA6,IA7,IA9,IA10,IGO
*DELETE CSMOD5.47,CSMOD5.48
  DIMENSION IPSTOP(5),AIC(5),PGA1T(5),IGE(5),RDSLP(5),ITRIP(5),
  1 ICU(5),IOCU(5),IXS(5),IA1(5),IA2(5),IA3(5),IA4(5),
  2 IA6(5),IA7(5),IA9(5),IA10(5),IGO(5)
*DELETE CSMOD2.78,CSMOD2.78
  IF(SD1(I,I).LT.0.0.AND.PGA1T(I).LE.-1600.0)PGA1T(I)=-1600.0
  IF(ITRIP(I).EQ.1) GO TO 101
*DELETE ALGGEAR.172,ALGGEAR.172
  IF(ABS(FT(I)).LE.FORCHT(I).AND.S(I,I).LE.ES(I)) GO TO 150
```

THE FOLLOWING LOGIC RETURNS THE GEARS DURING REBOUND TO INITIAL CONDITIONS IN THE EVENT THE GEAR CONTACTS THE SURFACE BEFORE THE LOGIC BETWEEN STATEMENTS 226 AND 421 IS FULLY EFFECTIVE.

289 IF (ISTROK(I) EQ 1 AND DDELTA(I) LT 0) IGO(I) = 1
227 IF (IGO(I) EQ 0) GO TO 226
220 IF (VCUM(I) GT 0 AND WFOR(I) EQ 0) 220, 226
221 QDI(I) = VCUM(I) / DSTOP
GO TO 226
222 PGA2T(I) = PGA1I(I)
QDI(I) = 0.0
IGO(I) = 0
226 IF (IGE(I) EQ 1) 290, 297
290 IF (ITRIPI(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)
XSVD3D(I) = 0.0
XSVD5D(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) / TAU
202 XMA1(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((PGA1T(I)-2000.0).LT.PGA1T(I)).AND.
    1 PGA1T(I).LT.(PGA1T(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.PGA1T(I)+2000.)292,293
292 IF(ISSV(I).LT.1)GO TO 311
XSVDDT(I)=XSVDMN(I).PERCNT(I)
PAS(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
XSVDDT(I)=0.0
IXSVL(I)=1
GO TO 294
293 IF((PGA1T(I).LT.PGA1T(I)-2000.))295,294
295 IF(ISSV(I).LT.1)GO TO 311
XSVDDT(I)=XSVDMX(I).PERCNT(I)
PAS(I)=2
XVALVE(I)=XVALVE(I)+XSVDMX(I).DELT*PERCNT(I)
IF(XVALVE(I).LE.0.13)302,304
304 XSV(I)=XVALVE(I)
GO TO 297
302 XVALVE(I)=0.13
IXSVH(I)=1
294 CONTINUE
XSVDDT(I)=0.0
ICOSV(I)=1
311 IF(PGA2T(I).LT.PGA1T(I)+4000.0)316,315
316 IF(QSV(I).LT.0D(I)).AND.ICOSV(I),EQ.1)317,318
317 XSVDDT(I)=0.0
XSV(I) = XVALVE(I)
GO TO 297

XSVDOT(I) = XSVDMN(I) * PERCNT(I)
XSV(I) = XVALVE(I)
GO TO 297

CONTINUE
IF(NAC(I) .EQ. 1) GO TO 307

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

XSV(I) = XVALVE(I)
GO TO 297

XVALVE(I) = XBIAS(I)
XSVDOT(I) = 0.0
IIXSVH(I) = 1
GO TO 400

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

XSV(I) = XVALVE(I)
GO TO 297

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
Q0(I) = 0.0
QSCVU(I) = 0.0
QSV(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
PGA1T(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
00(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(OSVCU(I) .GT. 0.00001 OR OSVCU(I) .LT. -0.00001) GO TO 433
QSV(I) = 0.0
IQCUI(I) = 1
GO TO 420
433 IF(OSVCU(I) .LT. -0.00001) GO TO 434
QSV(I) = -OSVCU(I) / DSTOP
GO TO 420
434 QSV(I) = -OSVCU(I) / DSTOP
420 IF(XSV(I) .GT. XBIAS(I) + 0.000001) 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) - 0.000001) 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) + IQCUC(I) + IXS(I) + IA1(I) + IA2(I) + IA3(I) + IA4(I) + IA6(I)
