SPACE SHUTTLE DIGITAL COMPUTER SIMULATION BENCHMARK

Man T. Ung, PhD
428 W. Acacia
El Segundo CA 90245

December 1978

Final Report

This document has been approved for public release and resale; its distribution is unlimited.

Prepared for:
6510 TESTW/TEEST
Edwards AFB CA 93523

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

79 03 05 105
This report was submitted by Man T. Ung, PhD, 428 West Acacia, El Segundo, California 90245, under contract Number FO4700-78-M-2539, Job Order Number 995030, with the Air Force Flight Test Center, Edwards AFB, California 93523. Mr Richard R. Hansen, Jr., was the AFPTC Project Engineer in charge of the procurement.

This report has been reviewed and cleared for open publication and/or public release by the AFPTC Office of Information in accordance with AFR 190-17 and DODD 5230.9. There is no objection to unlimited distribution of this report to the public at large, or by DDC to the National Technical Information Service where it may be released to foreign nationals.

Publication of this technical report does not constitute Air Force approval of its findings or conclusions. It is published only as a record of the technical effort and for the exchange and stimulation of ideas concerning aircraft flight simulation on digital computers.

Prepared by: This report has been reviewed and is approved for publication: December 1978

FOR THE COMMANDER

_________________________
MAN T. UNG, PhD

_________________________
John W. Keesler
Deputy, Development Division

_________________________
Alfred Phillips
Technical Director
Test Engineering

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

Do not return this copy; retain or destroy.
**Title**: SPACE SHUTTLE DIGITAL COMPUTER SIMULATION BENCHMARK

**Author**: Man T. Ung, PhD

**Performing Organization**: Man T. Ung, PhD
428 W. Acacia
El Segundo CA 90245

**Report Date**: December 1978

**Page Count**: 164

**Abstract**: This report was produced to serve as the benchmark to be used during the acceptance test for the Simulator for Flight Test and Development (SAFTD) at Edwards AFB. Afterwards, the document will be used to train new engineers to use the simulator. The writing is tutorial in nature and it stresses the orderly organization of a computer flight simulation. Model validation is explained in terms of static and dynamic checks. The report is not intended for public release and resale; its distribution is unlimited.
ABSTRACT (cont)

to arrive at any operational conclusion or new discoveries. This Space Shuttle Simulation, which is aimed at training and contractual acceptance of procured hardware, is different from the AFFTC Office of Advanced Manned Vehicles' Space Shuttle Simulation, which is used to support Shuttle development and flight testing.
PREFACE

The following document was intended to serve as a reference manual which will enable the reader to gain a quick understanding into the design and working arrangements of the Space Shuttle Entry and Approach Test simulation as used on the Simulator for Aircraft Flight Test and Development Simulation Benchmark. By reading through this report he is expected to know how to make use of the program and to find his way around in case any modification/addition is contemplated. In brief, the main purpose for this report is to insure a smooth transferability from the originators to the users with a minimum period of learning and associated delay.

The program was written in Fortran for the SEL 32/55 digital computer with 48K 600 nsec memory equipped with floating-point hardware and tied to a functioning mock-up of an aircraft cockpit. Some part of this program was coded in Assembly language to improve the execution speed. Therefore the package cannot be transferred to another type of machine without some re-programming efforts.

Significant contributions to all phases of the simulation and report-writing were made by the following personnel:

Mr. Richard Hansen, Supervisor
Mr. Steve E. Louton, Engineer
Mr. Larry V. LeDuc, Engineer
Capt. Charles L. Bozeman, USAF
Lt. Douglas V. Palmer, USAF
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF TABLES AND FIGURES</th>
<th>Page number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. INTRODUCTION</td>
<td>5</td>
</tr>
<tr>
<td>2. MATHEMATICAL MODEL</td>
<td>6</td>
</tr>
<tr>
<td>2.1 Aerodynamic Equations</td>
<td>7</td>
</tr>
<tr>
<td>2.2 Orbital &amp; Suborbital Equations</td>
<td>8</td>
</tr>
<tr>
<td>2.3 Rotational Equations</td>
<td>10</td>
</tr>
<tr>
<td>2.4 Other Pertinent Equations</td>
<td>13</td>
</tr>
<tr>
<td>3. MAIN PROGRAM</td>
<td>16</td>
</tr>
<tr>
<td>3.1 Initialization</td>
<td>16</td>
</tr>
<tr>
<td>3.2 Hi-speed Loop</td>
<td>16</td>
</tr>
<tr>
<td>4. FLIGHT CONTROL SYSTEM</td>
<td>26</td>
</tr>
<tr>
<td>4.1 Description</td>
<td>26</td>
</tr>
<tr>
<td>4.2 FCS Block Diagrams</td>
<td>26</td>
</tr>
<tr>
<td>4.3 FCS Computer Implementation</td>
<td>27</td>
</tr>
<tr>
<td>5. EASE PACKAGE</td>
<td>46</td>
</tr>
<tr>
<td>6. FUNCTION GENERATION PACKAGE</td>
<td>56</td>
</tr>
<tr>
<td>6.1 DATASTORE Subroutine</td>
<td>57</td>
</tr>
<tr>
<td>6.2 POINT Subroutine</td>
<td>60</td>
</tr>
<tr>
<td>6.3 DERIVE Subroutine</td>
<td>62</td>
</tr>
<tr>
<td>7. REPRESENTATIVE SOLUTIONS</td>
<td>64</td>
</tr>
<tr>
<td>7.1 General Procedures</td>
<td>64</td>
</tr>
<tr>
<td>7.2 Static Checks</td>
<td>66</td>
</tr>
<tr>
<td>7.2.1 Equations of Motion Checks</td>
<td>66</td>
</tr>
<tr>
<td>7.2.2 FCS Checks</td>
<td>70</td>
</tr>
<tr>
<td>7.3 Dynamic Checks</td>
<td>72</td>
</tr>
<tr>
<td>7.3.1 Equations of Motion Tests</td>
<td>73</td>
</tr>
<tr>
<td>7.3.2 FCS Tests</td>
<td>73</td>
</tr>
</tbody>
</table>
8. TERMINOLOGY, SYMBOLS AND DEFINITIONS 79
   8.1 List of Acronyms 79
   8.2 List of Symbols Used in Computer Programs 79

9. REFERENCES 88
   9.1 General Bibliography 88
   9.2 Shuttle-Related References 88

APPENDIX A - Main Program Listing 90
APPENDIX B - EASE Program Listing 121
APPENDIX C - Function Generation Package 128
APPENDIX D - Flight Control System Program Listing 150
# LIST OF TABLES AND FIGURES

<table>
<thead>
<tr>
<th>Figure Number</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1</td>
<td>Main Program Flowchart</td>
<td>17</td>
</tr>
<tr>
<td>3-2</td>
<td>Block Diagram (Equations of Motion)</td>
<td>25</td>
</tr>
<tr>
<td>4-1</td>
<td>FCS Pitch Channel</td>
<td>28</td>
</tr>
<tr>
<td>4-2</td>
<td>FCS Roll Channel</td>
<td>29</td>
</tr>
<tr>
<td>4-3</td>
<td>FCS Yaw Channel</td>
<td>30</td>
</tr>
<tr>
<td>4-4</td>
<td>FCS Actuators</td>
<td>31</td>
</tr>
<tr>
<td>4-5</td>
<td>Hysteresis Function (Flowchart)</td>
<td>35</td>
</tr>
<tr>
<td>4-6</td>
<td>SHTLFCS Flowchart</td>
<td>36</td>
</tr>
<tr>
<td>5-1</td>
<td>EASE Flowchart</td>
<td>48</td>
</tr>
<tr>
<td>6-1</td>
<td>Flowchart for DATASTORE subroutine</td>
<td>58</td>
</tr>
<tr>
<td>6-2</td>
<td>Subroutine POINT</td>
<td>61</td>
</tr>
<tr>
<td>7-1</td>
<td>Equations-of-Motion Check (Case a)</td>
<td>74</td>
</tr>
<tr>
<td>7-2</td>
<td>Equations-of-Motion Check (Case b)</td>
<td>75</td>
</tr>
<tr>
<td>7-3</td>
<td>Equations-of-Motion Check (Case c)</td>
<td>76</td>
</tr>
<tr>
<td>7-4</td>
<td>FCS Tests (Case d)</td>
<td>77</td>
</tr>
<tr>
<td>7-5</td>
<td>FCS Tests (Case e)</td>
<td>78</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table Number</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-1</td>
<td>Transfer Functions</td>
<td>34</td>
</tr>
<tr>
<td>8-1</td>
<td>Main Program Symbols</td>
<td>81</td>
</tr>
<tr>
<td>8-2</td>
<td>FCS Variables</td>
<td>85</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

The Shuttle software package is a general-purpose real-time digital computer program that simulates all aspects of the flight: aerodynamics, flight control system connected to a cockpit for man-in-the-loop operation. The Space Shuttle mission profile is composed of many modes. They are very briefly: ignition & lift-off, pitch initialization after about 6 seconds, staging 122 seconds after lift-off (in the neighborhood of 235-nautical-mile altitude), external tank separation & disposal then orbit insertion. On the way back to earth the mission can be subdivided into: deorbit phase at 160-nautical-mile altitude, entry mode starting at 400,000ft, terminal phase starting at 50,000ft and finally landing and touchdown. The simulation contained in this report is devoted to the last two phases: Entry (beginning at $10^5$ ft) and Terminal Area Energy Management (TALM).

Handling quality constitutes the main objective of this study. It does not address the Shuttle's full mission nor does it simulate the navigation to pinpoint a landing site. Actual touchdown will not be simulated because the model does not include ground effects, the lowering of landing gear assembly and nose wheel steering. The Space Shuttle simulation described in this report is designed for training and contractual acceptance of procured simulator hardware. It is different from the Shuttle simulation used by the AFFTC Office of Advanced Manned Vehicles which is used for development and flight testing of the Shuttle.
2. MATHEMATICAL MODEL

By writing the aerodynamic equations, three explicit coordinate systems are used. They are H-frame (flight-path axes for describing orbital mechanics), body-axes and earth axes. The choice of various coordinate systems is made to enhance the relative computational accuracy of each segment of the model. For example, it makes no sense writing the vehicle rotational equations in any coordinate system except the ones with an origin residing within the vehicle itself such as body-axes, stability axes, or wind axes.

The transformation from body-axes to earth-axes is accomplished with a matrix composed of direction-cosine terms. These direction cosines are listed below but they are not explicitly evaluated in the program.

\[
\begin{align*}
I_1 &= \cos \theta \cos \psi \\
I_2 &= \cos \theta \sin \psi \\
I_3 &= -\sin \theta \\
m_1 &= -\cos \phi \sin \psi + \sin \phi \sin \theta \cos \psi \\
m_2 &= \cos \phi \cos \psi + \sin \phi \sin \theta \cos \psi \\
m_3 &= \sin \phi \cos \theta \\
n_1 &= \sin \phi \sin \psi + \cos \phi \sin \theta \cos \psi \\
n_2 &= -\sin \phi \cos \psi + \cos \phi \sin \theta \sin \psi \\
n_3 &= \cos \theta \cos \phi
\end{align*}
\]

The angles \( \theta, \phi \) and \( \psi \) represent the pitch, roll and yaw angles. Together they are commonly referred to as the Euler angles.
For clarity, the model equations are arranged in groups, each group describing an aspect of the flight. Subscripts \( H \), \( L \), and \( E \) stand for the three coordinate systems mentioned earlier. Whatever symbol not explicitly named can be found in alphabetical order under Chapter 8.

### 2.1 Aerodynamic equations

Acceleration terms in body-axis reference frame are the results of thrust vectors \( T_x \), \( T_y \), and \( T_z \). The subscript "G" stands for landing gear and even though there is no engine, the thrust terms are included in the equations for completeness.

\[
X_B = \frac{-\theta G}{m} \left[ C_{c_{0}} + \frac{C_{c_{R}}(\delta e)}{2} + C_{c_{L}}(\delta e L) + C_{c_{DF}}(\delta DF) + C_{c_{SL}}(\delta SL) \right]
+ \frac{T_x}{m} \tag{2-1}
\]

\[
Y_B = \frac{-\theta G}{m} \left[ C_{c_{0}} + \frac{C_{c_{R}}(\delta e)}{2} + C_{c_{L}}(\delta e L) \right] + \frac{T_y}{m} \tag{2-2}
\]

\[
Z_B = \frac{-\theta G}{m} \left[ C_{c_{0}} + \frac{C_{c_{R}}(\delta e)}{2} + C_{c_{L}}(\delta e L) \right] + \frac{T_z}{m} \tag{2-3}
\]

Note that the subscript "N" refers to the normal force as opposed to subscript "n" which is associated with yawing-moment coefficients.

Resolved into the earth-axes reference frame, these acceleration terms become, after compensation for the earth's oblateness

\[
x_E = x_B^1 + \frac{Y_B}{m} + \frac{Z_B}{n_1} - 3J2f_0 \left( \frac{r_0}{r} \right)^4 \cos \mu \sin \mu
\]

\[
y_E = x_B^2 + \frac{Y_B}{m} + \frac{Z_B}{n_2} \tag{2-5}
\]

\[
z_E = x_B^3 + \frac{Y_B}{m} + \frac{Z_B}{n_3} \quad 7
\]
where $\bar{\mu}$ and $J_2$ are the colatitude ($\frac{\pi}{2} - L$) and the measure of earth oblateness respectively. The dynamic pressure $\bar{q}$ in the above equations and other related terms such as Mach number and the mass $m$ are defined below.

$$\bar{q} = \frac{1}{2} \rho V^2$$

$$\text{Mach} = \frac{V}{a}; \ a = \text{sonic speed} \quad (2-6)$$

$$m = \frac{V_0 + V_f}{G_0}$$

Here, $V_0$ and $V_f$ stand for the vehicle empty weight and fuel weight respectively. Also the airspeed $V$, which is normally obtained by

$$\sqrt{U_E^2 + V_L^2 + V_F^2}$$

can simply be established by

$$V = \frac{U_E}{\cos \alpha \cdot \cos \beta} \quad (2-7)$$

Latitude $L$ and longitude $\lambda$ are derived from the launching-point coordinate $(l_0, \lambda_0)$ and the earth's rotation $\omega_e$

$$L = L_0 + \int_0^t \frac{U_E}{r} \, dt$$

$$\lambda = \lambda_0 + \int_0^t \left[ \frac{V_L}{r \cos I} - \omega_e \right] \, dt \quad (2-8)$$

### 2.2 Orbital & Suborbital Equations

First consider the orbital phase. Let $\psi_H$ be the angle between the direction of travel and true north and $r$ the vehicle altitude measured from the earth center. Let $(x_H, y_H)$ be the local plane perpendicular to the line of $r$ (or $z_H$) such that $x_H$ is pointing in the general direction of travel and $y_H$ perpendicular to $(x_H, z_H)$ to form a right-handed triad (see ref. 9.1.3) Then

$$rU_H = (rU_H)_0 + \int_0^t rX_H \, dt \quad (2-9)$$

$$\psi_H = \psi_H_0 + \int_0^t \left( \frac{V_L}{r} \tan I + \frac{V_H}{U_E} \right) \, dt$$
\[
\begin{align*}
\Phi_{II} &= \Phi_{II}^0 + \int_{0}^{t} \left[g + Z_{II}\right] dt \\
\delta R &= \delta R_0 + \int_{0}^{t} (-\Phi_{II}) dt \\
U_{II} &= \frac{(rU_{II})}{r}
\end{align*}
\]

Where

\[
r = r_0 + \delta R \quad (\delta R \text{ measured from sea level})
\]

\[
X_{II} = X_E \cos \psi_{II} + Y_E \sin \psi_{II} \quad (2-10)
\]

\[
Y_{II} = -X_E \sin \psi_{II} + Y_E \cos \psi_{II}
\]

\[
Z_{II} = Z_{E}
\]

And the velocity and the gravity terms are obtained as follows

\[
U_E = U_{II} \cos \psi_{II}
\]

\[
V_E = U_{II} \sin \psi_{II} \quad (2-11)
\]

\[
W_E = W_{II}
\]

\[
\delta = \delta_0 \left(\frac{r_0}{r}\right)^2 - \left[\frac{3}{2} \delta_0 \left(\frac{r_0}{r}\right)^4 \left(3 \cos^2 \frac{\phi}{2} - 1\right) - \left(\frac{U_{II}}{r}\right)^2 \right]
\]

Next, consider the suborbital phase

\[
U_B = U_{EA1} + V_{EA1} + W_{EA1} \quad (2-12)
\]

\[
V_B = U_{EA2} + V_{EA2} + W_{EA2}
\]

\[
W_B = U_{EA3} + V_{EA3} + W_{EA3}
\]
where the effective airspeeds $U_{EA}$ and $V_{EA}$ are defined in terms of the north ($W_u$) and east ($W_v$) component of the wind shears. No updraft/downdraft are included in this model. Values of $W_u$ and $W_v$ are stored at 5,000 foot increments, from 0 to 100,000 feet altitude where this simulation starts.

\[ U_{EA} = U_E - W_u \tag{2-13} \]
\[ V_{EA} = V_E - (r \cos \theta) \omega_C - W_v \]

2.3 Rotational Equations: The moments acting on the vehicle are estimated about the body axes as shown below. Let $\ell_r$ be the reference length of the vehicle. Let $J_R$, $J_p$ and $J_Y$ be the dimensionless numbers of roll, pitch and yaw jet firings and $L_R$ and $L_Y$ be the rolling moments due to one roll and one yaw reaction jet firing, respectively. Then

\[ I_B = \frac{55}{2V} \left[ C_{\ell_p} (P_B) + C_{\ell_R} (P_B) \right] + \frac{q_{\ell_b}}{2V} \left[ C_{\ell_\beta} + C_{\ell_\gamma} (\delta \epsilon) \right. \]
\[ + \left. \left( C_{\ell_\beta} + C_{\ell_\gamma} \right) (\delta \ell_1 - 25^\circ) + C_{\ell_\beta} \delta \ell_2 \right] \]
\[ + \frac{q_{\ell_b}}{2V} \left[ C_{\ell_\beta} \delta \ell_3 \right] \]
\[ + \frac{q_{\ell_b}}{2V} \left[ C_{\ell_\beta} (\delta G) \right] \]
\[ + \frac{q_{\ell_b}}{2V} C_{\ell_\beta} (\delta a) + C_{\ell_\beta} (\delta r) \}

\[ + m \cdot \ell_r \left[ (x_{\text{NOM}} - \bar{x}) \cdot y_B - \bar{y} \cdot z_B \right] + (L_R) J_R \]

\[ I_B = \frac{55}{2V} \left[ C_{\ell_p} (P_B) + C_{\ell_R} (P_B) \right] + \frac{q_{\ell_b}}{2V} \left[ C_{\ell_\beta} + C_{\ell_\gamma} (\delta \epsilon) \right. \]
\[ + \left. \left( C_{\ell_\beta} + C_{\ell_\gamma} \right) (\delta \ell_1 - 25^\circ) + C_{\ell_\beta} \delta \ell_2 \right] \]
\[ + \frac{q_{\ell_b}}{2V} \left[ C_{\ell_\beta} \delta \ell_3 \right] \]
\[ + \frac{q_{\ell_b}}{2V} \left[ C_{\ell_\beta} \delta G \right] \]
\[ + \frac{q_{\ell_b}}{2V} C_{\ell_\beta} (\delta a) + C_{\ell_\beta} (\delta r) \}

\[ + m \cdot \ell_r \left[ (x_{\text{NOM}} - \bar{x}) \cdot y_B - \bar{y} \cdot z_B \right] + (L_R) J_R \]

\[ \frac{C_{\ell_\gamma}(\delta \epsilon)_{\text{RIGHT}} + C_{\ell_\gamma}(\delta \epsilon)_{\text{LEFT}}}{2} \]
\[ N_B = \frac{36L^2}{2V} \left[ C_{n_H} (\beta_L) + C_{n_H} (\beta_B) \right] + \bar{q} \bar{b} \left[ C_{n_b} \delta_e \right] \]

\[ + \left( C_{n_{\beta_1}} + C_{n_{\beta_2}} \right) (\delta \bar{b} - 25^\circ) + C_{n_{\beta_3}} \delta \bar{b} - 60^\circ \]

\[ + C_{n_{\beta_4}} (\delta \bar{G}) + C_{n_{\beta_5}} \delta \bar{G} + C_{n_{\delta \alpha}} \delta \bar{G} \]

\[ + C_{n_{\delta \rho \delta}} \delta \bar{G} \]

\[ + \left( |\beta| \right) (\delta \bar{r}) + \left( \bar{N}_C - \bar{N}_L \right) \left( \delta \bar{r} \right) + \left( L_Y \right) \bar{Y} \]

(2-16)

Appropriate reaction-jet terms can be added to the \( M_L \) and \( N_L \) equations if later vehicles are so equipped. The mean aerodynamic chord length is represented by \( \bar{c} \) in the above equation. The symbol \( \bar{x} \) represents the \( x \)-axis displacement of center-of-gravity from an arbitrarily picked reference point. Similar definitions apply to \( \bar{y} \) and \( \bar{z} \).

\[ \bar{x} = (c_{\bar{G}_X} \delta \bar{G}_0 - c_{\bar{G}_X} \text{NOM}) + \frac{\left( \frac{W_F}{W_{F_{\text{max}}}} \right)}{\sin \theta_F} \Delta \bar{G}_X \]

\[ = \frac{\sin \theta_F}{\sin \theta_F} (\Delta \bar{G}_X) \sqrt{\sin \frac{\theta_F}{\theta_F}} \]

\[ \bar{y} = (c_{\bar{G}_Y} \delta \bar{G}_0 - c_{\bar{G}_Y} \text{NOM}) \]

\[ \bar{z} = (c_{\bar{G}_Z} \delta \bar{G}_0 - c_{\bar{G}_Z} \text{NOM}) + \frac{\left( \frac{W_F}{W_{F_{\text{max}}}} \right)}{\sin \theta_F} \bar{Z}_F \]

Here, \( \theta_F \) the angle made by the surface of the fuel with respect to the \( X_B \) axis

\[ \theta_F = 90^\circ + \tan^{-1} \frac{Z_B}{X_B} \]

(2-18)
Using the moments $I_{B}$, $N_{B}$, and $N_{B}$, we can obtain the vehicle angular accelerations about its own axes $- \dot{P}_{B}$, $\dot{Q}_{B}$ and $\dot{R}_{B}$

$$
\dot{P}_{B} = \frac{1}{I_{XX}} \left[ I_{B} - I_{XY}(P_{B}^{R_{B}} + \dot{Q}_{B}) + I_{XZ}(P_{B}Q_{B} + \dot{R}_{B}) \\
+ I_{YZ}(Q_{B}^{2} - P_{B}^{2}) + Q_{B}^{R_{B}}(I_{YY} - I_{ZZ}) \right]
$$

$$
\dot{Q}_{B} = \frac{1}{I_{YY}} \left[ M_{B} + I_{XY}(Q_{B}^{R_{B}} + \dot{P}_{B}) - I_{XZ}(P_{B}^{2} - R_{B}^{2}) \\
- I_{YZ}(P_{B}Q_{B} - R_{B}) + P_{B}^{R_{B}}(I_{ZZ} - I_{XX}) \right] \tag{2-19}
$$

$$
\dot{R}_{B} = \frac{1}{I_{ZZ}} \left[ N_{B} - I_{XY}(Q_{B}^{2} - P_{B}^{2}) - I_{XZ}(Q_{B}^{R_{B}} - \dot{P}_{B}) \\
+ I_{YZ}(P_{B}Q_{B} + \dot{Q}_{B}) + P_{B}^{R_{B}}(I_{XX} - I_{YY}) \right]
$$

where the inertia terms are defined as follows

$$
I_{X} = I_{XX} = I_{XX_{0}} = \frac{W_{F}}{W_{F_{max}}} I_{XX_{F}}
$$

$$
I_{Y} = I_{YY} = I_{YY_{0}} + \frac{W_{F}}{W_{F_{max}}} I_{YY_{F}} \tag{2-20}
$$

$$
I_{Z} = I_{ZZ} = I_{ZZ_{0}} + \frac{W_{F}}{W_{F_{max}}} I_{ZZ_{F}}
$$

$$
I_{XZ} = I_{XZ_{0}} + \frac{W_{F}}{W_{F_{max}}} I_{XZ_{F}}
$$

$$
I_{XY} = I_{XY_{0}} + \frac{W_{F}}{W_{F_{max}}} I_{XY_{F}}
$$
\[
I_{YZ} = I_{YZ_0} + \left( \frac{W_F}{\frac{W_F}{W_F_{max}}} \right) I_{YZ_F}
\]

Fuel consumption and the weight of fuel are described by
\[
\dot{W}_F = -(F_T)T ; T = \text{Thrust}
\]
\[
W_F = W_{F_0} + \int_0^t \dot{W}_F dt
\]

By integrating the angular accelerations we acquire the angular velocities \( \alpha_B \), \( \beta_B \) and \( \gamma_B \) (also known as the rolling rate, pitching rate and yawing rate).
\[
P_B = P_{B_0} + \int_0^t \dot{P}_B dt
\]
\[
Q_B = Q_{B_0} + \int_0^t \dot{Q}_B dt \tag{2-22}
\]
\[
R_B = R_{B_0} + \int_0^t \dot{R}_B dt
\]

2.4 Other Pertinent Equations: Euler angles namely pitch, yaw and roll, are calculated below. Notice that this formulation will not tolerate a pitch angle \( \theta \) of \( \pm 90^\circ \).
\[
\theta = \theta_0 + \int_0^t (Q_B \cos \phi - R_B \sin \phi) dt
\]
\[
\phi = \phi_0 + \int_0^t (P_B + \tan \theta (R_B \cos \phi + Q_B \sin \phi)) dt \tag{2-23}
\]
\[
\psi = \psi_0 + \int_0^t \left( \frac{R_B \cos \phi + Q_B \sin \phi}{\cos \theta} \right) dt
\]
Stability-axis angles, namely angle of attack $\alpha$ and angle of sideslip $\beta$ can be computed as

$$
\alpha = \tan^{-1} \left( \frac{W_B}{U_B} \right) \tag{2-24}
$$

$$
\beta = \tan^{-1} \left( \frac{V_B \cos \alpha}{U_B} \right)
$$

Flight path angle

$$
\gamma = \delta - \alpha \cos \phi - \beta \sin \phi \tag{2-25}
$$

Let $\psi_R$ be the given angle between true north and the $R_D$ vector; then down range and cross range can be computed as:

$$
R_D = r_0 \left[ (L - L_R) \cos \psi_R + (\lambda - \lambda_0) \cos L \sin \psi_R \right] \tag{2-26}
$$

$$
R_X = r_0 \left[ (L - L_R) \sin \psi_R + (\lambda - \lambda_0) \cos L \cos \psi_R \right]
$$

Equivalent airspeed $KEAS$ and calibrated airspeed $KCAS$

$$
KEAS = \left( \frac{V \sqrt{\frac{p}{\rho_0}}} {6080} \right) \frac{3600}{6080} \text{ nautical miles}
$$

$$
KCAS = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, (KEAS) \tag{2-27}
$$

The quantities $v_1$ and $v_2$ are defined by the following empirical formulae:

$$
v_1 = f(x) \quad ; \quad x = \text{Mach number} \tag{2-28}
$$

$$
v_2 = f(x) \quad ; \quad x = (.001511) (KCAS)
$$

Where $f(x)$ assumes the following form:
\[
f(x) = \begin{cases} 
\sqrt{1 + \frac{x^2}{4} + \frac{x^4}{40} + \frac{x^6}{1600}} & \text{for } x < 1.0 \\
\sqrt{1.839 - \frac{.772}{x^2} + \frac{.164}{x^4} + \frac{.035}{x^6}} & \text{for } x \geq 1.0 
\end{cases}
\] (2-29)

Geometric altitude \( h \) and geopotential altitude:

\[ h = \delta r + (\sin^2 L) \delta h \] (2-30)

\[ h_p = \frac{(r_0)(h)}{r_0 + h} \]

Rate of climb:

\[ \frac{d}{dt} (h) = -v_H = -v_L \] (2-31)

Air density can be approximated as a function of altitude, based upon standard atmospheric data:

\[ \rho = \rho_0 e^{-\zeta h_p} \] (2-32)
3. MAIN PROGRAM

No efforts are spared to organize the main program in a self-explanatory manner. Parameters and data are grouped together according to their roles and the omnipresence of the comment cards should guide the reader through the program. It is suggested that subsequent users do not deviate from the existing logical organization when making modifications/additions, so not to defeat the purposes. In order to simplify the loading process, all parameters and data (including aerodynamic coefficients) are stored in the main program, thus avoiding the use of COMMON blocks.

As indicated by figure 3-1, the main program is composed of three parts: the initialization, the Hi-speed loop (starting from statement 47), and the EASE subroutine designed to display and modify any parameter of the main program. EASE was written to allow interchange of parameters without the use of COMMON blocks.

3.1 Initialization: Most data parameters and initial conditions are on punched cards to allow maximal flexibility on flight conditions and vehicle configurations. All aerodynamic derivatives are stored in the form of 16-bit normalized fixed point fractions. The detailed discussion of their handling will be covered under the heading of "Function Generation". During the initialization phase the program will ask the user if he is through with changing parameters before entering into the next phase, by typing the question "ARE FURTHER CHANGES DESIRED?" A "YES" answer will automatically call in the EASE program. A "NO" answer will set in motion the high-speed loop.

Not all parameters can be independently initialized. For instance, $V_E$ is called an "intermediate" variable because it is a function of $\psi_H$ and $U_H$. Thus the initial value of $V_E$ is determined once the initial conditions of $\psi_H$ and $U_H$ are specified, and vice versa.

3.2 Hi-Speed Loop: This loop is actuated by an interrupt which occurs at synchronous, regular intervals $\Delta t$ whose duration was set a-priori. During the time $\Delta t$, the computer must finish solving one frame of the dynamic equations before the occurrence
ENTER → Deactivate Interval Timer and Initialize Handler

Read from Disc
- Constants and Parameters
- Aerodynamic Derivatives
- Initial values of the state variables

RE-ENTER →

10 ABORT requested?
Y → Disable Interval Timer
N → STOP

1) Initialize digital input for mode control
2) Calculate intermediate parameters from given values

Set-up and Enable Interval Timer

47 Loop Start Point

Disable Interval Timer Wait for 2 seconds

Y → POTSET mode?
N → To be continued next page

Figure 3-1 - Main Program Flowchart
Figure 3-1 (Continued)

1) A-D conversion of FCS commands
2) Reading of control switches from cockpit (IC, HOLD, 3DOF, windy atmosphere, etc...)

Lock control surfaces IF corresponding FCS signals are disconnected

IC mode?

HOLD mode?

Y

N

Y

N

Y

N

Initialize Time Counter=0

Initialize P, Q, R, θ, ϕ & ψ
Initialize latitude, longitude, $U_H$, $Ψ_H$, $W_H$ and $δ_R$

170

1) Calculate vehicle body-axis moments $L_B$, $M_B$, $N_B$
2) Calculate $\dot{P}$, $\dot{Q}$, $\dot{R}$

Table lookup of aerodynamic coeff's $C_1$, $C_m$, $C_n$, $C_C$, $C_N$, $C_Y$

To Be Continued On Next Page
Integration to obtain body rate of rotation $P, Q, R$

IC or HOLD mode?

Calculate Euler-angle rates $\dot{\theta}, \dot{\phi}$ and $\dot{\psi}$

Want 3-DOF only?

Integration to obtain $\theta, \phi$ and $\psi$

Want 5-DOF only?

Set accelerations $\ddot{U}_B$, $\ddot{V}_B$, and $\ddot{W}_B$ to zero

Calculate body-axis acceleration $X_B$

Evaluate body-axis accelerations $Y_B$, $Z_B$

To Be Continued on Next Page
Figure 3-1 (Continued)

1) Coordinate transformation to obtain earth-axis accelerations $X_E$, $Y_E$, $Z_E$
2) Coordinate resolution to obtain H-frame accelerations $X_H$, $Y_H$, $Z_H$
3) Computation of gravity

Calculate H-frame derivatives $\dot{U}_H$, $\dot{Ψ}_H$
$\dot{W}_H$, $\dot{L}_H$, $\dot{λ}$, $\dot{δ}_R$

Want 5-DOF only?

Set $\dot{δ}_R = 0$

IC or HOLD mode?

1) Integrate to obtain $\dot{U}_H$, $\dot{Ψ}_H$, $\dot{W}_H$, $\dot{L}_H$, $\dot{λ}$, $\dot{δ}_R$
2) Calculate $r$, $U_H$ separately

1) Compute down range, cross range & altitude
2) Compute vehicle ground speeds $U_E$, $V_E$ and $W_E$

WIND=0

Windy atmosphere?

Wind model subroutine
1) Compute vehicle airspeed components $U_{EA}$; $V_{EA}$; $h$
2) Generate body-axis velocities $U_B$; $V_B$; $W_B$

Evaluate $\alpha$ and $\beta$

Want 5DOF only?

