(NASA-CR-168912) DEVELOPMENT OF A SIMPLE, SELF-CONTAINED FLIGHT TEST DATA ACQUISITION SYSTEM Progress Report (Kansas Univ. Center for Research, Inc.) 250 p HC A11/MF A01 CSCL 14B G3/06

N82-23245

Unclas 09881

THE UNIVERSITY OF KANSAS CENTER FOR RESEARCH, INC.

2291 Irving Hill Drive—Campus West
Lawrence, Kansas 66045

Progress Report on

PHASE III:

DEVELOPMENT OF A SIMPLE, SELF-CONTAINED FLIGHT TEST DATA ACQUISITION SYSTEM

KU-FRL-407-7

NASA Grant NSG-4019

Prepared by: Robert Clarke

Douglas Shane

Principal Investigators: Jan Roskam

Dale I. Rummer

Work Performed by: Nancy Fix

Alex Kotsabasis

Mark Mosser

University of Kansas Flight Research Laboratory

Lawrence, Kansas 66045

April 1982

ABSTRACT

The flight test system combines state-of-the-art microprocessor technology and high accuracy instrumentation with parameter identification technology which minimize data and flight time requirements. The system was designed to avoid permanent modifications of the test airplane and allow quick installation. It is capable of longitudinal and lateral-directional stability and control derivative estimation. This report presents details of this system, calibration and flight test procedures, and the results of the Cessna 172 flight test program. The system has proven easy to install, simple to operate, and capable of accurate estimation of stability and control parameters in the Cessna 172 flight tests.

ACKNOWLEDGEMENTS

We would like to thank the personnel of the NASA Dryden Flight Research Facility. Particular thanks go to Mr. Neil Matheny, Technical Monitor, and Mr. Ken Szalai for financial support of this program. Sincere thanks go to our secretary, Mrs. Nancy Hanson, for the many hours of work spent typing this report.

TABLE OF CONTENTS

TABLE OF CONTENTS, continued

TABLE OF CONTENTS, continued

LIST OF SYMBOLS

All parameters in this report are referenced to a system of body axes as shown in Figure C.1.

Symbol	Definition	Dimension
$A = -X$	Force in A direction	lb
A_x, A_X	Longitudinal acceleration	g
A_y, A_Y	Lateral acceleration	g
A_z, A_Z	Vertical acceleration	g
$A_N = -A_Z$	Normal acceleration	g
$[A]$	Stability matrix	
$[B]$	Control matrix	
b	Wing span	ft
$\{c\}$	Vector of unknowns for MMLE	
$\bar{c}$	Mean aerodynamic chord	ft
$C_A = \frac{A}{\bar{q}S} = -C_X$	Coefficient of force in A direction (A = -X)	
$C_{A_\alpha} = \frac{\partial C_A}{\partial \alpha}$	Variation of body A coefficient with angle of attack	rad^{-1}
$C_{A_u} = \frac{\partial C_A}{\partial (\frac{u}{U_1})}$	Variation of body A coefficient with speed	
$C_{A_{\delta_{E,c}}} = \frac{\partial C_A}{\partial \delta_{E,c}}$	Variation of body A coefficient with elevator or canard angle	rad^{-1}
C_{A_o}	Nondimensional longitudinal force equation bias	
$C_D = \frac{D}{\bar{q}S}$	Drag coefficient	

LIST OF SYMBOLS, continued

Symbol	Definition	Dimension
$C_{D_\alpha} = \frac{\partial C_D}{\partial \alpha}$	Variation of drag coefficient with angle of attack	rad^{-1}
$C_{D_u} = \frac{\partial C_D}{\partial (\frac{u}{U_1})}$	Variation of drag coefficient with speed	
$C_{D_{\delta_E}} = \frac{\partial C_D}{\partial \delta_E}$	Variation of drag coefficient with elevator angle	rad^{-1}
$C_L = \frac{L}{\bar{q}S}$	Lift coefficient	
$C_{L_\alpha} = \frac{\partial C_L}{\partial \alpha}$	Variation of lift coefficient with angle of attack	rad^{-1}
$C_{L_{\dot{\alpha}}} = \frac{\partial C_L}{\partial (\frac{\dot{\alpha}\bar{c}}{2U_1})}$	Variation of lift coefficient with rate of change of angle of attack	
$C_{L_q} = \frac{\partial C_L}{\partial (\frac{q\bar{c}}{2U_1})}$	Variation of lift coefficient with pitch rate	
$C_{L_u} = \frac{\partial C_L}{\partial (\frac{u}{U_1})}$	Variation of lift coefficient with speed	
$C_{L_{\delta_E}} = \frac{\partial C_L}{\partial \delta_E}$	Variation of lift coefficient with elevator angle	rad^{-1}
$C_\ell = \frac{L}{\bar{q}\ S\ b}$	Rolling moment coefficient	
$C_{\ell_\beta} = \frac{\partial C_\ell}{\partial \beta}$	Variation of rolling moment coefficient with sideslip angle	rad^{-1}

Symbol	Definition	Dimension
$C_{\ell_p} = \frac{\partial C_\ell}{\partial p}$	Variation of rolling moment coefficient with roll rate	rad^{-1}
$C_{\ell_r} = \frac{\partial C_\ell}{\partial r}$	Variation of rolling moment coefficient with yaw rate	rad^{-1}
$C_{\ell_{\delta_{A,R}}}$	Variation of rolling moment coefficient with aileron or rudder angle	rad^{-1}
$C_m = \frac{M}{\bar{q}S\bar{c}}$	Pitching moment coefficient	
$C_{m_\alpha} = \frac{\partial C_m}{\partial \alpha}$	Variation of pitching moment coefficient with angle of attack	rad^{-1}
$C_{m_{\dot{\alpha}}} = \frac{\partial C_m}{\partial (\frac{\dot{\alpha}\bar{c}}{2U_1})}$	Variation of pitching moment coefficient with rate of change of angle of attack	
$C_{m_q} = \frac{\partial C_m}{\partial (\frac{q\bar{c}}{2U_1})}$	Variation of pitching moment coefficient with pitch rate	
$C_{m_u} = \frac{\partial C_m}{\partial (\frac{u}{U_1})}$	Variation of pitching moment coefficient with speed	
C_{m_T}	Pitching moment coefficient due to thrust	
$C_{m_{T_\alpha}} = \frac{\partial C_{m_T}}{\partial \alpha}$	Variation of thrust pitching moment coefficient with angle of attack	rad^{-1}
$C_{m_{T_u}} = \frac{\partial C_{m_T}}{\partial (\frac{u}{U_1})}$	Variation of thrust pitching moment coefficient with speed	
$C_{m_{\delta_{E,c}}} = \frac{\partial C_m}{\partial \delta_{E,c}}$	Variation of pitching moment coefficient with elevator or canard angle	rad^{-1}

LIST OF SYMBOLS, continued

Symbol	Definition	Dimension
C_{m_o}	Nondimensional pitching moment equation bias	
$C_N = \frac{N}{\bar{q}S} = -C_Z$	Normal force coefficient. (N = -Z)	
$C_{N_\alpha} = \frac{\partial C_N}{\partial \alpha}$	Variation of normal force coefficient with angle of attack	rad^{-1}
$C_{N_u} = \frac{\partial C_N}{\partial (\frac{u}{U_1})}$	Variation of normal force coefficient with speed	
$C_{N_{\delta_{E,c}}} = \frac{\partial C_N}{\partial \delta_{E,c}}$	Variation of normal force coefficient with elevator or canard angle	rad^{-1}
C_{N_o}	Nondimensional normal force equation bias	
$C_n = \frac{N}{\bar{q}\ Sb}$	Yawing moment coefficient	
$C_{n_\beta} = \frac{\partial C_n}{\partial \beta}$	Variation of yawing moment coefficient with sideslip angle	rad^{-1}
C_{n_T}	Yawing moment coefficient due to thrust	
$C_{n_{T_\beta}} = \frac{\partial C_{n_T}}{\partial \beta}$	Variation of thrust yawing moment coefficient with sideslip angle	rad^{-1}
$C_{n_p} = \frac{\partial C_n}{\partial p}$	Variation of yawing moment coefficient with roll rate	rad^{-1}
$C_{n_r} = \frac{\partial C_n}{\partial r}$	Variation of yawing moment coefficient with yaw rate	rad^{-1}
$C_{n_{\delta_{A,R}}} = \frac{\partial C_n}{\partial \delta_{A,R}}$	Variation of yawing moment coefficient with aileron or rudder angle	rad^{-1}

LIST OF SYMBOLS, continued

Symbol	Definition	Dimension
$C_{T_x} = \frac{T_x}{\bar{q}S}$	Thrust force coefficient in X direction	
$C_{T_{x_u}} = \frac{\partial C_{T_x}}{\partial(\frac{u}{U_1})}$	Variation of thrust force coefficient with speed	
$C_X = \frac{X}{\bar{q}S}$	Force coefficient in X direction	
$C_{X_\alpha} = \frac{\partial C_X}{\partial \alpha}$	Variation of longitudinal force coefficient with angle of attack	rad^{-1}
$C_{X_u} = \frac{\partial C_X}{\partial(\frac{u}{U_1})}$	Variation of longitudinal force coefficient with speed	
$C_{X_{\delta_{E,c}}} = \frac{\partial C_X}{\partial \delta_{E,c}}$	Variation of longitudinal force coefficient with elevator or canard angle	rad^{-1}
C_{X_o}	Nondimensional longitudinal force equation bias	
$C_y = \frac{Y}{\bar{q}S} = C_Y$	Force coefficient in Y direction	
$C_{y_\beta} = \frac{\partial C_y}{\partial \beta}$	Variation of side force coefficient with sideslip angle	rad^{-1}
$C_{y_p} = \frac{\partial C_y}{\partial p}$	Variation of side force coefficient with roll rate	rad^{-1}
$C_{y_r} = \frac{\partial C_y}{\partial r}$	Variation of side force coefficient with yaw rate	rad^{-1}
$C_{y_{\delta_{A,R}}} = \frac{\partial C_y}{\partial \delta_{A,R}}$	Variation of side force coefficient with aileron or rudder angle	rad^{-1}

LIST OF SYMBOLS, continued

Symbol	Definition	Dimension
$C_Z = \frac{Z}{\bar{q}S}$	Force coefficient in Z direction	
$C_{Z_\alpha} = \frac{\partial C_Z}{\partial \alpha}$	Variation of vertical force coefficient with angle of attack	rad^{-1}
$C_{Z_u} = \frac{\partial C_Z}{\partial (\frac{u}{U_1})}$	Variation of vertical force coefficient with speed	
$C_{Z_{\delta_{E,c}}} = \frac{\partial C_Z}{\partial \delta_{E,c}}$	Variation of vertical force coefficient with elevator or canard angle	rad^{-1}
C_{Z_o}	Nondimensional vertical force equation bias	
C_c	Y axis force coefficient in wind tunnel axes	
D	Drag force	lb
[D]	MMLE weighting matrix	
g,G	Force of gravity	ft sec^{-2}
[G]	MMLE observation matrix	
[H]	MMLE observation matrix	
H_p	Pressure altitude	ft
[I]	Identity matrix	
I_{xx}, I_{yy}, I_{zz}	Moment of inertia about the X, Y, and Z axes respectively	slug ft^2
I_{xz}	Product of inertia	slug ft^2
J	MMLE cost function	
KTAS	True airspeed	knots
ℓ,L	Rolling moment (perturbed,total)	ft lb
L	Lift force	lb

LIST OF SYMBOLS, continued

Symbol	Definition	Dimension
L	Iteration number	
L_{β}	Dimensional variation of rolling moment with sideslip angle	sec^{-2}
L_p	Dimensional variation of rolling moment with roll rate	sec^{-1}
L_r	Dimensional variation of rolling moment with yaw rate	sec^{-1}
$L_{\delta_{A,R}}$	Dimensional variation of rolling moment with aileron or rudder angle	sec^{-2}
L_o	Rolling moment equation bias	sec^{-2}
m,M	Pitching moment (perturbed,total)	ft lb
m	Mass	slug
MP	Engine manifold pressure	
M_{α}	Dimensional variation of pitching moment with angle of attack	sec^{-2}
$M_{\dot{\alpha}}$	Dimensional variation of pitching moment with rate of change of angle of attack	sec^{-1}
M_q	Dimensional variation of pitching moment with pitch rate	sec^{-1}
M_u	Dimensional variation of pitching moment with speed	$ft^{-1}\ sec^{-1}$
$M_{T_{\alpha}}$	Dimensional variation of pitching moment due to thrust with angle of attack	sec^{-2}
M_{T_u}	Dimensional variation of pitching moment due to thrust with speed	$ft^{-1}\ sec^{-1}$

LIST OF SYMBOLS, continued

Symbol	Definition	Dimension
$M_{\delta_{E,c}}$	Dimensional variation of pitching moment due to elevator or canard angle	sec^{-2}
M_o	Pitching moment equation bias	sec^{-2}
M_θ	Dimensional variation of pitching moment with pitch angle	sec^{-2}
n, N	Yawing moment (perturbed, total)	ft lb
N = -Z	Normal force	lb
N_β	Dimension variation yawing moment with sideslip angle	sec^{-2}
N_{T_β}	Dimensional variation of yawing moment due to thrust with sideslip angle	sec^{-2}
N_p	Dimensional variation of yawing moment with roll rate	sec^{-1}
N_r	Dimensional variation of yawing moment with yaw rate	sec^{-1}
$N_{\delta_{A,R}}$	Dimensional variation of yawing moment with aileron or rudder angle	sec^{-2}
N_o	Yawing moment equation bias	sec^{-2}
p, P	Roll rate	rad sec^{-1}, deg sec^{-1}
P_D	Dynamic pressure	knots (speed) lb ft^{-2}
P_S	Static pressure	ft (altitude) lb ft^{-2}
P_T	Total pressure	lb ft^{-2}
q, Q	Pitch rate	rad sec^{-1}

Symbol	Definition	Dimension
$\bar{q}$	Dynamic pressure	lb ft^{-2}
r,R	Yaw rate	rad sec^{-1} deg sec^{-1}
RPM	Engine rotational speed	
[R]	Acceleration transformation matrix	
S	Wing area	ft^2
t,T	Time point	sec
T	Temperature	°F
T_X	Thrust force in X direction	lb
u,U	Speed (perturbed, total)	ft sec^{-1} mph
{u(t)}	Control vector	
v	Perturbed sideward velocity	ft sec^{-1}
{v}	MMLE variable bias vector	
V_X,V_x	Longitudinal velocity	ft sec^{-1}
V_Y,V_y	Lateral velocity	ft sec^{-1}
V_Z,V_z	Normal velocity	ft sec^{-1}
w	Perturbed downward velocity	ft sec^{-1}
{x(t)}	State vector	
X	Force in X direction	lb
$\bar{X}$	Distance in the X direction from the center of gravity	ft
X_α	Dimensional variation of X-force with angle of attack	ft sec^{-2}
X_u	Dimensional variation of X-force with speed	sec^{-1}
X_{T_u}	Dimensional variation of X-force due to thrust with speed	sec^{-1}

LIST OF SYMBOLS, continued

Symbol	Definition	Dimension
$X_{\delta_{E,c}}$	Dimensional variation of X-force with elevator or canard angle	ft sec^{-2}
X_o	Longitudinal force equation bias	ft sec^{-2}
$\{y(t)\}$	Computed observation vector	
$y_i = \{y(i)\}$	Computed observation vector at time i	
Y	Force in Y direction	lb
$\bar{Y}$	Distance in Y direction from the center of gravity	ft
Y_β	Dimensional variation of Y-force with sideslip angle	sec^{-1}, ft sec^{-2}
Y_p	Dimensional variation of Y-force with roll rate	ft sec^{-1}
Y_r	Dimensional variation of Y-force with yaw rate	ft sec^{-1}
$Y_{\delta_{A,R}}$	Dimensional variation of Y-force with aileron or rudder angle	ft sec^{-2} sec^{-1}
Y_o	Lateral acceleration equation bias	sec^{-1}
$\{z(t)\}$	Measured observation vector	
$z_i = \{z(i)\}$	Measured observation vector at time i	
Z = -N	Force in the Z direction	lb
$\bar{Z}$	Distance in Z direction from the center of gravity	ft
Z_α	Dimensional variation of Z-force with angle of attack	ft sec^{-2}

LIST OF SYMBOLS, continued

Symbol	Definition	Dimension
$Z_{\dot{\alpha}}$	Dimensional variation of Z-force with rate of change of angle of attack	ft sec^{-1}
Z_q	Dimensional variation of Z-force with pitch rate	ft sec^{-1}
Z_u	Dimensional variation of Z-force with speed	sec^{-1}
$Z_{\delta_{E,c}}$	Dimensional variation of Z-force with elevator or canard angle	ft sec^{-2}
Z_o	Vertical force equation bias	ft sec^{-1}

Greek Symbol		
α	Angle of attack	rad
β	Angle of sideslip	rad
ψ	Euler heading angle	rad
θ	Euler pitch angle	deg, rad
ϕ	Euler roll angle	deg, rad
$\dot{\theta}_o$	Bias in Euler pitch rate equation	rad sec^{-1}
δ_E , δ_e	Elevator angle	deg, rad
δ_A , δ_a	Aileron angle	deg, rad
δ_R , δ_r	Rudder angle	deg, rad
δ_c	Canard angle	deg, rad
ρ	Air density	slugs ft^{-3}
$\dot{\phi}_o$	Bias in Euler roll rate equation	rad sec^{-1}
$\omega_{n_{SP}}$	Undamped natural frequency of the short period mode	Hz

Symbol	Definition	Dimension
ω_{n_P}	Undamped natural frequency of the phugoid mode	Hz
ω_{n_D}	Undamped natural frequency of the dutch roll mode	Hz
$\{\eta\ (t)\}$	Noise vector	
∇_c	First gradient with respect to c	
$\nabla_c^{\ 2}$	Second gradient with respect to c	

Subscript	Definition
1	Initial
B	At body axis at center of gravity
M	As measured by transducer
I	As installed wrt body axis at center of gravity
L	Left hand
R	Right hand
,s	Flight stability axes
,w	Wind axes
,wt	Wind tunnel stability axes

Superscript	
†	Transpose
'	State vector derivatives

A dot over a quantity denotes the time derivative of that quantity.

LIST OF FIGURES

LIST OF FIGURES, continued

LIST OF FIGURES, continued

LIST OF TABLES

LIST OF TABLES, continued

1. INTRODUCTION

This report describes the results of work completed in the third phase of a continuing program sponsored by the NASA Dryden Flight Research Facility (NASA-DFRF). Funding was provided under NASA Grant NSG 4019 (CRINC/FRL 4070). This program began on January 21, 1979; and the current phase was completed on March 31, 1982. The program encompassed the design, development, and use of a simple, self-contained flight test data acquisition system. The program was divided into three phases:

Phase I

- A literature survey of flight test techniques (Reference 1).
- The development of a proof-of-concept system capable of longitudinal stability and control parameter estimation (Reference 2).

Phase II

- The development of a complete system capable of longitudinal and lateral-directional stability and control parameter estimation (Reference 3). This phase included sufficient flight testing to prove the flight test system.

Phase III

- The refinement of the completed system for a cleaner installation, overall package size reduction, and additional ease of use.

- The completion of flight tests on a Cessna 172 airplane at various cruise power settings.

This report describes in detail the system components, calibration and analysis procedures, as well as the results of the flight test program.

The purpose of the project is discussed in Chapter 2. Also discussed in that chapter are the design criteria and formal requirements for flight test instrumentation systems. The literature survey (Reference 1) was the primary instrument for establishment of the criteria.

Chapter 3 describes in detail the hardware that was selected and assembled into the final data acquisition system. Copies of the instrument specifications are included.

The data acquisition system is capable of longitudinal and lateral-directional stability and control parameter estimation. The limitations of the system are only those of sensor accuracy and range and limited computer program capability. The instrument ranges selected for this system were applicable to general aviation class airplanes. The computer program used for data analysis is applicable to linear, time invariant equations of motion.

The system was designed with the goals of ease of installation in an airplane and little or no need for permanent modifications to be made to the airplane. The data acquisition system is composed of four major modules:

(1) The power supply module, which consists of an independent battery and voltage regulation system. In this manner the

data acquisition system is totally isolated from the airplane electrical system.

(2) The transducer module, which consists of a package of inertial instruments combined with various external transducers (total temperature probe, to al pressure probe, static pressure probe, and three control position transducers). These external transducers are taped to the airplane with double-sided foam tape.

(3) The data management module, which consists of a micro-computer controller, a 16-channel multiplexing analog-to-digital converter, and a digital cassette recorder. This data system has proven to be a simple, accurate, and reliable method for recording flight test data.

(4) The operator's control module, which consists of the switches which control the data acquisition program as well as an independent digital voltmeter. This voltmeter is used for checking the output of the transducers (both inflight and during ground checkout).

Chapter 4 contains a description of the ground-based computer system and a discussion of the selection procedure that led to the purchase of this computer system.

The data analysis programs and linear airplane math model are discussed in Chapter 5. The heart of the parameter estimation technique used in this project is the Modified Maximum Likelihood Estimation (MMLE) program (Reference 5).

The actual results of the KU flight test program on a Cessna 172 are discussed in Chapter 6. These results are compared against estimated longitudinal and lateral-directional stability derivatives from NASA-Langley flight tests (Reference 4). Also included in this chapter are discussions of the flight test maneuver and test procedures. Figure 1.1 shows the experimental configuration of the tested Cessna 172.