       + IA7(I) + IA9(I) + IA10(I), .EQ. 11) ISTROK(I) = 0
       I2 = 2*I + INSTRUT-1
*DELETE ALGEAR.301, ALGEAR.301
C ZSSC IS A PERCENTAGE OF SB(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) = WLFORP
   INDEACT(I) = 2
   EPSILO(I) = EPSRSL(I)
   GO TO 451
131 IF(S(I, I), LE.ES(I)) GO TO 451
   IF(HMM(I), .EQ. 0) GO TO 451
*INSERT ALGEAR.330
   IF(S(I, 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) * OSVN(I) * DELT - QPUMPS(I) * DELT
*DELETE ALGEAR.354, ALGEAR.354
460 CALL LIMITS(XSV(I), XSVDOT(I), XSMAX(I), XSVMIN(I))
   CALL FLO2E2(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
*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 MOD329.10,MOD329.10
29 I9=I+9*NSTRUT
*DELETE CSMOD2.140,CSM0D2.140
27 I6=3*I+6*NSTRUT-2
*INSERT CSMOD2.145
   CALL LIMITS(XSV(I),XSVDDT(I),XSVMAX(I),XSVMIN(I))
   CALL LIMITS(XSVDDT(I),XSVDD(I),XSVDMX(I),XSVDMIN(I))
   CALL LIMITS(XSVDD(I),XSVDDD(I),XSDMAX(I),XSDMIN(I))
*INSERT GENFX.I8
   2 ,IGE, RDSLPS,ITRIP,ICU,IOCU,IXS,IA1,IA2,IA3,IA4,
   3 IA6,IA7,IA9,IA10,IGO
*DELETE ACTNG.103,ACTNG.103
   DIMENSION IPSTOP(5),AIC(5),PGLT(I),IGE(5),RDSLSP(5),ITRIP(5),
   1 ICU(5),IOCU(5),IXS(5),IA1(5),IA2(5),IA3(5),IA4(5),
   2 IA6(5),IA7(5),IA9(5),IA10(5),IGO(5)
*DELETE ACTNG.157,ACTNG.157
   IF(SD1(1,I).LT.0.0.AND.PGLT(I).LE.-1600.0)PGLT(I)=-1600.0
   IF(ITRIP(I).EQ.1)GO TO 101
*DELETE ACTNG.160,ACTNG.160
   IF(S(1,I).GT.ES(I))GO TO 421
*DELETE ACTNG.160,ACTNG.160
   IF(IMOTE(I).EQ.0.AND.S(1,I).LT.ES(I))GO TO 421
*DELETE ACTNG.160,ACTNG.160
   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(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(S(1,I).GT.ES(I))ISTROK(I)=1
*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))GO TO 287,289
287 ISTROK(I)=1
   ICU(I)=0
   IOCU(I)=0
   IXS(I)=0
C THE FOLLOWING LOGIC RETURNS THE GEARS, DURING REBOUND, TO INITIAL
CONDITIONS IN THE EVENT THE GEAR CONTACTS THE SURFACE BEFORE
THE LOGIC BETWEEN STATEMENTS 226 AND 421 IS FULLY EFFECTIVE

289 IF(ISTROK(I).EQ.1 .AND. DDELT(A(I)).LT.0.0) IGO(I)=1
   IF(IGE(I).EQ.1 .OR. ITRIP(I).EQ.1)227,297
   227 IGO(I).EQ.0) GO TO 226
   IF(DELT(A(I)).GT.0.0 .AND. WLFOR(I).EQ.0.0)220,226
   220 IF(VCM(U(I)).GT.0.00001 .OR. VCM(U(I)).LT.-0.00001)221,222
   221 QO(I)=VCUM(I)/DSTOP
      GO TO 226
   222 PGA2(I)=PGA1(I)
   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)
       XSVD(I)=0.0
       XSVDD(I)=0.0
       IF(DELT(X1(I)).GT.0.00001 .OR. DELT(X1(I)).LT.-0.00001) GO TO 210
       DLTX1(I)=0.0
       GO TO 202
   210 DLTX=--DELTX1(I)/TAUF
   202 XMA(I)=0.0
       XMA(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
       XMA10D(I)=0.0
       XMA20D(I)=0.0
       XMA30D(I)=0.0
       XMA40D(I)=0.0
       XMA60D(I)=0.0
       XMA70D(I)=0.0
       XMA90D(I)=0.0
       XMA10D(I)=0.0
   *DELETE ACTNG.214, ACTNG.215
       IF((PGA1(I)-2000.0).LT.PGA1T(I) .AND. 