Y → True airspeed $V=\text{constant}$

N → Evaluate $V$

1) Calculate $\rho$, $\tilde{q}$, equivalent airspeed, calibrated airspeed, Mach.
2) Digital FCS processing
3) Examine pulse flags (DEPULSE, DAPULSE and DRPULSE)
4) Apply pulse to the desired surface
5) Digital FCS Processing (bypassed if any surface is pulsed)
6) Examine surface flags (FCSDE, FCSDA, FCSDR, FCSSB, FCSBF)
7) Set surfaces that are flagged equal to their IC values (DELIC, DERIC, DRIC, DSBIC, DBFIC)
8) Preparation of output

To be continued on next page
of the next interrupt. In order to achieve the real-time (or time-critical) goal, great care was made to minimize the execution time. Even so, a number of calculations have to be delegated to the analog computer such as

- Sin/cos functions of angles $\theta$, $\phi$, $\psi$, $\alpha$ for instrumentation

- Generation of test pulses

In fact, if more equations are to be added into the dynamic model, some other existing equations must be deleted from the high-speed loop. An alternative would be solving the deleted equations on another digital or analog processor. The dynamic model is given in Chapter II, thus not repeated here. During run time, the operator can intervene by throwing a number of switches which in turn activate a number of interrupts in the digital computer. The switches are situated in the general cockpit area and they consist of
- IC Switch: sets the program back to Initial Condition mode. All information concerning the preceding run is lost.

- HOLD Switch: also stops execution but all state variable values are frozen at their respective values reached prior to activation of the switch. This mode allows the user to investigate the status of his program in post-mortem fashion.

- 3 DOF Switch: sets the yaw rate \( \dot{\psi} \) and the roll rate \( \dot{\theta} \) to zero, the vehicle can then be studied entirely in the pitch plane, having three degrees-of-freedom: \( \dot{x}, \dot{z}, \) and \( \theta \). This mode is useful in planning and performance analysis such as plotting the gliding envelope. In this case, we are interested only in getting from one point to another point.

- 5 DOF Switch: imposes a nearly constant altitude, constant total-velocity flight regime. This mode offers a convenient way to study vehicle handling quality and maneuverability. To different people, 5-DOF simulation means slightly different things. Thus we derive the 5-DOF situation used in this simulation for reference purposes.

Consider the acceleration equations in body-axes, excluding the thrust and aerodynamic force on the surfaces.

\[
\begin{bmatrix}
X_B \\
Y_B \\
Z_B
\end{bmatrix}
= \begin{bmatrix}
\dot{X}_B \\
\dot{Y}_B \\
\dot{Z}_B
\end{bmatrix}
= \begin{bmatrix}
0 & W_B & -V_B \\
-W_B & 0 & U_B \\
V_B & -U_B & 0
\end{bmatrix}
\begin{bmatrix}
P_B \\
Q_B \\
R_B
\end{bmatrix}
\]
\[
\begin{bmatrix}
\cos \psi \cos \theta & \sin \psi \cos \theta & -\sin \theta \\
(c\psi \sin \theta \cos \phi - s\psi \sin \phi) & (s\psi \sin \theta \cos \phi + c\psi \cos \phi) & c\theta \sin \phi \\
(c\psi \sin \theta \cos \phi + s\psi \sin \phi) & (s\psi \sin \theta \cos \phi - c\psi \sin \phi) & c\theta \cos \phi
\end{bmatrix}
\begin{bmatrix}
0 \\
0 \\
-g
\end{bmatrix}
\] (3-1)

Inertial-axis \rightarrow\text{ body-axis transformation}

Where \(c\psi\) and \(s\psi\) are the abbreviations for \(\cos \psi\) and \(\sin \psi\) respectively.

For 5DOF, assume that \(\dot{U}_B = \dot{V}_B = \dot{W}_B = 0\) and solve for the acceleration component \(X_B\)

\[
X_B = -\frac{Z_B W_B + Y_B V_B}{U_B} + g \left[ \sin \theta - \frac{\cos \phi (V_B \sin \phi - W_B \cos \phi)}{U_B} \right]
\] (3-2)

Equation (3-2) is used in lieu of \(X_B\) described in equations (2-4). The only other changes in the 5-DOF case are

\[
\frac{d(\Delta r)}{dt} = 0
\] (3-3)

and \(V = V_{IC}\), a constant

The closed-loop nature of the hi-speed loop is illustrated by the block diagram of figure 3-2. This block diagram depicts the main stream of the calculations and how different groups of calculations are related to each other functionally.
Figure 3-2: Block Diagram

**Euler-Axis Accelerations**

\[ X_E = X_B l_1 + Y_B m_1 + Z_B n_1 \]
\[ Y_E = X_B l_2 + Y_B m_2 + Z_B n_2 \]
\[ Z_E = X_B l_3 + Y_B m_3 + Z_B n_3 \]

**Directional Cosines**

\[ l_1 = \cos \theta \cos \psi \]
\[ l_2 = \cos \theta \sin \psi \]
\[ l_3 = -\sin \theta \]
\[ m_1 = -\cos \phi \sin \psi + \sin \phi \sin \theta \cos \psi \]
\[ m_2 = \cos \phi \cos \psi + \sin \phi \sin \theta \sin \psi \]
\[ m_3 = \sin \phi \cos \theta \]
\[ n_1 = \sin \phi \sin \psi + \cos \phi \sin \theta \cos \psi \]
\[ n_2 = -\sin \phi \cos \psi + \cos \phi \sin \theta \sin \psi \]
\[ n_3 = \cos \theta \cos \phi \]

**H - Axis Accelerations**

\[ X_H = X_E \cos \psi_H + Y_E \sin \psi_H \]
\[ Y_H = -X_E \sin \psi_H + Y_E \cos \psi_H \]
\[ Z_H = Z_E \]

**Winds and Rotating Earth**

\[ U_{EA} = U_E - \omega \times \mathbf{V} \]
\[ V_{EA} = V_E - \omega \times \mathbf{V} \]

**Body Axis Rotational Accelerations**

\[ \dot{P}_B = \frac{1}{Y_B} \left[ L_B - I_{XY} (P_B R_B - \dot{Q}_B) + I_{XZ} (P_B Q_B + \dot{R}_B) + (Q_B R_B) (Q_Y - I_Z) \right] \]
\[ \dot{Q}_B = \frac{1}{X_B} \left[ M_B + I_{XY} (Q_B R_B + \dot{P}_B) - I_{XZ} (P_B^2 - R_B^2) + (P_B R_B) (Q_Y - I_Z) \right] \]
\[ \dot{R}_B = \frac{1}{Z_B} \left[ N_B + I_{XY} (Q_B^2 - P_B^2) - I_{XZ} (Q_B R_B - \dot{P}_B) + (P_B Q_B) (Q_Y - I_Z) \right] \]
\[ g = g_0 \left( \frac{R}{R + h} \right)^2 \]

\[ \delta r = \delta r_0 + \int v e^2 dt \]

\[ \delta \theta = \delta \theta_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \theta dt \]

\[ \lambda = \lambda_0 + \int \left[ \frac{V^2 - 1}{H} \right] \sin \theta \cos \theta dt \]

\[ \psi = \psi_0 + \int \left[ \frac{V^2 - 1}{H} \right] \sin \theta \sin \phi dt \]

\[ \theta = \frac{g_0}{R} \frac{R^2}{H} \]

\[ \phi = \phi_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \theta \sin \phi dt \]

\[ \beta = \beta_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \theta \cos \phi dt \]

\[ \alpha = \alpha_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \alpha dt \]

\[ \psi = \psi_0 + \int \left[ \frac{V^2 - 1}{H} \right] \sin \theta \sin \phi dt \]

\[ \theta = \frac{g_0}{R} \frac{R^2}{H} \]

\[ \phi = \phi_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \theta \sin \phi dt \]

\[ \beta = \beta_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \theta \cos \phi dt \]

\[ \alpha = \alpha_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \alpha dt \]

\[ \psi = \psi_0 + \int \left[ \frac{V^2 - 1}{H} \right] \sin \theta \sin \phi dt \]

\[ \theta = \frac{g_0}{R} \frac{R^2}{H} \]

\[ \phi = \phi_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \theta \sin \phi dt \]

\[ \beta = \beta_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \theta \cos \phi dt \]

\[ \alpha = \alpha_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \alpha dt \]

\[ \psi = \psi_0 + \int \left[ \frac{V^2 - 1}{H} \right] \sin \theta \sin \phi dt \]

\[ \theta = \frac{g_0}{R} \frac{R^2}{H} \]

\[ \phi = \phi_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \theta \sin \phi dt \]

\[ \beta = \beta_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \theta \cos \phi dt \]

\[ \alpha = \alpha_0 + \int \left[ \frac{V^2 - 1}{H} \right] \cos \alpha dt \]

\[ \psi = \psi_0 + \int \left[ \frac{V^2 - 1}{H} \right] \sin \theta \sin \phi dt \]
4. FLIGHT CONTROL SYSTEM (FCS)

4.1 Description: In this simulation we investigate the Orbiter handling qualities under manual control of a pilot. There is no provision for an autopilot in this program. From the time the Orbiter enters the atmosphere at a nominal altitude of 400,000 ft to touchdown it is not power-assisted except for the Reaction-jet Control System (RCS). Thus the importance of the FCS can't be too much exaggerated.

The Orbiter is equipped with an all-digital fly-by-wire FCS which insures aerodynamic stability with the help of five on-board computers (four redundant primary and one secondary). The computers, in turn, feed signals to the actuators of the control surfaces. A picture of the control surfaces can be found on page 22 of reference 9.2.2.

   a/ Elevons (Elevator-aileron combination): They are full-span in construction, used for affecting both pitch and roll. This program does not exercise the inboard and outboard elevons separately.

   b/ Rudder/Speedbrakes: The conventional rudder is made up of two symmetrical halves which, when deployed, constitute the speed brakes.

   c/ Body flap is found on the very end of the fuselage below the main engines and serves mainly as augmentation to longitudinal trim.

Whenever the control surfaces prove to be ineffective (high altitude, low dynamic pressures) pitch and roll reaction-jets are fired to guaranty the orbiter's aerodynamic stability. The yaw reaction jets are used more often, including the cases just mentioned above.

4.2 FCS Block Diagrams: The pitch, roll and yaw modes of the FCS are depicted by the block diagrams on figures 4-1, 4-2 and 4-3 in that order. Let us discuss one of the modes in detail, say the pitch mode of figure 4-1. A command from the RHC (Rotational Hand Command) is fed through the deadband and pitch shaping networks then compared with the current Orbiter performance (a function of the pitch rate Q) to generate a pitch rate error signal called DPJ. Because of the coupling effects between the axes, the nose tends to dip during a turn. To compensate for this, we want an additional negative elevon deflection (trailing edge up, by convention) and the term R*\(\tan \theta\) does exactly this by contributing a negative influence to the elevator command \(\delta_{e\text{CMD}}\).
When the speedbrakes are deployed, they force the nose upward by contributing a moment around the Orbiter's center of gravity (CG). Note how the speedbrake increment (DSBPC) signal is used to negate the unwanted pitch-up by a direct reduction in the pitch-position trim.

The FSC roll channel is represented on figure 4-2. The switches EARLY and LATE correspond to MACH > 1.5 and MACH ≤ 1.5 respectively. Notice the roll-rate command main path. Starting with the roll stick position, the signal goes through deadband, shaping and first-order lag circuits before being added with the integral of the roll-rate trim term at SUM21. From there, the control signal is sent through SUM24 where the roll-rate feedback and other coupling terms are subtracted. All the feedback signals taken together are called PSTAB (roll-rate stability). SUM24 is directed to a switch whose poles are marked EARLY and LATE. The EARLY side of the switch allows only reaction jets to control the roll channel because, at that time, the Orbiter is still at high altitude. Only during the LATE stage can the control surfaces perform effectively.

The FCS yaw channel is illustrated on figure 4-3. The rudder pedal generates a signal which is fed through the dead-band and shaping networks. Then it is summed with the integrated yaw-trim signal. Before reaching the actuators proper, this signal is further modified by Mach number and dynamic pressure q. An arrangement similar to the roll-axis channel is implemented here to allow the yaw jets UZCMD to take over at high altitude (EARLY). Only at low altitude (LATE) can the rudder work the way it works on an airplane.

Figure 4-4 describes how the Orbiter's control surfaces are exercised by different actuators.

4.3 FCS Computer Implementation: In the block diagrams the transfer functions and filters are expressed in the Laplace s-domain. We could have used the s-plane if an analog computer were used. Since the FCS program is done digitally it is advantageous to map the transfer functions from the s-plane to another complex plane, the z-plane. In simple terms, z can be defined as \( z = e^{sT} \) where T is the sampling period. A comprehensive treatment of infinite-impulse-response digital filters can be found in reference 9.1.6, chapter 4.

It is a known fact that a stable analog filter might or might not map into a stable digital filter if we just go about replacing the differentials by finite
FIGURE 4-3: FCS YAW CHANNEL
FIGURE 4-4: FCS ACTUATORS
differences (backward, forward, or central differences). A more elaborate process is used in this program to prevent computational instability and aliasing problems. It is called Bilinear transformation (simple conformal mapping) and it is represented by the following transformation:

\[
s = \frac{2}{T} \frac{(1 - z^{-1})}{(1 + z^{-1})}
\]  

(4-1)

Consider a first-order transfer function as an example. Given

\[
H(s) = \frac{10}{s + 10}
\]  

(4-2)

Substituting \( s \) for \( z \) as dictated by equation (4-1)

\[
H(z) = \frac{10}{\frac{2}{T} \frac{(1 - z^{-1})}{(1 + z^{-1})} + 10} \frac{10T + 10Tz^{-1}}{2 - 2z^{-1} + 10T + 10Tz^{-1}}
\]  

(4-3)

In this program the frame time is \( T = 0.04 \) second. Any first-order transfer function can be, for our programming purposes, reduced to the form shown below:

\[
\frac{Y}{X} = \frac{G_1 + G_2 z^{-1}}{1 + G_3 z^{-1}}
\]  

(4-4)

Similarly we put all second-order transfer functions in the following form prior to digital coding:
Thus the z-transform of equation (4-2) finally becomes, after introducing the proper value for T into equation (4-3)

\[
\frac{Y}{X} = \frac{G_1 + G_2z^{-1} + G_3z^{-2}}{1 + G_4z^{-1} + G_5z^{-2}}
\]  

(4-5)

Computer mechanization of equations (4-4) and (4-5) are done in the form of function subroutines called FILT1 and FILT2 respectively. FILT1 and FILT2 stand for first-order filter and second-order filter. For FILT1 the expression comes directly from equation (4-4)

\[
Y_n = G_1X_n + X_{OLD}
\]

(4-7)

\[
X_{OLD} = G_2X_{n-1} - G_3Y_{n-1}
\]

For FILT2, equation (4-5) can be put into the format

\[
Y_n = G_1X_n + X_{node1}
\]

(4-8)

where

\[
X_{node1} = G_2X_{n-1} - G_4Y_{n-1} + X_{node2}
\]

\[
X_{node2} = G_3X_{n-2} - G_5Y_{n-2}
\]

Refer to Table 4-1 to correlate an s-domain transfer function to its z-domain counterpart. Besides the filters, the hysteresis transfer function also deserves a brief mention. Figure 4-5 depicts the flowchart of the HYSTER subroutine and it is self-explanatory.

A detailed flowchart of the Shuttle Flight Control System (SHTLFCS subroutine) is contained in Figures 4-6. The actual listing of SHTLFCS can be found in Appendix D.
**TABLE 4-1 : Transfer Functions**

<table>
<thead>
<tr>
<th>Transfer Functions (s-domain)</th>
<th>z-Transform Equivalent (for sampling period T = .04 sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{1}{s} )</td>
<td>( \frac{.04}{1 - z^{-1}} )</td>
</tr>
<tr>
<td>( \frac{s}{s + 1} )</td>
<td>( \frac{.9804 - .9608z^{-1}}{1 - .9608z^{-1}} )</td>
</tr>
<tr>
<td>( \frac{5}{s + 5} )</td>
<td>( \frac{.0909 + .0909z^{-1}}{1 - .8182z^{-1}} )</td>
</tr>
<tr>
<td>( \frac{7.5}{5s + 1} )</td>
<td>( \frac{.02988 + .02988z^{-1}}{1 - .992z^{-1}} )</td>
</tr>
<tr>
<td>( \frac{10}{s + 10} )</td>
<td>( \frac{.1667 + .1667z^{-1}}{1 - .6667z^{-1}} )</td>
</tr>
<tr>
<td>( \frac{1}{s + 1} )</td>
<td>( \frac{.01961 + .01961z^{-1}}{1 - .9608z^{-1}} )</td>
</tr>
<tr>
<td>( \frac{20s}{20s + 1} )</td>
<td>( \frac{.999 - .999z^{-1}}{1 - .998z^{-1}} )</td>
</tr>
<tr>
<td>( \frac{s + 1}{.5s + 1} )</td>
<td>( \frac{1.9615 - 1.8846z^{-1}}{1 - .9231z^{-1}} )</td>
</tr>
<tr>
<td>( \frac{15^2}{s^2 + 15s + 15^2} )</td>
<td>( \frac{.06475 + .1295z^{-1} + .06475z^{-2}}{1 - 1.309z^{-1} + .5683z^{-2}} )</td>
</tr>
</tbody>
</table>
Figure 4-5: Hysteresis Function

HYSTER

TEMP1 = |ARGT1|

TEMP1 ≥ LIM1

Y

20

LIM1

TEMP1 < LIM2?

N

30

ARGT1 ≥ 0?

Y

50

FUNCT=1

N

FUNCT=-1

80

ARGT1 ≥ 0?

Y

70

ARGTO ≥ LIM1?

N

10

ARGTO > -LIM1?

Y

10

N

ARGTO < LIM1?

Y

900

RETURN

35
Figure 4-6: SHTLFCS Flow Chart

- Generate SIN(ALPHA) Schedule Breakout
- Generate COS(ALPHA) , SIN_PHI
  - COS_PHI , COS_THETA
- Generate roll, pitch & yaw trim breakouts for DETMPAN, DETMRHC, DATMPAN
  - DATMRHC, DRT and DBFMAN
- Generate values of Mach-dependent function schedule for GPS, GJET, ALPHMAX,
  - ALPHMIN, KPI, GTRE, GDSB, KDAMPAR
  - KDAMLIN, GRS, CDAC, GP, GNYDRM,
  - GRAY, KDRC, GTRR

PITCH Processing

ESHAPE

Mach > 1.5
  - AND.
  - PRHCSOP > 19.5

Y

DEMSP

DEMSP = 2^DEMS

36
Figure 4-6: Continued

IC or HOLD?

Y

20

N

DETR

DPJET; UYCMD

Mach ≤ 1.5

or q > 20.

Y

UYCMD = 0

N

50

UYGJET

RTANPHI

N

Mach > 1.5

Y

RTANPHI = 0

B CSL; QFFIL

NZP; DCSLLO

DCSLHI; MIDPICK

To be continued on next page
Figure 4-6: Continued

Mach > 1.5

Y

105

Y

\( \bar{q} > 2 \)

N

DCSL = BCSL

Y

DCSL = 0

110

DQCT; GTREDQCT

Y

DSB \geq 30^\circ?

N

DSBXTRS = DSBXTR * GSBHI

Y

DSBXTRS = DSBXTR * GSBLO

120

DETP

To be continued on next page
Figure 4-6: Continued

IC or HOLD

N

ETRIM
ETRIM ELFBK

121

Y

Mach ≤ 12.

N

DETTRIM=ELFBK

DETTRIM=ETRIM

140

DECMD

ROLL Processing
Part One

DAMAN ; PROD22

IC or HOLD

N

PROD25

Y

171

DRPHI

To be continued on next page
Figure 4-6: Continued

\[ \text{Turncord} \]

- **DRPRM = R**
- **DRPRM = R - DRPHI**

174

**PROD29 ; BETAFILT**

- \( \bar{q} \leq 2 \)
  - **Y**
  - **N**
    - **\( \bar{q} > 20 \)**
      - **Y**
      - **N**
        - **BETAQ = 0**
        - **BETAQ = BETAFILT**
          - **SUM23 ; RSTAB**
          - **PSTAB ; DACM**

**YAW Processing**

- **DRMAN ; DRMS**
- **DRTMSF ; DRTMS**
- **PROD1**

To be continued on next page
Figure 4-6: Continued

N

TURNCORD
?
Y

SUM1 = -PROD1

NYP

SUM1 = NYP - PROD1

230

PROD2 ; PROD3
SUM2 ; GDRC
PROD5

Mach ≤ 1.5

Y

N

GDRE

DRCPF

235

DRCPF = 0

N

Mach ≤ 5.

Y

N

236

DRTTR

DRTRIM

GRXFDF

To be continued on the next page
Figure 4-6: Continued

\[ \text{DACMC} \leq 0 \]  
\[ \text{AND} \]  
\[ \text{TURNCORD} \]

\[ \text{Mach} \leq 1.5 \]

\[ \text{DACMC} = \text{GRXFD} \times \text{DACM} \]

\[ \text{SUM3} ; \text{DRCMD} \]

\[ \text{Mach} > 1.5 \]

\[ \text{DRJET} = \text{SUM23} \]

\[ \text{DRJET} = \text{GRCSA} \times \text{SUM2} \]

\[ \text{DRJET} = 0 \]

\[ \text{NR} \]  
\[ \text{(Low } \bar{q} \text{)} \]

\[ \text{NR} \]  
\[ \text{(High } \bar{q} \text{)} \]

\[ \bar{q} \leq 20. \]

262

264
Figure 4-6: Continued

ROLL Processing
Part Two

PROD41 = \min(\bar{q} > 2.

PROD41 = GP*YAWXFEED

GDA = 200/(\bar{q}+80)

DCSP = GDA*DACM

GDA = 150/\bar{q}

DCSP = GDA*PROD41

To be continued on next page
Figure 4-6: Continued
**Figure 4-6: Continued**

IC or HOLD

- **Y**
  - 300
  - DACMD
  - ACTUATOR Processing
    - DECDRR
    - DECDRL
    - \( DE = (DEL + DER) / 2 \)
    - DSBC
    - DRRP
    - DRLP
    - \( DSB = (DRLP - DRRP) / 2 \)
    - DBF
  
  - DER
  - DEL
  - \( DA = (DEL - DER) / 2 \)
  - DRCDRR
  - DRCDLR
  - \( DR = (DRRP + DRLP) / 2 \)
  - DBFRC

- **N**
  - DATSUMI

RETURN
5. EASE PACKAGE

EASE is an interactive package that allows the researcher easy access to the Simulation program in the format that is natural to him. To qualify as a general-purpose package, the program must readily accept different vehicle configurations and flight conditions which vary from case to case. The list of parameters and variables can be found in Chapter VI together with their units and computer symbols. Users of EASE are required to follow two simple formats corresponding to an interrogation and an input. EASE can be accessed through the Main program or by setting Control Switch 12 to .TRUE.

a/ Examples of Interrogation Mode

THETA? What is the current value of \( \theta \)? (in F25.10 format)

*12AF? What is the integer content of address 12AF? (In decimal format I25). The asterisk (*) specifies the integer mode.

43600? What is the floating-point content of address 43600?

b/ Examples of Input Mode

OMEGA = .000072923 Replace the content of OMEGA with .72923E-4.

*17777 = 50 Replace the content of cell 17777 with 50 decimal

1FF2C = .25 Put .25 (in floating-point format) into cell 1FF2C.

c/ To exit EASE, type an exclamation mark "!" followed by a carriage return (C/R).

For unusually large numbers or unusually small numbers, check the
accompanying listing to avoid truncation of the input data. For example, the I/O floating-point number is determined at format F25.10. Any digits beyond 10 places to the right of the decimal point will be ignored. The user should also safeguard against entering a value for a parameter that is larger than the largest allowable for the parameter. For example, the pitch rate $Q_B$ has a range of $\pm 2$ rad/sec and is scaled at $[Q_B/2]$ when appearing to the Flight Control System. If the user of EASE inadvertently sets $Q_B = 3$, the output of the DAC (digital-to-analog converter) corresponding to $Q_B$ is completely wrong.

When a wrong symbol was typed, EASE will respond with "NO SUCH SYMBOL IS DEFINED TO EASE". There are other interactive features built into EASE to help the users in case of mistakes. Here are a few common typing mistakes to serve as examples:

a/ $\text{OMEGA} = 2.6F19$

EASE will come up with the message "MEMORY DATA MUST BE A DECIMAL NUMBER" because there is no provision for EASE to accept hexadecimal numbers.

b/ $\ast \text{RHO}$? (even though $\rho$ was declared a floating-point in the main program). The resulting message is "WRONG INPUT FORMAT".

c/ $\text{BYTE} = 377$ (even though BYTE was declared as 8 bits)

In that case $255 \leq \text{BYTE} \leq 0$. EASE will alert the user with the warning "DATA OUTSIDE RANGE".

The simplified flowchart for the EASE package is contained in figure 5-1 for reference purposes.
Figure 5-1: EASE Flowchart, Executive

Enter EASE

Display "WELCOME TO EASE"

10 RETURN

1) Execute Carriage Return
2) Input 1 Line From CRT
3) Store Line in WORD(L)
4) L=L

15 L=L+1

Examine Character in WORD(L)

Y

Display "END OF EASE PROCESSING"

Exit EASE

N

To be continued on next page
To Number Address Processing

Examine Character in WORD(L)

To CRT Output Processing

To CRT Input Processing

---
Figure 5-1 (continued), Letter Processing

200

1) LETTER=.TRUE.
2) J=1
3) TEMP=""

TEMP(J)=WORD(L)

210

220

1) I=I+1
2) Examine Character in WORD(L)

RETURN

Display "VARIABLE NAME EXCEEDS 8 CHARACTERS"

N

Compare SYMBOL to LIST(K)

a blank?

Y

is J=9?

N

J=J+1

250

a "?"?

Y

Display "NO SUCH SYMBOL IS DEFINED TO EASE"

Store K For Use By Executive

RETURN

N

an "="?

N

Y

is SYMBOL in LIST(K)?
Figure 5-1 (continued), Number Address Processing

300

1) NUMB=''
2) J=1

310

NUMB(J)=WORD(L)

1) L=L+1
2) Examine Character in WORD(L)

320

Y

blank?

N

a

"?"

? Y

350

RETURN

Display "MEMORY ADDRESS MUST BE A HEX NUMBER"

N

a

Hex Number?

Y

Display "MEMORY ADDRESS EXCEEDS 5 DIGITS"

J=J+1

N
Figure 5-1 (continued), Number Address Processing

1) Shift NUMB right by SHIFT Bytes
2) Convert ASCII to Hex
3) Store Hex Address in ADDR

is

ADDR,GE,7FFFF?

Display "MEMORY ADDRESS IS OUTSIDE OF CORE LIMITS"

RETURN
Figure 5-1 (continued), CRT Output
Processing
Figure 5-1 (continued), CRT Input Processing

1) \( J = 1 \)
2) \( \text{NAMEIN} = ' ' \)

\[ \]

1) \( L = L + 1 \)
2) Examine Character in \( \text{WORD}(L) \)

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]

\[ \]
Figure 5-1 (continued), CRT Input Processing

520

is

INTEGR=TRUE

?  

Y  

1) Right Justify Number in NAMEIN  
2) Convert ASCII to Integer  

530

is

LETTER=TRUE

?  

N  

Y  

is

INTEGR=TRUE

?  

N  

1) Insert Decimal Point  
2) Convert ASCII to REAL  

Store Real or Integer Data in Proper Data Array Location

Store Integer Value in Memory Location Specified by ADDR

Store Real Value in Memory Location Specified by ADDR

599

RETURN
6. FUNCTION GENERATION PACKAGE (FGP)

Fast function generation constitute the main objective of this software package so it can successfully support a time-critical simulation. The package is Fortran compatible making it adaptable elsewhere if needed. On the other hand, because of speed consideration, some part of this program were written in Assemble language. Physically the FGP is embedded into the Main program and considered an integral part thereof. This feature also helps speed up the execution time. Aerodynamic data for this particular simulation are composed of about 40 functions having from one to three arguments. Overall, there exist more than 4000 data points to be handled by the FGP. If there is a shortage of memory space each data point could be converted into a fixed-point quantity, normalized then made to fit into a 16-bit half-word location. But in that case, the FGP must be greatly modified.

We are provided with data in the form of punched cards. The card image of a representative two-argument function is shown below. In this case the function is the pitching-moment coefficient (or derivative) bias, \(CM_0(a, Mach)\), which is a function of the angle of attack and the velocity.

<table>
<thead>
<tr>
<th>(CM_0)</th>
<th>(a)</th>
<th>(Mach)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.0083</td>
<td>-0.0023</td>
<td>0.0047</td>
</tr>
<tr>
<td>-0.0553</td>
<td>-0.0083</td>
<td>-0.0023</td>
</tr>
<tr>
<td>-0.0303</td>
<td>-0.0503</td>
<td>-0.0073</td>
</tr>
<tr>
<td>-0.0165</td>
<td>-0.0285</td>
<td>-0.0413</td>
</tr>
<tr>
<td>-0.0093</td>
<td>-0.0163</td>
<td>-0.0273</td>
</tr>
<tr>
<td>-0.0038</td>
<td>-0.0156</td>
<td>-0.0248</td>
</tr>
</tbody>
</table>

Note that the first card (called the Header card) gives the derivative name (CM0) and the number of data points (45). Derivative names must start with the letter "C" or they will be rejected. There is no mention of how many arguments are involved. Only when we make use of \(CM_0\) in the Main program that the type of argument and its breakpoints appear explicitly. For this example, the two arguments are arranged in the order \(CM_0(a_1, M_1)\);
CM₀(α₂, M₁); CM₀(α₃, M₁); . . . ; CM₀(α₅, M₁); CM₀(α₁, M₂);
CM₀(α₂, M₂); CM₀(α₃, M₂); . . . . until CM₀(α₅, M₉).

The FGP is made up of three separate sub-programs, each doing a specified task. They are: DATASTORE, POINT and DERIVE4. Their particular roles are described below.

6.1 DATASTORE Program: This program performs the following tasks:

- Read aerodynamic data from the card file as described earlier.
- Check the validity of data including the name.
- Rearrange the data in a predetermined order.
- When an end-of-file (EOF) card is encountered, write the whole data array on a disc file.

The flowchart for DATASTORE can be found on figure 6-1. We can analyze DATASTORE program by tracing through the handling of one stability derivative, say CM₀ again. When MAIN program calls DATASTORE, it reads and inspects the first card, also called the Header card. That card should contain an 8-character derivative name followed by a blank and followed by a 6-digit integers (blanks not included). For example

```
Derivative name
CM0   COMMENTS . . .
Columns #16 thru #80 reserved for comments
Number of data points
```

One header card is required per derivative by DATASTORE. Anytime an EOF card is encountered in place of a header card, control is reverted to the MAIN program. Thus the very last card must be an EOF or the program will hang up.
Figure 6.1: Flowchart for DATASTORE SUBROUTINE
Since all derivative names stored start with the letter "C", a test is made of the first character to eliminate any unnecessary search of the Data List. Only those names previously defined in the Main program's Data Table (such as $CM_0$, $C_{L}$, $C_{n}$, ..., ) are considered legitimate. If the first letter of the header card is a "C", control is passed to the "name list search" portion of DATASTORE program (beginning with statement #50). In the event the first character of the header card is not a "C", DATASTORE allows the user the choice of aborting the run (returning to Main program) or the choice of correcting the typographical mistake by inputting a new name on the CRT.

After the derivative name and address have been found in the tables, DATASTORE returns to the card reader and reads the information on the cards stacked behind the header card. The newly arrived data points are stored in the temporary array called DATA(i), (1 ≤ i, LENGTH). In our example LENGTH = 45 as indicated on the header card earlier. When all the expected data points are in (an EOF card is read) DATASTORE terminates the reading phase and all the aerodynamic data are written out on a disc file. Notice that DATASTORE is executed only once, during program initialization phase and the disc file will be utilized by the Derivative subroutines later on.

It should be aware that DATASTORE is a separate program that must be executed prior to all other programs in the normal sequence of operation. Also note that DATASTORE program must be modified and rerun for any change in the length or table arrangement of the aerodynamic data. These parameters must match those of the MAIN program to insure that the data are stored in the same relative location where they will be used. Specifically care and attention should be paid to the following:

- The order and dimension of the data-table declaration statements
- The DATA statements for variable SIZE and NUMBER
- The ASCII variables in NAMELIST
- The order of the data table ACW (Address Constant Word) array.
  These parameters effect the proper ordering of the data table in the array which is dumped to disc.

6.2 POINT Subroutine: This subroutine, which is called by the Main program, performs all the preliminary tasks prior to final interpolation. One call to POINT subroutine is required per argument list and a representative calling sequence applicable to our present example might appear as follows:

\[
\begin{align*}
\text{CALL POINT} & (\text{ALPHA}, \text{ALPHAPT}(2), \text{ALPHAPT}(1), \text{ALPHAT}(1), \text{ALPHAT}(3)) \\
\text{CALL POINT} & (\text{MACH}, \text{MACHPT}(2), \text{ALPHAPT}(1), \text{ALPHAT}(1), \text{ALPHAT}(3)) \\
\end{align*}
\]

The outgoing arrows are added to visually indicate arguments needed by the subroutine for processing the calls and, conversely, the incoming arrows show that information is being returned to the Main program. Two calls are made to POINT subroutine because in our current example, CM0 is a function of two arguments, \( \alpha \) and Mach. Let us inspect one of the calls above.