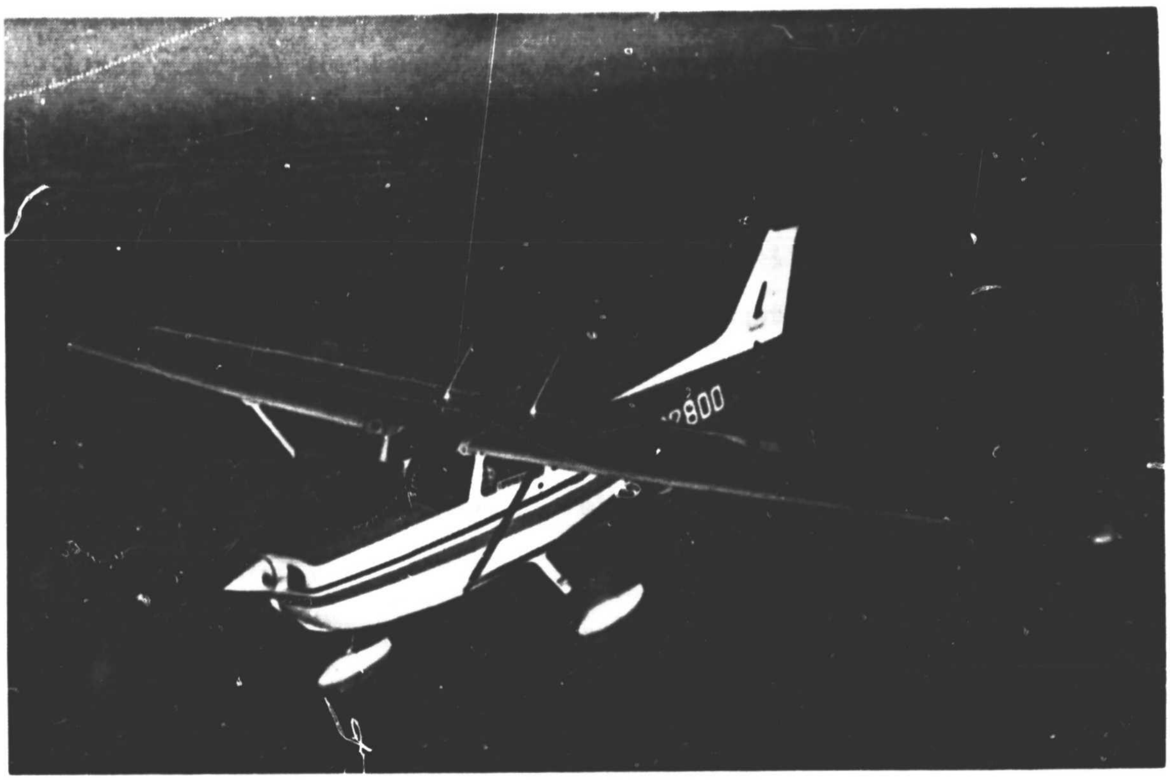


Figure 1.1 Experimental Configuration of the Airplane Tested

Chapter 7 presents the conclusions which were drawn from the experiences using this data acquisition system. Chapter 8 discusses recommendations for further work and for changes which can be made to improve the data acquisition system.

References and a list of all reports and formal papers which have been written about this research are presented in Chapters 9 and 10.

Appendix A has been included for a description and listing of all computer programs used in the flight testing and data analysis procedures.

A comparison of the KU-FRL MMLE output with the NASA-DFRF MMLE output has been included as Appendix B. The test case A (of Reference 5) was run on the KU-FRL computer system and compared with the output listed in Reference 5.

To allow for conversion of the estimated parameters among the various axes systems, Appendix C presents several sets of transformation equations. This appendix is taken directly from Reference 6.

2. PURPOSE OF PROJECT

Flight testing airplanes has previously required a high degree of complex instrumentation to get good results. This is particularly true in stability and control parameter estimation. In general, the normal approach has been to equip each individual flight test airplane with available instruments coupled with those specifically needed for the test program. This nonsystematic approach to the requirements of the flight test program can lead to high cost and needless complexity in the instrumentation system. This complexity typically leads to many man-hours in instrumenting the airplane and in system checkouts.

With recent advances in microcomputer technology, it was thought that an accurate, multipurpose data acquisition system could be developed which was applicable to flight testing airplanes. This system would combine state-of-the-art microcomputer technology with high-accuracy sensors in a systematic approach to the problem of flight testing. The system described within this report has been developed to meet those goals. The design criteria for the system are explained below.

- EASE OF INSTALLATION: This has been a major design consideration; if possible, no permanent modification should be done to the airplane. The system should be easy to install, should require few man-hours for installation, and should require no complex installation procedures. This should include the control surface position transducer (CPT) calibrations which are done after the system is installed.

- SELF-CONTAINED: The system should be totally independent from the airplane. This includes independent power as well as sensors. Likewise minimum ground support should be needed.
- SIMPLE: The system should be simple in operation. Complex instrumentation procedures, difficult calibration techniques, and specialized operator knowledge should be kept to a minimum wherever possible.
- FLIGHT TEST MANEUVER: The system and data analysis procedures should require no specialized piloting techniques to obtain good results.
- CLASS OF AIRCRAFT: The system and analysis methods should be applicable to all classes of aircraft. For this specific application, the transducer ranges and accuracies were chosen for general-aviation-class airplanes.
- FLEXIBILITY: The system and analysis methods have been proven for stability and control parameter estimation; but regardless of this, the system should be adaptable to other test requirements.
- COSTS: The system should meet all of the above objectives at a lower cost than current flight test methods and systems.

3. INSTRUMENTATION SYSTEM

The system designed and developed in this research program meets the criteria listed in Chapter 2. The flight system can be broken into four major parts:

(1) Data Management Computer,

(2) Transducers,

(3) Power Supply,

(4) Operator's Control Box.

A block diagram of the complete flight test package is shown in Figure 3.1. This system is used in two modes:

(1) airborne mode for recording flight data (Figures 3.2 and 3.3),

(2) ground-based mode for transferring flight test data to the data reduction computer (Figures 3.4 and 3.5).

In the following sections the requirements of the flight test data acquisition system, detailed descriptions of the system components, and design trade-offs are discussed.

3.1 Requirements

The requirements of the instrumentation system were accumulated through a literature survey (References 7 through 23) and through discussions with personnel at NASA-DFRF and in the aerospace industry.

Table 3.1 summarizes the sensors used in previous test programs. It also shows the sensors chosen for this system. The number of

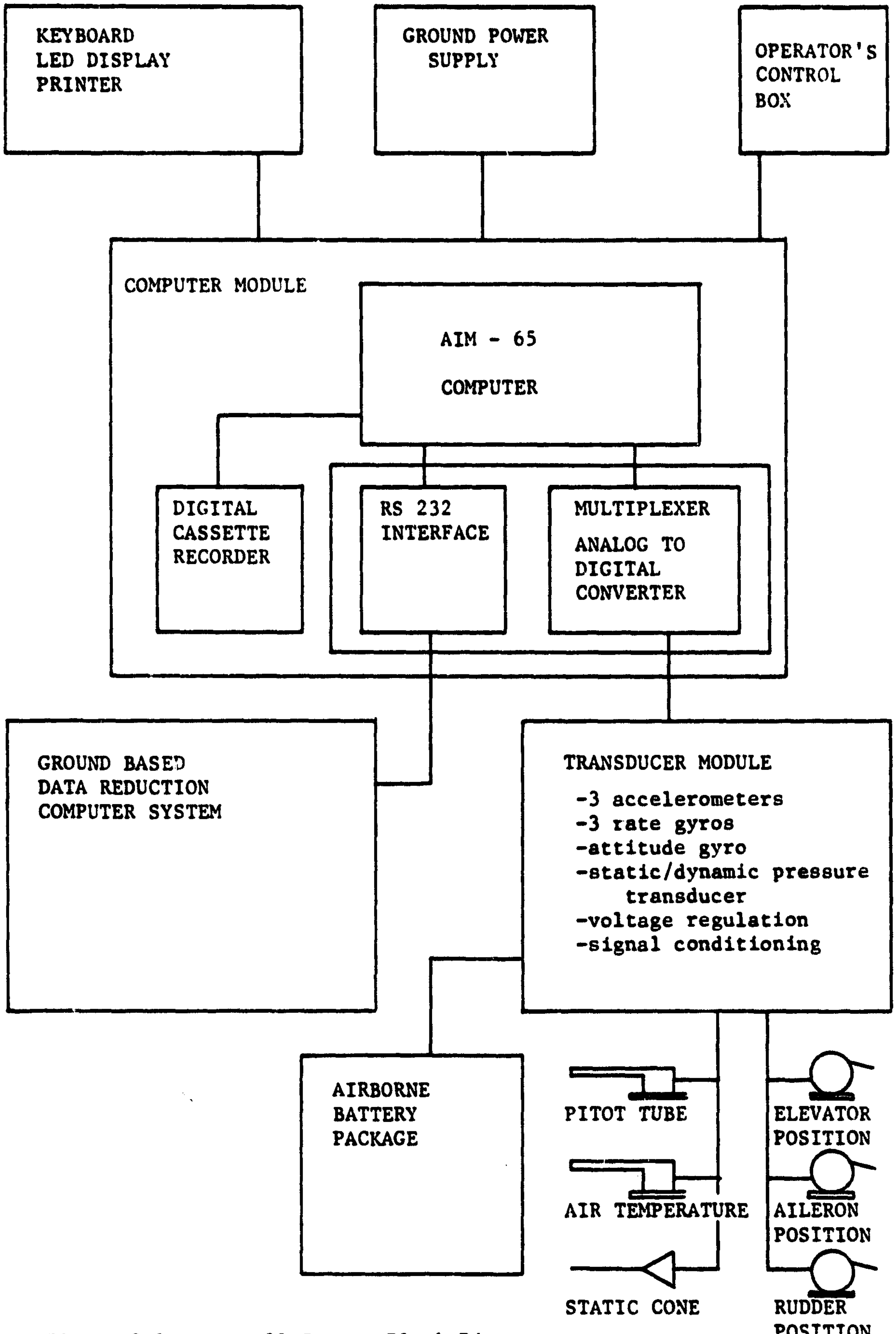
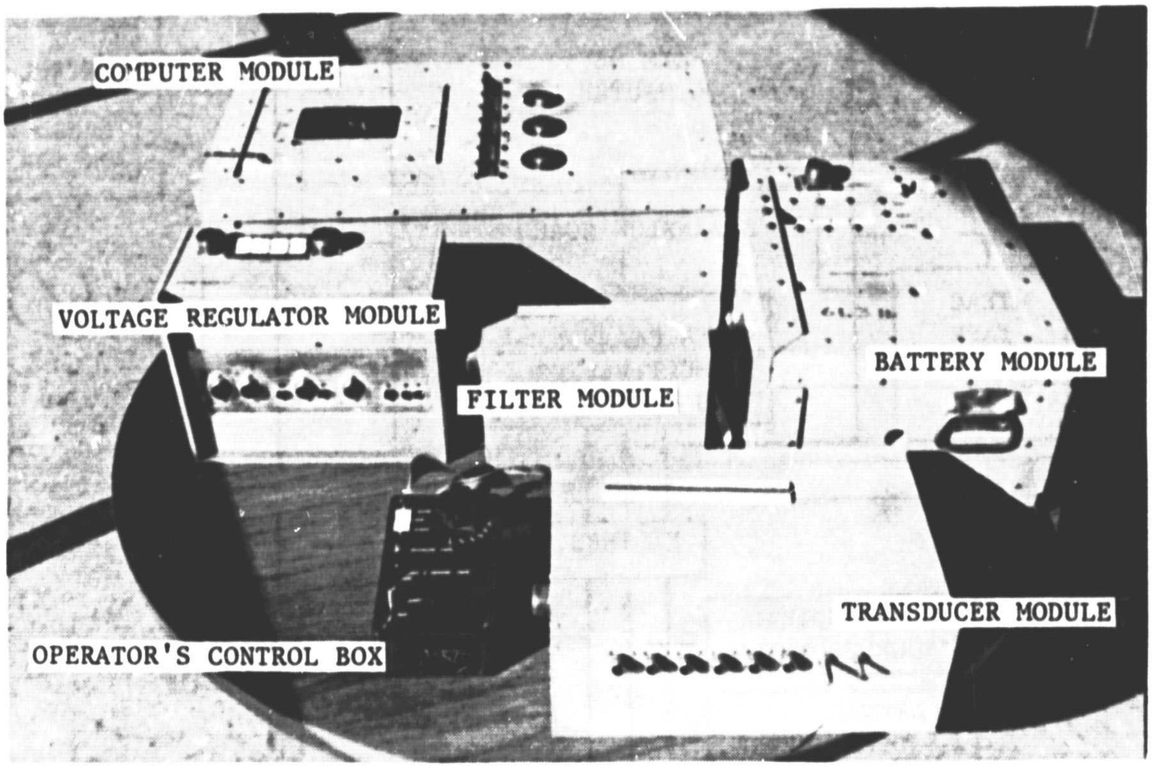


Figure 3.1 Overall System Block Diagram



WEIGHTS

BATTERY MODULE	61.3	(lb)
COMPUTER MODULE	37.3	
TRANSDUCER MODULE	17.0	
VOLTAGE REGULATOR MODULE	9.2	
FILTER MODULE	5.9	
PITOT PROBE*	0.2	
TEMPERATURE PROBE*	0.2	
OPERATOR'S CONTROL BOX	1.0	
MISCELLANEOUS (cables, clamps, etc.)*	2.0	
TOTAL:	134.1	

*not shown

Figure 3.2 Major Components of the Airborne System

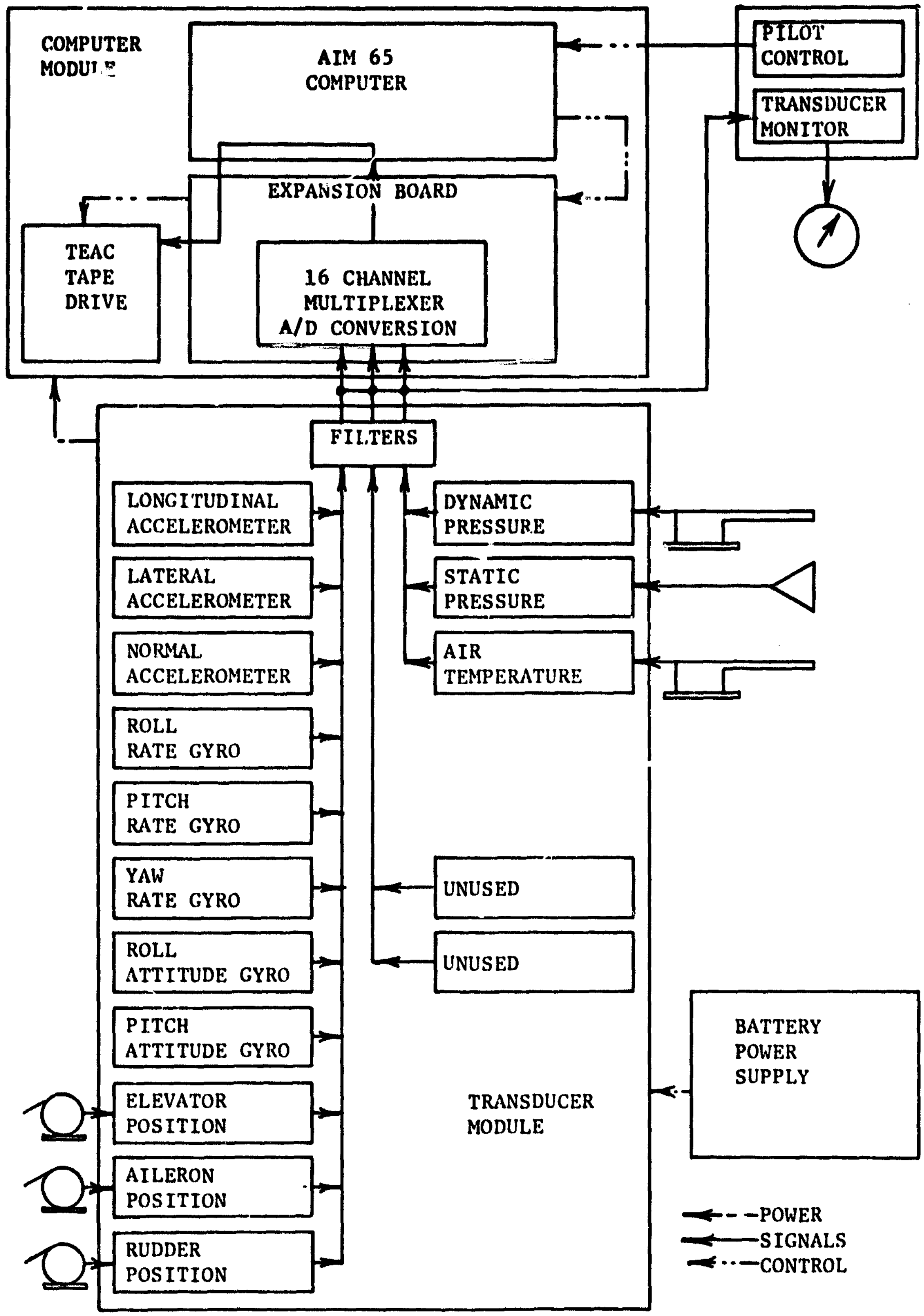


Figure 3.3 Airborne System Block Diagram

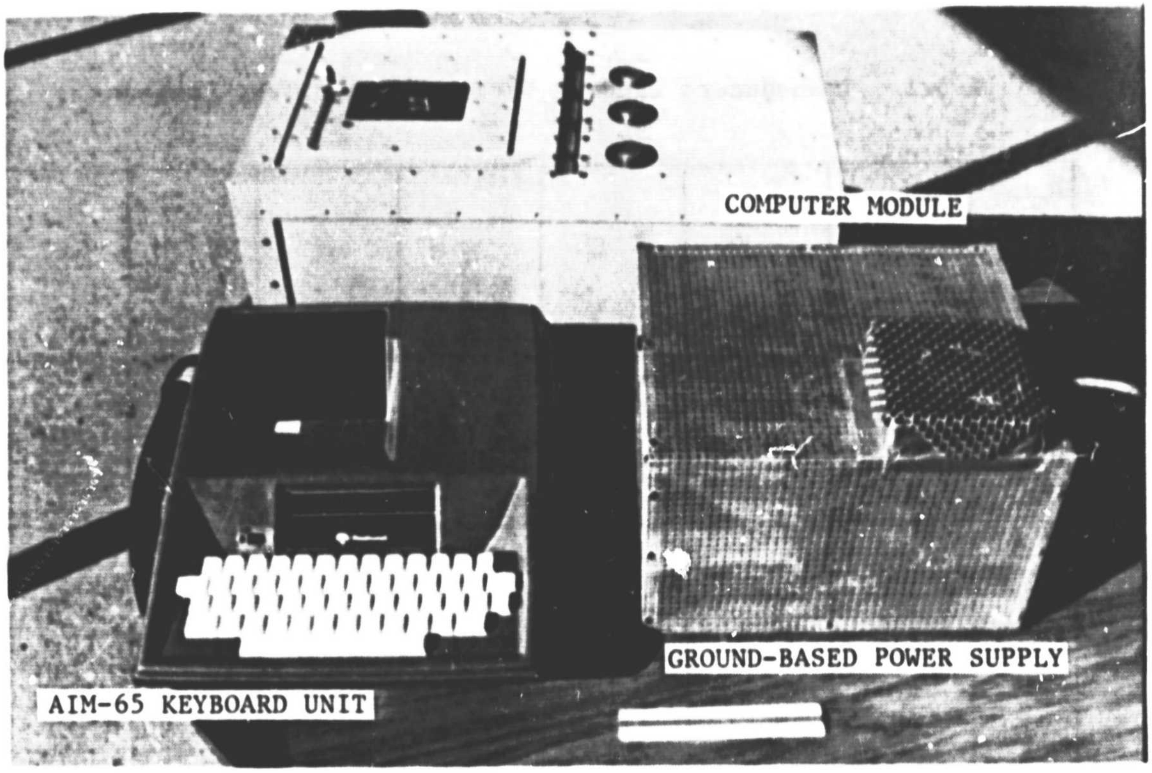


Figure 3.4 Major Components of the Data Transfer System

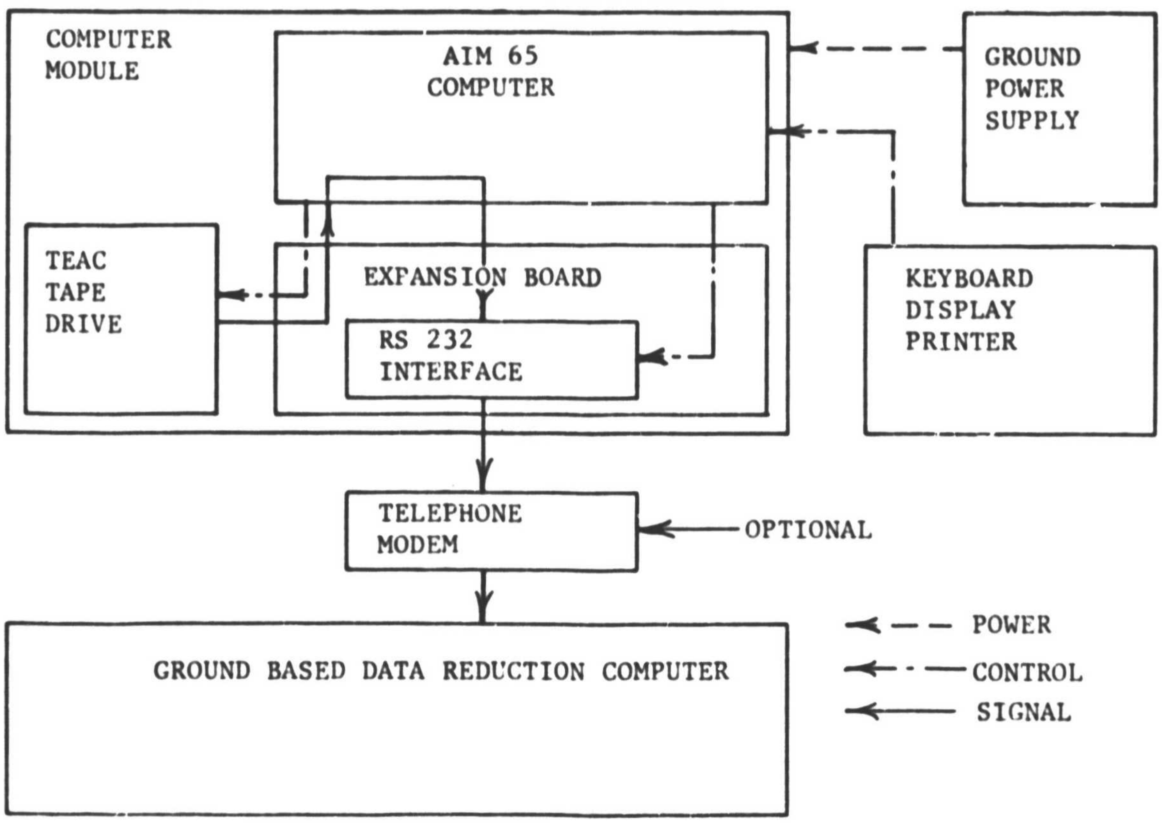


Figure 3.5 Data Transfer System Block Diagram

Table 3.1 Transducers Used in Various Flight Test Programs

	SMETANA Ref. 21	DELFT Ref. 15	SORENSON Ref. 22	BONES PROGRAM Ref. 19	KLEIN Ref. 20	SELECTED	
A_x	*	*	*	*	*	*	
A_y		*	*	*	*	*	
A_z		*	*	*	*	*	
V_x							
V_y							can be derived
V_z							
Alt.							
Temp.		*				*	
θ	*		*	*	*	*	
ϕ		*	*	*	*	*	
ψ		*					not normally needed
p			*	*	*	*	
q	*	*	*	*	*	*	
r		*	*	*	*	*	
$\dot{p}$			*	*			
$\dot{q}$			N.R.	*			can be derived
$\dot{r}$			*	*			
δ_E		*		*	*	*	
δ_A				*	*	*	
δ_R				*	*	*	
RPM		*					may be req'd or desireable for performance data
M.P.		*					
P_S		*			*	*	
P_T							can be derived
P_D	*	*	*	*	*	*	
α	*	*	*	*	*		
β			N.R.	*	*		can be derived
ρ	*						

channels to be recorded was set at fourteen with growth capability to sixteen. The application of the MMLE method (see Chapter 5 for details) does not require that all states be measured to get good parameter estimates of the equations of motion. Because of this fact, one of the important selection criteria for excluding measurement of a state was the complexity of the sensor installation.