           1 PGA1T(I).LT.(PGA1(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.PGA1(I)+2000.0)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) GO TO 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(PGAI1(I).LT.PGAI1(I)-2000.) GO TO 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) GO TO 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).LT.PGA11(I)+4000.0) GO TO 316, 315
316 IF(IXSVH(I).LT.QD(I).AND.ICOSV(I).EQ.1) GO TO 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(IXSVH(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)) GO TO 305, 306
306 XSV(I) = XVALVE(I)
    GO TO 297
305 XVALVE(I) = XBIAS(I)
    XSVDOT(I) = 0.0
    IXSVH(I) = 1
    GO TO 400
307 IF(IXSVL(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
IXSV(I)=1
*DELETE CSCMOD5.87,ACTNG.256
*INSERT ACTNG.257
DELTX(I)=0.0
*INSERT ACTNG.261
ICOSV(I)=0
QDI(I)=0.0
OSVCU(I)=0.0
QSV1(I)=0.0
QSV3(I)=0.0
VCUM(I)=0.0
VOL1(I)=VOL1(I)
VOL2(I)=VOL2(I)
VOL3(I)=VOL3(I)
ITRI(I)=1
PGA1T(I)=PGA1T(I)
PGA1T(I)=PGA1T(I)
PAG2T(I)=PGA2T(I)
PAG3T(I)=PGA3T(I)

*DELETE ACTNG.268,ACTNG.272
297 IF(ITRI(I).EQ.0.1)430,421

430 IF(VCUM(I).GT.0.00001.OR.VCUM(I).LT.-0.000001)GO TO 431
QDI(I)=0.0
ICU(I)=1
GO TO 440

431 IF(VCUM(I).LT.-0.000001)GO TO 432
QDI(I)=VCUM(I)/DSTOP
GO TO 440

432 QDI(I)=VCUM(I)/DSTOP
440 IF(OSVCU(I).GT.0.00001.OR.OSVCU(I).LT.-0.000001)GO TO 433
QSV(I)=0.0
IQCU(I)=1
GO TO 420

433 IF(OSVCU(I).LT.-0.000001)GO TO 434
QSV(I)=OSVCU(I)/DSTOP
GO TO 420

434 QSV(I)=OSVCU(I)/DSTOP
420 IF(XSV(I).GT.XBIAS(I)+0.00001)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)-0.000001)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(XMA5(I).GT.0.00001 .OR.XMA5(I).LT.-0.00001)GO TO 609
   XMA5DT(I)=0.0
   IA5(I)=1
   GO TO 610
609  XMA5DT(I)=-XMA5(I)/DSTOP
610  IF(XMA6(I).GT.0.00001 .OR.XMA6(I).LT.-0.00001)GO TO 611
   XMA6DT(I)=0.0
   IA6(I)=1
   GO TO 612
611  XMA6DT(I)=-XMA6(I)/DSTOP
612  IF(XMA7(I).GT.0.00001 .OR.XMA7(I).LT.-0.00001)GO TO 613
   XMA7DT(I)=0.0
   IA7(I)=1
   GO TO 614
613  XMA7DT(I)=-XMA7(I)/DSTOP
614  IF(XMA8(I).GT.0.00001 .OR.XMA8(I).LT.-0.00001)GO TO 615
   XMA8DT(I)=0.0
   IA8(I)=1
   GO TO 616
615  XMA8DT(I)=-XMA8(I)/DSTOP
   C  THESE SWITCHES ARE EITHER ZERO OR ONE
421  IF(IQCU(I)+IQCUC(I)+IXS(I)+IA1(I)+IA2(I)+IA3(I)+IA4(I)+IA6(I) +IA7(I)+IA9(I)+IA10(I) .EQ. 11) ISTROK(I)=0
   I2=2*I+ISTRUT-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)*RDSLPI(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(I,I).LE.ES(I))GO TO 451
  IF(HMM(I),EQ.0)GO TO 451
*INSERT  ACTNG.316
  IF(S(I,I).LE.ES(I).AND.ITRIP(I).EQ.1)GO TO 458
*DELETE  ACTNG.324,ACTNG.324
  458  P1(I)=PAG1T(I)/144.
  459  VOLANT(I)=VOLANT(I)+OSVN(I)*DELT-OPUMPS(I)*DELT
*DELETE  ACTNG.340,ACTNG.340
  410  CALL LIMITS(XSV(I),XSVDOT(I),XSVMA(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)*RDSLPI(I))
*IDENT  MOD3028
*DELETE  MOD2203.452,MOD2203.452
  1  (VMASS(I)*RDSLPI(I))
**Title and Subtitle**

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

**Author(s)**

G. H. Mall

**Performing Organization Name and Address**

Computer Sciences Corporation
3217 N. Armistead Avenue
Hampton, VA 23666

**Sponsoring Agency Name and Address**

National Aeronautics and Space Administration
Washington, DC 20546

**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.
End of Document