\text{ALPHA} = \text{an argument, the present value of } \alpha \\
\text{ALPHAT}(3) = \text{starting point of a list of breakpoints associated with the independent variable } \alpha. \text{ This list consist of floating-point values.} \\
\text{ALPHAPT}(1) = \text{returns to Main program as an integer value (called "pointer") indicating the smallest breakpoint adjacent to the argument } \alpha, \text{ say } \alpha_1. \text{ This information will be used during subsequent interpolation.}
Subroutine POINT

Acquire argument value
Load current POINTER

Update BREAKPOINT corresponding to current POINTER

(\text{BRKPT}) > \text{ARG}

\text{POINTER} = \text{POINTER} - 1

\text{POINTER} < \text{LIM}

\text{POINTER} = \text{LIM}

(\text{BRKPT}) = \text{ARG}

(\text{BRKPT}) < \text{ARG}

\text{POINTER} = \text{POINTER} + 1

\text{POINTER} < \text{MAXBRKPT}

\text{POINTER} = \text{POINTER} - 1

\text{POINTER} = \text{LIM}

(\text{BRKPT} + 1) = \text{ARG}

(\text{BRKPT} + 1) < \text{ARG}

\text{DIVI} = (\text{BRKPT} + 1) - (\text{BRKPT})

\text{FRAC} = \text{ARG} - (\text{BRKPT}) / \text{DIVI}

\text{FRAC} = 0

\text{RETURN} to Main program

Notes: Abbreviations used:
\text{BRKPT} = \text{Breakpoint number}
\text{-LIM} = 1, \text{the first brkpt}
\text{MAXBRKPT} = \text{no. of brkpts of an argument.}
\text{ARG} = \text{value of the argument}
(\text{BRKPT}) = \text{the value of the argument corresponding to BRKPT}

Figure 6-2: Subroutine POINT

61
ALPHAT(1) = returns to Main program representing the distance, \[ \text{FRAC} = \frac{\alpha - \alpha_1}{\alpha_{i+1} - \alpha_i} \]

Finally, ALPHAPT(2) contains the number of breakpoints associated with the argument. In this case there are 9 breakpoints as declared by a DATA statement in the Main program.

When subroutine POINT is entered, a test is made to compare the argument (\( \alpha \) in this instance) against the extrema of the argument values (\( \alpha_1 \) and \( \alpha_9 \) here). Should \( \alpha < \alpha_1 \) then we set the pointer ALPHAPT(1) to 1. Conversely whenever \( \alpha > \alpha_9 \) we set ALPHAPT(1) to 9. A simplified flowchart of the POINT subroutine can be found in figure 6-2.

6.3 Derive Subroutine: This section was written using the FUNCTION approach instead of the standard SUBROUTINE approach in order to reduce program length and minimize the transfer of arguments between different programs. There exist four DERIVE functions: DERIVE1, DERIVE2, DERIVE3 and DERIVE4. Together they can handle any function having from one to four arguments. A sample use of DERIVE1 is as follows:

\[ \text{FUNCT} = \text{DERIVE1(DATATABLE, MACHT)} \]

where DATATABLE is the starting location of the memory table containing the data which were were placed there earlier by the DATASTORE subroutine, MACHT is an argument table containing FRAC in the first word (see figure 6-2 for the significance of FRAC), the breakpoint pointer in the next half-word and the number of Mach breakpoints in the last half-word.

<table>
<thead>
<tr>
<th>MACHT</th>
<th>FRAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINTER</td>
<td>MACHT LENGTH</td>
</tr>
</tbody>
</table>

62
For our existing example, the pitching derivative bias CM0 appears at last, on the right-hand side of an assignment statement

\[
CM0 = \text{DERIVE2} \left( \text{CM0}_T, \text{ALPHAT}, \text{MACHT} \right)
\]
7. REPRESENTATIVE SOLUTIONS

No simulation can be completely trusted unless it has undergone preliminary STATIC and DYNAMIC CHECKS. It is imperative that such checks be carried out before the very first production run and especially if the simulation is just started up after some lay-off time. Check conditions need not be unique; any reasonable set of conditions will suffice. However, some considerations should be given to checking conditions commonly encountered in the flight regime.

7.1 General Procedures

The STATIC CHECK for the rotational equations could be performed as follows: an initial flight condition is established by fixing Mach number, dynamic pressure and angle of attack at pre-selected values. Input variables, angular rates, Euler angles, control surfaces, etc., to each of the equations are then perturbed one at a time or in pairs to excite particular terms of the equations to be checked. The outputs should be scrutinized for trends and for possible wrong signs.

By choice of certain parameters we can isolate each portion of an equation to see if it gives the correct answer. A case in point is illustrated by considering

\[
\dot{P}_B = \frac{\delta S_b}{I_X} \left[ (C_{pR} R_B + C_{bR} B_0 R_B + C_{dR}) \delta + C_{dR} \delta_a + C_{dR} \delta_r \right]
\]

\[
+ \frac{I_{XZ}}{I_X} \left( \dot{R}_B + P_B Q_B \right) + \left( \frac{I_Y - I_Z}{I_X} \right) Q_B R_B
\]

(7-1)

If we set \( P_B = Q_B = 0 \) and \( \delta = \delta_a = \delta_r = I_{XZ} = 0 \) we can observe the change in roll rate as a sole function of the yaw rate, or \( \dot{P}_B = f(R_B) \).

By resetting different parameters, we can gradually examine the influence of each of the terms in the \( P_B \) equation, remembering to take into account the sign of each term.
A similar procedure is used to check the aerodynamic coefficient equations for \(C_N\), \(C_C\), \(C_L\), \(C_m\), \(C_n\) and \(C_Y\) as functions of \(\alpha\), \(\beta\), Mach number and control surface configuration.

DYNAMIC CHECKS should be attempted only after the user is fully satisfied with the static tests. Again a few checks are suggested herein for illustration purposes. They are neither unique nor exhaustive. To test the dynamic behavior of the rotational equations, a stable flight condition is picked and initialized by setting \(\alpha\), \(\beta\), \(\delta_e\)\(_{\text{TRIM}}\), \(\delta_r\)\(_{\text{TRIM}}\), \(\delta_a\)\(_{\text{TRIM}}\), \(V\) and \(h\) (TRIM conditions are selected so that the initial moments are zero, i.e., \(\alpha\) and \(\beta\) remain unchanged). Each surface \((\delta_e, \delta_r, \delta_a)\) is pulsed separately and the resulting aerodynamic responses on \(P_B\), \(P_B\), \(Q_B\), \(Q_B\), \(R_B\), \(R_B\), \(\alpha\), \(\beta\), acceleration terms \(NZ\) & \(NY\), \(\theta\), \(\varphi\) and \(\psi\) are recorded on a strip-chart and compared to known correct responses. The suggested shape of the bidirectional ramp shown below could be used for forcing function (deflection varies from \(\pm 50^\circ\) for subsonic region to \(\pm 20^\circ\) for supersonic region)

![Control Surface Deflection Diagram](image)

It is recommended that this test be conducted on a daily basis to verify the operation worthiness of the equipment. DYNAMIC CHECKS should be carried out on the translational (orbital) equations also. If all aerodynamic effects are nullified by zeroing \(C_C\), \(C_Y\), \(C_N\), \(L_B\), \(M_B\), \(N_B\), \(P_B\), \(Q_B\), \(R_B\) at the same time fixing \(\alpha\), \(\beta\), \(\theta\), \(\psi\) and \(\varphi\), the vehicle can then be considered as a point mass. Additional checks of the orbital equations are also performed by initializing \(C_C\) and \(C_N\) such
that the aerodynamic effects of these equations can be assessed. It should be noted that dynamic checks for handling qualities are normally performed in five degrees-of-freedom. Dynamic checks for performance (point-mass) are performed in three and six degrees-of-freedom.

The control system mechanization is checked by varying the input variables (control stick, rudder pedal, angular rates, angle of attack, Mach number, dynamic pressure, etc) and observing the control surface responses.

The cockpit instruments, switches, control stick and rudder pedals must be checked for proper mechanization (magnitude and sign).

7.2 Static Checks

The values being established for Static Check can be completely arbitrary but they must be reasonable. For instance, we can't choose a test value for $\alpha = -10^\circ$ because the Shuttle will not operate in that range. On the other hand one can pick a test value of $\alpha = 28^\circ$ or $\alpha = 30^\circ$ and one value is as good as the other. In this program the Orbiter can assume either the ENTRY phase or the TAEM phase. Each phase should be tested individually.

7.2.1 Equations-of-Motion Checks: By setting certain flags the FCS can be effectively bypassed to allow us to check only the equations of motion. Specifically, to disengage the FCS, reset the flags concerning aileron and rudder and trim the elevon $\delta_{et} = -16.33^\circ$.

Then establish the following initial conditions

\[
\begin{align*}
\text{Mach} &= 3 \\
\text{h} &= 99480 \\
\text{b} &= 78.06 \\
\text{IX} &= 600,000 \\
\text{IY} = I_Z &= 5.5 \times 10^6 \\
\bar{X} &= 65\% \\
\bar{Z} &= 29.06\% \\
\delta_{SB} &= 25^\circ \\
\delta_{e} &= -16.33^\circ
\end{align*}
\]

With the above IC's, equation (2-19) become,
\[
\dot{P}_B = \left( \frac{I_y - I_z}{I_x} \right) Q_B R_B + \frac{I_{xz}}{I_x} \left( \dot{R}_B + P_B Q_B \right) + \frac{\delta S}{I_x} \left[ \left( C_{\delta_\beta} + C_{\delta_\beta \delta_e} \delta_e \right) \delta + C_{\delta a} \delta_a + C_{\delta r} \delta_r \right] + \frac{\delta S b^2}{2 \sqrt{I_x}} \left( C_{\delta a} P_B + C_{\delta r} R_B \right)
\]

Substituting known quantities into (8-2)

\[
\dot{P}_B = 0.1667 \left( \dot{R}_B + P_B Q_B \right) + 52.495 \left[ \left( -0.00168 \cdot 0.0001 \cdot (-15.33) \right) \delta + 0.000547 \delta_a + 0.0022 \delta_r \right] + 0.68986 \left( -0.261 P_B + 0.073 R_B \right)
\]

Next we manipulate the state variables to obtain a number of conditions. The series of checks appear lengthy at first glance, but they are necessary, because each set of conditions allows us to verify a different part of the set of equations of motion.

a/ For \( P_B = Q_B = 0 \); \( R_B = 0.4 \) rad/sec; \( \beta = 0^\circ \)
\[
I_{xz} = 0 \quad \text{resulting in} \quad \dot{P}_B = 0.0201 \text{ rad/sec}^2
\]

b/ For \( Q_B = R_B = 0 \); \( \beta = \delta_a = \delta_r = 0 \); \( P_B = 1 \)
\[
I_{xz} = 0 \quad \dot{P}_B = -0.1801
\]

c/ For \( P_B = Q_B = R_B = 0 \); \( \beta = -5^\circ \); \( \delta_a = \delta_r = 0 \)
\[
I_{xz} = 0 \quad \dot{P}_B = 0.3990
\]

d/ For \( P_B = Q_B = R_B = 0 \); \( \delta_a = 10^\circ \); \( \beta = \delta_r = 0 \)
\[
I_{xz} = 0 \quad \dot{P}_B = 0.2871
\]

e/ For \( P_B = Q_B = R_B = 0 \); \( \delta_r = 20^\circ \); \( \beta = \delta_a = 0 \)
\[
I_{xz} = 0 \quad \dot{P}_B = 0.2310
\]

f/ For \( P_B = 0 \); \( Q_B = R_B = 0.4 \); \( \beta = \delta_a = \delta_r = 0 \)
\[
I_{xz} = 0 \quad \dot{P}_B = 0.1534
\]
In the next two cases, g) and h), $I_{XZ}$ is not set to zero but to $10^5$ as indicated earlier.

**g/ For** $P_B = Q_B = R_B = 0$; $\delta_a = 10^\circ$; $\delta = \delta_r = 0$

$R_B = .0058$ Resulting in $\dot{P}_B = .2881$

**h/ For** $P_B = 1$; $Q_B = .4$; $R_B = 0$; $\delta_a = 10^\circ$; $\delta = \delta_r = 0$

$R_B = -.3541$ $\dot{P}_B = .1148$

Still under the IC's imposed earlier, calculate the static test concerning the pitch acceleration. According to equation (2-19)

$$\dot{Q}_B = \left(\frac{I_L - I_X}{I_Y}\right)P_B R_B + \frac{I_{XZ}}{I_Y} (R_B^2 - P_B) + \frac{q_{SC}}{I_Y} \left[ C_{m_0} + \Delta C_{m_e} + C_{m_{BF1}} \delta_{BF} + C_{m_{SB1}} (\delta_{SB} - 55^\circ) \right] + \frac{q_{SC}^2}{2VI_Y} (C_{m_Q} Q_B)$$

(7-4)

Plugging in the inertia terms

$$\dot{Q}_B = .8909 P_B R_B + .0182 (R_B^2 - P_B^2) + 2.903 \left[ - .023 + \Delta C_{m_e} - .00051 \delta_{BF} 
+ .00023 (\delta_{SB} - 55^\circ) \right] + .0193 (-2.3 Q_B)$$

(7-5)

As a consequence

- **i/ For** $P_B = Q_B = R_B = 0$; $\delta_e = 0$; $\delta_{BF} = 5^\circ$; $\delta_{SB} = 25^\circ$

  Whereupon the check value for the pitch acceleration is $\dot{Q}_B = -.0838$ rad/sec$^2$

- **j/ For** $P_B = Q_B = R_B = 0$; $\delta_e = -16.33^\circ$; $\delta_{BF} = 16.3^\circ$

  $\delta_{SB} = 25^\circ$ $\dot{Q}_B = -.0831$

- **k/ For** $P_B = Q_B = R_B = 0$; $\delta_e = -16.33^\circ$; $\delta_{BF} = -5^\circ$

  $\delta_{SB} = 87.2^\circ$ $\dot{Q}_B = .072$

68
1/ For \( P_B = R_B = 0; \) \( Q_B = .5; \) \( \delta_e = -16.33^\circ; \) \( \dot{\delta}_{BF} = -5^\circ \)
\( \delta_{SB} = 25^\circ \) \( \dot{Q}_B = .0222 \)

m/ For \( P_B = Q_B = 0; \) \( R_B = .4; \) \( \delta_e = -16.33^\circ; \) \( \dot{\delta}_{BF} = -5^\circ \)
\( \delta_{SB} = 25^\circ \) \( \dot{Q}_B = .0029 \)

n/ For \( P_B = 1.0; \) \( Q_B = 0; \) \( R_B = .4; \) \( \delta_e = -16.33^\circ \)
\( \dot{\delta}_{BF} = -5^\circ; \) \( \delta_{SB} = 25^\circ \) \( \dot{Q}_B = .3411 \)

Next let us turn our attention to the yaw acceleration by making use of equation (2-19) again

\[
\dot{R}_B = \left( \frac{I_Z - I_Y}{I_Z} \right) P_B Q_B + \frac{I_{XZ}}{I_Z} (\dot{P}_B - Q_B R_B) + \frac{\dot{Q}_B}{I_Z} (C_{n_8} + C_{n_{\delta_e}} \dot{\delta}_e) + C_{n_{\delta_a}} \delta_a + C_{n_{\delta_r}} \delta_r
\]

\[
+ (C_{n_{\delta_r}} \delta_r + C_{n_{\dot{\delta}_r}} [\dot{\delta}_r]) + \frac{\dot{Q}_B}{2V_{IZ}} (C_{n_p} P_B + C_{n_R} R_B)
\]

\( \dot{R}_B = -0.8909 P_B Q_B + .0182 (\dot{P}_B - Q_B R_B) + 5.727 \left[ -0.00096 \beta + .00001 \delta_a \right]
\]
\[+ (-.00042 + .000016 |\beta|) \delta_r + .0753 \left[ .01 P_B - .91 R_B \right] \]

Carrying on with the test

o/ For \( P_B = Q_B = 0; \) \( R_B = .4; \) \( \beta = \delta_a = \delta_r = 0 \)
\( I_{XZ} = 0, \) we obtain \( \dot{R}_B = -.0274 \)

p/ For \( P_B = 1.0; \) \( Q_B = R_B = 0; \) \( \beta = \delta_a = \delta_r = 0 \)
\( I_{XZ} = 0 \) \( \dot{R}_B = .0008 \)

q/ For \( P_B = Q_B = R_B = 0; \) \( \beta = -5^\circ; \) \( \delta_a = \delta_r = 0 \)
\( I_{XZ} = 0 \) \( \dot{R}_B = .0275 \)
\[ r/ \quad \text{For } P_B = Q_B = R_B = 0 ; \; \delta_a = 20^\circ ; \; \delta_r = \beta = 0 \]
\[ I_{XZ} = 0 \quad \dot{P}_B = .0011 \]
\[ s/ \quad \text{For } P_B = Q_B = R_B = 0 ; \; \delta_a = 20^\circ ; \; \beta = \delta_r = 0 \]
\[ I_{XZ} = 0 \quad \dot{R}_B = -.0481 \]
\[ t/ \quad \text{For } P_B = 1 ; \; Q_B = .4 ; \; R_B = 0 ; \; \beta = \delta_a = \delta_r = 0 \]
\[ I_{XZ} = 0 \quad \dot{R}_B = -.3556 \]

Now we again impose a value of \( 10^5 \) on \( I_{XZ} \)
\[ u/ \quad \text{For } P_B = Q_B = R_B = 0 ; \; \delta_a = 20^\circ ; \; \beta = \delta_r = 0 \]
\[ \dot{P}_B = .5761 \text{ rad/sec}^2 \quad \dot{R}_B = .0116 \]
\[ v/ \quad \text{For } P_B = 0 ; \; Q_B = R_B = .4 ; \; \delta_a = 20 ; \; \beta = \delta_r = 0 \]
\[ \dot{P}_B = .5913 \quad \dot{R}_B = -.0184 \]

7.2.2 FCS Checks : Set ALTITUDE = 92,800 ft ; VELTRUE = 2960 ft/sec ; TURNCORD = TRUE. (to allow the FCS to maintain the nose level during a turn) and Rudder Pedal Position RPTASOP = 0. Then proceed to parts a) and b) below.

a) Pitch Plane Test (ENTRY Phase) : Fix the Pitch RHC, PRHCSOP = -10° and \( Q_B = 2^\circ /\text{sec} \). Then the elevon deflection, the final object of our Pitch-plane test, should be \( \delta_e = -6.42^\circ \). The calculation of intermediate variables leading to \( \delta_e \) is left to the reader.

b) Roll and Yaw Test (ENTRY Phase) : Establish the Roll Stick Position RRHCSOP = 16.15° and also the following conditions

\[ \begin{align*} 
\text{PRHCSOP} &= 0^\circ \\
\varphi &= 30^\circ \\
\theta &= 20^\circ \\
\dot{P}_B &= 5^\circ /\text{sec} \\
R_B &= 1^\circ /\text{sec} \\
\alpha &= 15^\circ 
\end{align*} \]
We should obtain these final results

\[ \delta_a = -2.66^\circ \]
\[ \delta_T = -1.9^\circ \]

and Yaw-jet Command DRJET = .63 (dimensionless) corresponding to the firing of two reaction jets (UZCMD = 2).

Now we are ready to test the FCS in TAEM phase. Assume ALTITUDE = 48,600 ft and VELTRUE = 1017 ft/sec. This will result in Mach = 1.05 and \( \overline{q} = 200 \text{ lb/ft}^2 \). Proceed to parts c) and d) next.

c) **Pitch-Plane Test (TAEM Phase):** Choose PRHCSOP = -10°; RRHCSOP = 0°; RPTASOP = 0°; \( Q_B = -2^\circ/\text{sec} \); \( R_B = -2^\circ/\text{sec} \) and \( \varphi = 30^\circ \). With the above initial conditions we should command the following responses:

Pitch-jet command, \( DPJET = -0.79 \)
Pitch command signal, \( BCSL = 0.365 \)

d) **Roll & Yaw Test (TAEM Phase):** Select the conditions listed below.

<table>
<thead>
<tr>
<th>PRHCSOP</th>
<th>RRHCSOP</th>
<th>P_B</th>
<th>R_B</th>
<th>NY</th>
<th>( \dot{R}_B )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0°</td>
<td>11.15°</td>
<td>2°/sec</td>
<td>5°/sec</td>
<td>.25 g</td>
<td>0</td>
</tr>
</tbody>
</table>

Look for the following results out of the FCS

\[ \delta_a = .771^\circ \]
\[ \delta_T = 22.09^\circ \]

\[ \text{DRJET} = -10.68 \text{ (4 jets)} \]
7.3 Dynamic Checks

Again, two principal parts of the program are tested semi-independently of each other. We choose to test the simulation in 5 DOF's only, by setting FIVDEG = .TRUE.. In 5-DOF mode the Shuttle flies at a constant altitude at a constant axial velocity. To disturb the equilibrium (or steady-state) conditions, we introduce a forcing-function of short duration and observe the system response. The triangular-waveform forcing-function shown in paragraph 7.1 is generated on the analog computer and patched into an A-D convertor. It has its peaks at $\pm 1$. Once converted into a digital quantity, this waveform is further modulated to give the proper pulse amplitude (for example $\pm 16^\circ$ for $\delta_e$ pulse). During each test only one surface is disturbed. All other surface deflections are returned to initial-condition values or set to zero. Both the equations-of-motion test and the FCS test use the same standard conditions for altitude, true airspeed, weight, surface area, inertial coefficients, Euler angles and trim surface deflections, namely

$$
\begin{align*}
I_X &= 6 \times 10^5 \\
I_Y &= 5.5 \times 10^6 \\
I_Z &= 5.5 \times 10^6 \\
I_{XY} &= 0 \\
I_{XZ} &= 10^5 \\
\bar{\sigma} &= 39.56
\end{align*}
$$

$$
\begin{align*}
\bar{x} &= .65 \\
\bar{y} &= 0 \\
\bar{z} &= .2907 \\
\bar{h} &= 99,480 \\
W_0 &= 1.55 \times 10^5 \\
\Lambda_T &= 107.5
\end{align*}
$$

$$
\begin{align*}
V_T &= 2970 \\
\bar{q} &= 150 \\
\text{Mach} &= 3 \\
S &= 2690 \\
b &= 78.03
\end{align*}
$$

The following angular conditions are also arbitrarily imposed

$$
\begin{align*}
\delta_{SB} &= 25^\circ \\
\varphi(0) &= 0^\circ \\
L(0) &= 0^\circ \\
\delta_{BF} &= -5^\circ \\
\gamma(0) &= 0^\circ \\
\lambda(0) &= 0^\circ \\
\alpha(0) &= 20^\circ \\
\theta(0) &= 20^\circ \\
\delta_a &= 0^\circ \\
\beta(0) &= 0^\circ \\
\delta_e &= -16.33^\circ \\
\delta_T &= 0^\circ
\end{align*}
$$

72
7.3.1 Equations-of-Motion Tests: Three runs are made, all with the following conditions:

\[ \text{FCSDE} = \text{FCSDA} = \text{FCSDR} = \text{.TRUE.} \]

Various output quantities are fed to the DAC's and appear on the strip charts as analog signals. Under the best circumstances the traces can be read at an accuracy of 1% and thus can be used as a qualitative check only. Distinguish the three runs:

- Case a: \( \delta_e \) is pulsed with a triangular wave (see page 65) having \( \pm 16^\circ \) peaks. See figure 7-1.

- Case b: \( \delta_a \) is pulsed with the same waveform resulting in the traces on figure 7-2.

- Case c: \( \delta_r \) is pulsed with a triangular signal having \( \pm 20^\circ \) peaks. Consult figure 7-3 for the airframe responses.

7.3.2 Flight-Control-System Tests: The FCS tests help to verify the end-to-end performance of the simulation. Since we operate at Mach 3 and since rudder pedal inputs are bypassed by the FCS at Mach > 1.5, there is no need to pulse the rudder surface. In the two remaining test cases, a full stick input is applied to the aileron in one case and to the elevon in the other case. The testing is intended not only for dynamic system response, but also in assisting in the discovery of possible FCS discontinuities or errors. Both cases are made with the following conditions:

\[ \text{FCSDE} = \text{FCSDA} = \text{FCSDR} = \text{.FALSE.} \]

As before, the system response is observed through the use of strip-chart recorders. The two cases are perturbated in the following manner:

- Case d: \( \delta_{eRHC} \) is pulsed with a triangular wave having peaks at \( \pm 20^\circ \). See figure 7-4.

- Case e: \( \delta_{aRHC} \) is pulsed by the same signal. See figure 7-5.

It should be noted that in both cases the Flight Control System must be allowed to stabilize the airframe for approximately 3 to 5 seconds before applying the disturbances. In this manner the pitch axis will settle to a trim condition, reducing the inter-axis interaction.
Figure 7-1: Equations-of-Motion Check
(Case a)
Figure 7-2: Equations-of-Motion Check (Case b)
Figure 7-4: FCS Tests (Case d)
8. TERMINOLOGY, SYMBOLS AND DEFINITIONS

8.1 List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>American Standard Code for Information Interchange</td>
</tr>
<tr>
<td>A-D</td>
<td>Analog-to-Digital</td>
</tr>
<tr>
<td>CG</td>
<td>Center of Gravity</td>
</tr>
<tr>
<td>CMD</td>
<td>Command; Commanded</td>
</tr>
<tr>
<td>CRT</td>
<td>Cathode-Ray Tube (Computer terminal)</td>
</tr>
<tr>
<td>C/R</td>
<td>Carriage Return</td>
</tr>
<tr>
<td>DAC</td>
<td>Digital-to-Analog Convertor</td>
</tr>
<tr>
<td>DOF</td>
<td>Degree Of Freedom</td>
</tr>
<tr>
<td>EOF</td>
<td>End-Of-File</td>
</tr>
<tr>
<td>FCS</td>
<td>Flight Control System</td>
</tr>
<tr>
<td>FGP</td>
<td>Function-Generation Package</td>
</tr>
<tr>
<td>FILT</td>
<td>Filter</td>
</tr>
<tr>
<td>Hex</td>
<td>Hexadecimal number</td>
</tr>
<tr>
<td>IC</td>
<td>Initial Condition mode</td>
</tr>
<tr>
<td>I/O</td>
<td>Input / Output</td>
</tr>
<tr>
<td>NOM</td>
<td>Nominal value</td>
</tr>
<tr>
<td>RHC</td>
<td>Rotational Hand Control</td>
</tr>
<tr>
<td>RCS</td>
<td>Reaction-jet Control System</td>
</tr>
<tr>
<td>SEL</td>
<td>Systems Engineering Laboratories</td>
</tr>
</tbody>
</table>

8.2 Lists of Symbols Used in Computer Programs

The following abbreviation is used throughout this chapter: DL = dimensionless quantity. The subscripts on various quantities have the following significance:

- $\delta_a$ = aileron incremental deflection
- $B$ = written in Body coordinate system
- $\delta BF$ = Body Flap incremental deflection
$\delta e$ = elevon incremental deflection
$\delta r$ = rudder ""
$\delta SB$ = Speed Brake incremental deflection

E or e = Earth axes or relating to earth
F = Fuel
G = Gear extended

H = written in H-frame (orbital reference axes)
L, R = two halves (Left & Right) of the elevon
0 = bias value ; IC value ; sea-level value

Be aware that stability coefficients with various final subscripts may have different units. Consider $C_{N_0}$, $C_{N_{\delta e}}$, $C_{N_{\delta BF}}$ (bias value of normal force coefficient, normal force derivatives due to elevon and body flap incremental deflection); they will not appear as three separate entries on the table. Note that $C_{N_0}$ is dimensionless while $C_{N_{\delta e}}$ and $N_{\delta BF}$ have units of sec$^{-1}$. Other subscripts used are $\alpha$, $\beta$, $\rho$, $q$ and $r$.

Sometimes a quantity is represented by two computer variables which are not equivalent. For instance ALPHA stands for $\alpha$ in radians. At the same time ALFA also symbolizes $\alpha$ but in degrees. Similarly LAT and LA both stand for latitude, but LAT is a single-precision quantity while LA is computed in double precision. The computer symbols appear on two separate tables. The second table lists the names used in the FCS program and the first table covers the remaining programs (MAIN plus all subroutines)
### Table 8-1: MAIN PROGRAM SYMBOLS

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Explanation</th>
<th>Computer Representation</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Sonic speed</td>
<td>SOUND</td>
<td>ft/sec</td>
</tr>
<tr>
<td>(b)</td>
<td>Wing span</td>
<td>B</td>
<td>ft</td>
</tr>
<tr>
<td>(C_c)</td>
<td>Total trim chord coefficient</td>
<td>CC</td>
<td>DL</td>
</tr>
<tr>
<td>(c_{gX})</td>
<td>Longitudinal distance of CG measured from a reference point situated 238'' in front of the nose (expressed in % of Orbiter's length)</td>
<td>(not used)</td>
<td>%</td>
</tr>
<tr>
<td>(C_L)</td>
<td>Total rolling-moment derivative</td>
<td>CL</td>
<td>DL</td>
</tr>
<tr>
<td>(\bar{c})</td>
<td>Mean aerodynamic chord</td>
<td>C</td>
<td>ft</td>
</tr>
<tr>
<td>(C_{m\delta_e})</td>
<td>Pitching-moment derivative due to (\delta_e)</td>
<td>CMDE</td>
<td>deg^{-1}</td>
</tr>
<tr>
<td>(C_{m\delta_{SB}})</td>
<td>Pitching-moment derivative due to (\delta_{SB})</td>
<td>CMSB</td>
<td>deg^{-1}</td>
</tr>
<tr>
<td>(C_{n\delta_r})</td>
<td>Yawing-moment coeff. due to rudder</td>
<td>CNDR</td>
<td>deg^{-1}</td>
</tr>
<tr>
<td>(C_N)</td>
<td>Total normal force derivative</td>
<td>CN</td>
<td>DL</td>
</tr>
<tr>
<td>(C_Y)</td>
<td>Side-force coefficient due to (\beta)</td>
<td>CYB</td>
<td>deg^{-1}</td>
</tr>
<tr>
<td>(F_T)</td>
<td>Fuel consumption rate</td>
<td>(not used)</td>
<td>1/sec</td>
</tr>
<tr>
<td>(g)</td>
<td>Gravitational acceleration</td>
<td>GRAVITY or G</td>
<td>ft/sec^2</td>
</tr>
<tr>
<td>(h)</td>
<td>Geometric altitude</td>
<td>ALTITUDE</td>
<td>ft</td>
</tr>
<tr>
<td>(h_p)</td>
<td>Geopotential altitude (used for calculating air density &amp; temperature)</td>
<td>GEOALT</td>
<td>ft</td>
</tr>
<tr>
<td>(I_{XX})</td>
<td>Polar moment of inertia (along the longitudinal axis). Or (I_X)</td>
<td>IXX</td>
<td>slug*ft^2</td>
</tr>
<tr>
<td>(I_{XY})</td>
<td>Products of inertia</td>
<td>IXY</td>
<td></td>
</tr>
<tr>
<td>(I_{XZ})</td>
<td></td>
<td>IXZ</td>
<td></td>
</tr>
<tr>
<td>(I_{YZ})</td>
<td></td>
<td>IYZ</td>
<td></td>
</tr>
<tr>
<td>Symbol</td>
<td>Explanation</td>
<td>Computer Representation</td>
<td>Unit</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>-------------------------</td>
<td>------</td>
</tr>
<tr>
<td>$I_{YY}; I_{ZZ}$</td>
<td>Transverse moments of inertia</td>
<td>$I_{YY}; I_{ZZ}$</td>
<td>slug. ft$^2$</td>
</tr>
<tr>
<td>KCAS</td>
<td>Calibrated airspeed</td>
<td>KCAS</td>
<td>knot</td>
</tr>
<tr>
<td>KEAS</td>
<td>Equivalent airspeed</td>
<td>KEAS</td>
<td>knot</td>
</tr>
<tr>
<td>$J_2$</td>
<td>A constant characteristic of the earth mass distribution</td>
<td>$J_2$</td>
<td>DL</td>
</tr>
<tr>
<td>$J_P$, $J_R$, $J_Y$</td>
<td>Constants characteristic of the thrusts of the pitch, roll and yaw reaction jets.</td>
<td>$U_{YCMD}$, $U_{XCMD}$, $U_{ZCMD}$</td>
<td>DL</td>
</tr>
<tr>
<td>L</td>
<td>Latitude</td>
<td>LAT OR LA</td>
<td>radian</td>
</tr>
<tr>
<td>$L_B$</td>
<td>Total rolling-moment in body axes</td>
<td>$L_B$</td>
<td>lb.ft</td>
</tr>
<tr>
<td>$l_1; l_2; l_3$</td>
<td>Direction cosines</td>
<td>$L1; L2; L3$</td>
<td>DL</td>
</tr>
<tr>
<td>$\ell_T$</td>
<td>Length of the Orbiter (nose to tail)</td>
<td>LENGTH</td>
<td>ft</td>
</tr>
<tr>
<td>m</td>
<td>Orbiter's mass</td>
<td>MASS</td>
<td>slug</td>
</tr>
<tr>
<td>$m_1; m_2; m_3$</td>
<td>Direction cosines</td>
<td>$M1; M2; M3$</td>
<td>DL</td>
</tr>
<tr>
<td>$M_B$</td>
<td>Total pitching moment</td>
<td>MBODY</td>
<td>lb.ft</td>
</tr>
<tr>
<td>Mach</td>
<td>Mach number</td>
<td>MACH</td>
<td>DL</td>
</tr>
<tr>
<td>$n_1; n_2; n_3$</td>
<td>Direction cosines</td>
<td>$N1; N2; N3$</td>
<td>DL</td>
</tr>
<tr>
<td>$N_B$</td>
<td>Total rolling moment</td>
<td>NBODY</td>
<td>lb.ft</td>
</tr>
<tr>
<td>$P_B$</td>
<td>Rolling rate</td>
<td>PBODY</td>
<td>rad/sec</td>
</tr>
<tr>
<td>$q$</td>
<td>Dynamic pressure</td>
<td>QBAR</td>
<td>lb/ft$^2$</td>
</tr>
<tr>
<td>$Q_B$</td>
<td>Pitching rate</td>
<td>QBODY</td>
<td>rad/sec</td>
</tr>
</tbody>
</table>
Table 8-1: Continued