The accuracy required of the sensors was determined by study of these other systems. Table 3.2 summarizes this information. The required ranges of these sensors were determined by consideration of the performance characteristics of the class of airplanes to be tested. These ranges and the system accuracies for the selected sensors are summarized in Table 3.3.

For Phase III all transducer channels were measured at the same sample rate. This is quite different from earlier work which limited measurement of temperature, static pressure and dynamic pressure to the beginning and end of each maneuver.

The choice of data sample rate was based on the following considerations:

- The minimum sample rate must be higher than the highest undamped natural frequency of the airplane.
- The minimum sample rate must be high enough to avoid time skewing the data points.
- The minimum sample rate must be high enough to measure the control input completely.
- The minimum sample rate must be as low as possible within the above constraints to allow economy of the recording media.

Table 3.2 Transducer Accuracies Used in Various Flight Test Programs

	DELFT Ref. 15	ECKHOLD & WELLS Ref. 23	KLEIN Ref. 20	ILIFF & MAINE Ref. 5, 16-19	SELECTED
A_x	0.02% of full scale	.002 g	.005 g	0.1 % of full scale	.002 g
A_y		.02 g	or		.002 g
A_z		.02 g	2 %		.002 g
θ		1/2 °	.2° or		.5°
ϕ		1/2 °	2 %		.5°
p		.15°/sec	.2°/sec		.5° /sec
q		.15°/sec	or		.5° /sec
r		.15°/sec	2 %		.5° /sec
δ_E		.4°	.2°		.5°
δ_A		.4°	or		.5°
δ_R		.4°	2 %		.5°
T		2° F			2° F
P_S		10 ft			10 ft
P_D		5 knots	2 knots		2 knots

Table 3.3 Transducer Accuracy and Range Used

Symbol	Sensor	Resolution/ Accuracy*	Range
A_X	longitudinal acceleration	.0020 g	±2 g
A_Y	lateral acceleration	.0020 g	±.5 g
A_Z	normal acceleration	.0024 g	±5 g
θ	pitch angle	.5°	±30°
ϕ	roll angle	.5°	±30°
p	pitch rate	.5°/sec	±50°/sec
q	roll rate	.5°/sec	±50°/sec
r	yaw rate	.5°/sec	±50°/sec
δ_E	elevator position	.5°	
δ_A	aileron position	.5°	
δ_R	rudder position	.5°	
T	temperature	2°F	-65° to 120°F
P_S	static pressure	.010 psia (25 ft)	0 to 25 K feet
P_D	dynamic pressure	.005 psi (4 knots)	40 to 150 knots

*whichever value is larger

For data analysis a sample rate of at least five times the highest undamped natural frequency defines the minimum acceptable sample rate (Reference 24, Volume 1, Chapter 6). In this class of airplanes the following natural frequencies are typical (Reference 25).

$\omega_{n_{SP}}$ 0.05 - 1.00 Hz

ω_{n_P} 0.01 - 0.03 Hz

ω_{n_D} 0.25 - 0.60 Hz

This sets an absolute minimum on the sample rate of five samples per second (SPS). Reference 16 states that to measure the control input time history, 20 SPS should be a lower limit. For this reason, two sample rates were investigated in this phase, 10 and 20 SPS.

The use of a computer-controlled data acquisition system allows rapid scanning of the sensors (20 μsec/channel, 280 μsec total*) which eliminates the time skewing problems normally associated with these slow sample rates.

3.2 Data Management System

The data management module consists of a microprocessor-controlled data acquisition computer system. It uses a commercially available Rockwell AIM-65 microcomputer tied to a 16-channel multiplexing analog-to-digital converter and a digital cassette tape drive. The use of a commercially available computer greatly reduced the design task as well as the overall system complexity and cost. The versatility of the system is high, since the functions of the system are stored in firmware (erasable, programable read only memory--EPROM) which can be easily changed, rather than in the hardware, which cannot. There were many trade-offs in the choice of a system for on-board digital recording of the flight data. The trade-offs considered were those of analog vs. digital data recording and airborne vs. telemetry systems. The following is a discussion of these trade-offs as well as a detailed description of the system which was developed.

*Values for the KU-FRL system.

In the past, most systems which used true on-board recording (ignoring photopanel installations which must be manually recorded at a later time) used analog recording systems. This fact was due to the high cost and complexity of digital systems. In recent years, progress has been made, largely due to advances in silicon chip technology, which have reduced the size and cost of digital systems dramatically. These advances, coupled with the fact that the data analysis techniques usually require digital information, motivated a choice of a totally digital system.

Flight programs in the past have typically used telemetry for transmitting the digital data to the ground for recording. Telemetry has an important place in aircraft flight testing, specifically in high risk operations such as flutter or spin testing. The major disadvantages are the required ground station, high cost, and complexity of the on-board telemetry system. This method has been preferred in the past due to the large size and inaccuracies of older recording methods. The improvements discussed previously combined with advances made in digital recording devices have changed this. For the KU-FRL recording system, an on-board digital cassette recorder has been chosen.

The heart of the data management module is a Rockwell AIM-65 microcomputer. This computer is coupled through its expansion interfaces to the other components. These components are the Datel MDAS-16 multiplexing analog-to-digital converter, the TEAC MT2-02 digital cassette tape transport, and an RS232* interfacing port. A block diagram of these components is shown in Figure 3.6.

---

*RS232 = serial interfacing standard

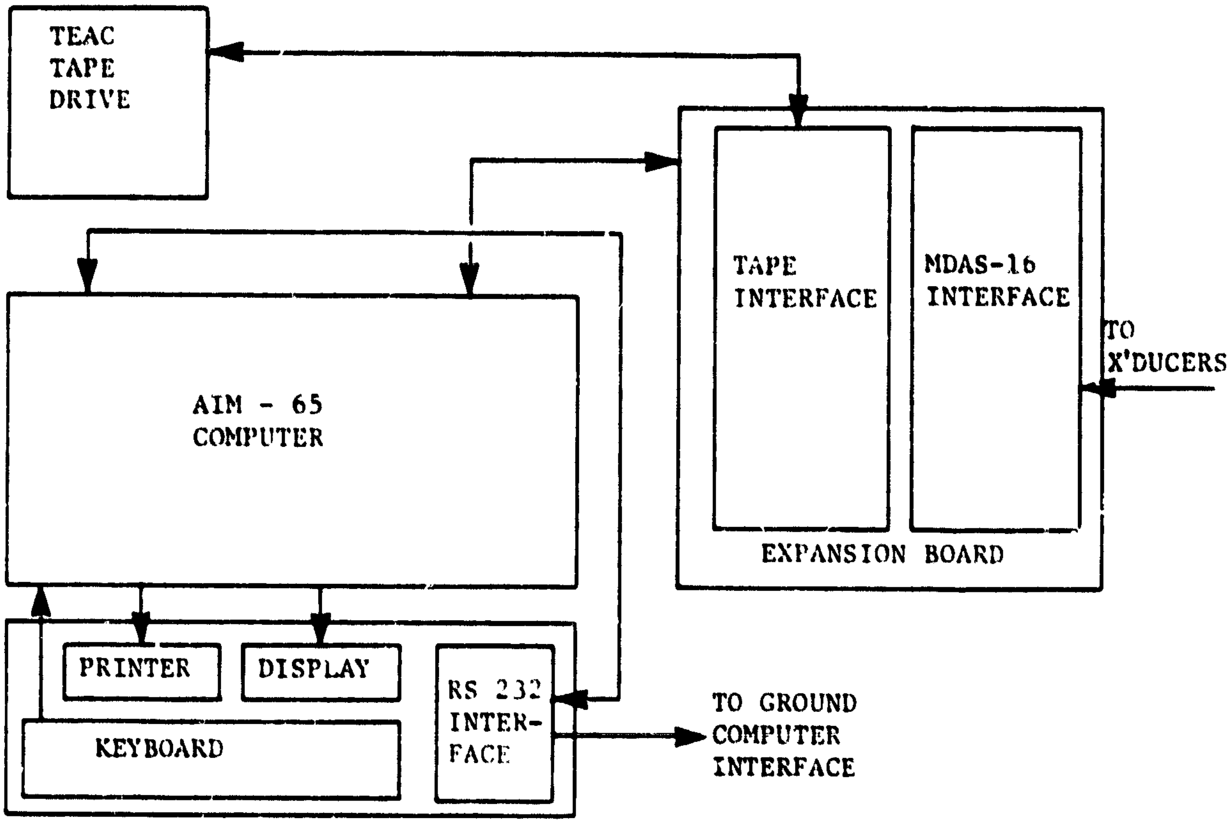


Figure 3.6 Computer Module Block Diagram

The AIM-65 is a single-board computer that uses an eight-bit 6502 microprocessor. The computer contains 4K bytes of random access memory (RAM), 4K bytes of EPROM (which contain the data acquisition programs), as well as a monitor and symbolic assembler. A 20 character display, 20 character printer, and keyboard unit allow the user to interact with the computer in the ground-based modes. Two application connectors greatly increase the versatility of the computer system. One expansion connector interfaces the computer with an RS232 port and to an audio cassett recorder which is used for software development. The second port is used for memory expansion and external computer interfaces. For the KU-FRL system this expansion port is used to interface the computer with the MDAS-16 and TEAC MT2-02.

The MDAS-16 module is shown in Figure 3.7. This device is a 16-channel multiplexing unit combined with a 12-bit (3 1/2 digit) analog-to-digital converter. The unit has random address as well as sequential address capability when tied to a microprocessor controller. The measurement voltage ranges are user-selectable (-5 to +5 VDC was selected for this system). The unit has an effective throughput of 50 KHz with 20μ seconds access time between channels. The specifications for the unit are shown in Figure 3.8.

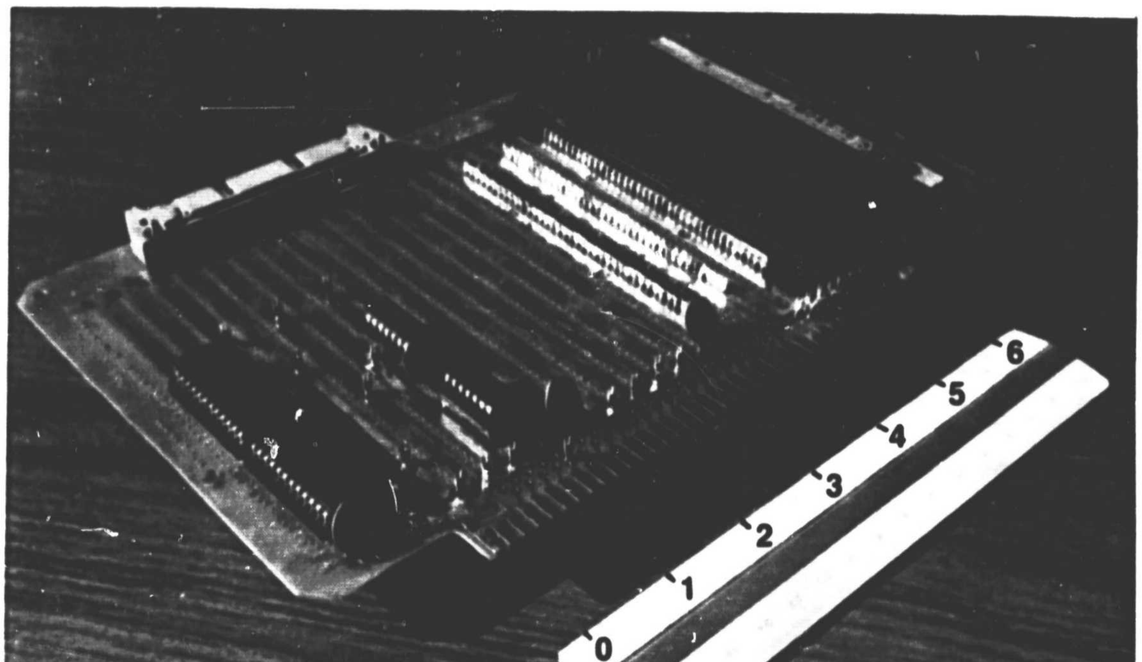
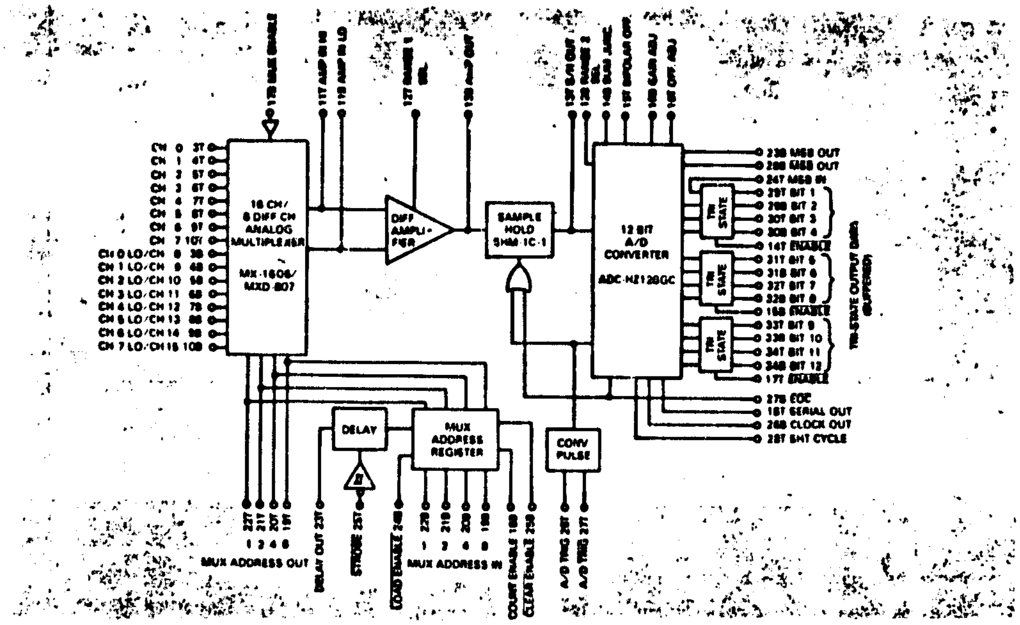


Figure 3.7 MDAS-16 Data Acquisition Module

ORIGINAL PAGE IS
OF POOR QUALITY



SPECIFICATIONS

ACCURACY

Resolution	12 Bits
Error, max. 50kHz sampling	±.025% of FSR
Nonlinearity, max.	±½ LSB
Diff. Nonlinearity, max.	±½ LSB
Gain Error	Adj. to zero
Offset Error	Adj. to zero
Temp. Coeff. of Gain, max.	±30ppm/°C
Temp. Coeff. of Offset, max.	±7ppm/°C of FS
Diff. Linearity Tempco, max.	±3ppm/°C of FS
Common Mode Rejec., min.	70 dB at 1 kHz
Monotonicity	0°C to 70°C
Power Supply Rejection	.01%/% Supply

POWER REQUIREMENT	+15VDC ±0.5V @ 65mA -15VDC ±0.5V @ 60 mA +5VDC ±0.25V @ 200mA

DYNAMIC CHARACTERISTICS

Throughput Rate, max.	50 kHz
Acquisition Time	12 μsec.
Conversion Time	8 μsec.
Aperture Time, max.	50 nsec
Sample-Hold Droop, max.	50μV/msec
Feedthrough, max.	.01%
Channel Crosstalk (Mux.)	-80 dB at 1 kHz

DIGITAL INPUTS

$\overline{\text{Enable}}$	Three separate inputs which enable tri-state outputs in 4 bit bytes 1 TTL load
Mux Address In	3 bit (MDAS-8D) or 4 bit (MDAS-16) binary address 1 LS TTL load
$\overline{\text{Strobe}}$	1 LS TTL load with 10K pull-up resistor
A/D Trigger	1 LS TTL load with 10K pull-up resistor
A/D $\overline{\text{Trigger}}$	1 LS TTL Load
Mux Enable	1 TTL load with 10K pull up resistor
Count Enable	1 LS TTL load with 10K pull up resistor
$\overline{\text{Load Enable}}$	1 LS TTL load with 10K pull-up resistor
$\overline{\text{Clear Enable}}$	1 LS TTL load with 10K pull-up resistor
MSB In	1 TTL load
Short Cycle	1 TTL load with 10K pull-up resistor

ANALOG INPUTS

Number of Channels	16 Single Ended (MDAS-16) 8 Differential (MDAS-8D)
Input Voltage Ranges	
unipolar	0 to +5V 0 to +10V
bipolar	±2.5V, ±5V, ±10V
Common Mode Range, min.	±10V
Max. Input Voltage, no damage	±15V
Input Impedance	100 megohms
Input Bias Current	3nA, 10nA max. 0 to 70°C
Input Capacitance	
OFF channel	10 pF
ON channel	100pF

DIGITAL OUTPUTS

Parallel Data Out	12 parallel lines of buffered tri-state output data Drives 12 TTL loads.
Coding	Straight binary, offset binary, and two's complement
Serial Out	Output data in MSB first, NRZ format Straight binary and offset binary coding. Drives 5 TTL loads
Mux Address Out	Buffered output of address register Drives 20 TTL loads
Delay Out	Drives 5 TTL loads
Clock Out	Drives 5 TTL loads
EOC (Status)	Drives 4 TTL loads
MSB Out	Drives 5 TTL loads
$\overline{\text{MSB}}$ Out	Drives 5 TTL loads

PHYSICAL ENVIRONMENTAL

Operating Temp. Range	0°C to 70°C
Storage Temperature Range	-25°C to +85°C
Package Size	4.6 x 2.5 x 0.375 inches (116.8 x 63.5 x 9.5 mm)
Package Type	Steel, shielded on 5 sides
Weight	6 oz (170 g)

NOTES: 1. All outputs are Vout ("0") ≤ +0.4V, Vout ("1") ≥ +2.4V
2. All inputs are Vin ("0") ≤ +0.8V, Vin ("1") ≥ +2.0V

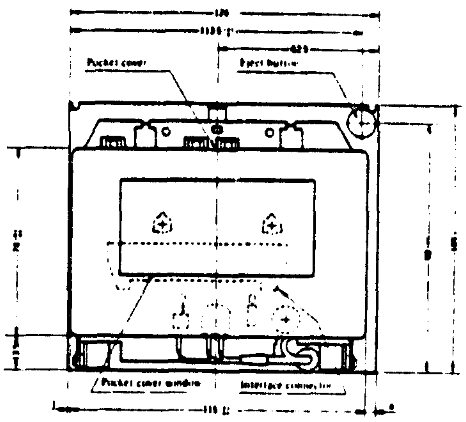
Figure 3.8 MDAS-16 Module Specifications

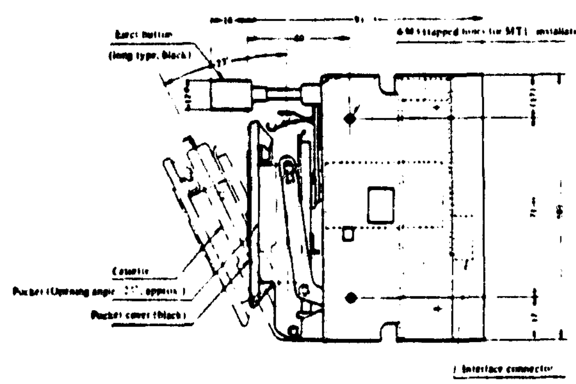
The calibration procedures for the MDAS-16 are described in detail in Reference 31.