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Explanation</th>
<th>Computer Representation</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_0$</td>
<td>Equatorial radius of earth</td>
<td>$R_0$</td>
<td>ft</td>
</tr>
<tr>
<td>$r$</td>
<td>Distance between Orbiter and earth center</td>
<td>$\text{RADIUS or RAD}$</td>
<td>ft</td>
</tr>
<tr>
<td>$\delta R$</td>
<td>Altitude measured from sea level</td>
<td>$\text{DELR}$</td>
<td>ft</td>
</tr>
<tr>
<td>$R_B$</td>
<td>Yawing rate</td>
<td>$R_{\text{BODY}}$</td>
<td>rad/sec</td>
</tr>
<tr>
<td>$R_D$</td>
<td>Down range</td>
<td>$\text{DOWNRNG}$</td>
<td>ft</td>
</tr>
<tr>
<td>$R_X$</td>
<td>Cross range</td>
<td>$\text{CROSSRNG}$</td>
<td>ft</td>
</tr>
<tr>
<td>$S$</td>
<td>Reference wing surface area Thrust</td>
<td>$S$</td>
<td>ft$^2$</td>
</tr>
<tr>
<td>$T$</td>
<td>(not used)</td>
<td>$\text{(not used)}$</td>
<td>lb</td>
</tr>
<tr>
<td>$T_X$, $T_Y$, $T_Z$</td>
<td>Thrust vector in the 3 body axes</td>
<td>(not used)</td>
<td>lb</td>
</tr>
<tr>
<td>$U_{EA}$, $V_{EA}$</td>
<td>Horizontal components of airspeed in earth axes</td>
<td>$\text{UAIR}(or \text{UA})$</td>
<td>ft/sec</td>
</tr>
<tr>
<td>$U_E$</td>
<td>Orthogonal velocity components expressed in earth axes</td>
<td>{ $\text{UEARTH}$, $\text{VEARTH}$, $\text{WEARTH}$ }</td>
<td>ft/sec</td>
</tr>
<tr>
<td>$V$</td>
<td>Orbiter's total velocity</td>
<td>$\text{VELTRUE or TAS}$</td>
<td>ft/sec</td>
</tr>
<tr>
<td>$W_0$</td>
<td>Orbiter's weight</td>
<td>$\text{WEIGHT}$</td>
<td>lb</td>
</tr>
<tr>
<td>$W_F$</td>
<td>Fuel weight</td>
<td>(not used)</td>
<td>lb</td>
</tr>
<tr>
<td>$W_u$</td>
<td>Wind component in North direction</td>
<td>$\text{NWIND}$</td>
<td>ft/sec</td>
</tr>
<tr>
<td>$W_v$</td>
<td>Wind component in East direction</td>
<td>$\text{EWIND}$</td>
<td>ft/sec</td>
</tr>
</tbody>
</table>
### Table 8-1: Continued

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Explanation</th>
<th>Computer Representation</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \ddot{x} )</td>
<td>Longitudinal location of Orbiter’s CG (expressed in percent of Orbiter’s length)</td>
<td>( \text{XBAR} )</td>
<td>%</td>
</tr>
<tr>
<td>( X_B )</td>
<td>Longitudinal acceleration component</td>
<td>( \text{XBODY, XB or AX} )</td>
<td>( \text{ft/sec}^2 )</td>
</tr>
<tr>
<td>( \dot{y} )</td>
<td>Lateral location of CG</td>
<td>( \text{YBAR} )</td>
<td>%</td>
</tr>
<tr>
<td>( Y_B )</td>
<td>Lateral acceleration component</td>
<td>( \text{YBODY, YB or AY} )</td>
<td>( \text{ft/sec}^2 )</td>
</tr>
<tr>
<td>( \ddot{z} )</td>
<td>Vertical location of the CG</td>
<td>( \text{ZBAR} )</td>
<td>%</td>
</tr>
<tr>
<td>( Z_B )</td>
<td>Normal acceleration component</td>
<td>( \text{ZBODY, ZB or AZ} )</td>
<td>( \text{ft/sec}^2 )</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>Angle of attack</td>
<td>( \text{ALPHA} )</td>
<td>( \text{radian} )</td>
</tr>
<tr>
<td>( \beta )</td>
<td>Angle of side slip</td>
<td>( \text{BETA} )</td>
<td>( \text{deg.} )</td>
</tr>
<tr>
<td>( \gamma )</td>
<td>Flight path angle</td>
<td>( ) (not used)</td>
<td>( \text{rad} )</td>
</tr>
<tr>
<td>( \delta_a )</td>
<td>Aileron total deflection</td>
<td>( \text{DAIL or DA} )</td>
<td>( \text{deg.} )</td>
</tr>
<tr>
<td>( \delta_{BF} )</td>
<td>Body-flap total deflection</td>
<td>( \text{FLAP or DBF} )</td>
<td>( \text{deg.} )</td>
</tr>
<tr>
<td>( \delta_e )</td>
<td>Elevon total deflection</td>
<td>( \text{ELEV or DE} )</td>
<td>( \text{deg.} )</td>
</tr>
<tr>
<td>( \delta_r )</td>
<td>Rudder total deflection</td>
<td>( \text{DRUD or DR} )</td>
<td>( \text{deg.} )</td>
</tr>
<tr>
<td>( \delta_{SB} )</td>
<td>Speed brake total deflection</td>
<td>( \text{BRAK or DSB} )</td>
<td>( \text{deg.} )</td>
</tr>
<tr>
<td>( \Delta t )</td>
<td>Time increment (time frame)</td>
<td>( \text{DELTA} )</td>
<td>( \text{sec} )</td>
</tr>
<tr>
<td>( \theta )</td>
<td>Pitch angle</td>
<td>( \text{THETA} )</td>
<td>( \text{rad.} )</td>
</tr>
<tr>
<td>( \theta_{F} )</td>
<td>Angle between surface of the fuel and ( X_B ) axis</td>
<td>( ) (not used)</td>
<td>( \text{deg.} )</td>
</tr>
</tbody>
</table>
Table 8-1: Continued

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Explanation</th>
<th>Computer Representation</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>λ</td>
<td>Longitude</td>
<td>LNG or LN</td>
<td>rad</td>
</tr>
<tr>
<td>μ</td>
<td>Colatitude ( \left( -\frac{\pi}{2} - L \right) )</td>
<td>MU</td>
<td>rad</td>
</tr>
<tr>
<td>ρ</td>
<td>Atmospheric density</td>
<td>RHO</td>
<td>slug/ft³</td>
</tr>
<tr>
<td>φ</td>
<td>Roll angle</td>
<td>PHI</td>
<td>rad</td>
</tr>
<tr>
<td>Ψ</td>
<td>Yaw angle</td>
<td>PSI</td>
<td>rad</td>
</tr>
<tr>
<td>( \omega_e )</td>
<td>Angular velocity of earth</td>
<td>OMEGA</td>
<td>rad/sec</td>
</tr>
</tbody>
</table>

Some variables appear both in the Main program and the SHT FCS. If they have been previously listed on Table 8-1, they will not be repeated here on Table 8-2.

Table 8-2: FCS Variables

<table>
<thead>
<tr>
<th>Computer Variable</th>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>DACMD</td>
<td>Aileron command</td>
<td>deg</td>
</tr>
<tr>
<td>DATMPAN</td>
<td>Aileron panel trim command</td>
<td>0, ±1</td>
</tr>
<tr>
<td>DATMRHC</td>
<td>Aileron stick trim command</td>
<td>0, ±1</td>
</tr>
<tr>
<td>DATSUMI</td>
<td>Aileron trim integrator command</td>
<td>deg</td>
</tr>
<tr>
<td>DBFMAN</td>
<td>Manual Body Flap command</td>
<td>0, ±1</td>
</tr>
<tr>
<td>DBFRC</td>
<td>Body Flap rate command</td>
<td>deg/sec</td>
</tr>
<tr>
<td>Computer Variable</td>
<td>Description</td>
<td>Unit</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>DCSP</td>
<td>Aileron command due to stick and rate feedback</td>
<td>deg</td>
</tr>
<tr>
<td>DECMRD</td>
<td>Elevator command</td>
<td>deg</td>
</tr>
<tr>
<td>DEL</td>
<td>Left elevon deflection</td>
<td>deg</td>
</tr>
<tr>
<td>DER</td>
<td>Right elevon deflection</td>
<td>deg</td>
</tr>
<tr>
<td>DETRIM</td>
<td>Elevator position trim command</td>
<td>deg</td>
</tr>
<tr>
<td>DETMPAN</td>
<td>Elevator panel trim command</td>
<td>0,±1</td>
</tr>
<tr>
<td>DETMRHC</td>
<td>Elevator stick trim command</td>
<td>0,±1</td>
</tr>
<tr>
<td>DPJET</td>
<td>Pitch Jet Command</td>
<td>DL</td>
</tr>
<tr>
<td>DQCT</td>
<td>Elevator Command due to stick and rate feedback</td>
<td>deg</td>
</tr>
<tr>
<td>DRCPF</td>
<td>Rudder command due to pedal</td>
<td>deg</td>
</tr>
<tr>
<td>DRCMD</td>
<td>Rudder command</td>
<td>deg</td>
</tr>
<tr>
<td>DRJET</td>
<td>Yaw Jet Command</td>
<td>DL</td>
</tr>
<tr>
<td>DRLP</td>
<td>Left-half rudder deflection</td>
<td>deg</td>
</tr>
<tr>
<td>DRRP</td>
<td>Right-half rudder deflection</td>
<td>deg</td>
</tr>
<tr>
<td>DRTRIM</td>
<td>Yaw Trim Integrator output</td>
<td>deg</td>
</tr>
<tr>
<td>DSBPC</td>
<td>Speedbrake command</td>
<td>deg</td>
</tr>
<tr>
<td>ELFBK</td>
<td>Elevator Feedback signal (Mach≤12)</td>
<td>deg</td>
</tr>
<tr>
<td>ETRIM</td>
<td>Elevator Position Trim (Mach&gt;12)</td>
<td>deg</td>
</tr>
<tr>
<td>NY</td>
<td>Lateral Acceleration</td>
<td>g's</td>
</tr>
<tr>
<td>NZ</td>
<td>Normal Acceleration</td>
<td>g's</td>
</tr>
</tbody>
</table>
Table 8-2: Continued

<table>
<thead>
<tr>
<th>Computer Variable</th>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>PE</td>
<td>Roll &amp; Yaw stability feedback</td>
<td>DL</td>
</tr>
<tr>
<td>PRHCSOP</td>
<td>Pitch Stick Command</td>
<td>deg</td>
</tr>
<tr>
<td>QDOT</td>
<td>Pitch Acceleration</td>
<td>rad/sec^2</td>
</tr>
<tr>
<td>RDOT</td>
<td>Yaw Acceleration</td>
<td>rad/sec^2</td>
</tr>
<tr>
<td>RPTASOP</td>
<td>Rudder Pedal Command</td>
<td>deg</td>
</tr>
<tr>
<td>RRHCSOP</td>
<td>Roll Stick Command</td>
<td>deg</td>
</tr>
<tr>
<td>SBHP</td>
<td>Speedbrake Handle Command</td>
<td>deg</td>
</tr>
<tr>
<td>SUM23</td>
<td>Roll &amp; Yaw Command due to stick</td>
<td>DL</td>
</tr>
</tbody>
</table>
9. REFERENCES

9.1 General Bibliography


9.2 Shuttle-Related References


APPENDIX A - MAIN PROGRAM LISTING

SAFID

SIMULATION BENCHMARK

**************

SHUTTLE

EQUATIONS OF MOTION

**************

PROGRAM SHILOFT

* SPACE SHUTTLE ORBITAL FLIGHT TEST SIMULATION.

IMPLICIT REAL *4 (A - Z)
REAL*8 DATA88L
LOGICAL*1 DATAY1
COMMON /ARAY1/DATAN8RM(400)
COMMON /ARAY2/DATAN8BL(17)
COMMON /ARAY3/DATABY8(17)
COMMON /ARAY4/ANGLE(3)
BIT IN818(16)
LOGICAL*1 IC,MU8D,PUTSET,A8ORT,IVEDG,FIVDEG
LOGICAL*1 FCSDA,FCSD8,FCSD88,FCSD8F,FCSS8,M8D
LOGICAL*1 READY,DUMPFLAG,AUTO8F
LOGICAL*1 TUNNCMD
LOGICAL*1 DAE8LSE,DAE8LSE,DWP8LSE
LOGICAL*4 EAS8R
LOGICAL*4 DISCHTIN
INTEGER*2 TEM81,TEM82,PREVTEM8
INTEGER*4 SAVFL
INTEGER*4 LINK,II,JJ
INTEGER*4 HANDLEM
REAL*8 DP112,DP113,COSLAT,COSMU,COSPSIM,LA,LN,MU,RAD,
* KCOSLAT,SHAKE,SINLAT,SINMU,SINPSIM,UA,UB,UE,VA,VB,VE,MB,ME,
* REAL*4 UDTAIE(5243)
* INTEGER*4 AEROSIZE

* THE FOLLOWING TYPING STATEMENTS DIMENSION THE DATA TABLE ARRAYS.
  INTEGER*2 DLNSIZE, RLNSIZE, LINSIZE, MAFSIZE, BYTSIZE, SIZE, PLACE
  INTEGER*2 ELEPT(2), DELPT(2), DENPT(2), ALPHAPT(2), MACHPT(2),
  INTEGER*8 LIST(434)

* THE FOLLOWING ARRAYS FORM THE I/U TABLES.
* DATAOUT AND DATAIN FORM THE 44 CHANNEL DATA OUTPUT AND 32 CHANNEL
  DATA INPUT TABLES, RESPECTIVELY.
* SCALOUT AND SCALIN ARE THE CORRESPONDING OUTPUT AND INPUT SCALE
  FACTORS, RESPECTIVELY.
  INTEGER*4 COUNT
  INTEGER*2 DATOUT1(44), DATAIN(32)
  REAL*4 SCALOUT(44), SCALIN(32)

  REAL*4 CCU1(63)
  REAL*4 UCET(378)
  REAL*4 CCDF1T(63)
  REAL*4 CLDBF2I(63)
  REAL*4 CDSU1T(63)
  REAL*4 CDSBH2T(63)
  REAL*4 CLBTABL(252)
  REAL*4 CLHDE1T(63)
  REAL*4 CLHUE2T(63)
  REAL*4 CLDSBH1T(63)
  REAL*4 CLDSBH2T(63)
  REAL*4 CUMT(252)
  REAL*4 CNBTABL(252)
  REAL*4 CNBDE1I(63)
  REAL*4 CNBDE2T(63)
  REAL*4 CNBSBH1T(63)
  REAL*4 CNBSBH2T(63)
  REAL*4 CNBTAT(378)
  REAL*4 CNDAT(252)
  REAL*4 CNMFT(63)
  REAL*4 CNP(63)
  REAL*4 CNNRT(63)
  REAL*4 CNMOT(63)
  REAL*4 CNDEF1T(63)
  REAL*4 CNDBF1T(63)
REAL*4 CNDBF2T(63)
REAL*4 CNDSB11T(63)
REAL*4 CNDSH2T(63)
REAL*4 CYBT(63)
REAL*4 CYDAT(378)
REAL*4 CYDNT(63)
REAL*4 CYDRSH1T(7)
REAL*4 CYDRSH2T(7)

REAL*4 SNDTADL(31), SNDARG(31), ZETABL(41), ZETARG(41)
REAL*4 WINDARG(20), NWTABL(20), ENTABL(20)
REAL*4 ELEV1(8), DELT(2), DELT(2), ALPHAT(11), MACHT(9), BETAT(6), DSBT(6)

* EQUIVALENCE (INBIT, DISCKTN)
* EQUIVALENCE (SNDARG(1), ZETARG(1))
* EQUIVALENCE (ELEV1(2), ELEV1T(1)), (DELT(2), DELPT(1)), (DELT(2),
  * DSBT(1)), (ALPHAT(2), ALPHAPT(1)), (MACHT(2), MACHPT(1)), (BETAT(2)
  * BETAPT(1)), (DSBT(2), DSBT(1))

* EQUIVALENCE
  * (OUTDATA(1), CCUT(1))
  * (OUTDATA(64), DCCEO(1))
  * (OUTDATA(442), CCDBET(11))
  * (OUTDATA(505), CCDBF2T(11))
  * (OUTDATA(568), CCDBSH1T(11))
  * (OUTDATA(631), CCDBSH2T(11))
  * (OUTDATA(694), CLBTABL(1))
  * (OUTDATA(946), CLBD11T(1))
  * (OUTDATA(1009), CLBDE1T(1))
  * (OUTDATA(1072), CLBDSH1T(1))
  * (OUTDATA(1135), CLBDSH2T(11))
  * (OUTDATA(1198), CLBDSH3T(11))
  * (OUTDATA(1261), CLDAT(1))
  * (OUTDATA(1394), CLDNT(1))
  * (OUTDATA(1891), CLPT(1))
  * (OUTDATA(1954), CLMT(1))
  * (OUTDATA(2017), CMUF(1))
  * (OUTDATA(2080), DCMET(1))
  * (OUTDATA(2458), CMDF1T(1))
  * (OUTDATA(2521), CMDF2T(1))
  * (OUTDATA(2584), CMDFSH1T(1))
  * (OUTDATA(2647), CMDFSH2T(1))
  * (OUTDATA(2710), CMUF(1))
  * (OUTDATA(2962), CNBTA1(1))
  * (OUTDATA(3214), CNBDE1T(1))
  * (OUTDATA(3277), CNBDE2T(1))
  * (OUTDATA(3340), CNBDSH1T(1))
  * (OUTDATA(3403), CNBDSH2T(1))
  * (OUTDATA(3466), CNBDSH3T(1))
  * (OUTDATA(3529), CNDAT(1))
  * (OUTDATA(3907), CNDNT(1))
  * (OUTDATA(4159), CNDMHT(1))
  * (OUTDATA(4222), CNNPT(1))
  * (OUTDATA(4285), CNRNT(1))
  * (OUTDATA(4348), CNUT(1))
  * (OUTDATA(4411), CNDRET(1))
  * (OUTDATA(4474), CNOMF1T(1))
EQUIVALENCE
* (DATAOBL(1), DELR)
* (DATAOBL(2), DELROOT)
* (DATAOBL(3), GRAVITY)
* (DATAOBL(4), LATDOT)
* (DATAOBL(5), LNGDOT)

THERE IS A HOLE AT DATAOBL(6)
* (DATAOBL(7), PS1HDOT)
* (DATAOBL(8), NUM)
* (DATAOBL(9), NUMDOT)
* (DATAOBL(10), HM)
* (DATAOBL(11), WH)
* (DATAOBL(12), XEARTH)
* (DATAOBL(13), XH)
* (DATAOBL(14), YEARTH)
* (DATAOBL(15), YH)
* (DATAOBL(16), ZEARTH)
* (DATAOBL(17), ZH)

* (DATAHUN(1), CALPHAN)
* (DATAHUN(2), AX)
* (DATAHUN(3), AY)
* (DATAHUN(4), A2)
* (DATAHUN(5), CALPHAN)
* (DATAHUN(6), ALTITUDE)
* (DATAHUN(7), H)
* (DATAHUN(8), CHETA0)
* (DATAHUN(9), HETA)
* (DATAHUN(10), BMAK)
* (DATAHUN(11), C)
* (DATAHUN(12), CC)
* (DATAHUN(13), CCLBF)
* (DATAHUN(14), CCBSB)
* (DATAHUN(15), CD)
* (DATAHUN(16), CL)
* (DATAHUN(17), CLB)
* (DATAHUN(18), CM)
* (DATAHUN(19), CMBSB)
* (DATAHUN(20), CMBSB)
* (DATAHUN(21), CN)
* (DATAHUN(22), CNBF)
* (DATAHUN(23), CNSB)
* (DATAHUN(24), CNH)
* (DATAHUN(25), CY)
* (DATAHUN(26), CMBST)
* (DATAHUN(27), CI)
* (DATAHUN(28), CNBSE)
* (DATAHUN(29), RUNTIME)
* (DATAHUN(30), TIME)
* (DATAHUN(31), CNBSB)
* (DATAHUN(32), CYMBSB)
*(DATAWRK(33), CMUSSRN)*
*(DATAWRK(34), DAIL)*
*(DATAWRK(35), DEL)*
*(DATAWRK(36), ALTU)*
*(DATAWRK(37), PHHCSP)*
*(DATAWRK(38), DEM)*
*(DATAWRK(39), UMIC)*
*(DATAWRK(40), DHUO)*
*(DATAWRK(41), DBFIC)*
*(DATAWRK(42), DELIC)*
*(DATAWRK(43), DACM)*
*(DATAWRK(44), UELRIC)*
*(DATAWRK(45), DELTA)*
*(DATAWRK(46), DERMIC)*
*(DATAWRK(47), DUWNRNG)*
*(DATAWRK(48), USBIC)*
*(DATAWRK(49), LEV)*
*(DATAWRK(50), EMIND)*
*(DATAWRK(51), FLAP)*
*(DATAWRK(52), GO)*
*(DATAWRK(53), GEAR)*
*(DATAWRK(54), DPJET)*
*(DATAWRK(55), DSEET)*
*(DATAWRK(56), MUUT)*
*(DATAWRK(57), IXY)*
*(DATAWRK(58), GEAULIC)*
*(DATAWRK(59), IXX)*
*(DATAWRK(60), IXY)*
*(DATAWRK(61), IZI)*
*(DATAWRK(62), IYMZ)*
*(DATAWRK(63), JUY)*
*(DATAWRK(64), IYY)*
*(DATAWRK(65), IZMX)*
*(DATAWRK(66), IZUZ)*
*(DATAWRK(67), IZZ)*
*(DATAWRK(68), IFUNC)*
*(DATAWRK(69), J2)*
*(DATAWRK(70), KLAS)*
*(DATAWRK(71), KEAS)*
*(DATAWRK(72), HHCSP)*
*(DATAWRK(73), KNO1CON)*
*(DATAWRK(74), HPTASOP)*
*(DATAWRK(75), CAT)*
*(DATAWRK(76), CLA1)*
*(DATAWRK(77), CLNG)*
*(DATAWRK(78), CLNGU)*
*(DATAWRK(79), UZCMO)*
*(DATAWRK(80), UXCMO)*
*(DATAWRK(81), L3BODY)*
*(DATAWRK(82), LENGTH)*
*(DATAWRK(83), UYCMO)*
*(DATAWRK(84), MACM)*
*(DATAWRK(85), MASS)*
*(DATAWRK(86), CM)*
*(DATAWRK(87), MBODY)*
*(DATAWRK(88), NBODY)*
*(DATAWRK(89), NWIND)*
*(DATAWRK(90), UBLATE)*
*(DATAWRK(91), OMEGA)*
* (DATAAWKDRH(92), PDLIC) 
* (DATAAWKDRH(93), PDUT) 
* (DATAAWKDRH(94), CPHI) 
* (DATAAWKDRH(95), CPH10) 
* (DATAAWKDRH(96), CPS1) 
* (DATAAWKDRH(97), CPS10) 
* (DATAAWKDRH(98), CPSIM) 
* (DATAAWKDRH(99), CPSIM) 
* (DATAAWKDRH(100), PBOUY) 
* (DATAAWKDRH(101), PBOUT) 
* (DATAAWKDRH(102), PE) 
* (DATAAWKDRH(103), UBAH) 
* (DATAAWKDRH(104), UBIC) 
* (DATAAWKDRH(105), UDUT) 
* (DATAAWKDRH(106), UHOUY) 
* (DATAAWKDRH(107), KO) 
* (DATAAWKDRH(108), KIC) 
* (DATAAWKDRH(109), KOOI) 
* (DATAAWKDRH(110), KHO) 
* (DATAAWKDRH(111), KHUO) 
* (DATAAWKDRH(112), SHMP) 
* (DATAAWKDRH(113), RADIUS) 
* (DATAAWKDRH(114), KBOUY) 

* THERE IS A HOLE AT DATAAWKDRH(115) 
* (DATAAWKDRH(116), HUMIC) 
* (DATAAWKDRH(117), S) 
* (DATAAWKDRH(118), CPSIM) 
* (DATAAWKDRH(119), SOUND) 
* (DATAAWKDRH(120), CTHEIAU) 
* (DATAAWKDRH(121), CTHETIA) 
* (DATAAWKDRH(122), THEDATAUT) 
* (DATAAWKDRH(123), UAIAK) 
* (DATAAWKDRH(124), UYEGES) 
* (DATAAWKDRH(125), UBODY) 
* (DATAAWKDRH(126), UEAIR) 
* (DATAAWKDRH(127), UVAIR) 
* (DATAAWKDRH(128), VBODY) 
* (DATAAWKDRH(129), VEAIR) 
* (DATAAWKDRH(130), VELMIC) 
* (DATAAWKDRH(131), VELTRUE) 
* (DATAAWKDRH(132), PITCHPAN) 
* (DATAAWKDRH(133), NHIC) 
* (DATAAWKDRH(134), PBODY) 
* (DATAAWKDRH(135), KEARTH) 
* (DATAAWKDRH(136), REIGHT) 
* (DATAAWKDRH(137), MHOUT) 
* (DATAAWKDRH(138), XAHM) 
* (DATAAWKDRH(139), XBKK) 
* (DATAAWKDRH(140), PITCHSIK) 
* (DATAAWKDRH(141), XBANKUM) 
* (DATAAWKDRH(142), XBODY) 
* (DATAAWKDRH(143), MOLLAPAN) 
* (DATAAWKDRH(144), YBAR) 
* (DATAAWKDRH(145), YBRR) 
* (DATAAWKDRH(146), MOLLSTIK) 
* (DATAAWKDRH(147), YBODY) 
* (DATAAWKDRH(148), ZHAK) 
* (DATAAWKDRH(149), ZBRK) 
* (DATAAWKDRH(150), ZETA) 

95
*, (DATAMHDK (151), YAMTH1M)
*, (DATAMHDR (152), ZBAMHDN)
*, (DATAMHDK (153), ZBODY)
*, (DATAMHDR (154), FLAPCMU)
*, (DATAMHDK (155), CCU)
*, (DATAMHDR (156), CCE)
*, (DATAMHDK (157), CCDBF1)
*, (DATAMHDR (158), CCDBF2)
*, (DATAMHDR (159), CCDSH1)
*, (DATAMHDK (160), CCDSH2)
*, (DATAMHDR (161), CLB)
*, (DATAMHDK (162), CLBOE1)
*, (DATAMHDR (163), CLBOE2)
*, (DATAMHDK (164), CLQDSB1)
*, (DATAMHDR (165), CLQDSB2)
*, (DATAMHDK (166), CLQDSB3)
*, (DATAMHDR (167), CLQA)
*, (DATAMHDK (168), CLQK)
*, (DATAMHDR (169), CLP)
*, (DATAMHDK (170), CLR)
*, (DATAMHDR (171), CMQ)
*, (DATAMHDR (172), CME)
*, (DATAMHDK (173), CMDBF1)
*, (DATAMHDR (174), CMDBF2)
*, (DATAMHDK (175), CMDSH1)
*, (DATAMHDR (176), CMDSH2)
*, (DATAMHDK (177), CMU)
*, (DATAMHDR (178), CNB)
*, (DATAMHDK (179), CNQOE1)
*, (DATAMHDR (180), CNQOE2)
*, (DATAMHDK (181), CNQSH1)
*, (DATAMHDR (182), CNQSH2)
*, (DATAMHDK (183), CNQSH3)
*, (DATAMHDR (184), CNDA)
*, (DATAMHDK (185), CNR)
*, (DATAMHDR (186), CNDRB)
*, (DATAMHDR (187), CNP)
*, (DATAMHDK (188), CNR)
*, (DATAMHDR (189), CNQ)
*, (DATAMHDK (190), CNQE)
*, (DATAMHDR (191), CNQBF1)
*, (DATAMHDK (192), CNQBF2)
*, (DATAMHDR (193), CNQSH1)
*, (DATAMHDK (194), CNQSH2)
*, (DATAMHDR (195), CYB)
*, (DATAMHDK (196), CYDA)
*, (DATAMHDR (197), CYUK)
*, (DATAMHDK (198), CYQDSB1)
*, (DATAMHDR (199), CYQDSB2)
* *, (DATABY 1(1), FCSUA)
* *, (DATABY 1(2), FCSUD)
* *, (DATABY 1(3), FCSUF)
* *, (DATABY 1(4), FCSOK)
* *, (DATABY 1(5), FCSOB)
* *, (DATABY 1(6), FIVEDG)
* *, (DATABY 1(7), FIVEDF)
* *, (DATABY 1(8), WIND)
* *, (DATABY 1(9), PUTFSET)
*/ (DATAYT(10),HULU)
*/ (DATAYT(11),IC)
*/ (DATAYT(12),AHUNT)
*/ (DATAYT(13),UKNCUKU)
*/ (DATAYT(14),AUTOF)
*/ (DATAYT(15),DEPULSE)
*/ (DATAYT(16),APULSE)
*/ (DATAYT(17),UKPULSE)
*/ (ANGLE(1),ALPHA)
*/ (ANGLE(2),PHI)
*/ (ANGLE(3),THETA)

DATA P1/3,141592654/
DATA AEROSIZE/5243/

DATA SCALUT/8727,3,491,8727,8727,1.745,4363,128,7,0.4,35,
* 64,35,7.45,25,0,2048,0,32768,0,16,0,3,142,3,142,3,142,
* 102400,0,10240,0,1024,0,512,0,10,24,0,0,0,0,3,142,3,142,
* 35,0,35,0,35,0,96,0,25,0,4,96,9,0,0/
DATA SCALE/25,0.20,0,30,0,90,0,1,0,1,0,1,0,1,0,1,0,0,0,
* 1,0,2,180,0/

* DATA ALPHAI/0.0,0.0,0.0,0.0,0.0,9893,0.1745,0.2618,0.3491,0.4365,
* 0.5236,0.6181,0,6727/
DATA MACHT/0.0,0.0,1.572,0.3,0,5,0.8,0.10,0/
DATA ELEV1/0.0,0.0,35.0,20.0,10.0,0,10.0,20.0/
DATA USHT0/0.0,0.0,0,25.0,35.0,87.22/
DATA BETAT/0.0,0.0,0,0.2,0.5,0.10,0/
DATA ALPHAT/1,4/
DATA MACHT/1,7/
DATA ELEV1/1,6/
DATA UDEPT/1,6/
DATA UDDEPT/1,6/
DATA DSDEPT/1,4/
DATA BETAT/1,4/
DATA ZETARG/1,10000,20000,30000,40000,50000,60000,70000,
* 1 80000,90000,100000,110000,120000,130000,140000,150000,
* 1 60000,170000,180000,190000,200000,210000,220000,230000,
* 240000,250000,260000,270000,280000,290000,300000,310000,
* 320000,330000,340000,350000,360000,370000,380000,390000,
* 400000,

DATA SMATAF/1110,1077,1037,995,968,968,968,971,978,985,
* 1 991,1004,1022,1040,1058,1075,1082,1082,1071,1058,1046,
* 1 1020,994,967,939,911,884,884,884,884,904,904,
DATA ZTAML/242330,50349,314973,327719,350427,376466,
* 543582,407519,417175,424352,429408,433938,437363,439442,
* 439268,435496,436120,433989,429458,426540,424377,421935,
* 414077,410275,420996,422748,425734,431314,436433,441318,
* 446949,451434,454810,457721,459620,460466,461186,461258,
* 469134,458847,457300/

DATA WINDAng/0,5000,10000,15000,20000,25000,30000,35000,
* 1 40000,45000,50000,55000,60000,65000,70000,75000,80000,
* 1 85000,90000,95000/

DATA DBLSIZE,HSIZE,ISIZE,MAFSIZE,BYISIZE,SIZE/17,400,0,0,17,434/

LIST CONTAINS THE ASCII DOUBLE NUMS WHICH DEFINE VARIABLE NAMES TO
* EASE AND DATASTORE.

DATA LIST/

REAL DOUBLEWORD VARIABLES ARE DEFINED HERE.

'DELR'
'DELRDOT'
'GAVITY'
'LATDOT'
'NGDOT'

THIS MOLVE CORRESPONDS TO DATADBL(6)

'PSIMDOT'
'KUMDOT'
'UM'
'MH'
'Xearth'
'Xm'
'Yearth'
'Ym'
'Zearth'
'Zm'

THE REAL MUNDO VARIABLES ARE DEFINED HERE.