The other major component of the data management module is the TEAC tape transport. This unit is shown in Figure 3.9. It is a low-cost magnetic tape unit designed specifically for digital applications. The unit makes use of standard audio-type digital cassette tapes for data storage. All interfacing required for the tape transport is included in the package. The tape unit requires only control signals and parallel data input, both provided by the AIM-65. All detailed control functions required by the tape unit are generated internally. Only simple control signals are required to initiate the various functions, such as data recording or data playback.

The data management module is also used for data transfer to the ground-based computer system. The use of the same recorder for in-flight recording and playback avoids possible tape head mismatch problems and reduces overall system costs. The data are transferred over an RS232 serial port to any computer with an RS232 port or dial-up time-sharing capability. The RS232 ports can be connected directly for transmission of the data at up to 9600 bits per second or through a modem across telephone lines at 300 bits per second. An appropriate software program is needed by the AIM-65 to interact with the ground-based computer system. Once the data are transferred to the other computer, the Rockwell AIM-65 computer is no longer needed.

ORIGINAL PAGE IS
OF POOR QUALITY





SPECIFICATIONS

DATA FORMAT

Data format complies with ISO-3407, ECMA-34, JIS-C6281 and other similar standards.

Recording Format:	Phase Encoding
Recording Density:	800 bpi(32 bits/mm, nominal)
Number of Tracks:	Single Track

CONSTRUCTION

Exterior Dimensions:	105(H) x 120(W) x 91(D)mm
Weight:	1.2kg, Max.

MECHANICAL CONSTRUCTION

Tape Drive System:	D.C. Motor Direct Reel Drive System
Tape Speed Detection System:	Encoder Detection
Cassette Insertion:	Pocket Holder type
Cassette Eject Mechanism:	Manual Ejection by depressing EJECT Button
BOT/EOT & Clear Leader Detector:	Photoelectric Detector
Magnetic Head:	Single Track, Single Gap Read/Write Head

POWER REQUIREMENTS

	Average Current Consumption	Maximum Current Consumption	Permanent Ripple Voltage
D.C.+12V±5%	1.0 A	1.8 A*	Less than 100mVp-p
D.C.+ 5V±5%	0.7 A	0.75A*	Less than 100mVp-p

* Except for surge current at power-on

GROUND ISOLATION

Insulation resistance between
0V power supply and frame ground: 5 Meg Ohms or more, at 150V D.C.

ENVIRONMENTAL CONDITIONS

Temperature range(Operating):	+5°C ~ +40°C
(Storage):	-15°C ~ +60°C
Relative Humidity Range(non-condensing)	
(Operating):	20% ~ 80%
(Storage):	10% ~ 90%
Vibration(Operating):	Less than 0.5G(less than 120Hz)
(Packaged Condition based on TEAC standard packing):	
Impact:	Less than 40G(less than 30msec)
Continuous:	Less than 3G

OPERATIONAL CHARACTERISTICS

Tape Speed(Slow):	15 ips(38.1cm/s) ± 3%
(Fast):	45 ips(114.3cm/s) ± 4%
IBG:	0.97 ± 0.24 inch(24.64 ± 6.10mm)
Erased Length	2.19 ± 0.27 inch(55.63 ± 6.86mm)
Initial Gap:	2.26 ± 0.40 inch(57.40 ± 10.16mm)
Readable IBG Length	0.7 inch(17.78mm) or more
Required tape length for Long IBG detection:	15.0 ~ 22.2 inch(381.0 ~ 563.9mm)
Start distance for HIGH SPEED SEARCH:	2.4 inch(60.96mm) or less
Stop distance for HIGH SPEED SEARCH:	1.9 inch(48.26mm) or less
Nominal Data Transfer Rate:	12k bits/sec
Recording Density	800 bpi ± 4%
Treshold Level(at reference read level)	
(LOW):	18 ± 3%
(HIGH):	40 ± 5%

INTERCHANGEABILITY

Cassette tapes which comply with the Information Interchange format standardized in ISO-3407, JIS-C6281, and etc. are securely read by the MTU. Also cassette tapes written with the MTU enable the perfect interchangeability with the Information Interchange format standardized in ISO-3407, JIS-C6281, and etc.

MTBF

10,000 hours or more at the 10% rate of operation

MAGNETIC TAPE

Magnetic tape cassette for Information Interchange which comply with ISO, ANSI, ECMA, JIS, and other similar standards. Also the cassette used for the MTU shall be previously approved between TEAC and the customer.
Recommended tape: TEAC CT-300

Figure 3.9 TEAC MT2-02 Tape Transport Specifications

3.3 Transducers

To keep the overall system size as small as possible, transducers were separated from the voltage regulators and signal filters. Figure 3.10 shows the transducer package assembly, and also the relative location of the sensors in the transducer package. Those transducers which must be externally mounted are connected to the transducer package through MIL-SPEC connectors.

The transducer package was firmly clamped to the seat tracks just behind the pilot and copilot seats in the Cessna 172 flight test program. Following are descriptions of the individual sensors used in this program.

3.3.1 Accelerometers

The accelerometers used in this package are force feedback (or closed-loop) type. This type of accelerometer derives its measurement by determining the force required to maintain its seismic mass at zero displacement. This technique reduces the errors caused by mass displacement and does not rely on springs (and their associated inaccuracies) as do displacement (or open-loop) type accelerometers. The disadvantage of the closed-loop accelerometer is its relatively high cost. It is essential to note that linear (as opposed to vibration) accelerometers must be used for this type of transducer package. This restriction is due to the low frequency characteristics of the vibration and linear accelerometers.

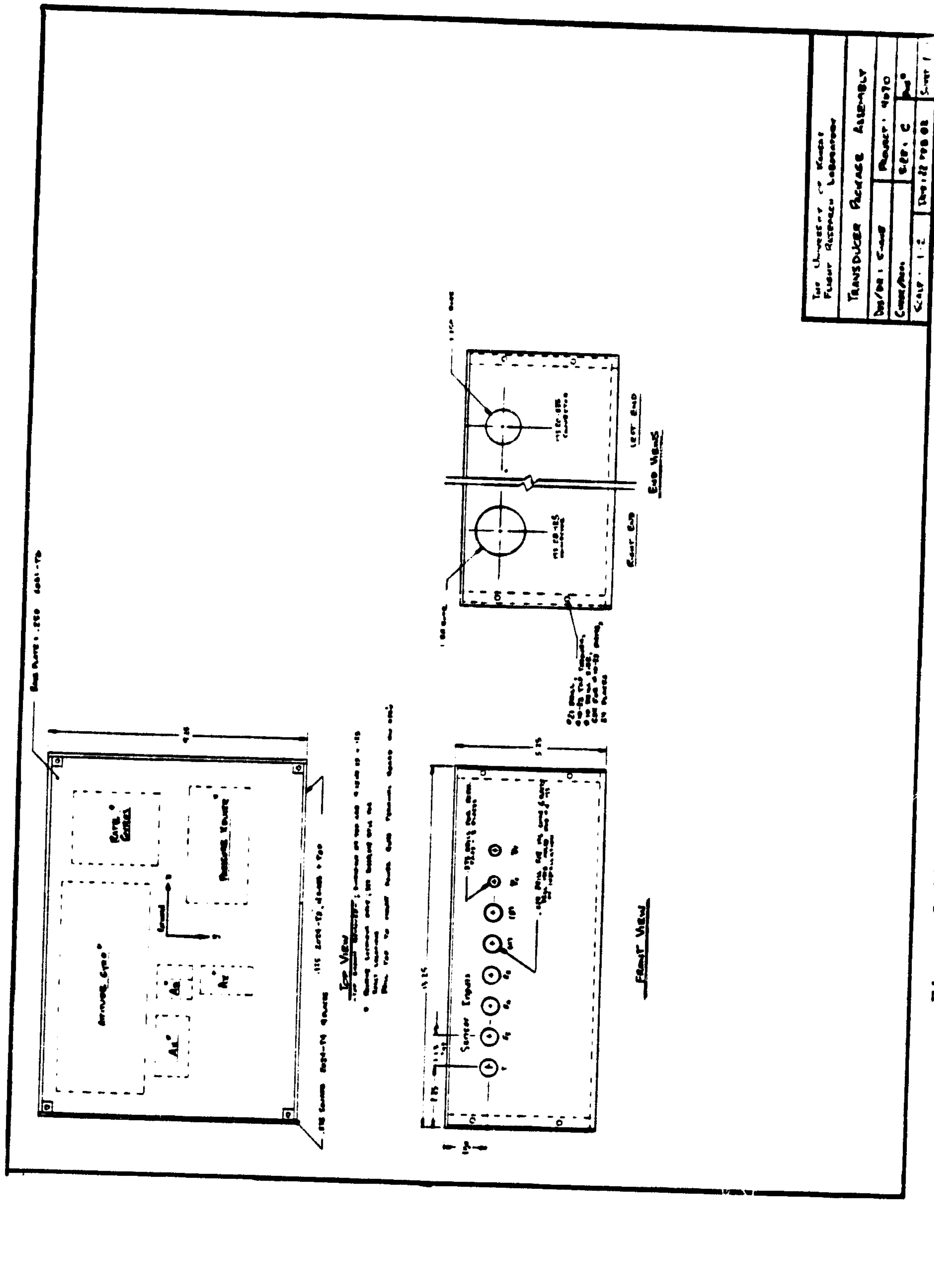


Figure 3.10 Transducer Package

The accelerometers chosen are manufactured by Schaevitz Engineering. Their specifications are shown in Figure 3.11. This type of accelerometer is intended for measurement of linear accelerations such as guidance and control systems or vehicle ride analysis. Both a precision sensor and its required electronics have been integrated into the accelerometer package. The sensor can be installed with relative ease; it requires only a DC input voltage and a measurement of the DC voltage output.

3.3.2 Attitude Gyro

Both roll attitude and pitch attitude are measured with a Humphrey VG-24 vertical gyroscope. The specifications of this gyro are shown in Figure 3.12. This gyroscope is a self-erecting DC type with potentiometers for determining the sensor outputs (+28 VDC used for the motor, and ±5 VDC used for the potentiometer excitation). This gyro has operated reliably throughout three phases of this test program.

3.3.3 Rate Gyros

A three-axis DC/DC rate gyro package was used for measurement of pitch, roll, and yaw rates. The advantage of using a three-axis package rather than three separate gyros is that orthogonal alignment of the gyros upon installation is assured. However, failure of a single gyro would require removal of the entire package for repair.

The gyros selected are displacement (or open-circuit) type.

Specifications at 20°C

LSB Linear

Range g	Nominal Natural Frequency Hz	Nominal Output Impedance kilohms
± 0.25	50	20
± 0.5	70	10
± 1.0	100	5
± 2.0	110	2.5
± 5.0	125	5
± 10.0	140	2.5
± 20.0	160	5
± 50.0	200	5

Input Voltage	±15V DC nominal
Input Current	10 mA DC maximum (6mA DC average)
Full-Range Open-Circuit Output Voltage	±5.0V DC
Damping Ratio	0.6 typical (0.3 to 1.0 on request)
Linearity (Notes 1 & 2)	±0.05% of full scale output
Hysteresis (Note 2)	0.02% of full scale
Resolution (Note 2)	0.0005% of full scale
Cross-Axis Sensitivity (Note 3)	±0.002 g per g up to ±10 g range, inclusive ±0.005 g per g over ±10 g range
Bias	Less than 0.1% of full scale
Sensitive Axis to Case Alignment	±1°
Noise Output	5mV rms maximum
Operating Temperature	−40°C to +95°C
Storage Temperature	−55°C to +105°C
Thermal Coefficient of Sensitivity	0.02% per °C
Thermal Coefficient of Bias	0.002% per °C
Shock Survival	100 g − 11 ms
Weight	3 oz.

SB Series

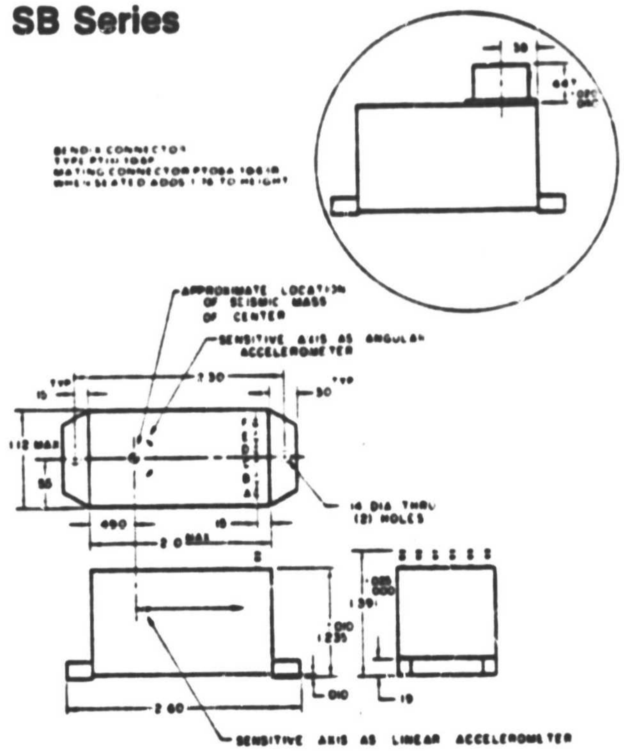
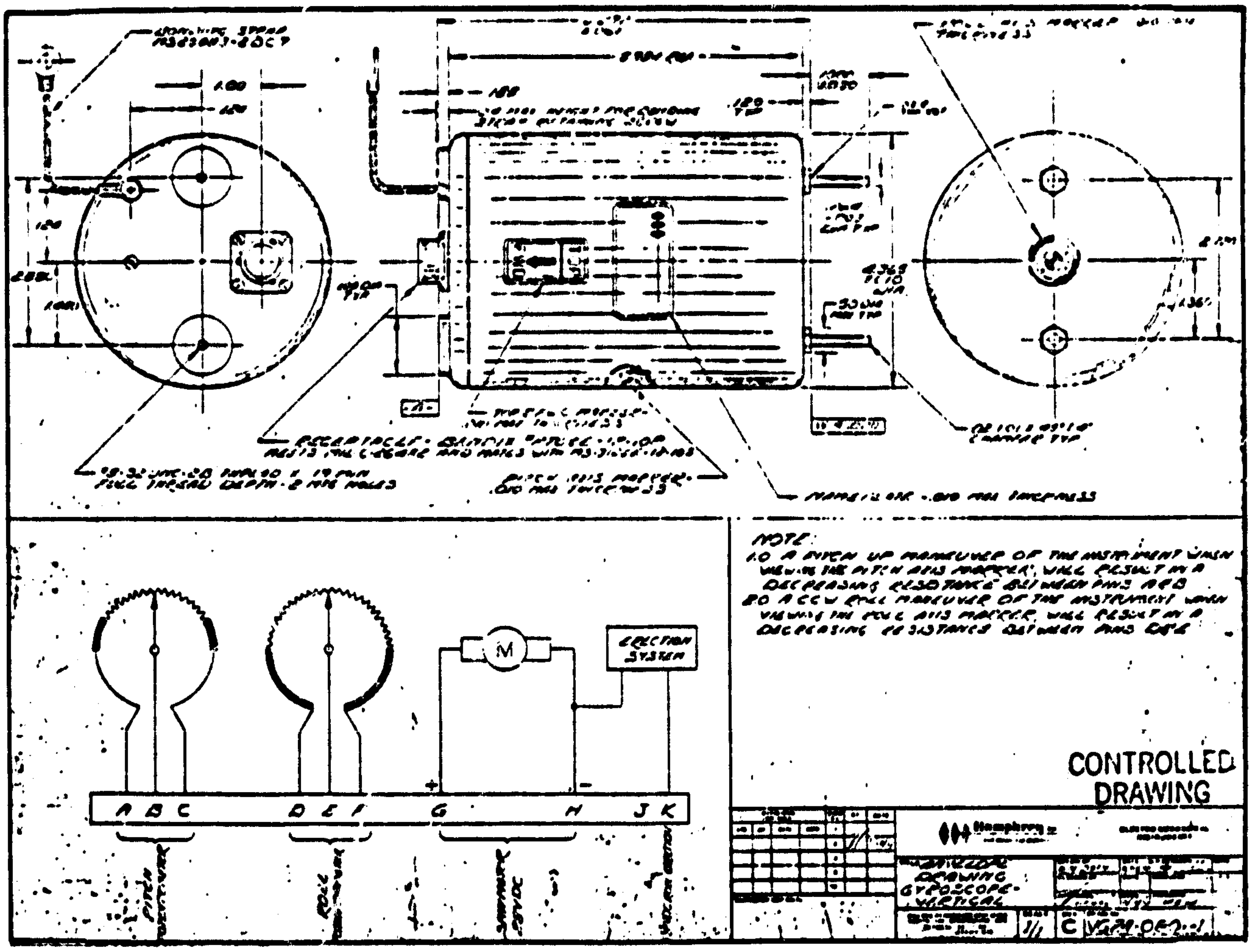


Figure 3.11 Schaevitz Engineering LSB Series Accelerometers



SPECIFICATIONS

RANGE - MECHANICAL	PITCH: ±60° minimum ROLL: 360° continuous
- ELECTRICAL	PITCH: ±60°, ±2.5° ROLL: ±90°, ±3.0°
OUTPUT	Potentiometer output
STATIC ERROR BAND	PITCH: ±1.25% of full scale at 0° expanding linearly to ±2.08% of full scale at 60° ROLL: ±0.83% of full scale at 0° expanding linearly to ±1.67% of full scale at 90°
RESISTANCE	1500 ±100 ohms
CONTACT RESISTANCE	2 ohms maximum at 20 mA
RESOLUTION	0.2% of full scale maximum
POWER DISSIPATION	1 watt at +165 °F
WIPER CURRENT	20 mA maximum
ELECTRICAL REQUIREMENTS	
SPIN MOTOR	
VOLTAGE	26 to 32 volts DC
CURRENT - STARTING	4.5 A maximum at 30 volts DC for 2.5 seconds
- RUNNING	1 A maximum at 30 volts DC
ERECTION	
VOLTAGE	26 to 32 volts DC
CURRENT	100 mA maximum intermittant
PERFORMANCE	
SPIN MOTOR TIME TO SPEED	5 minutes maximum
TIME TO ERRECT FROM MOTOR OFF	within 0.5° in 9 minutes
NORMAL OPERATING ERECTION RATE	2 to 9 °/minute after 3 min.
VERTICAL ACCURACY	within 0.5° of true vertical
FREE DRIFT RATE	0.5°/min. nominal; tested on ±3 1/2° ?torsby 6 min. run alternating

ENVIRONMENTAL CONDITIONS	
VIBRATION	vertical accuracy of ±2.0° shall be maintained during vibration of 0.01 inch D.A., 5 to 65 Hz; 2g, 65 to 500 Hz.
SHOCK	15g ; 11 msec; all axes
ACCELERATION - NON OPERATING	30 g; 1 min; vertical axis
- OPERATING	10 g; 1 min; applied in pitch or roll axis shall not produce a drift of greater than 10 °/min.
TEMPERATURE - OPERATING	-65 to +165 °F
- STORAGE	-80 to +185 °F
ALTITUDE	sea level to 40000 ft
SEA WATER IMMERSION	3 ft for 3 hr.
HUMIDITY	to 95% including condensation for 240 hrs.
SALT SPRAY	as encountered on shipboard or at coastal regions
SAND AND DUST	as encountered in desert regions
FUNGUS	external surfaces non-nutritive
EXPLOSION PROOF	shall not produce an explosion when operated in a fuel vapor rich area
RADIO NOISE INTERFERENCE	MIL-I-6181; paragraph 4.3.1 & 4.3.2
SERVICE LIFE	100 hrs minimum
SHELF LIFE	3 yrs minimum
INSULATED RESISTANCE	20 megohms minimum at 100 volts DC motor circuit exempt
WEIGHT	3.0 lb. maximum
SEALING	shall not leak under vacume equivalent to 40000 ft.