'ALPHA0'
'AX'
'AY'
'AZ'
'ALPHA'
'ALTITUDE'
'B'
'BETA0'
'BETA'
'BRAK'
'C'
'CCHF'
'CCSB'
'CD'
'CL'
'CLBT'
'CM'
'CMHF'
'CMSB'
'CN'
'CNHF'
'CNSB'
'CNBT'
'CY'
'CLHDE'
'CLHSB'
'CNHDE'
'HUNTIME'
'TIME'
'CHNSB'
'CYDURS8'
'CHOSHRNG'
'DAIL'
'DEL'

98
THIS HOLE CORRESPONDS TO DATAHIDR(115)

'PHIO'
'PSI'
'PSIO'
'PSIM'
'PSIR'
'PBODY'
'PHIDOT'
'PE'
'OMAN'
'UBIC'
'OQUOT'
'OBODY'
'RO'
'RHIC'
'RUOT'
'RHU'
'RHUO'
'SBMP'
'RADIUS'
'RBODY'

* THIS HOLE CORRESPONDS TO DATAHIDR(115)
* 'ZBUDY'
* 'FLAPCMD'

* THIS Point, REFERENCED BY THE DATSTORE SUBROUTINE, IS LIST(172)
* 'CCO'
* 'CCE'
* 'CCDBF1'
* 'CCDHF2'
* 'CCDSB1'
* 'CCDSB2'
* 'CLB'
* 'CLBDE1'
* 'CLBDE2'
* 'CLBDSB1'
* 'CLBDSB2'
* 'CLBDSB3'
* 'CLDA'
* 'CLDR'
* 'CLP'
* 'CMH'
* 'CLM'
* 'CLMDE1'
* 'CLMDE21'
* 'CLRL3DSB1'
* 'CLRL3DSB2'
* 'CLL' 
* 'CMO'
* 'CMOE'
* 'CMOBF1'
* 'CMOBF2'
* 'CMOSB1'
* 'CMDSB2'
* 'CMW'
* 'CNH'
* 'CNHDE1'
* 'CNHDE2'
* 'CNHDBS1'
* 'CNHDSB2'
* 'CNHDSB3'
* 'CNDA'
* 'CNUR'
* 'CNDBH'
* 'CLAP'
* 'CNK'
* 'CNU'
* 'CNDE'
* 'CNDBF1'
* 'CNDBF2'
* 'CNDBF3'
* 'CNDBS1'
* 'CNDBS2'
* 'CYB'
* 'CYDA'
* 'CYDK'
* 'CYDMDBS1'
* 'CYDMDSB2'
* 'DEMAN'
* 'ESHAPF'
* 'GPS'
* 'DEMS'
* 'Q'
* 'PHOO2b'
* 'JET'
* 'ETRIMIN'
* THE LOGICAL BYTE VARIABLES ARE DEFINED HERE.
  * 'FCSDA'
  * 'FCSDB'
  * 'FCSDC'
  * 'FCSDD'
  * 'FCSE1'
  * 'FCSEG'
  * 'FCSEF'
  * 'FCSDF'
  * 'FCSDE'
  * 'FCSDF'
  * 'FCSDG'
  * 'FCSDB'
  * 'FCSDE'
  * 'FCSDF'
  * 'FCSDD'
  * 'FCSDE'

*EXECUTIVE PORTION FOLLOWS
*THIS IS THE ENTRY POINT FOR PROGRAM ACTIVATION
**
*INITIALIZATION
* THIS SECTION INITIALIZES THE INTERVAL TIMER HANDLER.
*
  IN LINE
)
  1020 DI 43 DISABLE INTERVAL TIMER
  LM 7,428 FETCH ADDRESS OF OLD HANDLER
  STW 7,MHANDLER STORE FOR LATER USE
  LM 7,1000 FETCH ADDRESS OF NEW HANDLER
  STW 7,426 STORE ADDRESS IN HANDLER LOC.
ENDI
*
* THE AERODYNAMIC DATA IS PLACED IN CORE HERE
*
  II=1
  JJ=1152
  3 CALL BUFFERIN(5,1,OUTDATA(II),JJ)
  4 CALL STATUS(5,TEMP1)
  IF(TEMP1,NE,2)GOTO4
  II=II+1152
  IF(II.GT,AEROSIZE)GO TO 5
  IF(II+1152.GT,AEROSIZE)JJ=AEROSIZE+I-11
  GO TO 3
  5 CONTINUE
*
  IC=.TRUE.
  POTSET=.FALSE.
  HOLD=.FALSE.
*
* THE FOLLOWING VARIABLES ARE SET TO INDICATE A DESIRED CONDITION
* AS INDICATED. THESE VARIABLES ARE EASE ACCESSIBLE.
*
  AUTO HUFDY FLAP (NOT AVAILABLE)
  AUTOB=.FALSE.
  TURN COORDINATION
  TURNCORD=.TRUE.
ORDERLY PROGRAM TERMINATION

ABORT=.FALSE.

FIVE DEGREES OF FREEDOM

FIVED=.TRUE.

THREE DEGREES OF FREEDOM

THREE=.FALSE.

ATMOSPHERIC WIND

WIND=.FALSE.

CONTROL OF VEHICLES SURFACE DEFORMATIONS....TRUE. INDICATES THAT THE
VALUES ARE DETERMINED BY DELIC,DERIC,DRIC,DSBIC,DBFIC.

FCSU=FCSUR=FCSUR=.FALSE.

FCSUF=FCSUSH=.TRUE.

SURFACE PULSE FOR DYNAMIC RESPONSE CHECK....TRUE. INDICATES THAT
THE SURFACE IS TO BE PULSED.

DEPULSE=DEPULSE=DBPULSE=.FALSE.

PRESET PROGRAM RUN TIME IN SECONDS.

RUNTIME=5.175

CALPHA0=20.0
CBETA0=0.0
CTETA0=20.0
CPHIO=0.0
CPS1U=0.0
CLAT0=0.0
CLNGU=0.0
CPSIK=26.0
S=2649.0
B=78.03
C=39,56

LENGTH=107.5
WEIGHT=155000.0
IXX=600000.0
IYY=5580000.0
IZZ=5500000.0
IXY=0.0
IXZ=100000.0
XYB0=0.65
YB0=0.0
ZB0=0.2907
XBARNUM=0.65
ZBARNUM=0.2907
G0=32.146546
J2=0.0010823
KNOTCON=0.59210526
URLATE=70150.0
OMEGA0=0.00072921159
R0=20925738.0
RMU0=0.023769
DBFIC=5.0
DELIC=16.33
DERIC=16.33
DRIC=0.0
DSBIC=26.0
ALT0=99480.0
PBIC=0.0
QBIC=0.0
KBIC=0.0
VELIC=2470.0
DELTA=.04
HALFPI = PI/2
TWOP1 = 2*PI
*
* THIS IS THE EASE RETURN POINT
  10 IF (ABORT) GO TO 1100
*
* A DIGITAL INPUT IS PERFORMED HERE TO INITIALIZE THE MODE CONTROL
* PARAMETERS.
  15 CONTINUE
  INL1NE
  LW  4,8200003FFF
  LEA  1,920
  SIMW  1,256
  CD  0,428000
  EN1
*
  MASS=WEIGHT/32,174
  M*EX=UNPLATE
  HALFDUEL=DELFAT/2
*
  P15S1,P15S2,P15S3 ARE TEMPORARY LOCATIONS
  P151=MASS*LENGTH
  XBAR=X/MASS
  YBAR=Y/MASS
  ZBAR=Z/MASS
  XMY=1XX=1YY
  YMX=1YY-1ZZ
  ZMY=1ZZ-1XX
  IXY=1XY/1YY
  IYZ=1IZ/1ZZ
  IFUNC=1/(1XX+1ZZ*1XX+1YY*1XY)
  SC=S*C
  SC2BY2=.S*S*C**2
  SB=S*B
  SB2BY2=.S*B**2
*
* THE FOLLOWING SECTION CALCULATES THE INITIAL CONDITION VALUES
* FOR THE INTEGRAL EQUATIONS. EASE DEFINES THETA,PHI,PSI,ALPHA,
* BETA,VEL TRUE AND ALTITUDE. THESE VALUES ARE PROCESSED TO OBTAIN
* RHN,PSIN,PHI,A N D DELN.
*
* ALL ANGLES ARE INTERNALLY DEFINED IN RADIAN S. ALL COMMUNICATIONS
* WITH CARDS AND EASE ARE WITH ANGLES DEFINED IN DEGREES, A "C"
* PRECEDE S ALL DEGREE-DEFINED ANGLES.
  THE1A=CTHE1A/57.29577951
  PH10=CPH10/57.29577951
  PS10=CP510/57.29577951
  ALPH20=CALPH20/57.29577951
  BET20=CBET20/57.29577951
  PS1R=CP51R/57.29577951
  L10=CL10/57.29577951
  LNGO=CLNGO/57.29577951
  P111=SIN(THE1A0)
  P1T2=CUS(THE1A0)
  P1T3=SIN(PH10)
  P1T4=CUS(PH10)
  P1T5=SIN(PS10)
  P1T6=CUS(PS10)
  L1=P1T2*P1T6
  L2=P1T2*P1T5
  L3=P1T1

106
MIa.PjT4aPl lbsP113*P ITLaPIT 6
M2aPIUIaPII6 tPIT3 *PLTL*PII5
M3*P IT3aP1 2
NIsPI T3aPIT S+P jI4aPlT 1*PI lb
N2: ~PIT3 aP1Tb+PITUa P1T l*PI I5
N3*PLT2*PI T4
DE.LkIC:ALIU.RQ*(.00167628=.00167616*CUS(2*LA1O) -.00000211*
CUS(4*LA1O))
RAD = R0+DELHIC
UBODY*VELIC*CUS(ALPHA0)*CUS(dETA0)
VBOVFV*VEIFIC*SIN(BETAO)
HBOVFV*VEIFIC*SIN(ALPHA0)*COS(BETAO)
UA1M=UBODY*L1+VBOVFV*M1+HBOVFV*N1
VA1M=UBODY*L2+VBOVFV*M2+HBOVFV*N2
WEARHM=UBODY*L3+VBOVFV*M3+HBOVFV*N3
UEATHM=VA1M
VEMTHM=VA1M+HAD*CUS(LATU)*OMEGA
PSIMO=ATAN2(VEMTHM,UEATHM)
NEARHM=HAD*UEATHM/COS(PSIMO)
WMIC=WEATHM
CPSIMO=PSIMO*57.29577951

* THIS SECTION STARTS THE INTERVAL TIMER OPERATION.

* COUNT=DELTAP/.0000012
INLINE
Lw 0,COUNT LOAD FRAME TIME
CD 127,60 START TIMER
EI 43 ENABLE INTERVAL TIMER
ENDI

* PATCHES WILL GO HERE
* END OF INITIALIATION ROUTINE
* THIS IS THE LOOP RETURN ENTRY POINT
47 PUSET=INBIT(1)
IC=INBIT(2)
HOLD=INBIT(3)

* IF(PUSET) GO 10 1105
IF(HOLD) GO 10 170
IF(IC) TIME=0

* ANALOG INPUT IS STARTED HERE.
INLINE
Lw 4,820003FFFF
LEA 1,,910
STW 1,256
CD 0,428000
ENDI

* THIS SECTION TRANSFERS THE INPUT VALUES CONTAINED IN THE DATA
* HALFWORD TABLES TO THE APPROPRIATE REAL WORD VARIABLES.
JJ=1
INLINE
BL 1960
ENDI
PRHCSUP=DATA
JJ=2
INLINE

107
* IF( I4 ) GO TO 49
   GO TO 170
49  PHI0=PS1C; PHI05=BODY=OBIC; BODY=RBIC
THETA = THE1AU; PHI = PHI0; PSI = PS10
SH=HIC; DELM=DELMHC
PS1H=PSI0; I45=I45HC
LA=L45; LN=L450
170 CONTINUE
*
THIS STARTS THE DATA LOOK-UP.

FIND ARGUMENT BREAKPOINTS

\[ \text{ABS}\beta 1 = \text{ABS}(\beta) \]

CALL \text{POINTF} (\text{MACH}, \text{MACHPT}(2), \text{MACHPT}(1), \text{MACH}(1), \text{MACH}(3))

CALL \text{POINTF} (\text{ALPHA}, \text{ALPHAP}(2), \text{ALPHAP}(1), \text{ALPHA}(1), \text{ALPHA}(3))

CALL \text{POINTF} (\text{ELEV}, \text{ELEVPT}(2), \text{ELEVPT}(1), \text{ELEV}(1), \text{ELEV}(3))

CALL \text{POINTF} (\text{DEL}, \text{DELP}(2), \text{DELP}(1), \text{DELT}(1), \text{DELT}(3))

CALL \text{POINTF} (\text{EMK}, \text{EMKPT}(2), \text{EMKPT}(1), \text{EMK}(1), \text{EMK}(3))

CALL \text{POINTF} (\text{ABS}\beta 1, \text{ABS}\beta PT(2), \text{ABS}\beta PT(1), \text{ABS}\beta T(1), \text{ABS}\beta T(3))

FIND DERIVATIVE VALUES

\[ \text{CMO} = \text{DERIVE2} (\text{CMOT}, \text{ALPHAP}(2), \text{MACHPT}(2), \text{ALPHAP}(1), \text{MACHPT}(1), \text{ALPHA}(3)) \]

\[ \text{CMO} = \text{DERIVE2} (\text{CMOT}, \text{ALPHAP}(2)) \]

\[ \text{CCO} = \text{DERIVE2} (\text{CCOT}, \text{ALPHAP}(2)) \]

\[ \text{CLP} = \text{DERIVE2} (\text{CLPT}, \text{ALPHAP}(2)) \]

\[ \text{CLM} = \text{DERIVE2} (\text{CLMT}, \text{ALPHAP}(2)) \]

\[ \text{CNDB} = \text{DERIVE2} (\text{CNDBT}, \text{ALPHAP}(2)) \]

\[ \text{CNP} = \text{DERIVE2} (\text{CNPT}, \text{ALPHAP}(2)) \]

\[ \text{CNK} = \text{DERIVE2} (\text{CNKT}, \text{ALPHAP}(2)) \]

\[ \text{CYB} = \text{DERIVE2} (\text{CYSB}, \text{ALPHAP}(2)) \]

\[ \text{CYD} = \text{DERIVE2} (\text{CYSD}, \text{ALPHAP}(2)) \]

IF (FLAP, GT, 0.0) GOTO 220

\[ \text{CLB} = \text{CCDB} = \text{DERIVE2} (\text{CCDBF1}, \text{ALPHAP}(2)) \]

\[ \text{CMB} = \text{CCDBF2} = \text{DERIVE2} (\text{CCDBF2}, \text{ALPHAP}(2)) \]

\[ \text{CNF} = \text{CCDBF1} = \text{DERIVE2} (\text{CNDBF1}, \text{ALPHAP}(2)) \]

\[ \text{GOTO} 230 \]

220 CONTINUE

\[ \text{CCB} = \text{CCDBF2} = \text{DERIVE2} (\text{CCDBF2T}, \text{ALPHAP}(2)) \]

\[ \text{CMB} = \text{CCDBF2} = \text{DERIVE2} (\text{CCDBF2T}, \text{ALPHAP}(2)) \]

230 CONTINUE

IF (BAK, GT, 25.0) GOTO 240

\[ \text{CLB} = \text{CLBD} = \text{DERIVE2} (\text{CLBDST}, \text{ALPHAP}(2)) \]

\[ \text{CNB} = \text{CNDBS} = \text{DERIVE2} (\text{CNDBS1}, \text{ALPHAP}(2)) \]

\[ \text{GOTO} 245 \]

240 CONTINUE

\[ \text{CLB} = \text{CLBD} = \text{DERIVE2} (\text{CLBDST}, \text{ALPHAP}(2)) \]

\[ \text{CNB} = \text{CNDBS2} = \text{DERIVE2} (\text{CNDBS2}, \text{ALPHAP}(2)) \]

245 CONTINUE

\[ \text{CLB} = \text{CLBD} = \text{DERIVE2} (\text{CLBDST}, \text{ALPHAP}(2)) \]

IF (BAK, LT, 0.0) GOTO 249

\[ \text{CLB} = \text{CLBD} = \text{DERIVE2} (\text{CLBDST}, \text{ALPHAP}(2)) \]

\[ \text{CNB} = \text{CNDBS3} = \text{DERIVE2} (\text{CNDBS3}, \text{ALPHAP}(2)) \]

249 CONTINUE

IF (ELEV, GT, 0.0) GOTO 250

\[ \text{CLB} = \text{CNP} = \text{DERIVE2} (\text{CNPST}, \text{ALPHAP}(2)) \]

\[ \text{CNB} = \text{CNBD} = \text{DERIVE2} (\text{CNBS}, \text{ALPHAP}(2)) \]

\[ \text{GOTO} 255 \]

250 CONTINUE

\[ \text{CLB} = \text{CNP} = \text{DERIVE2} (\text{CNPST}, \text{ALPHAP}(2)) \]

\[ \text{CNB} = \text{CNBD} = \text{DERIVE2} (\text{CNBS}, \text{ALPHAP}(2)) \]

255 CONTINUE

IF (BAK, GT, 55.0) GOTO 258

\[ \text{CNS} = \text{CCNS} = \text{DERIVE2} (\text{CCNSST}, \text{ALPHAP}(2)) \]

\[ \text{CNM} = \text{CMNS} = \text{DERIVE2} (\text{CMNSST}, \text{ALPHAP}(2)) \]

\[ \text{CNB} = \text{CNDS} = \text{DERIVE2} (\text{CNDSST}, \text{ALPHAP}(2)) \]

\[ \text{CNB} = \text{CNDS} = \text{DERIVE2} (\text{CNDSST}, \text{ALPHAP}(2)) \]

\[ \text{CNB} = \text{CNDS} = \text{DERIVE2} (\text{CNDSST}, \text{ALPHAP}(2)) \]

\[ \text{GOTO} 259 \]
CONTINUE
CCSH=CCDSB2*DERIVE2(CCDSB2T,ALPHAPT(2))
CMSh=CMDSB2*DERIVE2(CMDSB2T,ALPHAPT(2))
CNSH=CNDSB2*DERIVE2(CNDSB2T,ALPHAPT(2))
CYNSh=CYDNSB2*DERIVE1(CYDNSB2T,MACHPT(2),MACHPT(1),MACHT(1))

CONTINUE
CMEL=DERIVE3(CMELT,ALPHAPT(2),MACHPT(2),DELPT(2),ALPHAPT(1),
* MACHPT(1),DELP(1),ALPHAT(1),MACH(1),DELT(1))
CCEL=DERIVE3(CCEL,ALPHAPT(2))
CMEH=DERIVE3(CMEHT,ALPHAPT(2),MACHPT(2),DERPT(2),ALPHAPT(1),
* MACHPT(1),DEHP(1),ALPHAT(1),MACH(1),DEHT(1))
CCNH=DERIVE3(CNHHT,ALPHAPT(2))
CLDA=DERIVE3(CLDAT,ALPHAPT(2),MACHPT(2),ELEVPT(2),ALPHAPT(1),
* MACHPT(1),ELEVPT(1),ALPHAT(1),MACH(1),ELEV(1))
CNDA=DERIVE3(CNDAT,ALPHAPT(2))
CYPD=DERIVE3(CYDAT,ALPHAPT(2))
CLB=DERIVE3(CLB T,ALPHAPT(2),MACHPT(2),BEIAPT(2),ALPHAPT(1),
* MACHPT(1),BEIAPT(1),ALPHAT(1),MACH(1),BEI(1))
CNB=DERIVE3(CNB T,ALPHAPT(2))
CMU=DERIVE3(CMUT,ALPHAPT(2),MACHPT(2),DSHPT(2),ALPHAPT(1),
* MACHPT(1),DSHPT(1),ALPHAT(1),MACH(1),DSH(1))
CLDU=DERIVE3(CLDT,ALPHAPT(2))
CNOU=DERIVE3(CNOU T,ALPHAPT(2))
CME=(CMEL+CMEL)/2.0
CCE=(CCEL+CCNH)/2.0

C THIS CONCLUDES THE DATA LOOK-UP.

*CALCULATION OF DERIVATIVE COEFFICIENTS
*CL AND LBODY
*CALCULATION OF LBOD
LBOD=CLB+CLBD*ELEV+CLBDSB*(BHAK=25)+CLBDSB3*(BHAK=60)

CALCULATION OF LBOD
LBOD=LBOD*WBAR*( (CLB*SHAPE+CLA*DAIL+CLDU*DRU) )*Sb

CALCULATION OF LBOD
LBOD=LBOD*WBAR*( (CLB+CLBD*ELEV+CLBDSB*(BHAK=25)+CLBDSB3*(BHAK=60))

CALCULATION OF LBOD
LBOD=LBOD*WBAR*( (CLB*SHAPE+CLA*DAIL+CLDU*DRU)*Sb+

CALCULATION OF CM
CM=(CMG+CMH+CMHBF)*FLAP+CMSB*(BHAK=55)

CALCULATION OF CM
CM=(CMG+CMH+CMHBF)*FLAP+CMSB*(BHAK=55)

CALCULATION OF NBDY
NBDY=NBODY*WBAR*( (CNH*BETA+CNH*DAIL+CNH*ABSB*UO)*SH+

CALCULATION OF NBDY
NBDY=NBODY*WBAR*( (CNH*BETA+CNH*DAIL+CNH*ABSB*UO)*SH+

CALCULATION OF CC
CC=(CCG+CCE+CCBF)*FLAP+CCSB*(BHAK=55)

CALCULATION OF CC
CC=(CCG+CCE+CCBF)*FLAP+CCSB*(BHAK=55)

CALCULATION OF CY
CY=(CYG*BETA+CYD*DAIL+CYD*CYDNSB*(BHAK=55))*DRU

CALCULATION OF CY
CY=(CYG*BETA+CYD*DAIL+CYD*CYDNSB*(BHAK=55))*DRU

CALCULATION OF CN (NORMAL FORCE COEFFICIENT)
CN=(CNH+CNHDE+CNHBF)*FLAP+CNHSB*(BHAK=55)

CALCULATION OF CN (NORMAL FORCE COEFFICIENT)
CN=(CNH+CNHDE+CNHBF)*FLAP+CNHSB*(BHAK=55)

CALCULATION OF CL (LIFT COEFFICIENT)
CL = CN*CSAFL + CN*SINAFL

CALCULATION OF CU
CU = CC*CSAFL + CN*SINAFL

*
*THE FOLLOWING IS A FOURTH ORDER RUNGE KUTTA INTEGRATION ROUTINE*
*FOR FORMATION OF PBODY, WBODY, AND RBODY*
*THIS SECTION USES FUNCTION TYPE SUBPROGRAMS FOR CALCULATION OF*
*DERIVATIVE TERMS*
*DELTAT = FRAME TIME IN SECONDS*

* POUT = POUTT(PBODY, WBODY, RBODY)  
  K0 = DELTA * POUT  
  QOUT = QOUTT(PBODY, WBODY, RBODY, POUT)  
  L0 = DELTA * QOUT  
  ROUT = ROUTT(PBODY, WBODY, RBODY, POUT)  
  M0 = DELTA * ROUT

* PIT1 = POUTT(PBODY + 5 * K0, WBODY + 5 * L0, RBODY + 5 * M0)  
  K1 = DELTA * PIT1  
  L1 = DELTA * QOUTT(PBODY + 5 * K0, WBODY + 5 * L0, RBODY + 5 * M0, PIT1)  
  M1 = DELTA * ROUTT(PBODY + 5 * K0, WBODY + 5 * L0, RBODY + 5 * M0, PIT1)

* PIT1 = POUTT(PBODY + 5 * K1, WBODY + 5 * L1, RBODY + 5 * M1)  
  K2 = DELTA * PIT1  
  L2 = DELTA * QOUTT(PBODY + 5 * K1, WBODY + 5 * L1, RBODY + 5 * M1, PIT1)  
  M2 = DELTA * ROUTT(PBODY + 5 * K1, WBODY + 5 * L1, RBODY + 5 * M1, PIT1)

* PIT1 = POUTT(PBODY + 5 * K2, WBODY + 5 * L2, RBODY + 5 * M2)  
  K3 = DELTA * PIT1  
  L3 = DELTA * QOUTT(PBODY + 5 * K2, WBODY + 5 * L2, RBODY + 5 * M2, PIT1)  
  M3 = DELTA * ROUTT(PBODY + 5 * K2, WBODY + 5 * L2, RBODY + 5 * M2, PIT1)

*STOP INTEGRATION IF IC ON HOLD MODES*
IF (IC = UR, HOLD) GO TO 210

* PBODY = PBODY + K0 / 6 + K1 / 2 + K2 / 3 + K3 / 6  
  WBODY = WBODY + L0 / 6 + L1 / 3 + L2 / 3 + L3 / 6  
  RBODY = RBODY + M0 / 6 + M1 / 3 + M2 / 3 + M3 / 6

210 CONTINUE

*THIS SECTION USES THE FOURTH ORDER RUNGE KUTTA TECHNIQUE TO GENERATE*
*THE EULER ANGLES THETA, PHI, AND PSI*
*THE RESULTANT VALUES FOR THETA, PHI, AND PSI ARE IN RADIANS*
*THE SIN AND COS VALUES FOR THE EULER ANGLES ARE ALSO CALCULATED*
  SINTHETA = SIN(THETA)  
  COSTHETA = COS(THETA)  
  SINPHI = SIN(PHI)  
  COSPHI = COS(PHI)  
  SINSI = SIN(PSI)  
  COSPSI = COS(PSI)

* THETADOT = WBODY * COSPHI * RBODY * SINPHI  
  KO0 = DELTA * THETADOT  
  PSIOUT = (WBODY * COSPHI + WBODY * SINPHI) / COSTHETA  
  L00 = DELTA * PSIOUT  
  PHIOUT = PBODY + PSIOUT * SINTHETA  
  M00 = DELTA * PHIOUT

* K01 = DELTA * THETADOT(PHI + 5 * K00)  
  PIT1 = PSIOUTT(PHI + 5 * K00, THETA + 5 * K00)  
  L01 = DELTA * PIT1  
  M01 = DELTA * PHIDOT1(PHI + 5 * K00, THETA + 5 * K00, PIT1)
* K02=DELTA*THETADOTT(PHI+.5*M01)
P1T1=PS1DOTT(PHI+.5*M01,THETA+.5*K01)
L02=DELTA*P1T1
M02=DELTA*PHIDOTT(PHI+.5*M01,THETA+.5*K01,P1T1)

* K03=DELTA*THETADOTT(PHI+M02)
P1T1=PS1DOTT(PHI+M02,THETA+K02)
L03=DELTA*P1T1
M03=DELTA*PHIDOTT(PHI+M02,THETA+K02,P1T1)

* INTEGRATION IS NOT PERFORMED IN THE IC OR MOLD MODES
  IF(IC,UR,MOLD) GO TO 270

* THREE DEGREES OF FREEDOM MODE REQUIRES PMIDOT AND PSI DOT = 0
  IF(INDEG) L00=L01=L02=L03=M00=M01=M02=M03=0

* 270 CONTINUE

* CALCULATION OF GRAVITY, INCLUDING EARTH'S OBLATENESS
  MU=HALFPI—LA
  COSMU=DCOS(MU)
  SINMU=DSIN(MU)

* THIS TERM, SHARE, IS USED BOTH HERE AND IN THE CALCULATION OF XEARTH
  *J2,GC0N1, AND GC0N2 ARE CONSTANTS
  SHARE=-3*J2*G0*(R0/RAD)**4
  GRAVITY=G0*(R0/RAD)**2+.5*SHARE*(3*COSMU**2-1)-UM**2/RAD

* GRAVITY HAS UNITS FT/SEC**2

* BODY AXIS ACCELERATION CALCULATIONS
  DPT1=UBAH/MAS

* TEST FOR DESIRED FIVE DEGREES OF FREEDOM
  IF (FIVDEG) GO TO 290
  XB=DPT1*CC*S
  GO TO 320

290 DPT2=(XB*ZH+VB*YB)/UB
DPT3=SHINTETA*CuSTMETA*(XB*COSPHI+VB*SINPHI)/UB
XD=GRAVITY*DPT3*DPT2

320 AX=XB
XBODY=XB
YB=DPT1*CY*S
AT=YT
YBODY=YB
ZB=DPT1*CN*S
AZ=ZB
ZBODY=ZB

* EARTH AND H-FRAME ACCELERATIONS
  DPT1=ZB*COSPHI+YB*SINPHI
  DPT2=ZB*SINPHI+YB*COSPHI
  DPT3=XB*SHINTETA+DPT1*SHINTETA
  XEARTH=DPT2+SINPSI+DPT3*COSPSI+SHARE*COSMU*SINMU
  YEARTH=DPT1*COSPSI+DPT3*SINPSI
  ZEARTH=XB*SHINTETA+DPT1*COSTMETA*GRAVITY
  XH = XEARTH*COSPSIM + YEARTH*SINPSIM
  YH = XEARTH*SINPSIM + YEARTH*COSPSIM
  ZH = ZEARTH

112
* THE FOLLOWING SECTION CALCULATES THE M-FRAME DERIVATIVE VALUES.
  RHODOT=RAD*XM
  PSIMDOT=VE*SINLAT/RHOCOSLAT+YH/UM
  RHOD=ZH
  RHODT=RD
  LATDOT=UE/RAD
  LNGDOT=VE/RHOCOSLAT-OMEGA
* OMEGA IS THE EARTH'S ANGULAR VELOCITY
* FOR FIVE DEGREES OF FREEDOM DELRDOT MUST BE ZERO
  DELRDOT=0
  IF(FIVEDEG) DELRDOT=0
*
* THE M-FRAME DERIVATIVE VALUES ARE INTEGRATED BY A FIRST-ORDER
* EULER INTEGRATION ROUTINE.
* INTEGRATION IS BYPASSED IN IC AND HOLD MODES.
*
  IF(IC,UR,HOLD) GO TO 370
  R1M=R1M+RHODOT*DELTA
  PSIM=PSIM+PSIMDOT*DELTA
  R1M=R1M+RHODT*DELTA
  LAT=LAT+LATDOT*DELTA
  LNG=LNG+LNGDOT*DELTA
  DELR=DELR+DELRDOT*DELTA
*
  370 LAT=LAT
  LNG=LNG
  RAD=RO*DEL
  RADIUS=RAD
  UM=RHU/RAD
*
* THE FOLLOWING SECTION CALCULATES TRANSLATION OF THE BODY PROJECTED
* ONTO THE EARTH'S SURFACE, IN FEET.
* LAT0 AND LNG0 ARE REFERENCE LATITUDE AND LONGITUDE OF THE STARTING
* POINT OF FLIGHT.
* PS1R IS THE ANGLE IN RADIANS BETWEEN TRUE NORTH AND TOP OF MAP.
* POSITIVE PS1R IS WITH MAP ROTATED EAST, (CLOCKWISE) FROM NORTH.
* NOTE THAT THE COSLAT TERM ACCOUNTS FOR CHANGE IN DISTANCE BETWEEN
* LONGITUDE LINES AS A FUNCTION OF LATITUDE
  SINLAT=OSIN(LA)
  COSLAT=OCOS(LA)
  RHO=OSIN(LA)+COSLAT
  SINPSIM=OSIN(PS1R)
  COSPSIM=OCOS(PS1R)
  DOWN=UM*XH*((LAT=LAT0)*COSP3IR+(LNG-LNG0)*SINPSIM*COSLAT)
  CROSS=PUX=((LAT=LAT0)*SINPSIM+(LNG=LNG0)*COSPSIM*COSLAT
  ALTITUDE=DELH+RU* (.00167828+.0016726*Cos(2*LAT)>.0000211*
  * C0S(4*LAT))
  COSPSIM=OCOS(PS1R)
  SINPSIM=OSIN(PS1R)
  UE=UM*COSPSIM
  UE=UM*COSPSIM
  WE=UM*PS1R
  V=UM*PS1R
  V=UM*PS1R
  WE=UM
  WE=UM
*
* WINDS
* NORTH WINDS(NWINU) AND EAST WIND (EWINU) ARE TABLE LOOKUPS.