Figure 3.12 Humphrey VG-24 Vertical Gyro

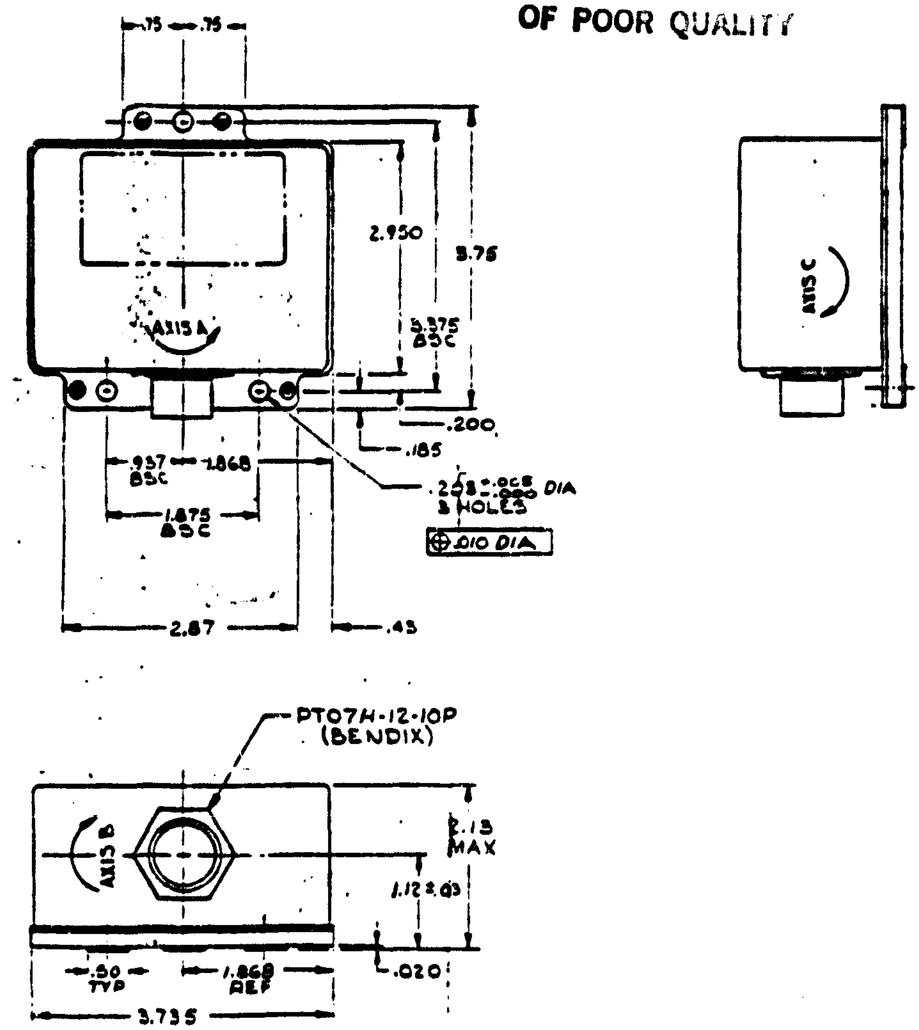
Integrating (or closed-circuit) gyros provide better accuracy; however, they cost approximately ten times as much. The accuracy of a displacement gyro meets the requirements (see Tables 3.2 and 3.3) and therefore was chosen for the sensor package.

The gyros selected are manufactured by Northrop Corporation, Precision Products Division. Required input power is 28 VDC, while the output signal is -5 to +5 VDC. The gyros are model G5 subminiature rate sensors. The gyro package and specifications are shown in Figure 3.13.

3.3.4 Control Position Transducers

Linear displacement transducers manufactured by Space-Age Control, Inc., were used to measure rudder, aileron, and elevator positions. This transducer is shown in Figure 3.14. Due to the relatively small size of the transducer, it was considered acceptable to mount it externally to the airframe. Chapter 6 shows the installation locations used for the Cessna 172 flight tests.

A novel technique for the attachment of the control position transducers (CPT's) has been used. Double-sided foam tape attaches these external devices (as well as the temperature and pitot probes) to the airframe. The mounting technique is depicted in Figure 3.15. This technique was first tested in the KU-FRL subsonic wind tunnel at speeds up to 119 mph. These tests were run for two hours with no degradation in rigidity of the mount (Reference 27). This method has been shown to give excellent results in three flight test programs



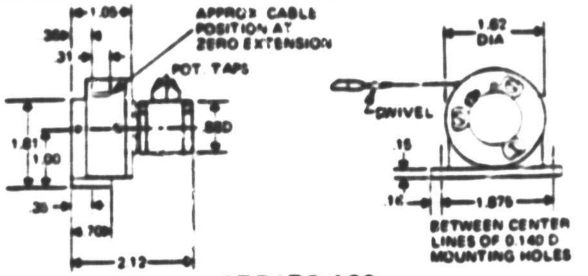
Weight	2.0 lb. (max)
Outline dimensions	3.75 x 3.75 x 2.13 in
Power input	15 w.(max)(31 vdc)
Input voltage limits	28 ±3vdc
Full-scale output	±5 vdc
Output impedance	5000 ohms (max)
Output load resistance	500K ohms (nominal)
Ripple	25 mv.peak-peak(max)
Zero rate setting	±1/2 % FS
Input range (roll/pitch/yaw)	50°/sec
Maximum input rate	600°/sec
Output voltage (at overrange limits)	±7 vdc
Output stability (input voltage variations)	1/2 % FS.
Repeatability	1 % FS.
Threshold	0.01 °/sec
Resolution	0.01 °/sec
Hysterisis	0.1 °/sec
Operating temperature	0 - 160 °F
Temperature sensitivity	Zero output 1% FS/100°F Scale Factor 3% FS/100°F

Warm up time	10 min
Motor acceleration time	30 sec (max)
Gimble deflection angle	±2° typical
Acceleration sensitivity	
Linear	0.05 °/sec/g
Angular	0.08 °/sec/rad/sec^2
Linearity	1/2 % FS, 0-1/2 scale 2 % FS, 1/2 - FS
Service life	100 hr(typical 14000 hr)
Insulation resistance	10 megohms (min),50 vdc
Damping ratio	0.5 to 0.9
Natural frequency	35 Hz (min)
Environments	
Shock	250g peak sawtooth, 5 msec
Vibration	0.1 g /Hz, 20-2000 Hz
Storage temperature	-65 - 200 °F
Radio interference	MIL-I-8161D

Figure 3.13 Northrop G5 3-Axis Rate Gyro Package

ORIGINAL PAGE IS
OF POOR QUALITY

TECHNICAL INFORMATION



SERIES 160

MODEL NO.	DASH NO.	RANGE 0 TO (INCHES)	RESOLUTION INCHES
160 –	161	2	0.0033

Potentiometer life, 1-turn units, 1,000,000 cycles
3-turn units, 600,000 cycles
or 900 hours at rated power

Cable static tension at zero extension 10-16-oz

Op. temp., -65°F to +255°F

Resistance, 1000 Ω · Other resistances available on special order are 5, 10, 20, 50, 100, 200, 500, 2K, 5K, 10K, & 20K. 45K (3 turn only), 50K and 100K (1 turn only)

Standard pots are used unless otherwise specified. Specials are available on special order only.

	Standard	Special
Resistance	±3%	±1%
Linearity (3 turn)	±0.25%	±0.20%
Linearity (1 turn)	±0.5%	±0.35%

Max current at 155°F (ambient) is 31.6 milliamps

Max. voltage across coil is 31.6 volts

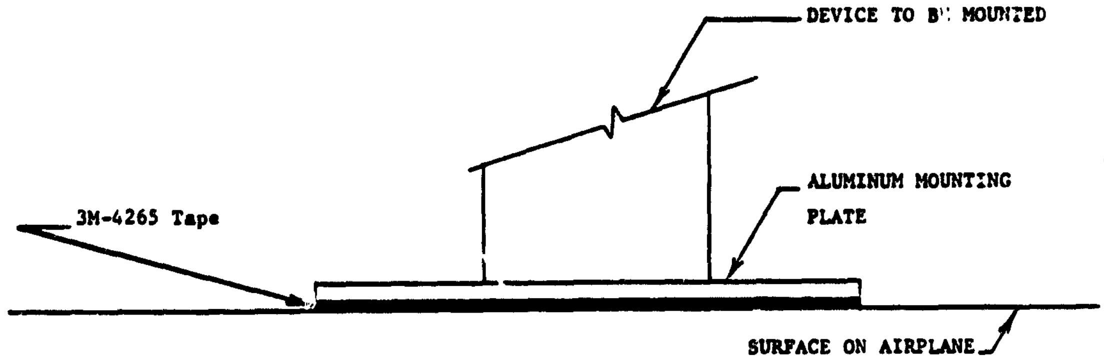
Power rating, 1.0 watts at 155°F derated to 0.0 watts at 255°F

Insulation resistance, 1000 megohms min. at 500 VDC

Dielectric strength, 1000 volts RMS min. at 60 CPS

SAC Linear Displacement Transducers (LDT) consist of an extension cable, spirally wound on a spring-loaded rewind drum, which is coupled to a precision, wire-wound, rotary potentiometer. The cable end is attached to the object whose movements are to be monitored. As the cable is extended or retracted, the cable drum rotates the potentiometer wiper, varying the voltage at the wiper tap (No. 2) of the potentiometer. The voltage may be measured to reflect the position, direction, or rate of motion of the object attached to the cable.

Figure 3.14 Space Age Controls Linear Displacement Transducer



NOTE:-

- lightly sand surface of airplane
- clean with isopropyl alcohol
- surface must be room temperature during attachment
- fair with duct tape

3M #4265 -DOUBLE COATED NEOPRENE FOAM TAPE

Adhesive	A-20 Firm Acrylic
Thickness	3/64 in.
Tensile	60 psi
Static Shear	66 psi
Temp max.	225 °F
Temp min	-20 °F

Figure 3.15 External Device Mounting Technique

(with no failures of any type). The tape used is 3M number 4265 neoprene foam tape. Its properties are included in Figure 3.15.

A minor inconvenience of the mounting locations for the CPT's is the resultant nonlinear calibration curves. All other sensor calibrations were linear.

3.3.5 Static and Dynamic Pressure Transducer

A B&D Instruments Company model 2504 series transducer was used for static and dynamic pressure measurements. This transducer and its specifications are shown in Figure 3.16. This device incorporates its own signal conditioning and utilizes semiconductor pressure transducers. This type of pressure transducer is particularly temperature sensitive; the B&D unit compensates for this by heating its case to maintain a constant temperature.

The pitot tube was designed and constructed according to specifications taken from Reference 28 (see Figure 3.17). The pitot tube is attached to the underside of the wing using the method shown

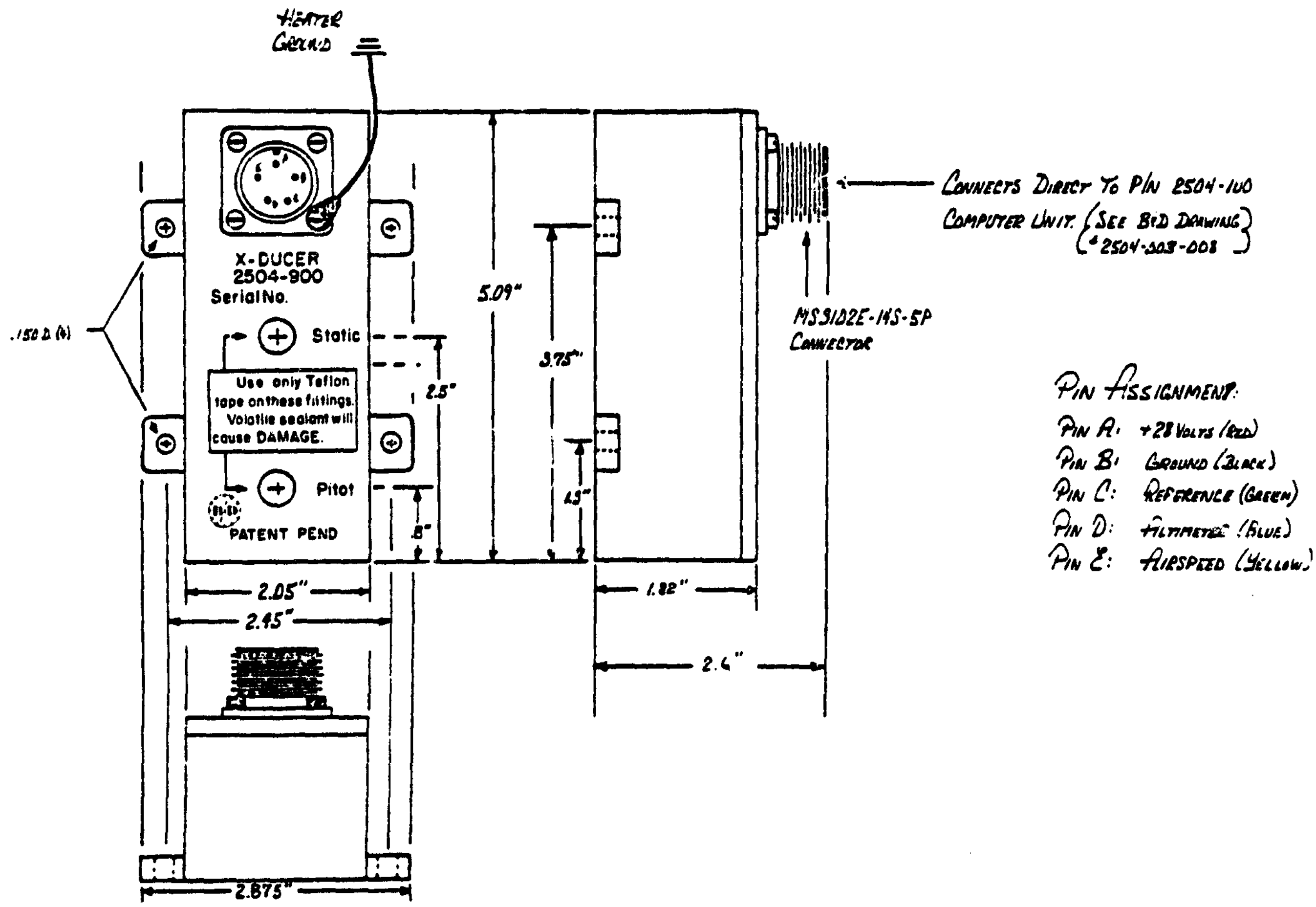


Figure 3.16 B&D Instruments 2054 Pressure Tran :ducer

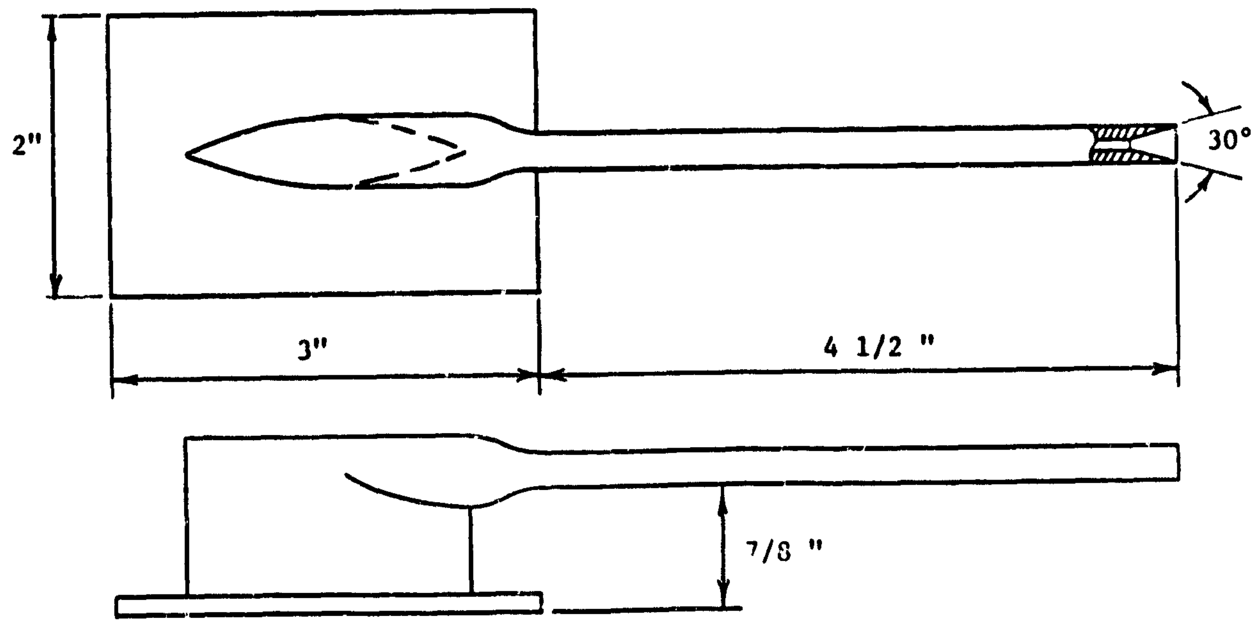


Figure 3.17 Pitot Tube

in Figure 3.15. It allows a high angularity of flow while still maintaining true readings. The pitot tube has been constructed to assure that it remains outside of the boundary layer. This in turn assures that true readings are maintained as long as the axis of the tube is close to the direction of the airflow (±15°). The tube was mounted under the right wing, halfway between the propeller arc and the wing tip (see Figure 3.18). This location minimizes flow and energy effects of the wing tip vortices and propeller slip stream.

For accurate measurement of static pressure, a trailing cone can be used (see Reference 29). Initial flight tests showed difficulty in deploying this cone after takeoff. For this reason the cone was used in its retracted position for the Phase III flight tests of the Cessna 172. This has been found to be sufficient for stability and control parameter estimation; however, a more accurate

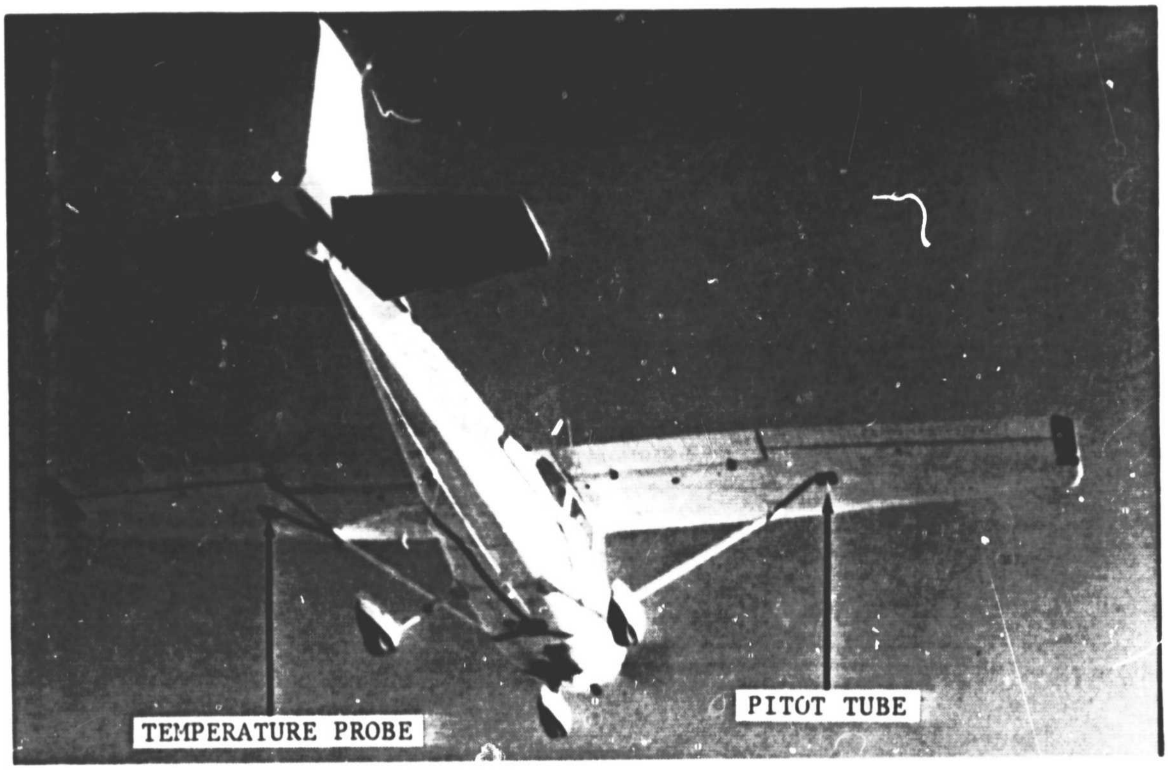
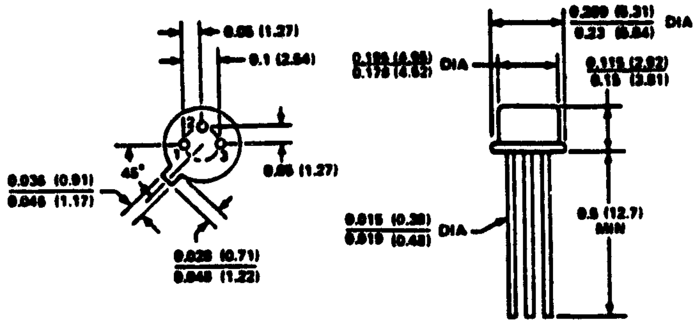


Figure 3.18 Location of Pitot Tube and Temperature Probe

static pressure measurement would be required for any performance testing.

3.3.6 Temperature Transducer

An Analog Devices Company semiconductor temperature transducer was used for measurement of air temperature. Specifications for this unit are shown in Figure 3.19. The transducer is mounted in the probe as shown in Figure 3.20. The sleeve was added to the temperature probe to assure that the air is sufficiently slowed to provide accurate temperature measurement. The temperature probe is mounted to the airplane in the same manner as the pitot tube.