113
* AS A FUNCTION OF ALTITUDE AND ARE ADDED TO CORRESPONDING EARTH
* AXIS VELOCITIES TO GET AIR SPEED
  IF (.NOT. WIN) GO TO 410
  WINALT = ALTITUDE
  IF (WINALT.GT. 95000) WINALT = 95000
  CALL POINTF(WINALT,20,PLACE,FRACTION,WINARG)
  NWIND = NTABL(PLACE) + (NTABL(PLACE+1) - NTABL(PLACE)) * FRACTION
  NWIND = NTABL(PLACE) + (NTABL(PLACE+1) - NTABL(PLACE)) * FRACTION
  GO TO 420
  410 NWIND = EWIND = 0
* EARTH AXIS AND AIR VELOCITIES CONSIDER H-FRAME VELOCITIES,
* WINDS, AND ROTATING EARTH
  420 UA=UE-NWIND
       UA=UE-WIND
       UA=VE-EWIND-RCOSLAT*OMEGA
       VAIR=VA
       H001 = -NH
* H001 REFERENCES POSITIVE UPWARDS
* CALCULATION OF BODY-AXIS VELOCITIES, ALPHA, BETA
* SIGN CONVENTIONS FOR VELOCITIES ARE, U IS POSITIVE FOR FORWARD
* V IS POSITIVE RIGHT, AND W IS POSITIVE DOWNWARD
  DPI1 = VA*CUSPSI-UA*SINPSI
  DPI2 = UA*CUSPSI+VA*SINPSI
  DPI3 = W*CUSSTHETA+UPI2*SINTHETA
       UBODY=UB
       VB=UPI1*CUSPHI+UPI2*SINPHI
       VBODY=VB
       WB=UPI3*CUSPHI+UPI1*SINPHI
       WBODY=WB
* ALPHA IS ANGLE OF ATTACK IN RADIANS
  ALPHA=ATAN2(WBODY,UBODY)
  SINALF=SIN(ALPHA)
  COSALF=COS(ALPHA)
  PI1 = SQRT(UBODY**2 + WBODY**2)
* BETA IS ANGLE OF SIDESLIP IN DEGREES
  BETA=ATAN2(VBODY,PI1)
  BETA=29577951
* VELTRUE IS TRUE AIRSPEED IN FT/SEC
* TEST FOR FIVE DEGREE OF FREEDOM
  IF (FIVEG) GO TO 430
  VELTRUE=SQRT(VBODY**2+PI1**2)
  GO TO 440
  430 VELTRUE=VELIC
  CONTINUE
* RHO AND QBAR CALCULATIONS REQUIRE TABLE LOOKUP OF ZETA
  GEOALT=RE*ALTITUDE/(RE+ALTITUDE)
  RH0ALT=GE0ALT
  CALL POINTF(RH0ALT,41,PLACE,FRACTION,ZETAARG)
  ZETA=ZETABL(PLACE)+(ZETABL(PLACE+1)-ZETABL(PLACE)) * FRACTION
  RHO=RH00*EXP(-ZETA*GE0ALT/10000)
  GBRN = .5*RHO*VELTRUE**2
* EQUIVALENT AND CALIBRATED AIRSPEEDS ARE CALCULATED IN KNOTS
* KEAS IS EQUIVALENT AIRSPEED, KCAS IS CALIBRATED AIRSPEED
  KEAS = VELTRUE*SQRT(RHO/RH00)*KNOTCUN
  THE CALCULATION OF MACH REQUIRES A TABLE LOOKUP OF SPEED OF
* SOUND BASED ON ALTITUDE
CALL POINTF(;/OAI,31,P/LACE,FRACTION,SN/DA/HG)
SOUND=SN/DA/HG(P/LACE)+(SN/DA/HG(P/LACE+1)-SN/DA/HG(P/LACE))*FRACTION
MACH = VE/LO/VE/SOUND

*REAL FUNCTION SPEED IS USE/DED TO DERIVE KCAS FROM KEAS, AND
*IS PRESENTED IN THE FUNCTION DEFINITION SECTION
PIT1 = SPEED(MACH)
PIT2 = SPEED(.001511*KCAS)
KCAS = PIT1*KEAS/PIT2

* ANALOG OUTPUT AND DIGITAL INPUT ARE STARTED HERE.

INLINE
LW 4800003FFFF
LEA 1,915
STM 1,25b
CO 0,428000
ENDI

* IF DYNAMIC CHECKS ARE DESIRED THE APPROPRIATE SURFACES ARE DEFLECTED
* AND FCS PROCESSING IS BYPASSED.

IF(.NOT.,D/EPULSE) GO TO 443
DEL=DELIC+PULSE*16.0
DER=DERIC+PULSE*16.0
GO TO 448
443 IF(.NOT.,DA/PULSE) GO TO 444
DEL=DELIC+PULSE*16.0
DER=DERIC+PULSE*16.0
GO TO 448
444 IF(.NOT.,OR/PULSE) GO TO 445
DRUD=DMIC+PULSE*20.0
DER=DERIC
DEL=DELIC
GO TO 449
445 CONTINUE

*EXECUTE SHUTTLE FLIGHT CONTROL SYSTEM

CALL SHTLFCS

IF(.NOT.,(FCSDE,0R,FCSDA)) GO TO 448
DEL=DELIC
OEM=VEMIC
448 IF(FCSDR) DRUD=DMIC
449 IF(FCSDF) FLAP=VEMIC
IF(FCSSB) BMAK=VSBIC
ELEV=(DEL+DER)/2
DAIL=(DEL+DER)/2

* THIS SECTION BUILDS THE DATA HALFWORD TABLES TO BE OUTPUT THRU
* THE D-A CONVERTER.
JJ=1
DATA=D0D1
INLINE
BL950
ENDI
JJ=2
DATA=PDOT
INLINE BL 1950
ENDI
JJ=3
DATA=HDOT
INLINE BL 1950
ENDI
JJ=4
DATA=UBODY
INLINE BL 1950
ENDI
JJ=5
DATA=PBODY
INLINE BL 1950
ENDI
JJ=6
DATA=-RBODY
INLINE BL 1950
ENDI
JJ=7
DATA=AZ
INLINE BL 1950
ENDI
JJ=8
DATA=AY
INLINE BL 1950
ENDI
JJ=9
DATA=AX
INLINE BL 1950
ENDI
JJ=10
DATA=ALPHA
INLINE BL 1950
ENDI
JJ=11
DATA=BETA
INLINE BL 1950
ENDI
JJ=12
DATA=UBAK
INLINE BL 1950
ENDI
JJ=13
DATA=VELTRUE
INLINE BL 1950
ENDI
<table>
<thead>
<tr>
<th>JJ=14</th>
<th>DATA=MACH</th>
<th>INLINE</th>
<th>BL</th>
<th>1950</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JJ=15</td>
<td>DATA=PHI</td>
<td>INLINE</td>
<td>BL</td>
</tr>
<tr>
<td></td>
<td>JJ=16</td>
<td>DATA=PSI</td>
<td>INLINE</td>
<td>BL</td>
</tr>
<tr>
<td></td>
<td>JJ=17</td>
<td>DATA=THETA</td>
<td>INLINE</td>
<td>BL</td>
</tr>
<tr>
<td></td>
<td>JJ=18</td>
<td>DATA=ALTITUDE</td>
<td>INLINE</td>
<td>BL</td>
</tr>
<tr>
<td></td>
<td>JJ=19</td>
<td>DATA=KEAS</td>
<td>INLINE</td>
<td>BL</td>
</tr>
<tr>
<td></td>
<td>JJ=20</td>
<td>DATA=VELTRUE</td>
<td>INLINE</td>
<td>BL</td>
</tr>
<tr>
<td></td>
<td>JJ=21</td>
<td>DATA=HDOT</td>
<td>INLINE</td>
<td>BL</td>
</tr>
<tr>
<td></td>
<td>JJ=22</td>
<td>DATA=MHDOT</td>
<td>INLINE</td>
<td>BL</td>
</tr>
<tr>
<td></td>
<td>JJ=23</td>
<td>DATA=CHUSSRNG</td>
<td>INLINE</td>
<td>BL</td>
</tr>
<tr>
<td></td>
<td>JJ=24</td>
<td>DATA=DUNRNG</td>
<td>INLINE</td>
<td>BL</td>
</tr>
<tr>
<td></td>
<td>JJ=25</td>
<td>DATA=LAT</td>
<td>INLINE</td>
<td>BL</td>
</tr>
</tbody>
</table>
ENDI
JJ=26
DATA=LNG
INLINE
BL )950
ENDI
JJ=27
DATA=DER
INLINE
BL )950
ENDI
JJ=28
DATA=ELEV
INLINE
BL )950
ENDI
JJ=29
DATA=DAIL
INLINE
BL )950
ENDI
JJ=30
DATA=ALTITUDE
INLINE
BL )950
ENDI
JJ=31
DATA=DEL
INLINE
BL )950
ENDI
JJ=32
DATA=DRUD
INLINE
BL )950
ENDI
JJ=33
DATA=BRAK
INLINE
BL )950
ENDI
JJ=34
DATA=FLAP
INLINE
BL )950
ENDI
JJ=35
DATA=UZCMD
INLINE
BL )950
ENDI

* "READY" IS USED TO INDICATE THAT A NEW LOOP IS TO START.
* IT IS SET TO TRUE BY THE INTERVAL TIMER INTERRUPT HANDLER.
  READY=.FALSE.
*
*TEST FOR DESIRED EASE
*EASE IS REQUESTED BY SETTING SENSE SWITCH 0 = TRUE.

118
* 500 CALL SSWITCH(12,EASER)
    IF(.NOT.EASER) GO TO 550
505 CONTINUE
  IN LIN I
  DI 43 DISABLE INTERVAL TIMER
ENDI
* C ANGLES ARE FOR COMMUNICATIONS TO AND FROM THE OPERATOR VIA EASE.
* THEY ARE EXPRESSED IN DEGREES.
  CTHETA=THETA*57.29577951
  CPHI=PHI*57.29577951
  CPSI=PSI*57.29577951
  CALPHA=ALPHA*57.29577951
  CPSIH=PSIH*57.29577951
  CPSIK=PSIK*57.29577951
  CMU=MU*57.29577951
  CLAT=LAT*57.29577951
  CLNG=NG*57.29577951
  CALL M:LOAD ('EASEFCS ',11)
  CALL EASE(DATADBL, DBLSIZE, DATAHDR, RSIZE, DATAHOU, ISIZE,
  * DATAMF, HAFSIZE, DBLBYI, BYTISIZE, LIST, SIZE)
  GO TO 10
*
**WAIT FOR INTERVAL TIMER INTERRUPT
  550 TIME=TIME+DELTA
  IF(TIME,GT, RUNTIME) GO TO 505
  551 CONTINUE
  IF(READY) GO TO 47
  GO TO 551
* 
** THIS IS THE ANALOG INPUT IUCB
  IN LIN I
  )910 DATAM BZC0010020
  ACH )911
  ACH DATAIN
  RES 1W
  )911 RES 1W
*
** THIS IS THE LINKED IUCB FOR BOTH ANALOG INPUT AND DIGITAL OUTPUT
  )915 DATAM BZ42000002C
  ACH )916
  ACH DATAOUT
  RES 1W
  )920 DATAM BZEO000001
  ACH )921
  ACH DISCRTIN
  RES 1W
  )916 RES 1W
  )921 DATAM BZ40000000
ENDI
*
** IN LIN I
  )950 STM 0,0,1NK
ENDI
  IF(SCALOUT(JJ),LT,0) GO TO 954
  IF(DATA.LE,SCALOUT(JJ)) GO TO 951
  DATAOUT(JJ)=32767
  GO TO 953
  951 IF(DATA,GE, SCALOUT(JJ)) GO TO 952

119
DATAOUT(JJ)=32767
GO TO 953
952 DATAOUT(JJ)=DATA/SCALOUT(JJ)*32767
INLINE
 )953
  LW 3, JJ
  SLA 3, 1
  LM 7, DATAOUT=2, 3
  SRL 7, 4
  STM 7, DATAOUT=2, 3
  LW 0, LINK
  TKSW 0
ENDI
954 IF (DATA .GE. SCALOUT(JJ)) GO TO 955
DATAOUT(JJ)=32767
GO TO 953
955 IF (DATA .LE. SCALOUT(JJ)) GO TO 952
DATAOUT(JJ)=32767
GO TO 953
*
INLINE
 )960
  STM 0, LINK
  LW 3, JJ
  SLA 3, 1
  LM 7, DATAIN=2, 3
  SLL 7, 1
  STM 7, DATAIN=2, 3
ENDI
DATA=DATAIN(JJ)*SCALIN(JJ)/32767
INLINE
  LW 0, LINK
  TKSW 0
ENDI
*
INLINE
 )1000
  ACM 1010
 )1010
  RES 1
  STM 7, SAVFL
  LB 7, 22FF
  STB 7, READY
  LW 7, SAVFL
  BHI 1010
ENDI
*
INLINE
 )1100
  DI 43
  LW 7, HANDLER
  STM 7, 428
ENDI
STOP
*
INLINE
 )1105
  DI 45
  LEA 6, )1105
  LW 7, 82FFFFFFFFEC
  CALM 84
ENDI
GO TO 10
END
120
**APPENDIX B - EASE PROGRAM LISTING**

**EASE**

*SUBROUTINE EASE(DATADBL,DDBSIZE,DATAHDRK,HSIZE,DATAHDRI,ISIZE, 1 DATAAF,HAFSIZE,DATABYI,BYTSIZE,LIST,SIZE)*

THE PURPOSE OF THE EASE PROGRAM IS TO ALLOW INPUT AND OUTPUT OF DATA VALUES. DATA MAY BE REFERENCED BY EITHER VARIABLE NAME OR NUMBERED MEMORY LOCATION.

DATA IS COMMUNICATED IN THE FOLLOWING FASHION—

NAMED VARIABLES ARE ADDRESSED BY NAME, AND MEMORY LOCATIONS ARE ADDRESSED BY A FULLWORD ADDRESS.

TO EXAMINE A DATA VALUE A QUESTION MARK SHOULD FOLLOW THE LOCATION DESCRIPTION.

TO CHANGE THE VALUE OF THE SPECIFIED LOCATION AN EQUAL SIGN SHOULD FOLLOW THE LOCATION DESCRIPTION.

INQUIRIES OF OR INPUTS TO MEMORY LOCATIONS THAT ARE INTEGRAL VALUES SHOULD BE PRECEDED BY AN ASTERISK.

INQUIRIES OF OR INPUTS TO MEMORY LOCATIONS THAT ARE REAL VALUES MAY BE LEFT UNPREFIXED.

IMBEDDED BLANKS ARE IGNORED IN ALL CASES.

TO RETURN TO THE MAIN PROGRAM AN EXCLAMATION MARK SHOULD BE TYPED.

HEX ADDRESSES BEGINNING WITH A LETTER (C.F. A72C) MUST BE PRECEDED BY A ZERO (C.F. 0A72C)

THE FOLLOWING INPUT WILL CAUSE ALFA TO BE SET TO 30 DEGREES—

ALFA=30

THE FOLLOWING WILL CAUSE THE PROGRAM TO OUTPUT ZETA—

ZETA?

THE FOLLOWING WILL CAUSE THE PROGRAM TO OUTPUT A REAL VALUE FROM LOCATION 12345

12345?

THE FOLLOWING WILL INPUT THE INTEGER VALUE OF 546 TO LOCATION 12345

*12345#456

THE DATA STORAGE ARRAYS ARE SET UP AS FOLLOWS

*SUBROUTINE EASE(DATADBL,DDBSIZE,DATAHDRK,HSIZE,DATAHDRI,ISIZE, 1 DATAAF,HAFSIZE,DATABYI,BYTSIZE,LIST,SIZE)*

THE DATA STORAGE ARRAYS ARE SET UP AS FOLLOWS

***********************

*DATADBL(K)=REAL DOUBLEWORD ARRAY*

121
**DATAHDL(K) = REAL WORD ARRAY**

**DATAWDL(K) = INTEGER WORD ARRAY**

**DATAHAF(K) = INTEGER HALFWORD ARRAY**

**DATABYT(K) = INTEGER BYTE ARRAY**

THE LIST SYMBOL ARRAY CONTAINS A DOUBLEWORD CHARACTER NAME FOR EACH VARIABLE IN THE DATA(K) ARRAY. THE VALUE OF K IN LIST(K) CORRESPONDS TO THE VALUE OF K IN THE DATA(K) ARRAYS.

THE FOLLOWING VARIABLES MUST BE DIMENSIONED AS INDICATED TO INSURE PROPER OPERATION OF THE PROGRAM IN EACH CASE:

- DATAHDL(VALUE OF KMAXHDL, SIZE)
- DATAHAF(VALUE OF KMAXHAF, VALUE OF KMAXHAF)
- DATAWDL(VALUE OF KMAXWDL, VALUE OF KMAXWDL)
- DATABYT(VALUE OF KMAXBYTE, VALUE OF KMAXBYTE)
- LIST(TOTAL NUMBER OF VARIABLES)

ALSO, THE FOLLOWING PARAMETERS MUST BE INITIALIZED IN A DATA STATEMENT:

- SIZE = TOTAL NUMBER OF VARIABLES
- KMAXHDL = VALUE OF K AT END OF DOUBLEWORD ARRAY
- KMAXHAF = VALUE OF K AT END OF INTEGER HALFWORD ARRAY
- KMAXWDL = VALUE OF K AT END OF INTEGER WORD ARRAY
- KMAXWDL = VALUE OF K AT END OF INTEGER HALFWORD ARRAY

BIT INTEGER, LETTER, DECIMAL
CHAR MNUH(18), TEMP(8), NUMB(5), OUTBUF(25), NAME1N(25)
CHAR MSSG1(15), MSSG2(18), MSSG3(34), MSSG4(33), MSSG5(35), MSSG6(31)
CHAR MSSG7(37), MSSG8(36), MSSG9(29), MSSG10(22)
INTEGER*2 DBLSIZE, RSIZE, ISIZE, HAFSIZE, BYTSIZE, SIZE
INTEGER*1 DATABYT(BYTSIZE)
INTEGER*2 DATAHAF(HAFSIZE)
INTEGER*4 DATAMDI(ISIZE)
INTEGER*4 ADDR,BUXI
INTEGER*8 LIST(SIZE),INBUF1,SYMBOL
REAL*4 DATAWRDR(RSIZE),BOXR
REAL*8 DATAUBL(DBLSIZE),INBUF2
EQUIVALENCE (SYMBOL,TEMP(1)),(TEMP(1),NUMB(1)),(BOXR,BUXI)
DATA MSSG1/'WELCOME TO EASE'/
DATA MSSG2/'WRONG INPUT FORMATT'/
DATA MSSG3/'VARIABLE NAME EXCEEDS 8 CHARACTERS'/
DATA MSSG4/'NO SUCH SYMBOL IS DEFINED TO EASE'/
DATA MSSG5/'MEMORY ADDRESS MUST BE A HEX NUMBER'/
DATA MSSG6/'MEMORY ADDRESS EXCEEDS 5 DIGITS'/
DATA MSSG7/'MEMORY ADDRESS IS OUTSIDE CORE LIMITS'/
DATA MSSG8/'MEMORY DATA MUST BE A DECIMAL NUMBER'/
DATA MSSG9/'MEMORY DATA EXCEEDS 25 DIGITS'/
DATA MSSG10/'END OF EASE PROCESSING'/
KMAXDBL=DBLSIZE
KMAXREAL=DSIZE+RSIZE
KMAXMD=DSIZE+RSIZE+ISIZE
KMAXHAF=DSIZE+RSIZE+ISIZE+HAFSIZE
CALL CARRIAGE
CALL MTELEW(MSSG1,15)
CALL CARRIAGE
1 CONTINUE
LETTEH=INTGR=.FALSE.
CALL CARRIAGE
CALL MTELEW(MSSG1,15)
L=2
* TEST FOR BLANKS IN INPUT LINE
15 IF(WORD(L),NE.' ') GO TO 20
L=L+1
GO TO 15
20 IF(WORD(L),EQ.'1') GO TO 100
IF((WORD(L),LT.'A'),OR.(WORD(L),GT.'Z')) GO TO 22
GO TO 200
22 IF(WORD(L),NE.' ') GO TO 24
INTGR=.TRUE.
L=L+1
23 IF(WORD(L),NE.' ') GO TO 24
GO TO 23
24 IF((WORD(L),LT.'0'),OR.(WORD(L),GT.'9')) GO TO 30
GO TO 300
40 CONTINUE
IF(WORD(L),EQ.'2') GO TO 400
IF(WORD(L),EQ.'3') GO TO 500
GO TO 30
*
* THE FOLLOWING SECTION CONTAINS ERROR MESSAGES
*
30 CALL CARRIAGE
CALL MTELEW(MSSG2,18)
GO TO 10
*
50 CALL CARRIAGE
CALL MTELEW(MSSG3,34)
GO TO 10
*
60 CALL CARRIAGE  
  CALL MIELEW(MSSG4,33)  
  GO TO 10  
*  
70 CALL CARRIAGE  
  CALL MIELEW(MSSG5,35)  
  GO TO 10  
*  
80 CALL CARRIAGE  
  CALL MIELEW(MSSG6,31)  
  GO TO 10  
*  
90 CALL CARRIAGE  
  CALL MIELEW(MSSG7,37)  
  GO TO 10  
*  
55 CALL CARRIAGE  
  CALL MIELEW(MSSG8,36)  
  GO TO 10  
*  
65 CALL CARRIAGE  
  CALL MIELEW(MSSG9,29)  
  GO TO 10  
*  
* NAMED VARIABLE PROCESSING  
200 CONTINUE  
* LETTER SIGNIFICATES TO THE I/O SECTION THAT THE EFFECTIVE VARIABLE  
* ADDRESS IS DEFINED BY LOCATION IN THE VARIABLE TABLE.  
* LETERN='TRUE'.  
* INITIALIZE TEMP TO BLANKS  
  TEMP=' '  
  J=1  
210 TEMP(J)=WORD(L)  
220 L=L+1  
  IF(WORD(L),NE,' ') GO TO 250  
  GO TO 220  
230 IF(WORD(L),EQ,'?'),OR,(WORD(L),EQ,'?')) GO TO 250  
  J=J+1  
  IF(J,LE,9) GO TO 50  
  GO TO 210  
* TEST FOR SYMBOL IN SYMBOL LIST AND FIND RELATIVE LOCATION IN DATA  
* ARRAYS  
250 K=1  
255 IF(SYMBOL,EQ,LIST(K)) GO TO 260  
  K=K+1  
  IF(K,GT,SIZE) GO TO 60  
  GO TO 255  
260 IF(K,GT,KMAXREAL) INTGK='TRUE'.  
* RETURN TO MAIN  
  GO TO 40  
*  
* NUMBERED ADDRESS PROCESSING  
300 CONTINUE  
* FILL NUMB WITH BLANKS  
  NUMB=' '  
124
J=1
310 NUMB(J) = WORD(L)
320 L = L + 1
   IF (WORD(L).NE. ' ') GO TO 330
   GO TO 320
330 IF ((WORD(L).EQ. '?') .OR. (WORD(L).EQ. ' ')) GO TO 350
   IF ((WORD(L).GE. '0') .AND. (WORD(L).LE. '9')) GO TO 340
   IF ((WORD(L).GE. 'A') .AND. (WORD(L).LE. 'F')) GO TO 340
   GO TO 70
340 J = J + 1
   IF (J .LE. 6) GO TO 30
   GO TO 310
* NUMB MUST BE RIGHT JUSTIFIED FOR CONVERSION
350 IF (J .EQ. 5) GO TO 357
   MOVE = J
   DO 355 N = 1, MOVE
   NUMB(N) = NUMB(N+1)
   NUMB(N+1) = NUMB(N+2)
   NUMB(N+2) = NUMB(N+3)
   NUMB(N+3) = NUMB(N+4)
   NUMB(N+4) = NUMB(N+5)
   NUMB(N+5) = ' '
* CONVERT ASCII INPUT TO HEX ADDRESS
357 DECODE(5, 360, NUMB) ADDR
360 FORMAT(25)
* TEST FOR EXISTENT MEMORY ADDRESS
   IF (ADDR .GT. 527FFC) GO TO 90
* RETURN TO MAIN
   GO TO 40

* CRT OUTPUT PROCESSING
* TEST TO DETERMINE IF NUMBERED LOCATION OR NAMED VARIABLE OUTPUT IS DESIRED
400 CONTINUE
   IF (LETTER) GO TO 450
* RETRIEVE VALUE FROM EFFECTIVE ADDRESS
   INLINE
   LW 7, *ADDR
   STW 7, BOXR
ENDI
* TEST FOR INTEGER OR REAL CONVERSION
   IF (INTGR) GO TO 410
   ENCODE(25, 405, OUTBUF) BOXR
405 FORMAT(25.10)
   GO TO 490
410 ENCODE(25, 415, OUTBUF) BOXI
415 FORMAT(125)
   GO TO 490
450 IF (INTGR) GO TO 470
* TEST TO DETERMINE WHICH ARRAY TYPE IS TO BE ACCESSED, AND PROCESS
* ACCORDINGLY
   IF (K .GT. KMAXDBL) GO TO 460
   ENCODE(25, 405, OUTBUF) DATADBL(K)
   GO TO 490
460 ENCODE(25, 405, OUTBUF) DATAMKDBL(K = KMAXDBL)
   GO TO 490
470 IF (K .GT. KMAXBUF) GO TO 485
   IF (K .GT. KMAXBUF) GO TO 480
ENCODE(25,415,OUTBUF)DATAMUI(K=KMAXHEAL)
GO TO 490
480 ENCODE(25,415,OUTBUF)DATAMAF(K=KMAXHDO)
GO TO 490
485 ENCODE(25,415,OUTBUF)DATAMYT(K=KMAXHAF)
* OUTPUT ANSWER
490 CALL CAMKIAGE
CALL MTELEM(OUTBUF,25)
* RETURN TO MAIN
GO TO 10
*
*
* CRT INPUT PROCESSING
* FINISH INPUT OF DESIRED VALUE AND STORE
500 CONTINUE
DECIMAL=.FALSE.
J=1
NAMEIN=''
510 L=L+1
* TEST FOR BLANKS
   IF(MORD(L).EQ.' ') GO TO 510
* TEST FOR CAMKIAGE RETURN
   IF(MORD(L).EQ.2200) GO TO 520
* TEST FOR VALID NUMBER
   IF((MORD(L).GE.'0') .AND. (MORD(L).LE.'9')) GO TO 515
   IF(MORD(L).EQ.'-') GO TO 515
   IF(MORD(L).NE.' ') GO TO 55
   DECIMAL=.TRUE.
515 NAMEIN(J)=MORD(L)
   J=J+1
   IF(J.EQ.26) GO TO 65
   GO TO 510
* NAMEIN NOW CONTAINS THE VALUE TO BE STORED IN ASCII
* TEST FOR INTEGER OR REAL CONVERSION
520 IF(INTEGR) GO TO 525
   IF(.NOT.DECIMAL) NAMEIN(J)='.'
   DECODE(25,405,NAMEIN)INBUF
   GO TO 530
525 IF(J.EQ.25) GO TO 529
* NAMEIN MUST BE RIGHT JUSTIFIED FOR CONVERSION
   MOVE=26-J
   DO 527 I=1,MUE
   DO 526 N=1,24
526 NAMEIN(26-N)=NAMEIN(25-N)
527 NAMEIN(1)=''
529 DECODE(25,415,NAMEIN)INBUF
* INBUF CONTAINS THE REAL DATA VALUE, AND INBUF CONTAINS THE INTEGER
* DATA VALUE
* TEST WHETHER DATA IS FOR NAMED VARIABLE OR NUMBERED MEMORY LOCATION
530 IF(LETTER) GO TO 550
   IF(INIGH) GO TO 540
* STORE REAL VALUE IN EFFECTIVE ADDRESS
   INLINE
   LW 7,INBUF
   STW 7,*ADDR
   END
   GO TO 599
* STORE INTEGER VALUE IN EFFECTIVE ADDRESS

126
* CONTINUE
  INLINEx
  LM  7, INBUF1 + 1W
  SIW  7, * ADDR
  ENDI
  GO TO 599
* CHOOSE DATA TABLE IN WHICH VARIABLE IS TO BE STORED
  IF(K. GT. KMAXHAF) GO TO 560
  IF(K. GT. KMAXHDF) GO TO 562
  IF(K. GT. KMAXREAL) GO TO 564
  IF(K. GT. KMAXDBL) GO TO 566
  DATADBF(K) = INBUF1
  GO TO 570
  DATABYT(K = KMAXHAF) = INBUF1
  GO TO 570
  DATAHDF(K = KMAXHDF) = INBUF1
  GO TO 570
  DATAFDK(K = KMAXREAL) = INBUF1
  GO TO 570
  DATAFUL(K = KMAXDBL) = INBUF1
* THE FOLLOWING SECTION TEST FOR VALUES OF K THAT REQUIRE ADDITIONAL
* CALCULATIONS FOR EACH CHANGE IN VALUE
  570 CONTINUE
* *
* INSERT ADDED CALCULATIONS HERE  **********************
***
**
* RETURN TO MAIN
  599 GO TO 10
* *
  100 CALL CARRIAGE
      CALL M; T; LEA; (MSSG10, 22)
      RETURN
* *
END OF EASE PROGRAM
END
APPENDIX C - FUNCTION GENERATION PACKAGE

```plaintext
PROGRAM DATASTUK
REAL DATA(1000)
INTEGER*8 NAME,NAMELIST(45)

* NOTE THAT OUTDATA MUST BE DIMENSIONED TO 'SIZE', OR THE TOTAL
* NUMBER OF WORDS IN ALL DATA ARRAYS.
REAL*4 OUTDATA(5243)

* INTEGER*4 LENGTH,NUMBER,DIFF,ADDRESS,SIZE,PIT,STAT

<table>
<thead>
<tr>
<th>REAL*4</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCOT</td>
<td>63</td>
</tr>
<tr>
<td>OCCET</td>
<td>378</td>
</tr>
<tr>
<td>CODBFT</td>
<td>63</td>
</tr>
<tr>
<td>CDBF2T</td>
<td>63</td>
</tr>
<tr>
<td>CDBS1T</td>
<td>63</td>
</tr>
<tr>
<td>CDBSB2T</td>
<td>63</td>
</tr>
<tr>
<td>CLBTA</td>
<td>252</td>
</tr>
<tr>
<td>CLBED1T</td>
<td>63</td>
</tr>
<tr>
<td>CLBED2T</td>
<td>63</td>
</tr>
<tr>
<td>CLBOSB1T</td>
<td>63</td>
</tr>
<tr>
<td>CLBOSB2T</td>
<td>63</td>
</tr>
<tr>
<td>CLBOSB3T</td>
<td>63</td>
</tr>
<tr>
<td>CLDAT</td>
<td>378</td>
</tr>
<tr>
<td>CLDRT</td>
<td>252</td>
</tr>
<tr>
<td>CLPT</td>
<td>63</td>
</tr>
<tr>
<td>CLRT</td>
<td>63</td>
</tr>
<tr>
<td>CMUT</td>
<td>63</td>
</tr>
<tr>
<td>DCMET</td>
<td>378</td>
</tr>
<tr>
<td>CMDHF1T</td>
<td>63</td>
</tr>
<tr>
<td>CMDHF2T</td>
<td>63</td>
</tr>
<tr>
<td>CMDSB1T</td>
<td>63</td>
</tr>
<tr>
<td>CMDSB2T</td>
<td>63</td>
</tr>
<tr>
<td>CMQI</td>
<td>252</td>
</tr>
<tr>
<td>CNBITABL</td>
<td>252</td>
</tr>
<tr>
<td>CNBDE1T</td>
<td>63</td>
</tr>
<tr>
<td>CNBDE2T</td>
<td>63</td>
</tr>
<tr>
<td>CNBDSB1T</td>
<td>63</td>
</tr>
<tr>
<td>CNBDSB2T</td>
<td>63</td>
</tr>
<tr>
<td>CNBDSB3T</td>
<td>63</td>
</tr>
<tr>
<td>CNDAT</td>
<td>378</td>
</tr>
<tr>
<td>CNDHT</td>
<td>252</td>
</tr>
<tr>
<td>CNDHBT</td>
<td>63</td>
</tr>
<tr>
<td>CNP</td>
<td>(63)</td>
</tr>
<tr>
<td>CNRT</td>
<td>(63)</td>
</tr>
<tr>
<td>CNOT</td>
<td>(63)</td>
</tr>
<tr>
<td>CNDE1T</td>
<td>(63)</td>
</tr>
<tr>
<td>CNDHF1T</td>
<td>(63)</td>
</tr>
<tr>
<td>CNDHF2T</td>
<td>(63)</td>
</tr>
<tr>
<td>CNDHSB1T</td>
<td>63</td>
</tr>
<tr>
<td>CNDHSB2T</td>
<td>63</td>
</tr>
<tr>
<td>CYBT</td>
<td>(63)</td>
</tr>
<tr>
<td>CYDAT</td>
<td>(378)</td>
</tr>
<tr>
<td>CYDT</td>
<td>(63)</td>
</tr>
<tr>
<td>CYDRSB1T</td>
<td>(7)</td>
</tr>
<tr>
<td>CYDRSB2T</td>
<td>(7)</td>
</tr>
</tbody>
</table>
```

128
CHAR BUFFER(81), COMMENTS(65), MSG(80), TEST
CHAR MSG1(45), MSG2(44), MSG3(64)
EQUIVALENCE (BUFFER(2), MSG(1), TEST)
EQUIVALENCE
* (OUTDATA(1), CCOT(1))
* (OUTDATA(64), DCCET(1))
* (OUTDATA(442), CCDBF1T(1))
* (OUTDATA(505), CCDBF2T(1))
* (OUTDATA(568), CCDSB1T(1))
* (OUTDATA(631), CCDSB2T(1))
* (OUTDATA(694), CLBTABL(1))
* (OUTDATA(746), CLBDE1T(1))
* (OUTDATA(1009), CLBUE2T(1))
* (OUTDATA(1072), CLBUSB1T(1))
* (OUTDATA(1135), CLBDSB2T(1))
* (OUTDATA(1198), CLBDSB3T(1))
* (OUTDATA(1261), CLDAT(1))
* (OUTDATA(1639), CLDRT(1))
* (OUTDATA(1891), CLPT(1))
* (OUTDATA(1954), CLR(1))
* (OUTDATA(2017), CMUT(1))
* (OUTDATA(2080), DCMET(1))
* (OUTDATA(2458), CMDBF1T(1))
* (OUTDATA(2521), CMDBF2T(1))
* (OUTDATA(2584), CMDSB1T(1))
* (OUTDATA(2647), CMDSB2T(1))
* (OUTDATA(2710), CMGT(1))
* (OUTDATA(2962), CNBABL(1))
* (OUTDATA(3214), CNBDE1T(1))
* (OUTDATA(3277), CNBDE2T(1))
* (OUTDATA(3340), CNBDSB1T(1))
* (OUTDATA(3403), CNBDSB2T(1))
* (OUTDATA(3466), CNBDSB3T(1))
* (OUTDATA(3529), CNDAT(1))
* (OUTDATA(3907), CNDRT(1))
* (OUTDATA(4159), CNDRTT(1))
* (OUTDATA(4222), CNPT(1))
* (OUTDATA(4285), CNRT(1))
* (OUTDATA(4348), CNUT(1))
* (OUTDATA(4411), CNDET(1))
* (OUTDATA(4474), CNDBF1T(1))
* (OUTDATA(4537), CNDBF2T(1))
* (OUTDATA(4600), CNDSB1T(1))
* (OUTDATA(4663), CNDSB2T(1))
* (OUTDATA(4726), CYBT(1))
* (OUTDATA(4789), CYDAT(1))
* (OUTDATA(5167), CYDRT(1))
* (OUTDATA(5230), CYDRSb1T(1))
* (OUTDATA(5237), CYDRSb2T(1))
DATA MSG1/'IF HEADER CARD CORRECT TYPE C, IF NOT TYPE A?'/
DATA MSG2 /'PLEASE INPUT CORRECT HEADER CARD INFORMATION'/
DATA MSG3/'NAME WAS NOT FOUND IN NAMELIST PLEASE ENTER CORRECT INFORMATION'/
DATA SIZE/5243/
DATA NUMBER/45/

DATA NAMELIST/
* 'CCO' 
* 'CCE' 
* 'CCDBF1' 
* 'CCDBF2' 
* 'CCDSB1' 
* 'CCDSB2' 
* 'CLB1' 
* 'CLBDE1' 
* 'CLBDE2' 
* 'CLBDSB1' 
* 'CLBDSB2' 
* 'CLD1' 
* 'CLDDE1' 
* 'CLDDE2' 
* 'CLDDBF1' 
* 'CLDDBF2' 
* 'CLDDBS1' 
* 'CLDDBS2' 
* 'CLDDBS3' 
* 'CLDA' 
* 'CLDK' 
* 'CLP' 
* 'CLM' 
* 'CMO' 
* 'CME' 
* 'CMDBF1' 
* 'CMDBF2' 
* 'CMDBS1' 
* 'CMDBS2' 
* 'CMG1' 
* 'CNH1' 
* 'CNBDE1' 
* 'CNBDE2' 
* 'CNBDSB1' 
* 'CNBDSB2' 
* 'CNBDSB3' 
* 'CNDA' 
* 'CNDR' 
* 'CNDRB' 
* 'CNP' 
* 'CNR' 
* 'CMG' 
* 'CNDE' 
* 'CNDBF1' 
* 'CNDBF2' 
* 'CNDBS1' 
* 'CNDBS2' 
* 'CYB' 
* 'CYDA' 
* 'CUDR' 
* 'CYDRE1' 
* 'CYDRE2' 
* 'CYDRE3' 
* 'CYDDBFI' 
* 'CYDDBF2' 
* 'CYDDBS1' 
* 'CYDDBS2' /
10 DATA = 0,0
READ (4,20,END=150) NAME, LENGTH, COMMENTS
20 FORMAT (A8,1X,1b,65A1)
30 ENCODE (80,20,MSG) NAME, LENGTH,COMMENTS
IF (TEST.EQ.'C') GO TO 50
* THIS SECTION OUTPUTS THE HEADER CARD TO THE CRT

CALL CARRIAGE
CALL M:TELEW (MSSG, 80)
CALL CARRIAGE
CALL M:TELEW (MSSG1, 45)
CALL M:TELEW (BUFFER, 80)
IF (TEST equ 'I') GO TO 150
IF (TEST equ 'C') GO TO 50
CALL CARRIAGE
CALL M:TELEW (MSSG2, 44)
40 CALL CARRIAGE
CALL M:TELEW (MSSG, 80)
CALL M:TELEW (MSSG, 80)
CALL CARRIAGE
CALL M:TELEW (BUFFER, 81)
DECODE(80, 20, MSSG) NAME, LENGTH, COMMENTS
GO TO 30

* SEARCH EASE TABLE OF DERIVATIVE NAMES FOR NAME AND RETURN POSITION

50 J = 0
51 J = J + 1
IF (NAME equ NAMELIST(J)) GO TO 55
IF (J .LT. NUMBER) GO TO 51
CALL CARRIAGE
CALL M:TELEW (MSSG3, 64)
GO TO 40
55 CONTINUE
INLINE
LW 3, J
SLA 3, 2
LM 7, 200-4, 3
STW 7, ADDRESS
ENDI

* THIS SECTION READS THE DATA CARDS UNTIL THE NUMBER OF POINTS READ = LENGTH

59 READ (4, 60) (DATA(I), I = 1, LENGTH)
60 FORMAT (BF10, 6)

DO 160 I = 1, LENGTH
DATAW = DATA(I)
INLINE
LW 3, I
SLA 3, 2
ADM 3, ADDRESS
LM 7, DATAW
STW 7, = 4, 3
ENDI
160 CONTINUE
GO TO 1010
150 CALL CARRIAGE
PIT = 1152
180 CALL BUFFEROUT(S,1,OUTDATA(I),PIT)
185 CALL STATUS(S,STAT)
   IF (STAT .NE. 2) GO TO 185
   I = I + 1152
   IF (I .GT. SIZE) GO TO 186
   IF (I + 1152 .GT. SIZE) PIT = SIZE + 1 - I
   GO TO 180
186 CONTINUE
STOP
END
DERIVE FUNCTIONS

FUNCTION DERIVE1(ARRAY1, I, ARGPN1, ARFRAC1)
INTEGER*2 ARGPN1, I
DIMENSION ARRAY1(I)
DERIVE1 = (ARRAY1(ARGPN1 + 1) - ARRAY1(ARGPN1)) * ARFRAC1
RETURN
END

FUNCTION DERIVE2(ARRAY2, I, J, ARGPN1, ARGPN2, ARFRAC1, ARFRAC2)
INTEGER*2 ARGPN1, ARGPN2
TEMP3 = (ARRAY2(ARGPN1 + 1, ARGPN2 + 1) - ARRAY2(ARGPN1, ARGPN2 + 1)) * ARFRAC1
TEMP2 = (ARRAY2(ARGPN1 + 1, ARGPN2) - ARRAY2(ARGPN1, ARGPN2)) * ARFRAC1 + ARFRAC2
DERIVE2 = (TEMP3 - TEMP2) * ARFRAC2 + TEMP2
RETURN
END

FUNCTION DERIVE3(ARRAY3, I, J, K, ARGPN1, ARGPN2, ARGPN3, ARFRAC1, ARFRAC2, ARFRAC3)
INTEGER*2 ARGPN1, ARGPN2, ARGPN3
TEMP7 = (ARRAY3(ARGPN1 + 1, ARGPN2 + 1, ARGPN3 + 1) - ARRAY3(ARGPN1, ARGPN2, ARGPN3 + 1)) * ARFRAC1
TEMP6 = (ARRAY3(ARGPN1 + 1, ARGPN2, ARGPN3 + 1) - ARRAY3(ARGPN1, ARGPN2, ARGPN3)) * ARFRAC1 + ARFRAC2
TEMP5 = (ARRAY3(ARGPN1 + 1, ARGPN2, ARGPN3) - ARRAY3(ARGPN1, ARGPN2 + 1, ARGPN3)) * ARFRAC1 + ARFRAC3
TEMP3 = (TEMP7 - TEMP6) * ARFRAC2 + TEMP6
TEMP2 = (TEMP5 - TEMP4) * ARFRAC2 + TEMP4
DERIVE3 = (TEMP3 - TEMP2) * ARFRAC3 + TEMP2
RETURN
END

FUNCTION DERIVE4(ARRAY4, I, J, K, L, ARGPN1, ARGPN2, ARGPN3, ARGPN4, ARFRAC1, ARFRAC2, ARFRAC3, ARFRAC4)
INTEGER*2 ARGPN1, ARGPN2, ARGPN3, ARGPN4
DERIVE4 = THE VALUE RETURNED TO THE MAIN PROGRAM.
ARRAY4 IS A FOUR DIMENSIONAL DUMMY ARRAY CONTAINING FUNCTION DATA
I IS AN INTEGER VALUE REPRESENTING THE SIZE OF THE FIRST DIMENSION OF
J IS AN INTEGER VALUE REPRESENTING THE SIZE OF THE SECOND DIMENSION OF
K IS AN INTEGER VALUE REPRESENTING THE SIZE OF THE THIRD DIMENSION OF
L IS AN INTEGER VALUE REPRESENTING THE SIZE OF THE FOURTH DIMENSION
**S U B P R O G R A M  D E F I N I T I O N S**

THE FOLLOWING ARE THE FUNCTION TYPE SUBPROGRAM DEFINITIONS

REAL FUNCTION PDOT(A, B, C)

NOTE THAT A, B, AND C REPRESENT P, Q, AND R RESPECTIVELY

IMPLICIT REAL*4 (A-Z)

COMMON /ARRAY4/ARRAY(4),IT(XY),IXZ,A4(4),B4, MBODY,NDBODY

PDOT=IFUNC((LBODY-1XY*A*C+1XZ*A*B+IYMZ*B*C)
1+IYUY*(MBODY+1XY*A*C+1XZ*A*B+IYMZ*B*C)
1+IZOZ*(NDBODY-1XY*(B**2-A**2)-1XZ*B*C+IXMY*A*B))

RETURN

END
* REAL FUNCTION wLALT(A,H,C,U)
  * A, B, C, D REPRESENT P, W, R, PDI RESPECTIVELY
  * IMPLICIT REAL*4 (A-Z)
  * COMMON /ARRAY1/DATA*MUR(400)
  * EQUIVALENCE (DATAMUR(60),1XY)
  *,(DATAMUR(61),1XZ)
  *,(DATAMUR(64),1YY)
  *,(DATAMUR(65),1ZMX)
  *,(DATAMUR(87),MBODY)
  *wLALT=(1/1YY)*(MBODY*1XY*(B*C+D)-1XZ*(A**2-C**2)+1ZMX*A*C)
  * RETURN
END
*
* REAL FUNCTION RUOTT(A,B,C,O)
  * A, B, C, D REPRESENT P, W, R, PDI RESPECTIVELY
  * IMPLICIT REAL*4 (A-Z)
  * COMMON /ARRAY1/DATA*MUR(400)
  * EQUIVALENCE (DATAMUR(57),1XY)
  *,(DATAMUR(60),1XY)
  *,(DATAMUR(61),1XZ)
  *,(DATAMUR(67),1IZ)
  *,(DATAMUR(88),MBODY)
  *RUOTT=(1/1ZZ)*(MBODY*1XY*(B**2-A**2)-1XZ*(B*C+D)+1XY*A*B)
  * RETURN
END
*
* REAL FUNCTION THETDUTT(A)
  * A REPRESENTS PHI
  * IMPLICIT REAL*4 (A-Z)
  * COMMON /ARRAY1/DATA*MUR(400)
  * EQUIVALENCE (DATAMUR(106),WBODY),(DATAMUR(114),RBODY)
  * THETDUTT=WBODY*COS(A)-WBODY*SIN(A)
  * RETURN
END
*
* REAL FUNCTION PHIDOTT(A,H,C)
  * A, B, C REPRESENT PHI, THETA, PSIDOTT RESPECTIVELY
  * IMPLICIT REAL*4 (A-Z)
  * COMMON /ARRAY1/DATA*MUR(400)
  * EQUIVALENCE (DATAMUR(100),MBODY)
  * PHIDOTT=MBODY*C*SIN(B)
  * RETURN
END
*
* REAL FUNCTION PSIDOTT(A,B)
  * A, B REPRESENT PHI, THETA RESPECTIVELY
  * IMPLICIT REAL*4 (A-Z)
  * COMMON /ARRAY1/DATA*MUR(400)
  * EQUIVALENCE (DATAMUR(106),WBODY),(DATAMUR(114),RBODY)
  * PSIDOTT=WBODY*COS(A)+WBODY*SIN(A)/COS(B)
  * RETURN
END
*
* REAL FUNCTION SPEED(A)
  * A REPRESENTS AN UNNAMED FUNCTION/VARIABL
  * IMPLICIT REAL*4 (A-Z)
  * IF(A,G.E,1) GO TO 10
  * STUKE=I+A**2/4+A**4/40+A**6/1600
  * RETURN
END
GO TO 20
10 STORE=1.839-.772/A**2+.164/A**4+.035/A**6
20 SPEED=SQR(STORE)
    RETURN

END

******************************************************************************

FUNCTION

******************************************************************************

SUBROUTINE POINTF(ARGVALUE,ARGLTH,ARGPN1,ARFRAC,ANGLIST)
    THIS IS THE FLOATING POINT VERSION OF POINT
    SUBROUTINE POINT C COMPARES THE FLOATING POINT VALUE IN ARGVALUE WITH
    A LIST OF FLOATING POINT VALUES STARTING IN LOCATION ARGLIST
    AND RETURNS ARGPN1 AND ARFRAC TO THE MAIN PROGRAM
    WHERE ARGLIST IS A LIST OF (ARGLTH) REAL NUMBERS IN ASCENDING ORDER
    AND WHERE ARGPN1 IS THE POSITION OF THE LARGEST VALUE IN ARGLIST THAT
    IS LESS THAN ARGVALUE AND ARFRAC IS A SCALED FRACTION VALUE REPRESE
    THE DISTANCE BETWEEN THE TWO POINTS ARGPN1 AND ARGPN1PLUS ONE

INTEGER*2 ARGPN1
REAL ARGVALUE,ANGLIST(ARGLTH)
2  IF (ARGPN1,L.T.1) GOTO 150
4  IF (ARGPN1,G.T.ARGLTH) ARGPN1=ARGLTH
10 IF (ARGVALUE=ANGLIST(ARGPN1)) 70,130,20
20 ITEMP=ARGPN1+1
30 IF (ITEMP,G.T.ARGLTH) GO TO 130
40 IF (ARGVALUE=ANGLIST(ITEMP)) 100,120,50
50 ARGPN1=ITEMP
60 GO TO 20
70 IF (ARGPN1=1) 150,130,80
80 ARGPN1=ARGPN1+1
90 GO TO 10
100 ARFRAC=(ARGVALUE-ANGLIST(ARGPN1))/(ANGLIST(ITEMP)-ANGLIST(ARGPN1))
110 GO TO 140
120 ARGPN1=ITEMP
130 ARFRAC=0
132 IF (ARGPN1,L.T.ARGLTH) GOTO 140
135 ARGPN1=ARGLTH+1
137 ARFRAC=1.0
140 RETURN
150 ARGPN1=1
160 GO TO 10

END

# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #

NOTE: Pages 136 to 148 inclusive contain all the aerodynamic coefficients
used within this program.
<table>
<thead>
<tr>
<th>CMO</th>
<th>63</th>
</tr>
</thead>
<tbody>
<tr>
<td>.0450</td>
<td>.0040</td>
</tr>
<tr>
<td>.0630</td>
<td>.0270</td>
</tr>
<tr>
<td>.0660</td>
<td>.0700</td>
</tr>
<tr>
<td>.0490</td>
<td>.0806</td>
</tr>
<tr>
<td>.0250</td>
<td>.0370</td>
</tr>
<tr>
<td>.0080</td>
<td>.0115</td>
</tr>
<tr>
<td>.0041</td>
<td>.0035</td>
</tr>
<tr>
<td>.0080</td>
<td>.0028</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CME</th>
<th>378</th>
</tr>
</thead>
<tbody>
<tr>
<td>.1155</td>
<td>.1100</td>
</tr>
<tr>
<td>.1110</td>
<td>.0770</td>
</tr>
<tr>
<td>.0860</td>
<td>.0940</td>
</tr>
<tr>
<td>.0608</td>
<td>.0800</td>
</tr>
<tr>
<td>.0420</td>
<td>.0500</td>
</tr>
<tr>
<td>.0300</td>
<td>.0360</td>
</tr>
<tr>
<td>.0150</td>
<td>.0215</td>
</tr>
<tr>
<td>.0150</td>
<td>.0150</td>
</tr>
<tr>
<td>.0760</td>
<td>.0720</td>
</tr>
<tr>
<td>.0560</td>
<td>.0530</td>
</tr>
<tr>
<td>.0730</td>
<td>.0310</td>
</tr>
<tr>
<td>.0650</td>
<td>.0830</td>
</tr>
<tr>
<td>.0423</td>
<td>.0620</td>
</tr>
<tr>
<td>.0300</td>
<td>.0390</td>
</tr>
<tr>
<td>.0163</td>
<td>.0240</td>
</tr>
<tr>
<td>.0120</td>
<td>.0163</td>
</tr>
<tr>
<td>.0120</td>
<td>.0163</td>
</tr>
<tr>
<td>.0425</td>
<td>.0440</td>
</tr>
<tr>
<td>.0310</td>
<td>.0292</td>
</tr>
<tr>
<td>.0140</td>
<td>.0160</td>
</tr>
<tr>
<td>.0120</td>
<td>.0085</td>
</tr>
<tr>
<td>.0048</td>
<td>.0052</td>
</tr>
<tr>
<td>.0028</td>
<td>.0039</td>
</tr>
<tr>
<td>.0016</td>
<td>.0235</td>
</tr>
<tr>
<td>.0110</td>
<td>.0165</td>
</tr>
<tr>
<td>.0000</td>
<td>.0000</td>
</tr>
<tr>
<td>.0000</td>
<td>.0000</td>
</tr>
<tr>
<td>.0000</td>
<td>.0000</td>
</tr>
<tr>
<td>.0000</td>
<td>.0000</td>
</tr>
<tr>
<td>.0000</td>
<td>.0000</td>
</tr>
<tr>
<td>.0538</td>
<td>.0616</td>
</tr>
<tr>
<td>.0354</td>
<td>.0410</td>
</tr>
<tr>
<td>.0158</td>
<td>.0190</td>
</tr>
<tr>
<td>.0120</td>
<td>.0175</td>
</tr>
<tr>
<td>.0070</td>
<td>.0109</td>
</tr>
<tr>
<td>.0740</td>
<td>.0053</td>
</tr>
<tr>
<td>.0580</td>
<td>.0740</td>
</tr>
<tr>
<td>.0430</td>
<td>.0580</td>
</tr>
<tr>
<td>.1190</td>
<td>.1280</td>
</tr>
<tr>
<td>.0935</td>
<td>.1050</td>
</tr>
<tr>
<td>.0590</td>
<td>.0640</td>
</tr>
<tr>
<td>.0438</td>
<td>.0585</td>
</tr>
<tr>
<td>.0260</td>
<td>.0345</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CMDBF1</th>
<th>63</th>
</tr>
</thead>
<tbody>
<tr>
<td>.1380</td>
<td>.0120</td>
</tr>
<tr>
<td>Value 1</td>
<td>Value 2</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>-0.0064</td>
<td>-0.0077</td>
</tr>
<tr>
<td>-0.0040</td>
<td>-0.0039</td>
</tr>
<tr>
<td>-0.0094</td>
<td>-0.0138</td>
</tr>
<tr>
<td>-0.0063</td>
<td>-0.0084</td>
</tr>
<tr>
<td>-0.0040</td>
<td>-0.0057</td>
</tr>
<tr>
<td>-0.0026</td>
<td>-0.0034</td>
</tr>
<tr>
<td>-0.0028</td>
<td>-0.0026</td>
</tr>
<tr>
<td>-0.0007</td>
<td>-0.0010</td>
</tr>
<tr>
<td>-0.0036</td>
<td>-0.0040</td>
</tr>
<tr>
<td>-0.0063</td>
<td>-0.0084</td>
</tr>
<tr>
<td>-0.0036</td>
<td>-0.0057</td>
</tr>
<tr>
<td>-0.0026</td>
<td>-0.0034</td>
</tr>
<tr>
<td>-0.0028</td>
<td>-0.0026</td>
</tr>
<tr>
<td>-0.0036</td>
<td>-0.0057</td>
</tr>
<tr>
<td>-0.0026</td>
<td>-0.0034</td>
</tr>
<tr>
<td>-0.0028</td>
<td>-0.0026</td>
</tr>
<tr>
<td>-0.0036</td>
<td>-0.0057</td>
</tr>
<tr>
<td>-0.0026</td>
<td>-0.0034</td>
</tr>
<tr>
<td>-0.0028</td>
<td>-0.0026</td>
</tr>
<tr>
<td>-0.0036</td>
<td>-0.0057</td>
</tr>
<tr>
<td>-0.0026</td>
<td>-0.0034</td>
</tr>
<tr>
<td>-0.0028</td>
<td>-0.0026</td>
</tr>
<tr>
<td>CNO</td>
<td>0.0351</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
</tr>
<tr>
<td>2.212</td>
<td>0.0475</td>
</tr>
<tr>
<td>1.6690</td>
<td>2.1300</td>
</tr>
<tr>
<td>0.9710</td>
<td>1.4220</td>
</tr>
<tr>
<td>0.6950</td>
<td>0.9060</td>
</tr>
<tr>
<td>0.4690</td>
<td>0.6520</td>
</tr>
<tr>
<td>0.2518</td>
<td>0.4199</td>
</tr>
<tr>
<td>0.1130</td>
<td>0.2429</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CNOE</th>
<th>0.0061</th>
<th>0.0061</th>
<th>0.0060</th>
<th>0.0055</th>
<th>0.0050</th>
<th>0.0050</th>
<th>0.0050</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0050</td>
<td>0.0040</td>
<td>0.0040</td>
<td>0.0040</td>
<td>0.0040</td>
<td>0.0040</td>
<td>0.0040</td>
<td>0.0040</td>
</tr>
<tr>
<td>0.0040</td>
<td>0.0039</td>
<td>0.0039</td>
<td>0.0039</td>
<td>0.0039</td>
<td>0.0039</td>
<td>0.0039</td>
<td>0.0039</td>
</tr>
<tr>
<td>0.0039</td>
<td>0.0038</td>
<td>0.0038</td>
<td>0.0038</td>
<td>0.0038</td>
<td>0.0038</td>
<td>0.0038</td>
<td>0.0038</td>
</tr>
<tr>
<td>0.0038</td>
<td>0.0037</td>
<td>0.0037</td>
<td>0.0037</td>
<td>0.0037</td>
<td>0.0037</td>
<td>0.0037</td>
<td>0.0037</td>
</tr>
<tr>
<td>0.0037</td>
<td>0.0036</td>
<td>0.0036</td>
<td>0.0036</td>
<td>0.0036</td>
<td>0.0036</td>
<td>0.0036</td>
<td>0.0036</td>
</tr>
<tr>
<td>0.0036</td>
<td>0.0035</td>
<td>0.0035</td>
<td>0.0035</td>
<td>0.0035</td>
<td>0.0035</td>
<td>0.0035</td>
<td>0.0035</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CND6F1</th>
<th>0.00068</th>
<th>0.00077</th>
<th>0.00085</th>
<th>0.00091</th>
<th>0.00097</th>
<th>0.00111</th>
<th>0.00152</th>
<th>0.00188</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00030</td>
<td>0.00030</td>
<td>0.00030</td>
<td>0.00030</td>
<td>0.00030</td>
<td>0.00030</td>
<td>0.00030</td>
<td>0.00030</td>
<td>0.00030</td>
</tr>
<tr>
<td>0.00017</td>
<td>0.00017</td>
<td>0.00017</td>
<td>0.00017</td>
<td>0.00017</td>
<td>0.00017</td>
<td>0.00017</td>
<td>0.00017</td>
<td>0.00017</td>
</tr>
<tr>
<td>0.00013</td>
<td>0.00013</td>
<td>0.00013</td>
<td>0.00013</td>
<td>0.00013</td>
<td>0.00013</td>
<td>0.00013</td>
<td>0.00013</td>
<td>0.00013</td>
</tr>
<tr>
<td>0.00009</td>
<td>0.00009</td>
<td>0.00009</td>
<td>0.00009</td>
<td>0.00009</td>
<td>0.00009</td>
<td>0.00009</td>
<td>0.00009</td>
<td>0.00009</td>
</tr>
<tr>
<td>0.00005</td>
<td>0.00005</td>
<td>0.00005</td>
<td>0.00005</td>
<td>0.00005</td>
<td>0.00005</td>
<td>0.00005</td>
<td>0.00005</td>
<td>0.00005</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CND6F2</th>
<th>0.00147</th>
<th>0.00162</th>
<th>0.00176</th>
<th>0.00195</th>
<th>0.00214</th>
<th>0.00250</th>
<th>0.00271</th>
<th>0.00372</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00045</td>
<td>0.00062</td>
<td>0.00081</td>
<td>0.00100</td>
<td>0.00120</td>
<td>0.00140</td>
<td>0.00170</td>
<td>0.00200</td>
<td>0.00248</td>
</tr>
<tr>
<td>0.00034</td>
<td>0.00045</td>
<td>0.00056</td>
<td>0.00068</td>
<td>0.00081</td>
<td>0.00100</td>
<td>0.00120</td>
<td>0.00140</td>
<td>0.00170</td>
</tr>
<tr>
<td>0.00017</td>
<td>0.00031</td>
<td>0.00045</td>
<td>0.00064</td>
<td>0.00083</td>
<td>0.00100</td>
<td>0.00120</td>
<td>0.00140</td>
<td>0.00170</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CND5B1</th>
<th>0.00044</th>
<th>0.00036</th>
<th>0.00028</th>
<th>0.00018</th>
<th>0.00011</th>
<th>0.00007</th>
<th>0.00003</th>
<th>0.00000</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CND5B2</th>
<th>0.00074</th>
<th>0.00064</th>
<th>0.00052</th>
<th>0.00041</th>
<th>0.00030</th>
<th>0.00022</th>
<th>0.00014</th>
<th>0.00006</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>.006</td>
<td>.011</td>
<td>.016</td>
<td>.024</td>
<td>.034</td>
<td>.048</td>
<td>.057</td>
<td>.070</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>.113</td>
<td>.006</td>
<td>.011</td>
<td>.016</td>
<td>.024</td>
<td>.034</td>
<td>.048</td>
<td>.057</td>
<td></td>
</tr>
<tr>
<td>.070</td>
<td>.113</td>
<td>.006</td>
<td>.011</td>
<td>.016</td>
<td>.024</td>
<td>.034</td>
<td>.048</td>
<td>.057</td>
</tr>
</tbody>
</table>

**CCDBF1**

<table>
<thead>
<tr>
<th>63</th>
<th>63</th>
<th>63</th>
</tr>
</thead>
<tbody>
<tr>
<td>-.00024</td>
<td>-.00020</td>
<td>-.00015</td>
</tr>
<tr>
<td>-.00038</td>
<td>-.00009</td>
<td>-.00006</td>
</tr>
<tr>
<td>-.00028</td>
<td>-.00042</td>
<td>-.00000</td>
</tr>
<tr>
<td>-.00013</td>
<td>-.00019</td>
<td>-.00032</td>
</tr>
<tr>
<td>-.00008</td>
<td>-.00010</td>
<td>-.00016</td>
</tr>
<tr>
<td>-.00000</td>
<td>-.00003</td>
<td>-.00005</td>
</tr>
<tr>
<td>-.00000</td>
<td>-.00000</td>
<td>-.00000</td>
</tr>
</tbody>
</table>

**CCDBF2**

<table>
<thead>
<tr>
<th>63</th>
<th>63</th>
<th>63</th>
</tr>
</thead>
<tbody>
<tr>
<td>-.00048</td>
<td>-.00055</td>
<td>-.00061</td>
</tr>
<tr>
<td>-.00144</td>
<td>-.00268</td>
<td>-.00035</td>
</tr>
<tr>
<td>-.00100</td>
<td>-.00121</td>
<td>-.00010</td>
</tr>
<tr>
<td>-.00073</td>
<td>-.00102</td>
<td>-.00012</td>
</tr>
<tr>
<td>-.00053</td>
<td>-.00069</td>
<td>-.00013</td>
</tr>
<tr>
<td>-.00034</td>
<td>-.00050</td>
<td>-.00006</td>
</tr>
<tr>
<td>-.00024</td>
<td>-.00030</td>
<td>-.00004</td>
</tr>
<tr>
<td>-.00017</td>
<td>-.00023</td>
<td>-.00002</td>
</tr>
</tbody>
</table>

**CCDBS1**

<table>
<thead>
<tr>
<th>63</th>
<th>63</th>
<th>63</th>
</tr>
</thead>
<tbody>
<tr>
<td>-.00075</td>
<td>-.00070</td>
<td>-.00066</td>
</tr>
<tr>
<td>-.00000</td>
<td>-.00042</td>
<td>-.00042</td>
</tr>
<tr>
<td>-.00021</td>
<td>-.00014</td>
<td>-.00031</td>
</tr>
<tr>
<td>-.00014</td>
<td>-.00007</td>
<td>-.00001</td>
</tr>
<tr>
<td>-.00009</td>
<td>-.00007</td>
<td>-.00001</td>
</tr>
<tr>
<td>-.00004</td>
<td>-.00003</td>
<td>-.00001</td>
</tr>
<tr>
<td>-.00006</td>
<td>-.00003</td>
<td>-.00002</td>
</tr>
<tr>
<td>-.00008</td>
<td>-.00006</td>
<td>-.00003</td>
</tr>
</tbody>
</table>

**CCDBS2**

<table>
<thead>
<tr>
<th>63</th>
<th>63</th>
<th>63</th>
</tr>
</thead>
<tbody>
<tr>
<td>-.00072</td>
<td>-.00071</td>
<td>-.00067</td>
</tr>
<tr>
<td>-.00004</td>
<td>-.00062</td>
<td>-.00061</td>
</tr>
<tr>
<td>-.00021</td>
<td>-.00008</td>
<td>-.00048</td>
</tr>
<tr>
<td>-.00027</td>
<td>-.00016</td>
<td>-.00003</td>
</tr>
<tr>
<td>-.00018</td>
<td>-.00012</td>
<td>-.00007</td>
</tr>
<tr>
<td>-.00017</td>
<td>-.00011</td>
<td>-.00006</td>
</tr>
<tr>
<td>-.00011</td>
<td>-.00006</td>
<td>-.00005</td>
</tr>
<tr>
<td>-.00016</td>
<td>-.00011</td>
<td>-.00006</td>
</tr>
</tbody>
</table>