Model	AD590M	Absolute error (rated range)	
Absolute Maximum Ratings		No external adjustment	±1.7°C max
Forward voltage	+44 v	+25°C calib error = 0	±1.0°C max
Reverse voltage	-20 v	Nonlinearity	±3.0°C max
Breakdown voltage(to case)	±200 v	Repeatability	±0.1°C max
Rated temperature range	-55°C to +150°C	Long term drift	±0.1°C/month max
Storage temperature	-65°C to +155°C	Current noise	40 pA$\sqrt{Hz}$
Lead temperature(soldering)	+300°C	Power supply rejection	
PowerSupply		+4v <Vs <+5v	0.5μA/v
Operating voltage range	+4v to +30v	+5v <Vs <+15v	0.2μA/v
Output		+15v <Vs <+30v	0.1μA/v
Nominal current (+25°C)	298.2 μA	Case isolation	10^{10} ohms
Nominal temp. coefficient	1μA/°C	Effective shunt capacitance	100pF
Calibration error (+25°C)	±0.5°C max	Turn on time	20 μs
		Reverse bias leakage (reverse voltage =10 v)	10 pA

Figure 3.19 Temperature Transducer Specifications

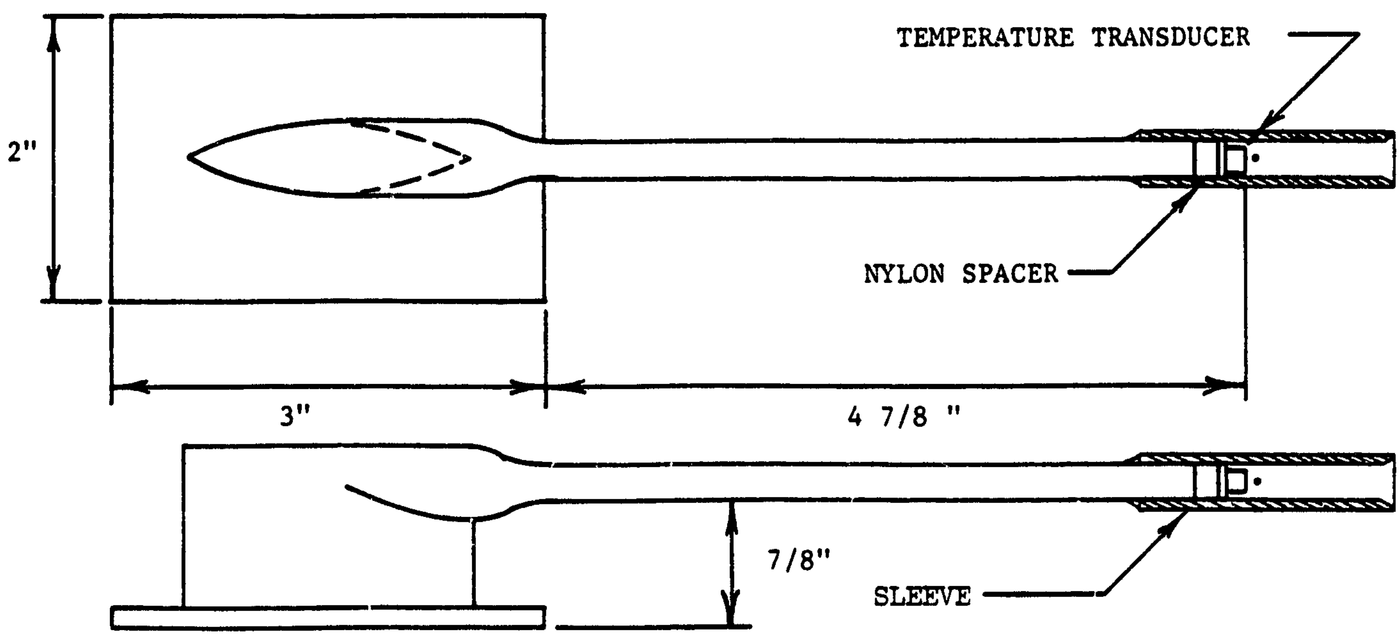


Figure 3.20 Temperature Probe

It is mounted to the underside of the left wing using the double-sided tape (see Figure 3.18).

3.4 Calibration

Calibration of the various inertial transducers used in the transducer package was accomplished with a pendulum-type calibration stand specifically constructed for that purpose. Figure 3.21 shows the calibration pendulum. This pendulum was capable of calibrating the attitude gyro, the 3-axis rate gyros, and each of the three linear accelerometers. For these calibration tests, the entire data acquisition system measured the output of the transducer module, which was mounted at the end of the pendulum arm. This gave a complete system calibration as opposed to individual sensor calibrations. Figures 3.22, 3.23, and 3.24 show the construction drawings for this calibration stand.

The calibration of the transducers was accomplished in two modes: (1) a static mode for angular calibration of the vertical gyro and for ±1 g calibration of the accelerometers and (2) a dynamic mode for calibration of the rate gyros and for calibration of the accelerometers at elevated-g levels. In the dynamic mode the data acquisition system was used to record the sensor time histories at a sample rate of 100 SPS.

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

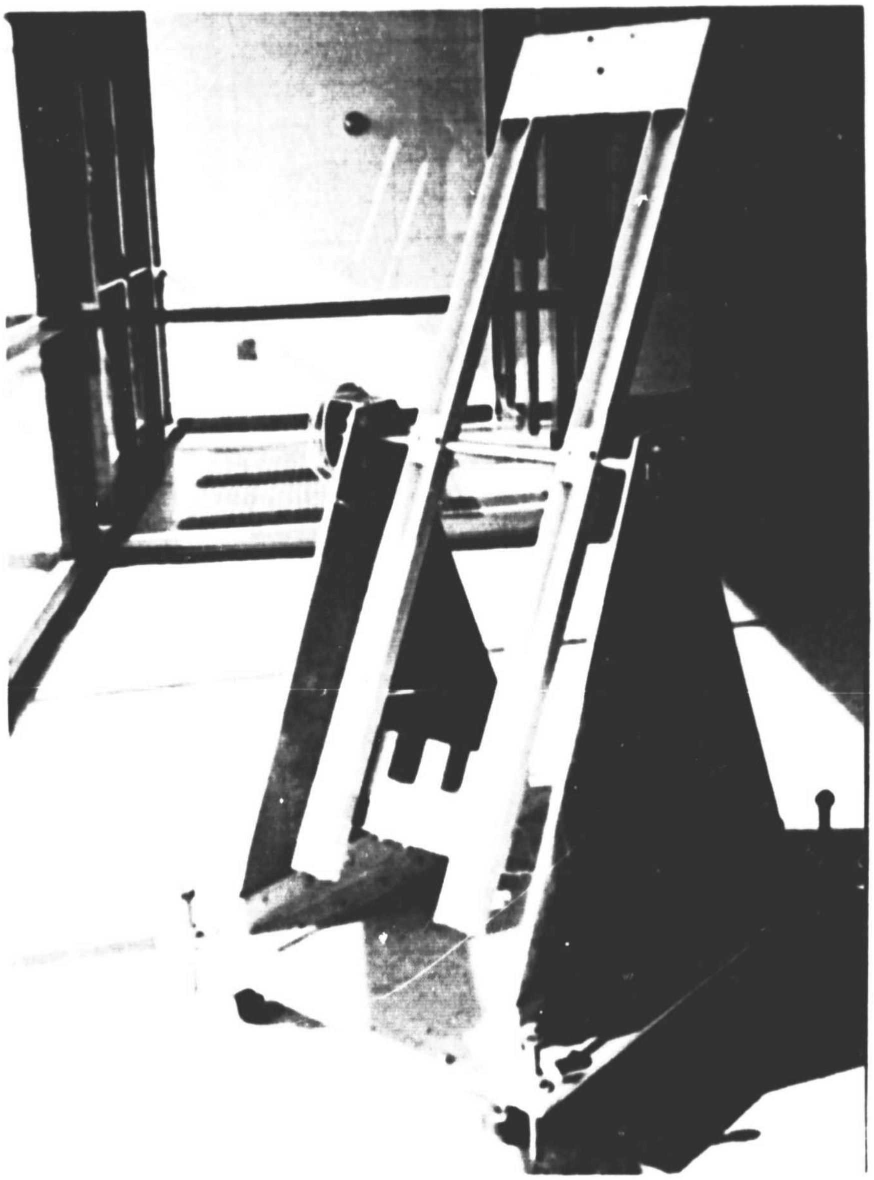


Figure 3.21 Calibration Pendulum Stand

ORIGINAL PAGE IS
OF POOR QUALITY

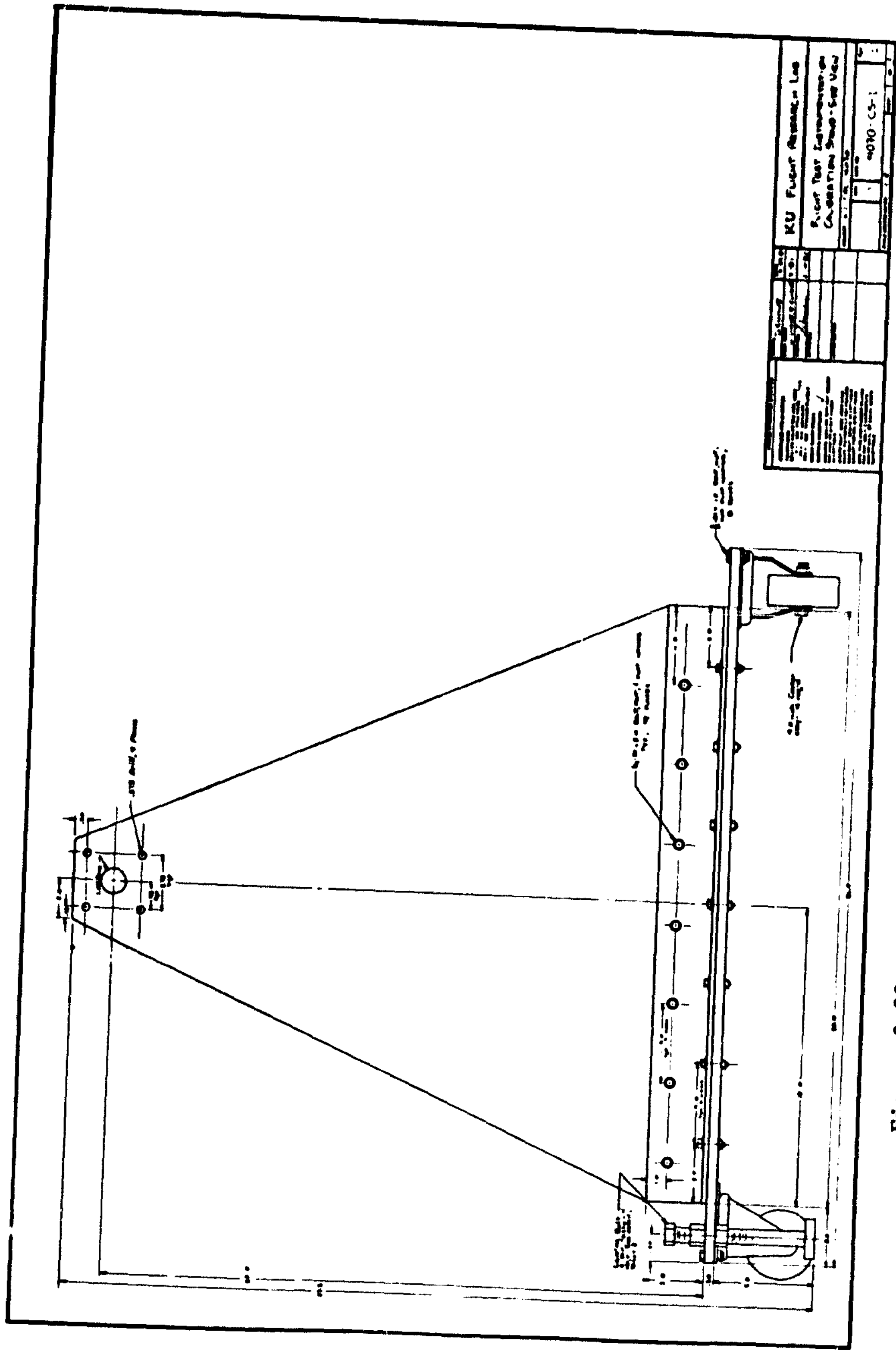


Figure 3.22 Calibration Stand (Side View)

ORIGINAL PAGE IS
OF POOR QUALITY

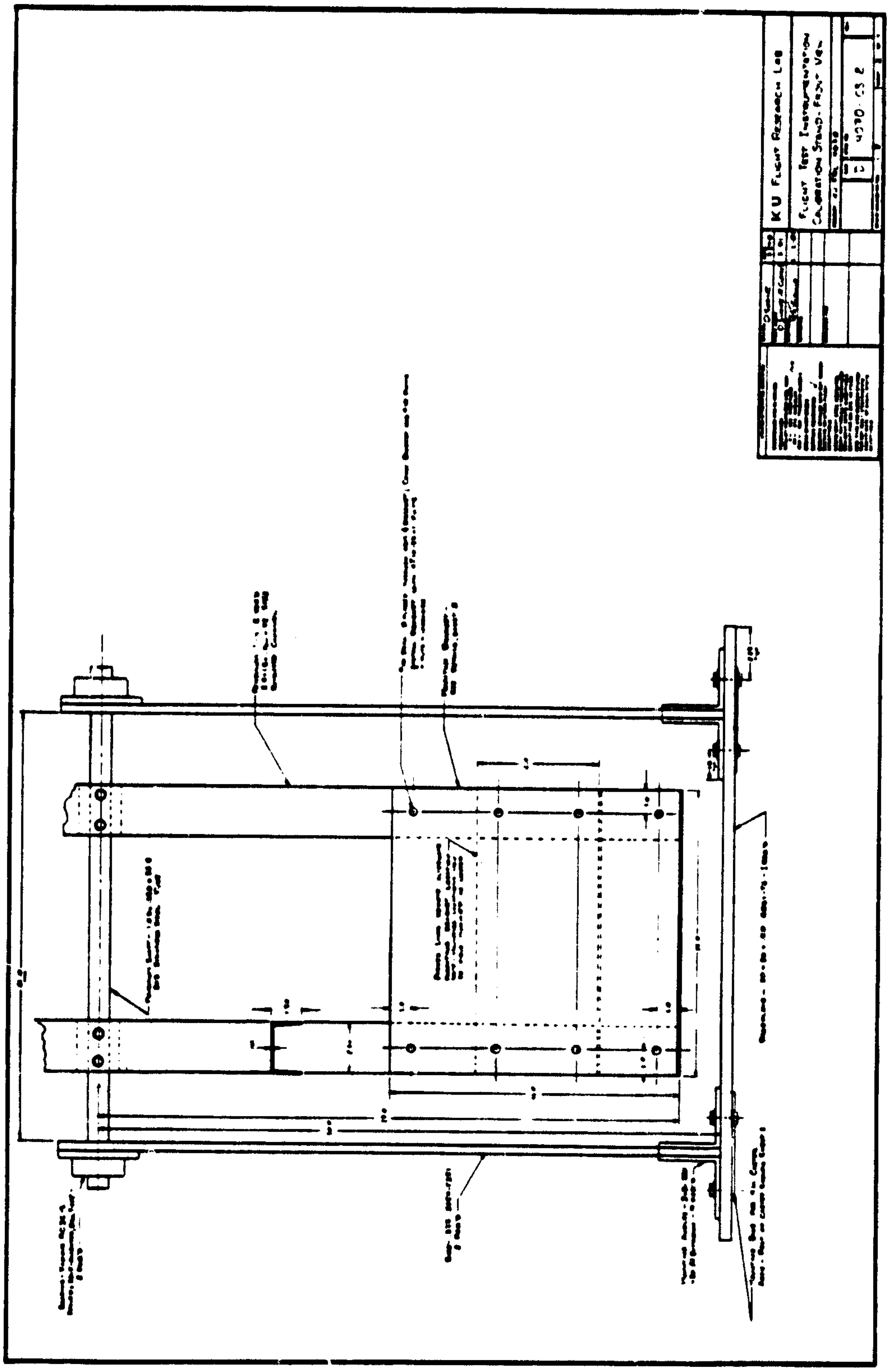


Figure 3.23 Calibration Stand (Front View)

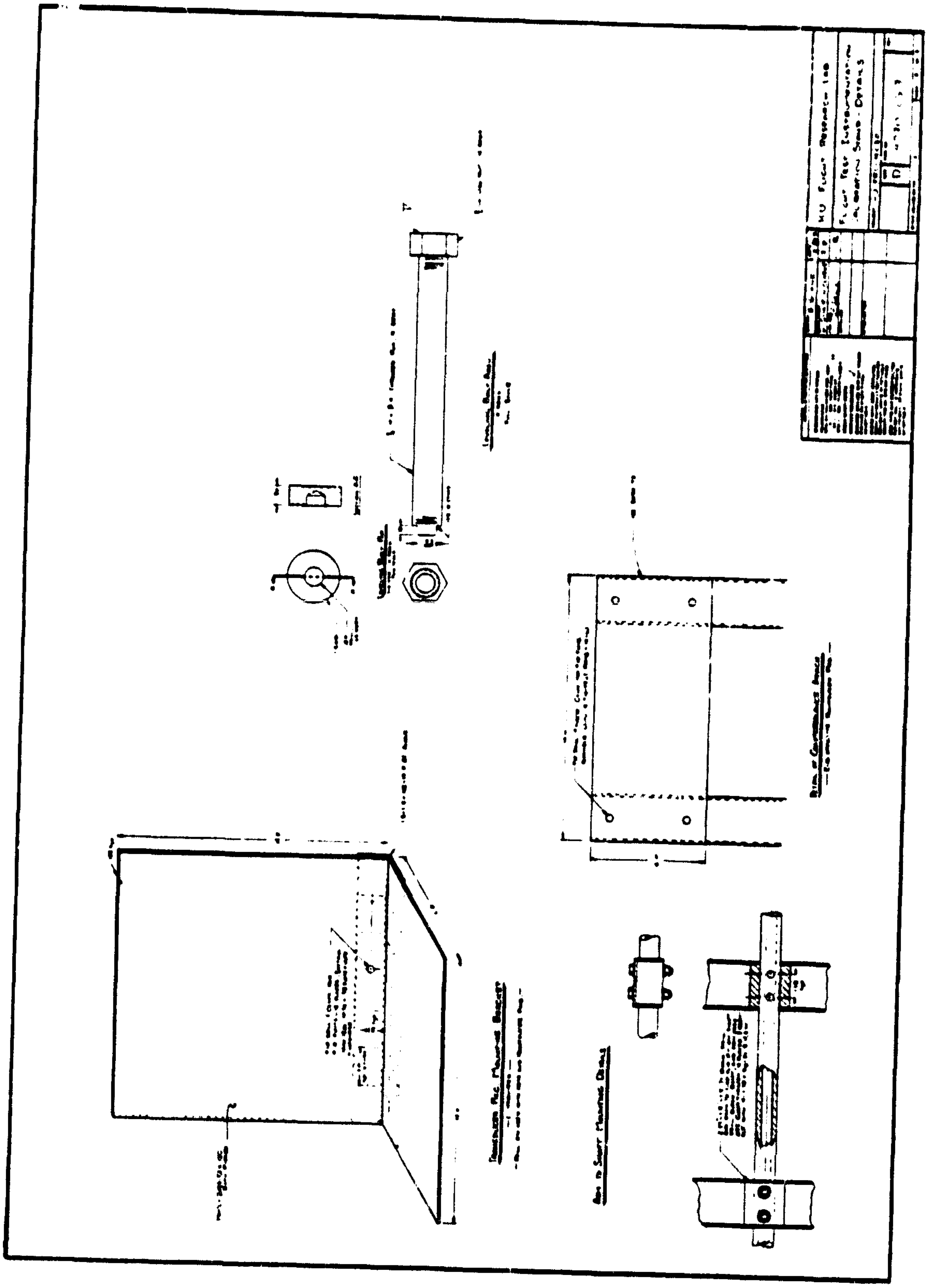


Figure 3.24 Calibration Stand (Details)

3.5 Filtering

Early flight tests showed that the response characteristics of the accelerometers were such that they picked up aircraft engine vibrations. The graph of Figure 3.25 shows the airframe vibration characteristics (measured using an accelerometer and observing its output on an oscilloscope trace) as a function of engine speed. It is obvious from these curves that the vibration is primarily caused by the engine and is a function of engine speed. This vibration is almost entirely at frequencies greater than 100 Hz.

A low-pass filter with a break frequency of 20 hertz was used to eliminate this unwanted high-frequency vibration from the measurement signal. A two-pole active filter with the response characteristics of Figure 3.26 virtually eliminated the noise and left the desired measurement (occurring at a frequency near 1 Hz) unchanged. Measurements of the A_Z accelerometer with filter and without (see Figure 3.27) show this to be true.

To avoid phase shifts associated with filter lags, all channels were filtered with the same design two-pole filter. The filter box details are shown in Figure 3.28.

3.6 Power Supply

Two options were considered as power sources for this instrumentation system:

(1) Tap the aircraft electrical system, or

(2) Carry a separate battery package during the flight tests.