**CLB**

<table>
<thead>
<tr>
<th>252</th>
<th>252</th>
<th>252</th>
</tr>
</thead>
<tbody>
<tr>
<td>-.00141</td>
<td>-.00157</td>
<td>-.00161</td>
</tr>
<tr>
<td>-.00092</td>
<td>-.00115</td>
<td>-.00117</td>
</tr>
<tr>
<td>-.00095</td>
<td>-.00079</td>
<td>-.00074</td>
</tr>
<tr>
<td>-.00132</td>
<td>-.00312</td>
<td>-.00342</td>
</tr>
<tr>
<td>-.00118</td>
<td>-.00161</td>
<td>-.00296</td>
</tr>
<tr>
<td>-.00136</td>
<td>-.00143</td>
<td>-.00170</td>
</tr>
<tr>
<td>-.00106</td>
<td>-.00120</td>
<td>-.00145</td>
</tr>
<tr>
<td>-.00087</td>
<td>-.00106</td>
<td>-.00120</td>
</tr>
<tr>
<td>-.00157</td>
<td>-.00161</td>
<td>-.00147</td>
</tr>
<tr>
<td>-.00115</td>
<td>-.00117</td>
<td>-.00136</td>
</tr>
<tr>
<td>-.00079</td>
<td>-.00074</td>
<td>-.00094</td>
</tr>
<tr>
<td>-.00161</td>
<td>-.00296</td>
<td>-.00331</td>
</tr>
<tr>
<td>-.00143</td>
<td>-.00170</td>
<td>-.00267</td>
</tr>
<tr>
<td>-.00120</td>
<td>-.00120</td>
<td>-.00145</td>
</tr>
<tr>
<td>-.00086</td>
<td>-.00098</td>
<td>-.00120</td>
</tr>
<tr>
<td>CLP</td>
<td>63</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>0.320</td>
<td>0.277</td>
<td></td>
</tr>
<tr>
<td>0.000</td>
<td>0.275</td>
<td></td>
</tr>
<tr>
<td>0.000</td>
<td>0.262</td>
<td></td>
</tr>
<tr>
<td>0.175</td>
<td>0.213</td>
<td></td>
</tr>
<tr>
<td>0.180</td>
<td>0.241</td>
<td></td>
</tr>
<tr>
<td>0.238</td>
<td>0.220</td>
<td></td>
</tr>
<tr>
<td>0.190</td>
<td>0.180</td>
<td></td>
</tr>
<tr>
<td>0.149</td>
<td>0.185</td>
<td></td>
</tr>
<tr>
<td>0.140</td>
<td>0.185</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CLR</th>
<th>63</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.086</td>
<td>0.112</td>
</tr>
<tr>
<td>0.160</td>
<td>0.064</td>
</tr>
<tr>
<td>0.112</td>
<td>0.081</td>
</tr>
<tr>
<td>0.069</td>
<td>0.043</td>
</tr>
<tr>
<td>0.048</td>
<td>0.031</td>
</tr>
<tr>
<td>0.046</td>
<td>0.062</td>
</tr>
<tr>
<td>0.041</td>
<td>0.062</td>
</tr>
<tr>
<td>0.035</td>
<td>0.062</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CNB</th>
<th>252</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0110</td>
<td>0.0040</td>
</tr>
<tr>
<td>0.0050</td>
<td>0.0012</td>
</tr>
<tr>
<td>0.0050</td>
<td>0.0008</td>
</tr>
<tr>
<td>0.0050</td>
<td>0.0017</td>
</tr>
<tr>
<td>0.00196</td>
<td>0.00178</td>
</tr>
<tr>
<td>0.00121</td>
<td>0.00180</td>
</tr>
<tr>
<td>0.00112</td>
<td>0.00132</td>
</tr>
<tr>
<td>0.00033</td>
<td>0.00020</td>
</tr>
<tr>
<td>0.00122</td>
<td>0.00039</td>
</tr>
<tr>
<td>0.00500</td>
<td>0.00087</td>
</tr>
<tr>
<td>0.00170</td>
<td>0.00196</td>
</tr>
<tr>
<td>0.00173</td>
<td>0.00180</td>
</tr>
<tr>
<td>0.00132</td>
<td>0.00132</td>
</tr>
<tr>
<td>0.00040</td>
<td>0.00020</td>
</tr>
<tr>
<td>0.00068</td>
<td>0.00039</td>
</tr>
<tr>
<td>0.00087</td>
<td>0.00019</td>
</tr>
<tr>
<td>0.00170</td>
<td>0.00196</td>
</tr>
<tr>
<td>0.00132</td>
<td>0.00132</td>
</tr>
<tr>
<td>0.00020</td>
<td>0.00046</td>
</tr>
<tr>
<td>0.00018</td>
<td>0.00019</td>
</tr>
<tr>
<td>0.00087</td>
<td>0.00066</td>
</tr>
<tr>
<td>0.00170</td>
<td>0.00198</td>
</tr>
<tr>
<td>0.00130</td>
<td>0.00132</td>
</tr>
<tr>
<td>0.00020</td>
<td>0.00046</td>
</tr>
<tr>
<td>0.00016</td>
<td>0.00012</td>
</tr>
<tr>
<td>0.00022</td>
<td>0.00022</td>
</tr>
<tr>
<td>0.00019</td>
<td>0.00019</td>
</tr>
<tr>
<td>0.00018</td>
<td>0.00027</td>
</tr>
<tr>
<td>0.00019</td>
<td>0.00019</td>
</tr>
<tr>
<td>0.00018</td>
<td>0.00018</td>
</tr>
<tr>
<td>0.00017</td>
<td>0.00017</td>
</tr>
<tr>
<td>0.00016</td>
<td>0.00016</td>
</tr>
<tr>
<td>0.00015</td>
<td>0.00015</td>
</tr>
<tr>
<td>0.00014</td>
<td>0.00014</td>
</tr>
<tr>
<td>0.00013</td>
<td>0.00013</td>
</tr>
<tr>
<td>0.00012</td>
<td>0.00012</td>
</tr>
<tr>
<td>0.00011</td>
<td>0.00011</td>
</tr>
<tr>
<td>0.00010</td>
<td>0.00010</td>
</tr>
<tr>
<td>0.00009</td>
<td>0.00009</td>
</tr>
<tr>
<td>0.00008</td>
<td>0.00008</td>
</tr>
<tr>
<td>0.00007</td>
<td>0.00007</td>
</tr>
<tr>
<td>0.00006</td>
<td>0.00006</td>
</tr>
<tr>
<td>0.00005</td>
<td>0.00005</td>
</tr>
<tr>
<td>0.00004</td>
<td>0.00004</td>
</tr>
<tr>
<td>0.00003</td>
<td>0.00003</td>
</tr>
<tr>
<td>0.00002</td>
<td>0.00002</td>
</tr>
<tr>
<td>0.00001</td>
<td>0.00001</td>
</tr>
<tr>
<td>0.00000</td>
<td>0.00000</td>
</tr>
</tbody>
</table>
SUBROUTINE SMILFC
IMPLICIT HEAL *1(A-Z)
INTEGER *2 IAI,IP1,IDE,IGH
LOGICAL*1 DATAYS
COMMON /ARKAY1/DATANRDH(400)
COMMON /ARKAY3/DATASYT(17)
COMMON /ARKAY4/ANGLE(5)
LOGICAL*1 AUTOBF
LOGICAL*1 IC,MJLU
LOGICAL*1 TURNCORD
EQUIVALENCE 
* (DATAROH(3), AY)
* (DATAROH(4), AZ)
* (DATAROH(9), BETA)
* (DATAROH(10), DSB)
* (DATAROH(34), DA)
* (DATAROH(55), DEL)
* (DATAROH(37), PHHCSP)
* (DATAROH(38), DER)
* (DATAROH(40), DM)
* (DATAROH(43), DACM)
* (DATAROH(49), DE)
* (DATAROH(51), DBF)
* (DATAROH(54), DNPJET)
* (DATAROH(55), DMPJET)
* (DATAROH(72), RNHCSP)
* (DATAROH(74), HFTPAS)
* (DATAROH(79), UZCMO)
* (DATAROH(80), UXCMO)
* (DATAROH(83), UYCMO)
* (DATAROH(84), MACT)

150
*, (DATAWK0H(311), SUM2)
*, (DATAWK0H(312), PRU05)
*, (DATAWK0H(313), DMCPF)
*, (DATAWK0H(314), KURC)
*, (DATAWK0H(315), GDRC)
*, (DATAWK0H(316), GNYDRM)
*, (DATAWK0H(317), UMRHR)
*, (DATAWK0H(318), PHOU7)
*, (DATAWK0H(319), GMTK)
*, (DATAWK0H(320), ORHIM)
*, (DATAWK0H(321), SUM3)
*, (DATAWK0H(322), UMCMD)
*, (DATAWK0H(323), DACMC)
*, (DATAWK0H(324), GRXFD)
*, (DATAWK0H(325), GDRE)
*, (DATAWK0H(327), NR)
*, (DATASYT(10), MOLD)
*, (DATASYT(11), IC)
*, (DATASYT(13), URNCORD)
*, (DATASYT(14), AUT4BF)
*, (ANGLE(1), ALFA)
*, (ANGLE(2), PHI)
*, (ANGLE(3), THETA)

DIMENSION
*GPST(4),
*ALPHMIN(4),
*TALPHMAX(4,2),
*TKPIIT(4,2),
*GTHET(4),
*GOSBT(4),
*KUAMLINT(4),
*KDAMPT(I),
*GRST(4),
*GDACT(4),
*GPT(4),
*GNYDORM(4),
*GRAYT(4),
*GDRCT(4),
*GTHXT(4),

DATA DEG2RAD/57.29577951/
DATA GPST/0.4,0.125,2.0,2.5/
DATA ALPHMIN/-4.0,0.0,6.0,8/
DATA TALPHMAX/20.0,15.0,15.0,30.0,6.0,8.0,1.5,3.0/
DATA TKPIIT/15.0,81.5,81.5,60.0,1.0,3.0,3.5,5.0/
DATA GTHET/1.0,0.3,10.0,12.0/
DATA GOSBT/0.1,0.25,1.5,3.0/
DATA KUAMLINT/0.08,0.9324,0.6,1.5/
DATA KDAMPT/0.0636,0.008,0.6,1.5/
DATA GRST/1.0,0.25,0.6,1.5/
DATA GDACT/0.45,2.5,1.15,3.0/
DATA GPT/10.0,1.5,10.0,45.0/
DATA GNYDORM/0.05,0.015,250.0,900.0/
DATA GRAYT/10.0,3.2,2.0,3.0/
DATA GDRCT/1200.0,3600.0,1.15,5.0/
DATA GTHXT/0.2,1.0,2.0,5.0/
DATA DAMAX/20.0/
DATA PRMCUB/1.15/
DATA GPRHCT/0.5/
DATA KPIAUB/1.125/
DATA NL00/2.0/
DATA NGMI/4.0/
DATA OUPCUTOFF/1.0/
DATA OPIJET1/0.0/
DATA OLOPE/0.0/
DATA UOJET1/0.0/
DATA OBDB/0.25/
DATA QADB/0.2/
DATA IGJET12/0.5/
DATA TGJET21/5.0/
DATA GHPAN1/0.5/
DATA GHPAN2/0.5/
DATA GHPANTE/0.5/
DATA GHPANTE/0.6/
DATA GHPANCIL/0.2/
DATA GHPANCIL/1.5/
DATA GQA/1.0/
DATA NZMAX/3.75/
DATA GUNL/4.0/
DATA NZMIN/-1.0/
DATA GSHML/25.0/
DATA GSHML/1.0/
DATA GSHMU/10.0/
DATA GPPANT/2.0/
DATA GPP/1.0/
DATA GINA/0.2/
DATA G/32.174/
DATA RPTAD/1.125/
DATA GBX/1.0/
DATA YBBFILT/1.0/
DATA OBFAD/0.5/
DATA OBFDD/1.0/
DATA KURMLIN/0.0131/
DATA KURMPAR/0.042/
DATA DBFMA/1.0/
DATA DBFMB/2.0/
DATA DFADH/0.9/
DATA OFKHD/1.0/
DATA PCHD/0.2/
DATA PEAD/0.1/
DATA NH1/1.0/
DATA NH2/2.0/
DATA NH3/3.0/
DATA NH4/4.0/
DATA MHCSA/-2.0/
DATA DBFUCS1/0.0/
DATA DBFDC/0.0/
DATA OLD25/0.0/
DATA OSHC1/0.0/
DATA KHPHIN/0.0/
DATA BCSLN/0.0/
DATA WFFILN/0.0/
DATA NZPN1/NZPN2/0.0/0.0/
DATA PWIND2N/0.0/
DATA PWIND2N/0.0/
DATA BETAN/0.0/
DATA NHIN1/NHIN2/0.0/0.0/
DATA NHIND/0.0/
DATA (IC) GO TO 1000
3 CONTINUE
NYAT/32.174
NZ = "AZ/32,174
ALPHA = ALFA * DEGRAD
CGPUS = 1.293.36 * xBAR + 236.0

PS = BODY * DEGRAD
QS = BODY * DEGRAD
WS = BODY * DEGRAD
SINALF = SIN(ALFA)
SINALF1 = SINALF

IF (SINALF_LI,0.087155) SINALF1 = 0.087155
COSALF = COS(ALFA)
SINPHI = SIN(PHI)
COSPHI = COS(PHI)

COSTHETA = COS(THETA)

DETMPAN = 0.0
IF (PITCHPAN,LTE,0.5) DETMPAN = 1.0
IF (PITCHPAN_GTE,0.5) DETMPAN = -1.0
DETMKC = 0.0
IF (PITCHSTK,LTE,0.5) DETMKC = 1.0
IF (PITCHSTK_GTE,0.5) DETMKC = -1.0
DATMPAN = 0.0
IF (ROLLPAN_GTE,0.5) DATMPAN = 1.0
IF (ROLLPAN_LTE,-0.5) DATMPAN = -1.0

DATMPIC = 0.0
IF (YAWTHM_GTE,0.5) DATMPIC = 1.0
IF (YAWTHM_LTE,-0.5) DATMPIC = 1.0

* THE FOLLOWING SECTION CALCULATES ALL TABLE LOOKUP FUNCTION VALUES *

GPS = FUNCTION (MACH, GPST)
GJET = 0.0
IF (UBAM_GTE,TGJET12) GJET_TGJET21
CALL POINTF (MACH, 1, IAT, FRAC, TALPHMAX (1,2))
ALPHMAX = TALPHMAX (1,1) + (TALPHMAX (1AT +1,1) - TALPHMAX (1AT,1)) * FRAC
ALPHMIN = FUNCTION (MACH, ALPHMIN)
CALL POINTF (MACH, 4, IPT, FRAC, TKPIT (1,2))
KPIT = TKPIT (IPT,1) + (TKPIT (IPT +1,1) - TKPIT (IPT,1)) * FRAC
GTHET = FUNCTION (MACH, GTHET)
GDSB = FUNCTION (MACH, GDSBT)
KDAMPK = FUNCTION (MACH, KDAMPK)
KDAMLINT = FUNCTION (MACH, KDAMLINT)
GRS = FUNCTION (MACH, GRST)
GOACT = FUNCTION (MACH, GDAC1)
GP = FUNCTION (UBAR, GPT)
GNYTM = FUNCTION (TAS, GNYTMRT)
GMAY = FUNCTION (MACH, GMAY)
KDHC = FUNCTION (MACH, GUNCI)
GTH = FUNCTION (MACH, GTHRT)

* PITCH FCS PROCESSING *
* DEMSP *
DEMAN = 0
IF (PRHCSOP_LI, PRHCD8) DEMAN = PRHCSOP + PRHCD8

155
IF(PWMCSOP.GT.PWMCDB)DEMAN=PWMCDB
ESHAPE=(0.36+0.0484*ABS(DEMAN))*DEMAN
IF(ESHAPE.GT.23.0)ESHAPE=23.0
IF(ESHAPE.LT.-23.0)ESHAPE=-23.0
DEMS=ESHAPE*GAPS
DEMS=DEMS*DEMS
IF((MACH,GT,1.5).AND.(PWMCSOP.GT,19.5))DEMS=DEMS*2.0

*DETR
20 DETRGPN=DETRMHC*GPNMCT
IF(IC,OR,MULD) GO TO 29
DETR=FLUP1(DETRGPN,.04,.0,.0,1.0,DETRN)
IF(DETR.GT.1.5)DETR=1.5
IF(DETR.LT.-1.5)DETR=-1.5
29 CONTINUE

*DPJET
DPJET=DEMS*Q

*UVCMD
AHSUPJET=ABS(DPJET)
NG=NGHI
IF(AHSUPJET.LE.DPCTOUFF) NG=NGLO
CALL HYSTP1(GADB,QBDM,DPJET1,DPJET,OUTDPJET)
DPJET1=DPJET
UVCMD=NG*OUTDPJET
IF((MACH,LE,1.5).OR.(QBAR,GT,20.0))UVCMD=0.0

*UYGJET
50 UYGJET=GJET*UVCMD

*RTANP
TANPHI=SINPHI/COSPHI
IF(TANPHI,GT,1.0)TANPHI=1.0
IF(TANPHI,LT,-1.0)TANPHI=-1.0
60 RTANPHI=RTANPHI
IF(HOLD) GO TO 62
RTANPHI=FLIP1(RTANPHI,.01961,.01961,-.9608,RTANPHI)
62 CONTINUE

*BCLS
IF(MACH,GT,1.5)RTANPHI=0.0
70 DJRTP=DPJET1-RTANPHI-DETR
IF(HOLD) GO TO 72
BCLS=FILT1(DJRTP,.9615,-1.8846,.9231,BCLS)
72 CONTINUE

*DCSSLU
IF(HOLD) GO TO 74
WFFIL=FILT1(W,.02988,.02988,-.992,WFFILN)
74 CONTINUE
DCSUL=WFFIL*GWMNGP
DWLOA=(ALPHA-ALPHMAX)*GQA
NZP=NZ+1.8649*UDOT
IF(HOLD) GO TO 76
NZP=FILT2(NZP,.06475,.1295,.06475,-1.3094,.5683,NZPN1,NZPN2)
76 CONTINUE
DQLUN=(NZP,NZMAX)*GWNL
DGLU=DWLOA
IF(DQLUN,GT,DWLDA) DULU=DQLUN
80 DCSL=DLU+DCSW

*DCSHMI
DQMIA=(ALPHA-ALPHMIN)*GQA
DQMIA=(NZP,NZMIN)*GWNL
DQMIN=DQMIA
IF(DQMIA,LE,DQMIA)DQMI=DQMIA
90 DCSLMI=DQHI+DCSW
*MIDPICK
   IF(DCSLL0,GE,DCSLMI) GO TO 98
   IF(DCSLHI,LT,BCSL) GO TO 91
   GO TO 93
91 IF(DCSLHI,GT,DCSLL0) GO TO 97
   GO TO 98
93 IF(BCSL,GE,DCSLL0) GO TO 98.
   GO TO 99
97 MIDPICK=DCSLHI
   GO TO 100
98 MIDPICK=BCSL
   GO TO 100
99 MIDPICK=DCSLL0

*DCSL
100 IF(MACH,GT,1.5)GO TO 105
   DCSL=MIDPICK
   GO TO 110
105 IF(QBAR,GT,2.0) GO TO 106
   DCSL=0.0
   GO TO 110
106 DCSL=BCSL

*DUCT
110 CONTINUE
   GDW=KPI/T/STM(QBAR+4.0)
   IF(GDW,GE,6.0)GDW=6.0
   IF(GDW,LE,0.2)GDW=0.2
   DUCT=DCSL*GDW

*GTREDUCT
   GTREDUCT=GTRE*DUCT

*DSBXTNS
   DSBXTS=DSBPC*GOSH
   IF(DSB,GE,30.0) GO TO 115
   DSBXTS=DSBXTK*GSBLO
   GO TO 120
115 DSBXTS=DSBXTS*GSBH1

*DETP
120 DETP=DETMPAN*GPPANT
   DETDSB=DSBXTNS-DETP

*ETRIM
   ETRIM=GTFREDUCT-DETP=UYGJET
   IF(IC,OR,HOLD) GO TO 121
   ETRIM=FIL11(ETRIM,.04,0.0,-1.0,ETRIM)
121 CONTINUE
   IF(ETRIM,GE,20.0)ETRIM=20.0
   IF(ETRIM,LE,-35.0)ETRIM=-35.0

*ELFBK
   ELFBKIN=(DETDOSB/GTRE)+DE
   IF(IC,OR,HOLD) GO TO 122
   TEMP=50.0/GTRE
   ELFBK=FIL11(ELFBKIN,1.0/(TEMP+1.0),1.0/(TEMP+1.0),
   *(+TEMP+1.0)/(TEMP+1.0),ELFBKN)
122 CONTINUE

*DETRIM
   IF(MACH,LE,12.0) GO TO 125
   DETRIM=ETRIM
   GO TO 130
125 DETRIM=ELFBK
130 CONTINUE

157
*DECMD
  140 DECMD=DETKIM+DUCT
  IF(DECMD,GT,20.0) DECMD=20.0
  IF(DECMD,LT,-35.0) DECMD=-35.0
*
* END OF PITCH PROCESSING
*
* ROLL FCS PROCESSING, PART 1
*
* DAMAN
  DAMAN=0.0
  IF(RHCSOP,LT,-1.15) DAMAN=RHCSOP+1.15
  IF(RHCSOP,GT,1.15) DAMAN=RHCSOP+1.15
*
* PROD21
  PROD21=DAMAN*(KDAMLIN+KDAMPAX*ABS(DAMAN))
  IF(PROD21,GT,DAMAX)PROD21=DAMAX
  IF(PROD21,LT,-DAMAX)PROD21=-DAMAX
  IF(HOLD) GO TO 155
  PROD21=FIT11(PROD21,.0909,.0909,=.8182,PROD22N)
  155 CONTINUE
*
* GNS
  PROD23=GNS*PROD22
  IF(MACH,GT,1.5) PROD24=DAMAXKHC*.6
  IF(MACH,LE,1.5) PROD24=DAMAXKHC*.2
  IF(C,OK,HOLD) GO TO 171
  PROD25=FIT11(PROD24,.04,.0,-1.0,PROD25N)
  171 CONTINUE
  IF(PROD25,GT,2.5)PROD25=2.5
  IF(PROD25,LT,-2.5) PROD25=-2.5
  SUM21=PROD25+PROD23
  PROD27=SUM21*.2,0
  IF(PROD27,GT,10.0)PROD27=10.0
  IF(PROD27,LT,-10.0)PROD27=-10.0
  PROD27=PROD27+SINALF
*
* DRPHI
  DRPHI=57.3*G*SINPHI*COSTHETA/TAS
  IF(TNJNCURV)GOTO 174
  DRPHI=0
*
* DRPKM
  174 DRPKM=R-DRPHI
  IF(HOLD) GO TO 178
  PROD24=FIT11(DRPKM,.9804,.9804,.9608,PROD29N)
  178 CONTINUE
*
* GNM
  GNM=28.8/(QBAAK+10.0)
  IF(GNM,LT,.05) GNM=.05
  IF(HOLD) GO TO 185
  BETAFILT=FIT11(BETA,.999,.999,.998,BETAN)
  185 CONTINUE
  BETAK=BETAFILT
  IF(QBAR,LE,2.0)BETAK=BETA
  IF(QBAR,GT,20.0)BETAK=0,0
  SUM22=COSALF*BETAK=PROD29*GRM=2*DPREM
  SUM23=PROD26+SUM22
*
* RSTAB
  RSTAB=DRPKM*COSALF=P*SINALF
*
* PSTAT
  PSTAT=DRPKM*SINALF+P*COSALF
  PSTAT=0=PSTAT*GPP

158
END OF ROLL, PART I PROCESSING

YAW FCS PROCESSING

DRMAN

DRMAN=0.0
IF (9PTASUP.LT.1.125) DRMAN=RPTASOP+1.125
IF (9PTASOP.GT.1.125) DRMAN=9PTASOP+1.125

DRMS = (KDRLN+KDRRPA*ABS(DRMAN))*DRMAN
IF (DRMS.GT.22.5) DRMS=22.5
IF (DRMS.LT.-22.5) DRMS=-22.5

DRTGDN=DRT*GDRT1
IF (IC.MH.HOLD) GO TO 221
DRTMSF=FILT1(DRTGDN,.04,0,1,0,DRTMSF)
221 CONTINUE
IF (DRTMSF.LT.2.0) DRTMSF=2.0
IF (DRTMSF.GT.2.0) DRTMSF=2.0
UHTMS=DRTMSF+DRMS

SUM1

PROD1= GNY*DRM*DRMS
IF (TUMNCURD) GO TO 225
SUM1=PROD1
GOTO230
225 NYA#NY+ROD1*1.8649
IF (HOLD) GO TO 232
NYP=FILT2(NYA,.06475,.1295,.06475,.1309,.5683,NYP1,NYP2)
SUM1=NYP#PROD1
230 PROD2=FILT1(SUM1,.0909,.0909,.8182,PROD2N)
232 CONTINUE

PROD3=GRAY*PROD2

SUM2

SUM2=PROD3+KSTAB

GDRC

GDRC=KDRC/(QBAR+80)
IF (GDRC.GT.15.0) GDRC=15.0
IF (GDRC.LT.1.2) GDRC=1.2

234 PROD5=SUM2*GDRC
IF (MACH.LE.1.5) GOTO 235

GDRE (EARLY)

GDRE=.000,.0/QBAR
IF (GDRE.GT.1.0) GDRE=1.0
IF (GDRE.LT.5.0) GDRE=5.0
PROD7=GDRE*SUM23
DRCPF=YB8FILT*PROD7
IF (MACH.LE.5.0) GOTO 236
PROD7=0.0
DRCPF=0.0
GOTO 236

235 DRCPF=YB8FILT*PROD5
236 CONTINUE

DTRR

DTRR=UNCPF*GTRK
IF (IC.OR.HOLD) GO TO 244
DRTRIM=FLT1(DRTRIM,0,4,0,=1,0,DRTRIMN)

244 CONTINUE
IF(DRTRIM,GT,9,0)DRTRIM=9,0
IF(DRTRIM,LT,9,0)DRTRIM=9,0
GRXFD=150/WBAR
IF(GRXFD,GT,1,0)GRXFD=1,0
IF(GRXFD,LT,1,1)GRXFD=1
DACMC=0
IF((MACH,LE,1,5),AND,(TURNCOND)) DACMC=GRXFD*DACMC
SUM3=DRCPF+DRTRIM+DACMC

* DRCMD
IF(SUM3,GT,22,8)SUM3=22,8
IF(SUM3,LT,-22,8)SUM3=-22,8
DRCMD=SUM3
DRJET=SUM23
IF(MACH,GT,1,5)GO TO 262

* DKJET
DRJET=0
IF(MACH,GT,1,0)DKJET=GRCSA*SUM2

* NR
* YA
* YB
262 ABSDRJET=ABS(DKJET)
IF(QBAH,LE,2,0)GO TO 263
NR=NR1
IF(ABSDRJET,GT,5,0)NR=NR2
IF(ABSDRJET,GT,1,0)NR=NR3
IF(ABSDRJET,GT,1,5)NR=NR4
GO TO 264
263 NR=NR1
IF(ABSDRJET,GT,5,0)NR=NR2
264 YADB=.1
YDB=.2
IF(MACH,GT,1,5)GO TO 265
YADB=.5
YDB=.8
265 CALL HYSTER(YADH,YDB,DRJET1,DRJET,OUTDRJ1)
DRJET1=DRJET
UZCMD=NR*OUTDRJ

* END OF YAK PROCESSING
* NULL FCS PROCESSING, PART 2
* PE
COTALF=COSALF/SINALF
IF(COTALF,GT,11,43)COTALF=11,43
IF(COTALF,LT,-11,43)COTALF=-11,43
PE=DKPRM*COTALF=BETA*SINALF=P
ABSPE=ABS(PE)

* YAXXXFEED
CALL HYSTER(PEADB,PEBDB,OLDPE,PE,PEOUT)
OLDPE=PE
YAXXXFEED=PEOUT*ABSPE
PROD41=0
IF(QBAH,GT,2)PROD41=GP*YAXXXFEED

* DLSP
IF(MACH,GT,1,5)GO TO 270
GDA=200/(WBAR+80)
IF(GDA.LT.1) GDA=1
IF(GDA.GT.(1.2/GDAC)) GDA=1.2/GDAC
DCSP=GDA*DACM
GO TO 275

270 CONTINUE
GDA=150/QBAH
IF(GDA.LT.1) GDA=1
IF(GDA.GT.1) GDA=1
DCSP=GDA*PRD41

275 CONTINUE
PROD42=DCSP*GTRA

* UXCMD
NP=2
IF(ABSPE.GT.1) NP=4
UXCMD=0
IF((QBAR.LE.10).AND.(MACH.GT.1.5)) UXCMD=NP*PEOUT

* PROD43
SUM25=SIGN(1.0,UXCMU)=SIGN(1.0,UXCMD)
SUM25=SIGN(1.0,SUM25)
ABS25=ABS(SUM25S)
SUM26=SUM25=UL025
UL025=SUM25S
PROD43=SIGN(1.0,SUM26)*ABS25

* GTX,PROD44
GTX=1.25
IF(MACH.GT.10) GTX=50/(QBAH+10)
IF(GTX.LT.1.25) GTX=1.25
IF(GTX.GT.5) GTX=5
PROD44=PROD43*GTX

* DATR
DATR=0
IF(MACH.LE.1.5) DATR=PRD42
IF((MACH.GT.5).AND.(QBAR.GT.2)) DATR=PRD44

* DATP
DATP=DATMPAN*.5
IF(MACH.GT.1.5) DATP=5*DATMPAN

* DATSUMI
DATSUMI=DATK+DATP
IF(1C.UK,MOLD) GO TO 300
DATSUMI=FLATI(DATSUM,.04,.0,.1,.0,DATSUMN)

300 CONTINUE

* DACMD
DACMD=DCSP+DATSUMI
IF(DACMD.GT.10) DACMD=10
IF(DACMD.LT.-10) DACMD=-10

* END OF MOLL PROCESSING

* ACTUATOR PROCESSING FOLLOWS

* DCDRM
DECRRH=((DECRM-DACMD)-DER)*20.0
IF(DECRRH.GE.20.0) DECRRH=20.0
IF(DECRRH.LE.-20.0) DECRRH=-20.0

* DER
430 DER=.04*DECDRRH+DER
IF(DER.GE.20.0) DER=20.0
IF(DER.LT.-35.0) DER=-35.0
* DEC0DL
433 DEC0DL=((DECMD+DACMD)-DEL)*20.0
  IF(DEC0DL,GE,20.0)DEC0DL=20.0
  IF(DEC0DL,LE,-20.0)DEC0DL=-20.0
* DEL
440 DEL=.04*DEC0DL+DEL
  IF(DEL,GT,20.0)DEL=20.0
  IF(DEL,LT,-35.0)DEL=-35.0
* DE AND DA
  DE=(DEL+DEK)/2.0
  DA=(DEL-DEK)/2.0
* DSBK
450 DSBP=SBKP=USBK1
  IF(DSBP,GT,6.1)USBP=6.1
  IF(DSBP,LT,-10.86)USBP=-10.86
  USBK1=DSBK1+USBK1
  USBK1=5*USBK1
* DCDR
  DCDHR=((DCM+DSBK)-DRKP)*10.0
  IF(DCDHR,GE,10.0)DCDHR=10.0
  IF(DCDHR,LT,-10.0)DCDHR=-10.0
* DRRP
460 DRRP=.04*DCDHR+DRKP
  IF(DRRP,GT,54.88)DRRP=54.88
  IF(DRRP,LT,-54.88)DRRP=-54.88
* DDC0DL
463 DDC0DL=((DCM+DSBK)-DLRP)*10.0
  IF(DDC0DL,GE,10.0)DDC0DL=10.0
  IF(DDC0DL,LT,-10.0)DDC0DL=-10.0
* DLRP
470 DLRP=.04*DDC0DL+DLRP
  IF(DLRP,GT,54.88)DLRP=54.88
  IF(DLRP,LT,-54.88)DLRP=-54.88
* DR
  DR=DDRP+DLRP)*0.5
* DSBK
  DSBK=ULPK=DRKP
* DBFRC
  IF(DBFRC,GE,3.0)DBFRC=3.0
  IF(DBFRC,LE,-1.0)DBFRC=-1.0
* DBF
490 DBF=DBF+0.04*DBFM
  IF(DBF,GT,22.5)DBF=22.5
  IF(DBF,LT,-11.7)DBF=-11.7
RETURN

* THE FOLLOWING SECTION INITIALIZES ALL INTEGRATOR OUTPUTS AND
  INTERMEDIATE TRANSFER FUNCTION NODES TO ZERO.
* 1000 DETR=EETHM=ELFBK=0
  PH0DZ5=DKIMSF=DRTHLM=0
  DATSUMI=0.0
* THE TRANSFER FUNCTION NODES ARE INITIALIZED TO ZERO HERE.
* DEIKN=0.0
  ETHIM=ELFBKN=0
  PH0DZ5N=DKIMSNF=DRTHIMN=0.0
  DATSUMN=0
GO TO 5
END
************************************************************
**
**
FUNCTION FILT1(XIN,GX1,GX2,GX3,XNODE)
FILT1=XNODE+XIN*GX1
XNODE=XIN*GX2=FILT1*GX3
RETURN
END
************************************************************
**
**
FUNCTION FILT2(XIN,GX1,GX2,GX3,GX4,GX5,XNODE1,XNODE2)
FILT2=XIN*GX1+XNODE1
XNODE1=XIN*GX2=FILT2*GX4+XNODE2
XNODE2=XIN*GX3=FILT2*GX5
RETURN
END
************************************************************
**
**
SUBROUTINE HYSTER (LIM1,LIM2,ARGTO,ARGT1,FUNCT)
REAL LIM1,LIM2

THIS SUBROUTINE PERFORMS THE HYSTERESIS FUNCTION USING
PARAMETERS FROM THE CALLING PROGRAM. THE ARGUMENTS ARE AS FOLLOWS:

LIM1 AND LIM2 ARE THE POSITIVE BREAK POINT VALUES,
(LIM1 LESS THAN LIM2). IT IS ASSUMED THAT THE FUNCTION
IS SYMMETRICAL ABOUT THE ORIGIN.

ARGTO AND ARGT1 ARE THE INPUT VALUES FOR THE TWO MOST
RECENT TIME FRAMES WITH ARGT1 THE MOST RECENT. THE CALLING
ROUTINE MUST CORRECTLY UPDATE THESE VALUES BEFORE CALLING
HYSTER.

FUNCT IS THE OUTPUT VALUE RETURNED TO THE CALLING PROGRAM.
FUNCT IS SET TO -1, 0, OR +1

TEMP1 = ABS(ARGT1)
10 IF (TEMP1 = LIM1) 10,20,20
   10 FUNCT = 0.0
   20 RETURN
900 RETURN
20 IF ( TEM T1 = LIM2 ) 40,30,30
30 IF ( ARGTO ) 60,50,50
50 FUNCT = 1.0
   60 GO TO 900
60 FUNCT = -1.0
   70 GO TO 900
40 IF ( ARGTO ) 80,70,70

163
COME HERE IF INPUT IS BETWEEN -LIM1 AND -LIM2.
CHECK PREVIOUS INPUT VALUE AND SET FUNCTION ACCORDINGLY OR
LEAVE IT UNCHANGED.

80 IF (ARGTO + LIM1) 900,900,10

COME HERE IF INPUT IS BETWEEN LIM1 AND LIM2.
CHECK PREVIOUS INPUT VALUE

70 IF (ARGTO = LIM1) 10,900,900
END

******************************************************************************

FUNCTION

******************************************************************************

C 'FUNCTION' IS USED TO DETERMINE THE OUTPUT OF A FUNCTION SCHEDULE THAT
C HAS MINIMUM, MAXIMUM, AND INTERMEDIATE LINEAR VALUES. THE FUNCTION
C SCHEDULE CAN HAVE A MAXIMUM OF TWO BREAKPOINTS. TABLE(3) IS THE
C SMALLEST ARGUMENT BREAKPOINT VALUE AND TABLE(4) IS THE LARGEST.
C TABLE(1) IS THE OUTPUT VALUE FOR THE CORRESPONDING TABLE(3) (SMALL)
C ARGUMENT AND TABLE(2) IS THE CORRESPONDING VALUE FOR THE TABLE(4)
C (LARGE) ARGUMENT.
FUNCTION FUNCTION(ARG,TABLE)
DIMENSION TABLE(4)
IF(ARG.GT.TABLE(3))GOTO3000
FUNCTION=TABLE(1)
GOTO 3002
3000 IF(ARG.LT.TABLE(4))GOTO 3001
FUNCTION=TABLE(2)
GOTO 3002
3001 FUNCTION=TABLE(1)+(ARG-TABLE(3))*((TABLE(2)-TABLE(1))/(TABLE(4)-
+TABLE(3)))
3002 RETURN

******************************************************************************

******************************************************************************

END