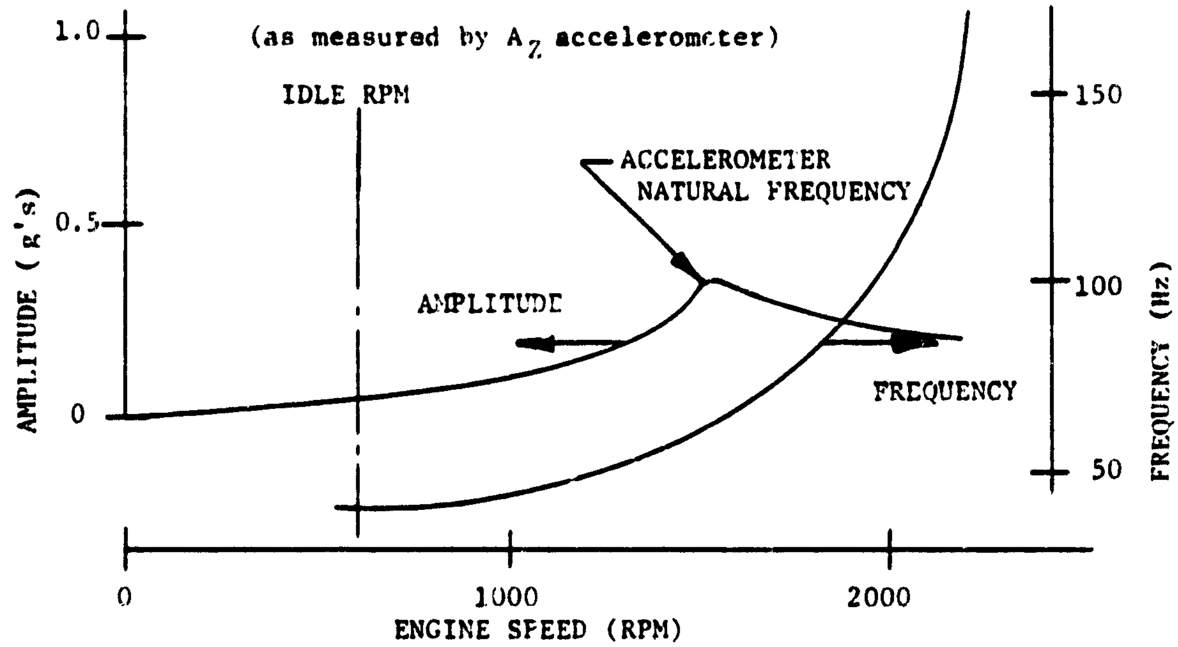


Figure 3.25 Measured Airframe Vibration

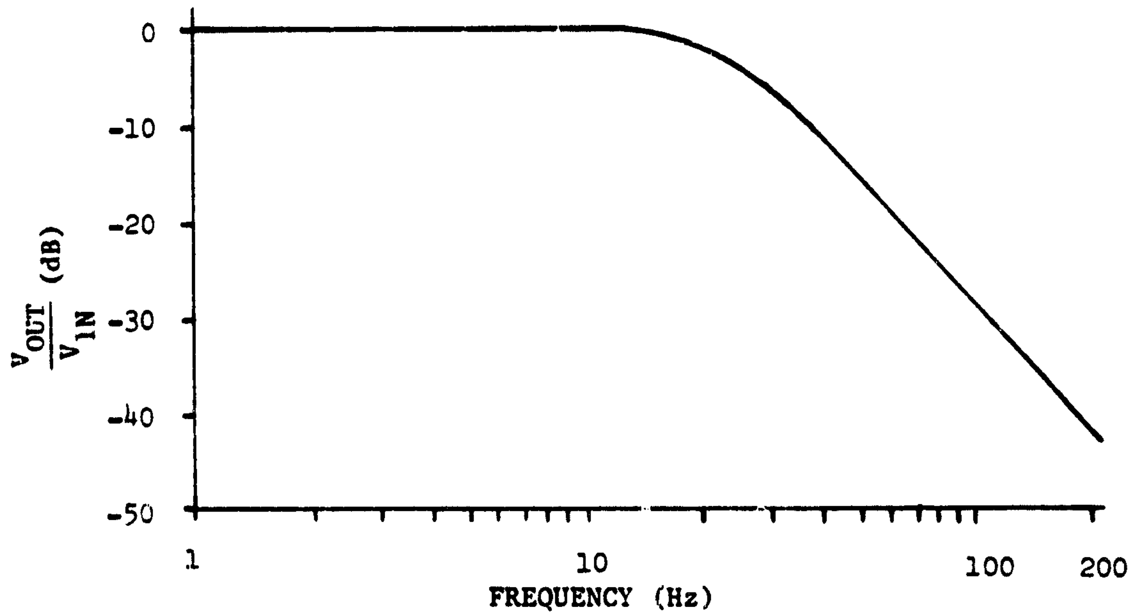


Figure 3.26 Measured Filter Frequency Response

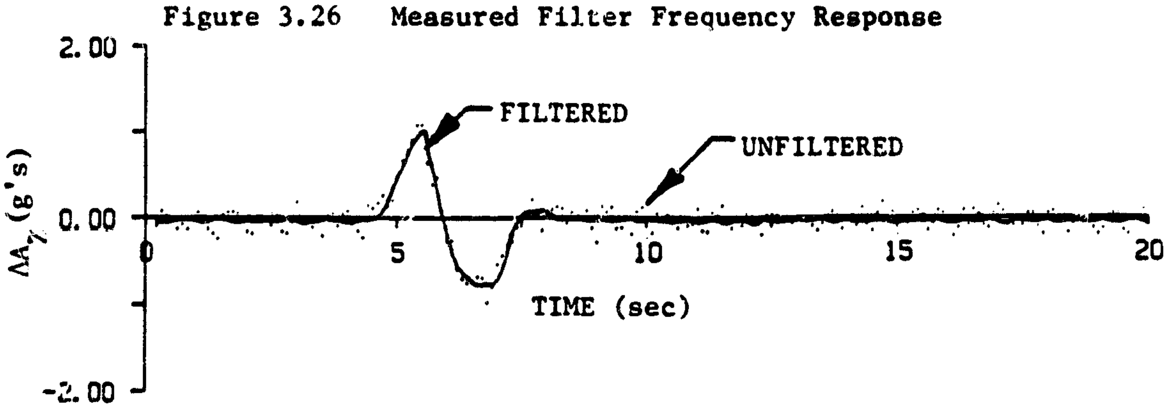


Figure 3.27 Filtered and Unfiltered in-Flight Measurements

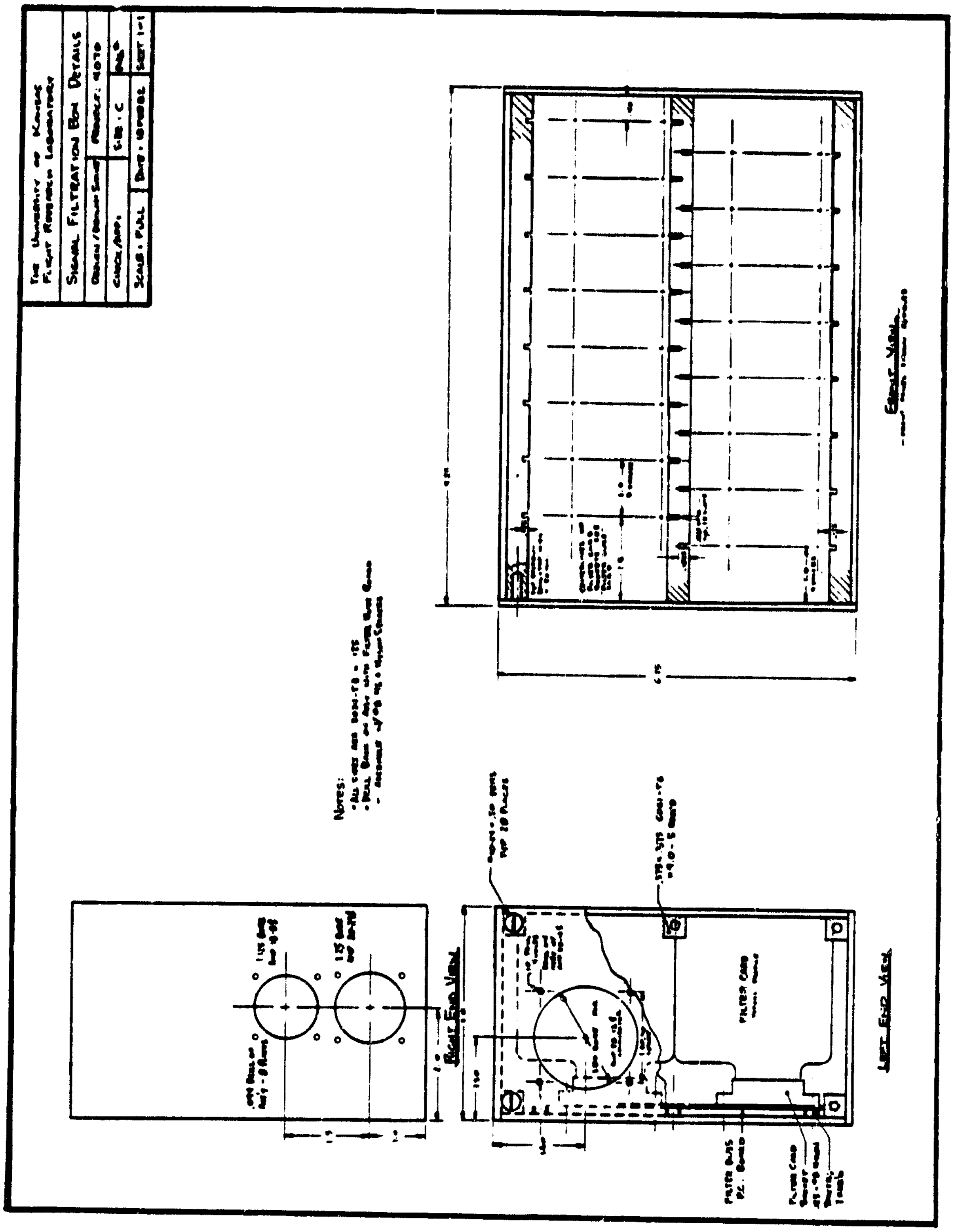


Figure 3.28 Filter Box Details

Option one offered several advantages over the second option. These included a significant size and weight saving as well as no limitations to flight time due to depleted batteries. It was realized, however, that two disadvantages also existed. Due to the fact that two separate voltage standards exist in current general-aviation-class airplanes, a complex voltage control system or several single-input voltage systems would be required. Furthermore, modifications to the electrical system of each airplane would be required during system installation.

Further research into the second option uncovered a suitable rechargeable battery, manufactured by Eagle-Picher. These lead-acid batteries are sealed, rechargeable, and maintenance free. A typical discharge curve is shown in Figure 3.29. For flight test purposes the batteries were used in a deep cycle regime; i.e., removing 50-100% of the rated battery capacity prior to recharge. In this type of application, the recharge time is 12 to 20 hours, and the battery expected lifetime is 100 to 150 complete discharge/charge cycles. The batteries are usable in any position. The cost of these batteries is low enough that several battery packages could be purchased for less than the price of a regulated voltage divider (of the type required if the airplane electrical systems were used).

The voltages required for the on-board data acquisition system are shown in Table 3.4. Batteries were sized to match the power requirements at each of the voltages. The wiring diagram and battery specifications are shown in Figure 3.30. The voltage regulation box

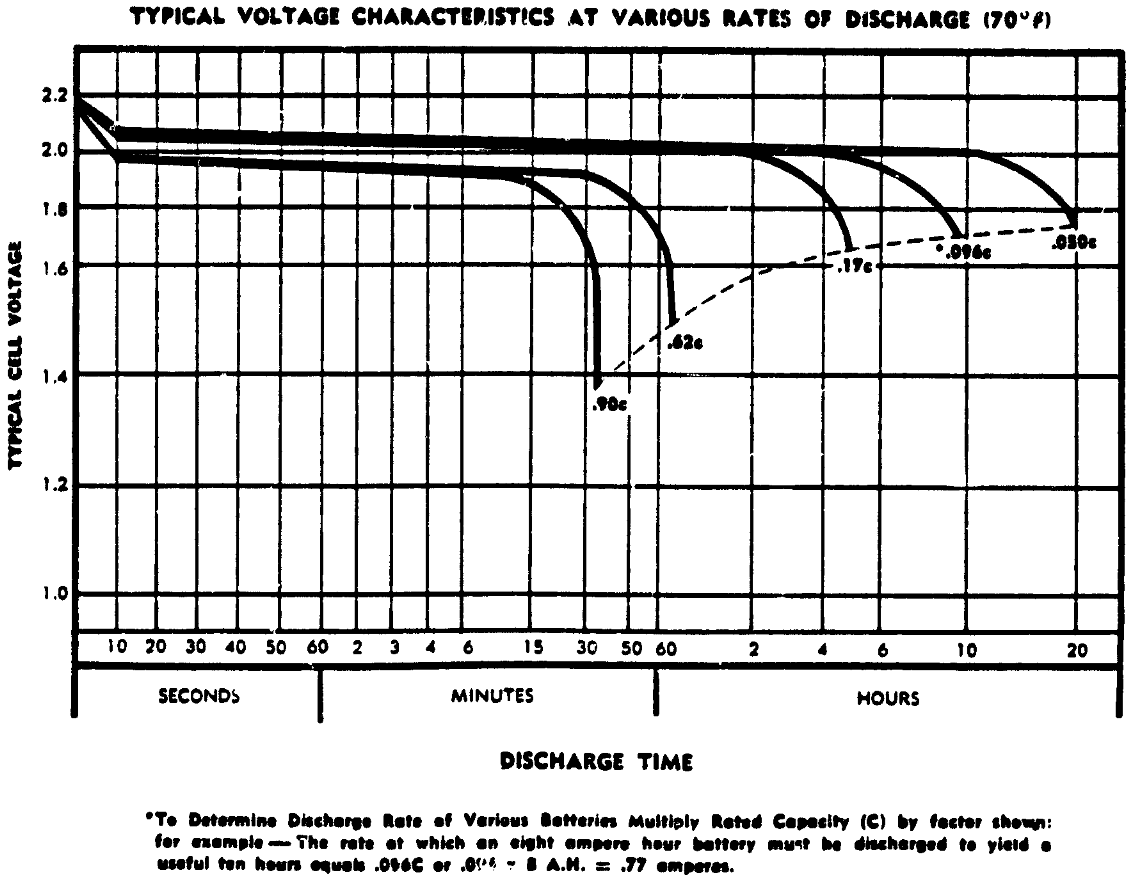


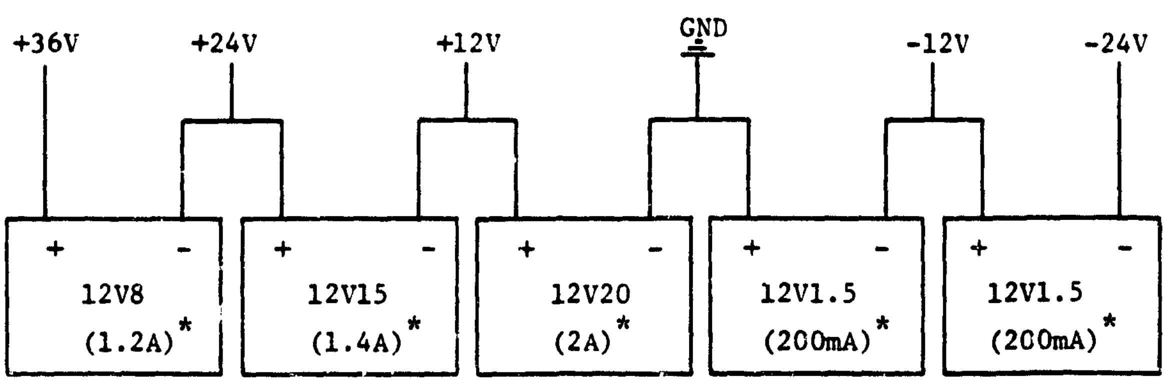
Figure 3.29 Typical Eagle-Picher Battery Discharge Curve

is shown in Figure 3.31. This system supplies a maximum of 2 1/2 hours of running time between recharges.

The largest disadvantage to using batteries for power is the weight. The battery module weighs 61.3 lbs. As the table in Figure 3.2 shows, this is the heaviest component in the data acquisition system. For the majority of general-aviation-class airplanes, this weight is not a problem.

Table 3.4 Power Requirements for the Data Acquisition Package

BATTERY VOLTAGE	REGULATED VOLTAGE	REQUIREMENT
+36	+28	Heater(P_S,P_D) Gyro motors(θ,ϕ.p,q,r)
+24	+15	Accelerometers(A_X,A_Y,A_Z) Filters, MDAS-16
	+12	TEAC tape drive
+12	+5.5	P_S and P_D reference voltage
	+5	Potentiometers (θ,ϕ,δ_E,δ_A,δ_R) AIM 65 computer Temperature transducer
-12	-5	Potentiometers (θ,ϕ,δ_E,δ_A,δ_R)
-24	-15	Accelerometers(A_X,A_Y,A_Z) Filters, MDAS-16



* maximum current requirement

		NOMINAL CAPACITY				DIMENSIONS (INCHES)				
BATTERY NUMBER	NOMINAL VOLTAGE	20 HR	10 HR	5 HR	1 HR	LENGTH	WIDTH	HEIGHT	TO TERMINAL	WEIGHT (LB)
CF12V20	12	20.0	19.0	17.5	12.5	6.51	4.91	6.53	6.75	16.2
CF12V15	12	15.0	14.5	13.0	9.0	7.22	3.34	6.50	6.75	12.8
CF12V8	12	8.0	7.7	7.0	5.0	6.00	4.00	3.75	3.97	7.5
CF12V1.5	12	1.5	1.4	1.3	0.9	7.02	1.33	2.40	2.60	' 9

Figure 3.30 Battery Module Specifications

ORIGINAL PAGE IS
OF POOR QUALITY

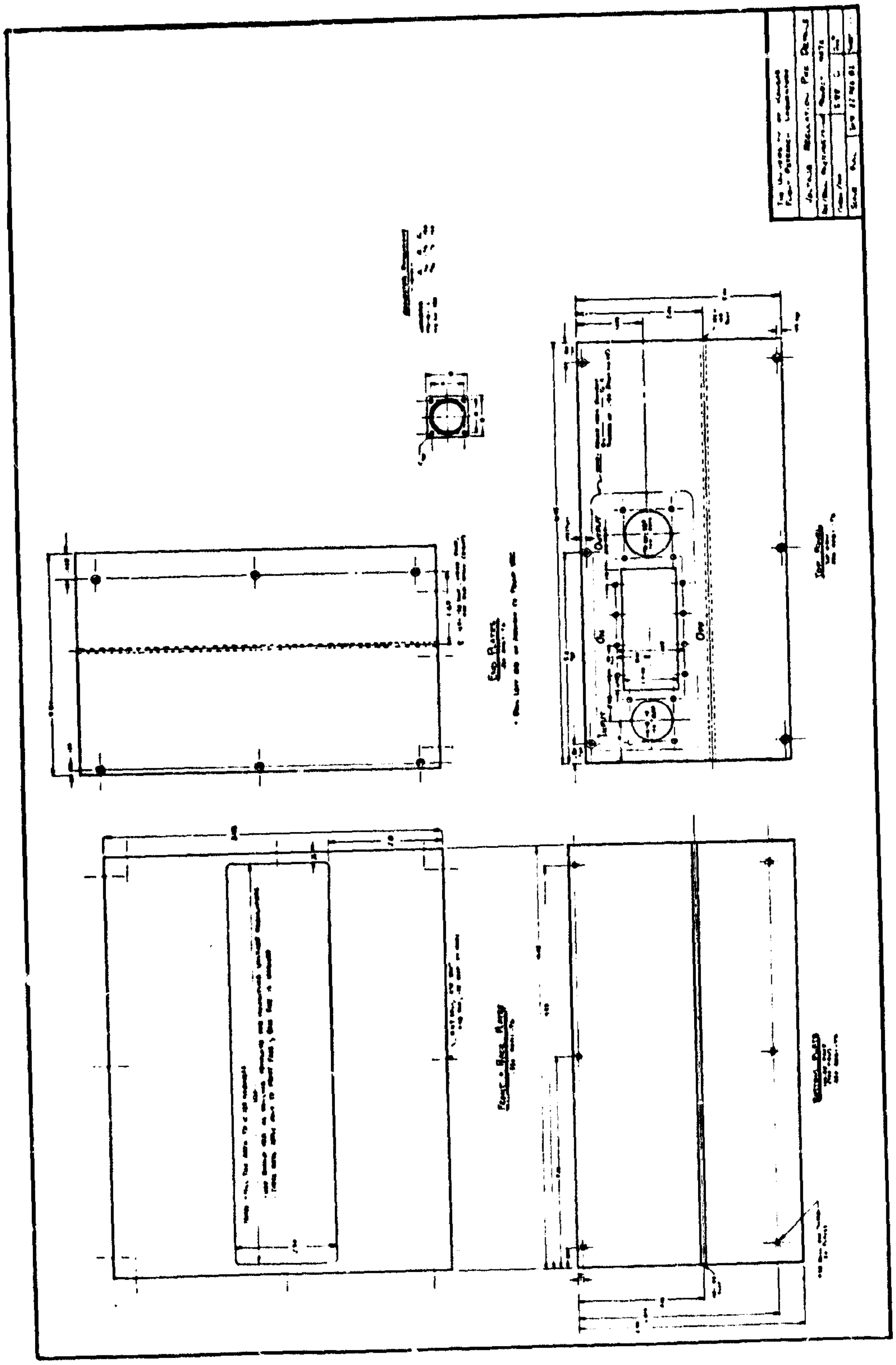


Figure 3.31 Voltage Regulation Box Details

3.7 Operator's Control

The flight engineer or pilot controls the data acquisition system through a small control box shown in Figure 3.32. The control box performs the same functions as the keyboard unit, allowing user input to the data acquision programs.

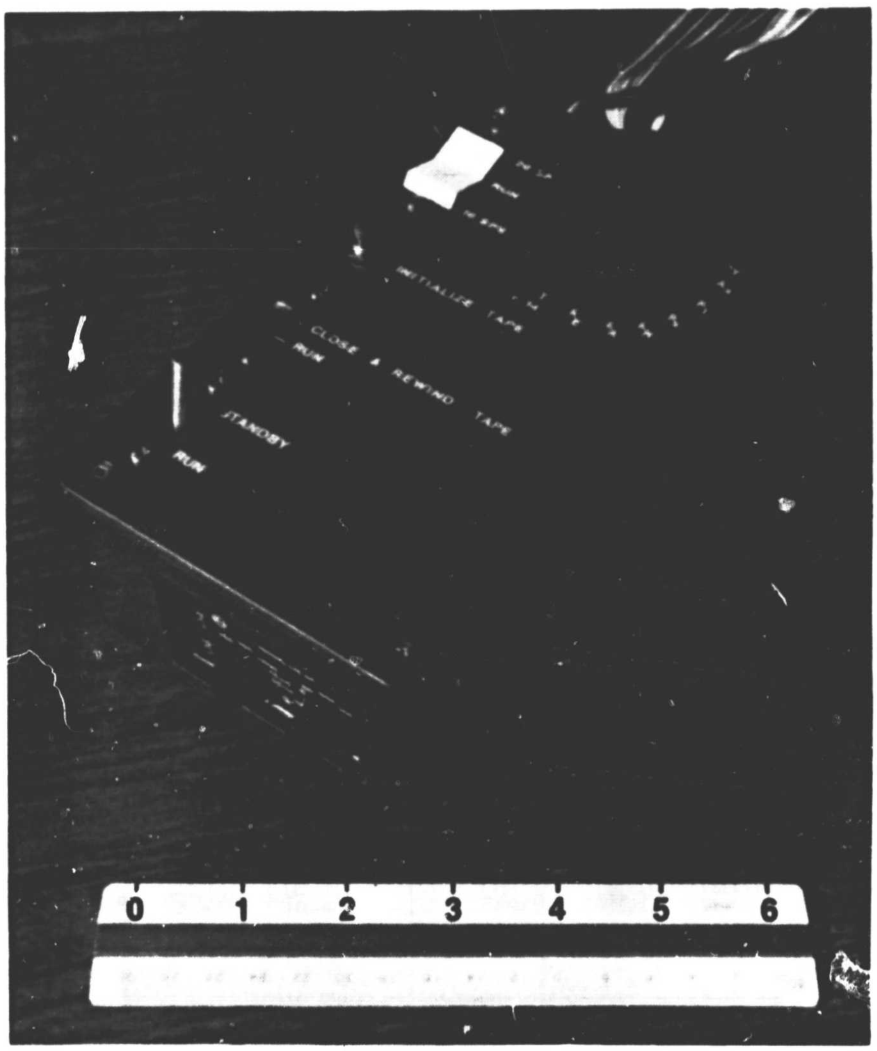


Figure 3.32 Operator's Control Box

The control box has four control switches:

- Program Selection Switch: This switch loads either the 10 or 20 SPS data acquisition program. This switch allows the operator to change sample rates during the flight test.
- Initialize Tape Switch: This switch initializes a fresh tape for proper use by the data acquisition program.
- Run/Standby Switch: This switch controls the data acquisition process. When it is in the run mode, data will be recorded. In the standby mode the system is not active.
- Rewind Tape Switch: This switch is used to place an "end" mark on the data tape and rewind the tape.

A high-impedance digital voltmeter is provided to the pilot or flight engineer so that he can observe a particular transducer output. The meter is installed in the control box, as shown in Figure 3.33. A rotary switch on the control box selects the signal to be observed.

This voltmeter also allows verification of correct functioning of all transducers prior to flight test

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

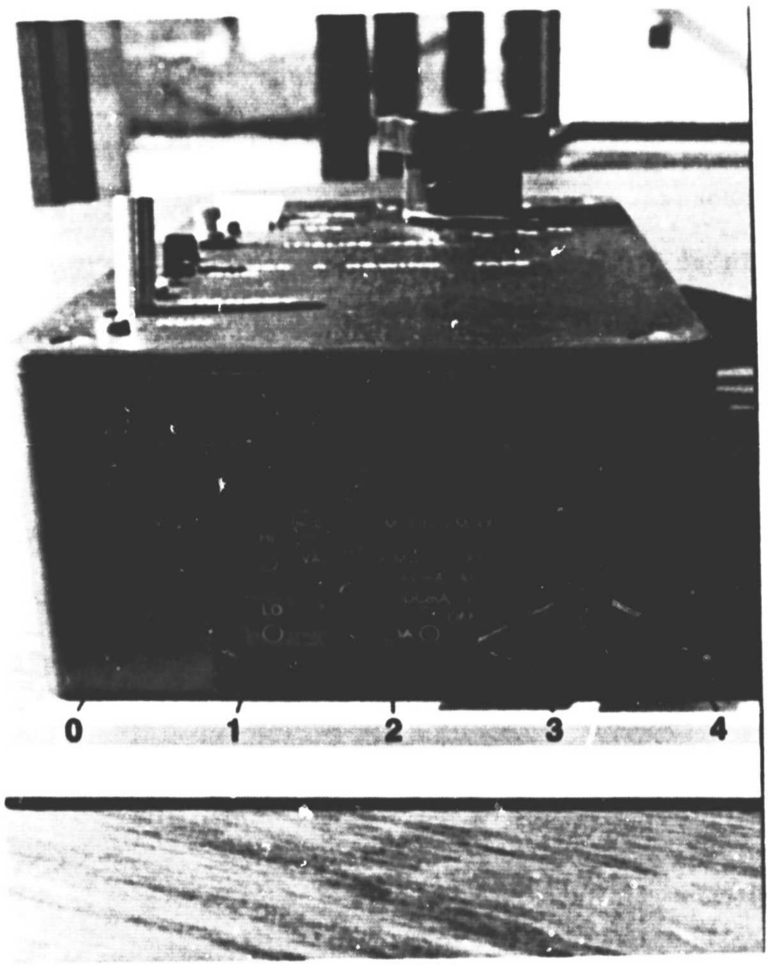


Figure 3.33 Operator's Control Box

4. GROUND-BASED COMPUTER SYSTEM

The Modified Maximum Likelihood Estimation (MMLE) data reduction process described in Chapter 5 requires a computer system capable of being programmed in a high-level language. Phase I (Reference 2) pointed out the requirement for a compiled language capability. This requirement is due to the long execution time associated with an interpretive language. (A Hewlett-Packard 9825A was used in Phase I. This desk-top calculator could only be programmed in interpretive Basic.) This chapter presents the results of a benchmark process undertaken in Phase II (Reference 3) used to evaluate the capability of numerous computer systems. Also included in the second section is a detailed description of the selected computer system.

4.1 Benchmark Routines

A two-step evaluation process was used to evaluate computer system speed. The first benchmark program is the INTEGER SPEED ROUTINE. This routine was easily implemented and gave a good speed estimate for each computer system. The second benchmark program which was run was the FLOATING POINT SPEED ROUTINE. This program more closely matched the kind and number of operations which took place in the MMLE program. Each of these programs is described in detail below. The INTEGER SPEED ROUTINE is a program that calculates the prime numbers up to 10,000. The FLOATING POINT ROUTINE is a program that performs operations similiar to those in the MMLE program. The routine was made to have the same run time on the HP 9825A as one iteration of the MMLE program (300 time points).

4.1.1 Integer Speed Benchmark

This routine was taken from Reference 30. A listing of this program is contained in Table 4.1. Although the program does not realistically reflect the MMLE program, it was a simple algorithm which was easy to program and gave a good ball-park estimate of computer speed.

This benchmark showed that all acceptable computers used a compiled language as well as a floating point processor. Table 4.2 contains the results of the integer speed benchmark. This routine shows no eight-bit computers capable of meeting the desired speed of 8 minutes maximum for completion of the routine. This study narrowed the number of computer systems which were evaluated further.

Table 4.1 Integer Speed Routine (FORTRAN Listing)

```
          DO 100 M=5,10000,2
          I=M/2
          DO 200 K=3,I,2
          J=(M/K)*K
          IF(J.EQ.M) GO TO 100
    200   CONTINUE
          PRINT,M
    100   CONTINUE
          STOP
          END
```

Table 4.2 Integer Speed Comparison

PROCESSOR	MACHINE	LANGUAGE	INTERPRETER	COMPILER	FLOATING POINT HARDWARE	HRS:MIN:SEC	ACCEPTABLE
8-BIT MICRO	AIM 65	BASIC (PRINTER OUTPUT)	*			4: 14: 44	
		ASSEMBLY (LED OUTPUT)				0: 23: 36	
		ASSEMBLY (PRINTER OUTPUT)				0: 33: 40	
	TRS 80	LEVEL I BASIC	*			7: 12: 27	
		LEVEL II BASIC	*			6: 31: 10	
		ASSEMBLY				0: 21: 55	
		FORTRAN		*		0: 54: 18	
		MODEL II BASIC	*			3: 15: 00	
	APPLE II	INTEGER BASIC	*			2: 24: 31	
		FLOATING POINT BASIC	*			3: 56: 23	
16-BIT MICRO	TERAC 8510	PASCAL (COMPILE TO P CODE)	*	*		0: 30: 35	
	TEKTRONIX (4052)	BASIC	*			1: 23: 00	
	HP 9825	BASIC	*			1: 41: 17	
	HP 1000	FORTRAN RTE IV B (CRT OUTPUT)		*	*	0: 01: 23	✓
		" (NO OUTPUT)		*	*	0: 00: 48	✓
		FORTRAN RTEM (CRT OUTPUT)		*	*	0: 00: 57	✓
		" (NO OUTPUT)		*	*	0: 00: 44	✓
	IBM SERIES I	FORTRAN (NO OUTPUT)		*	*	0: 01: 30	✓
		" (PRINTER OUTPUT)		*	*	0: 04: 30	✓
	PDP 11/34*	FORTRAN (RSX 11 M) (CRT OUTPUT)		*	*	0: 07: 10	✓
		" (PRINTER OUTPUT)		*	*	0: 11: 20	
	MINC 11/23	FORTRAN RT11-IV (CRT OUTPUT)		*	*	0: 03: 36	✓
		" (DISC OUTPUT)		*	*	0: 03: 29	✓
		FORTRAN RT11-IV PLUS (CRT OUTPUT)		*	*	0: 03: 10	✓
		" (DISC OUTPUT)		*	*	0: 03: 00	✓
MAIN FRAME	HONEYWELL 60/66	FORTRAN		*	*	0: 00: 44	✓
		PL/1		*	*	0: 02: 13	✓
	CDC CYBER 70	FORTRAN (NON OPTIMIZED)		*	*	0: 00: 39	✓
		FORTRAN (OPTIMIZED)		*	*	0: 00: 37	✓
	IBM 370-148	PL/1 (OPTIMIZED)		*	*	0: 01: 19	✓

*The PDP 11/34 was operating in a multi-user mode. Its performance is estimated to be approximately 2-3 times faster than the 11/23 series computer in single-user mode.

4.1.2 Floating Point Speed Benchmark

An attempt was made to model more closely the processes which took place in the MMLE program. This routine is shown in Table 4.3. The program primarily contains floating point matrix mathematics.

Table 4.4 shows the run time for various computer systems. A maximum acceptable speed was 6 minutes for this routine (our speed requirement).

Table 4.3 Floating Point Speed Routine
(FORTRAN Listing)

```
      REAL A(20,20),B(20,20),C(20,20),E(20,20),T(20,20),D(20,20),F
      INTEGER I,J,K,M
      PRINT,"START"
      F=.098625
      DO 400 M=1,40
              DO 200 I=1,20
              DO 200 J=1,20
                      E(I,J)=0
                      A(I,J)=F*I*J
                      B(I,J)=F*I
                      C(I,J)=F*J
                      D(I,J)=F
                      T(I,J)=0
200           CONTINUE
              DO 300 I=1,20
              DO 300 J=1,20
              DO 300 K=1,20
                      T(I,J)=T(I,J)+(A(I,K)*B(K,J))
                      E(I,J)=E(I,J)+(E(I,K)*D(K,J))
300           CONTINUE
              DO 400 I=1,20
              DO 400 J=1,20
400                   E(I,J)=E(I,J)+T(I,J)
              PRINT,"E="
              DO 100 I=1,20
              DO 100 J=1,20
                      PRINT,E(I,J)
100   CONTINUE
      PRINT,M
      STOP
      END.
```

Table 4.4 Floating Point Speed Comparison

MACHINE		MIN:SECS
HP9825	(BASIC)	48:15
HONEYWELL 60/66		0:20.6
HP1000	(NO OUTPUT)	1:08.7
	(DISC OUTPUT)	2:07
IBM SERIES 1	(DISC OUTPUT)	0:58
MINC 11/03	(DISC OUTPUT)	5:35
MINC 11/23	(DISC OUTPUT)	4:00

4.2 Computer System Description

Several factors besides speed were considered in the evaluation of computer systems. The following factors were considered:

- Cost
- Floating Point Hardware availability
- RS232 ports/General Purpose Interface Buss (GPIB) availability
- CRT graphics capability
- Hard/Flexible disk storage capability
- Programming language availability
- Existence of a user's group
- Delivery time.

After evaluation of all acceptable computers, the Digital Equipment Corporation (DEC) MINC 11/03 computer was selected as best meeting the requirements. A description of this computer follows.

The MINC 11/03 is shown in Figure 4.1. The block diagram of Figure 4.2 shows the basic features and some of the options available.

The computer uses a 16-bit DEC LSI 11/03 processor, capable of addressing 64K bytes of memory, and contains a floating point hardware package, four RS232 ports, and a GPIB port.

Data and program storage is handled using the dual RX02 flexible disk drives. These use 8" flexible disks, capable of holding 500 K bytes of information each.

Computer and program interaction is handled using the DEC-VT 105 graphics terminal. This permits inputting and outputting of data, as well as allowing graphical representation of the flight test results.

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

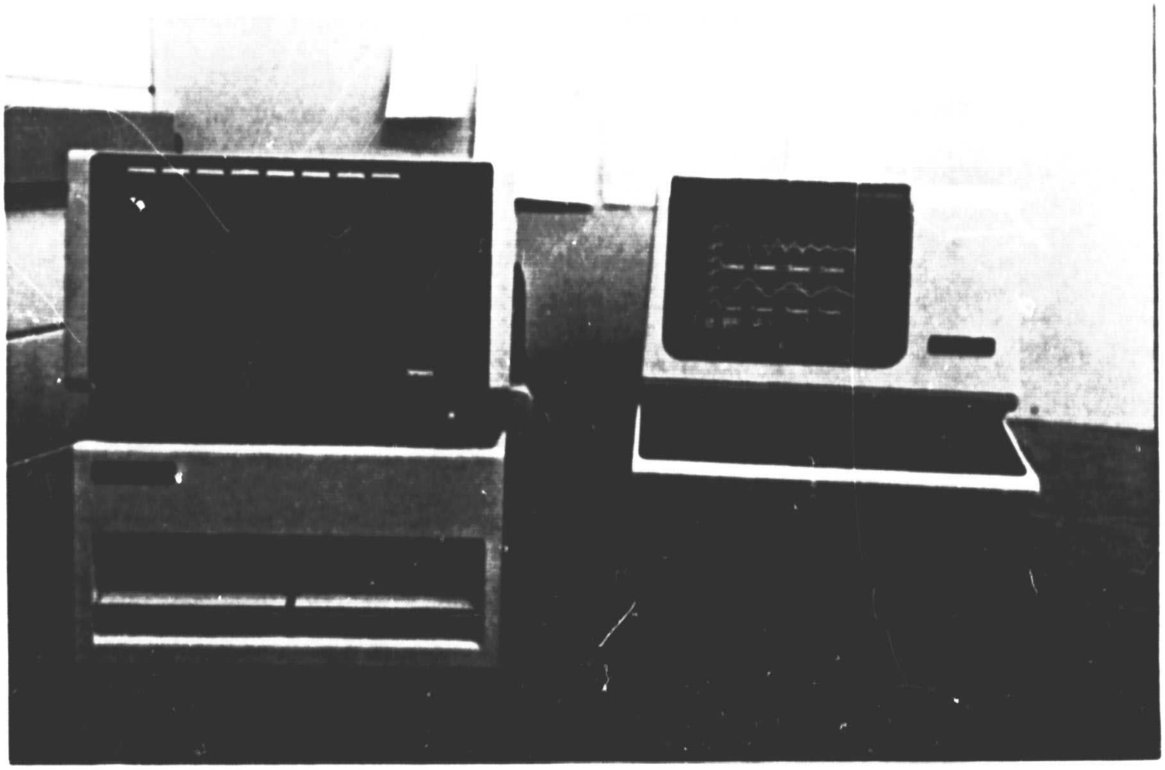
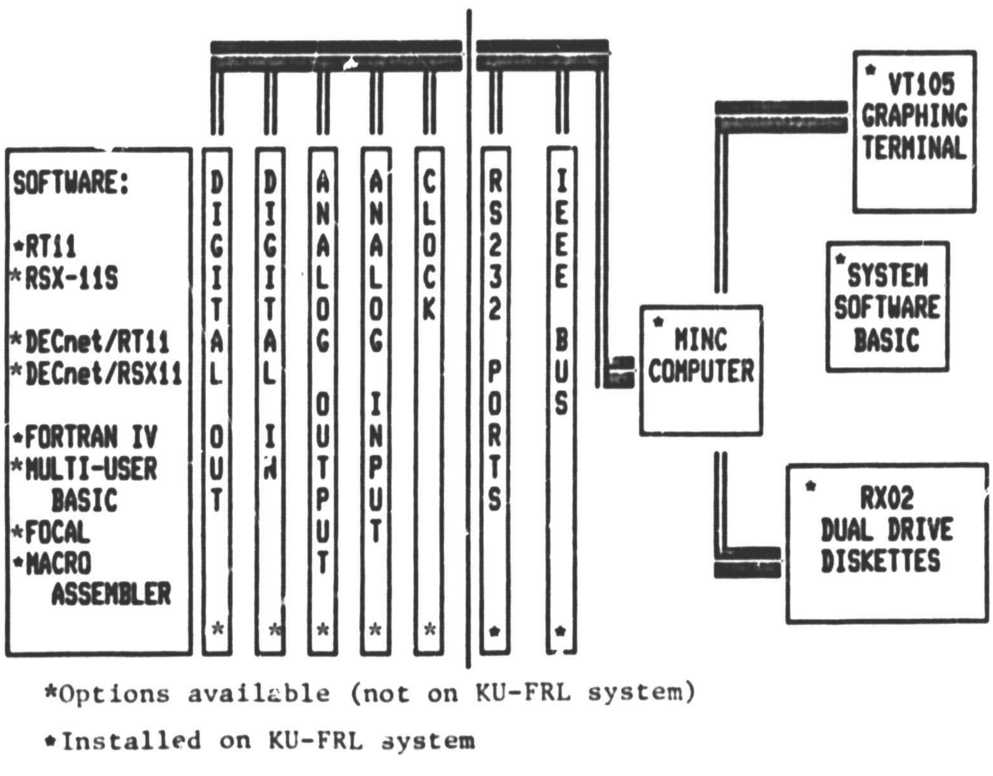


Figure 4.1   Digital Equipment Corporation MINC 11/03 Computer



*Options available (not on KU-FRL system)

•Installed on KU-FRL system

Figure 4.2   MINC 11/03 Computer System Block Diagram

The RS232 ports are used for input and output of the data. Four are provided. One is used for the VT 105 terminal, one is configured to allow data transfer from the Rockwell AIM-65, one is configured to be modem compatible, and one is used to control a hard-copy printer.

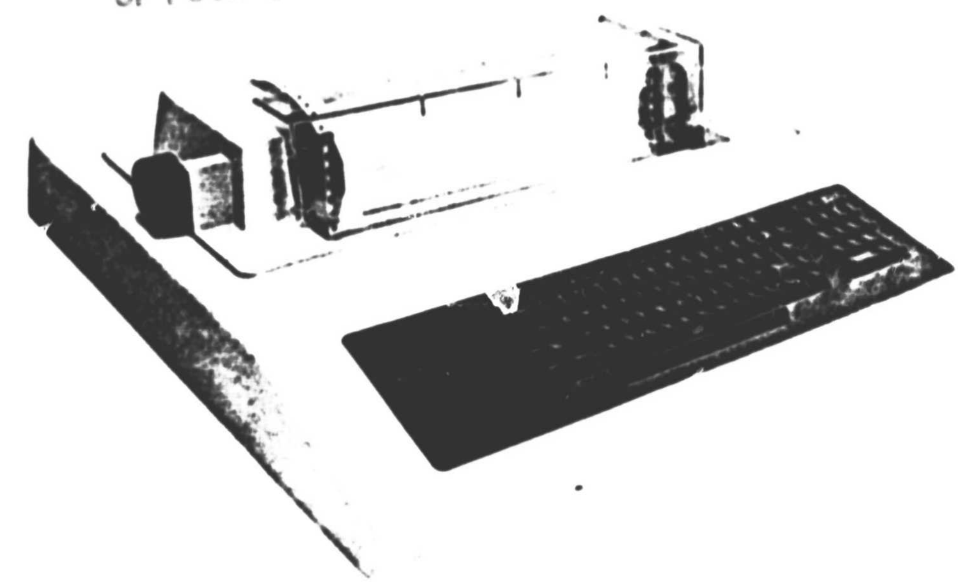
The GPIB port allows ease of interfacing to many industry standard components. This port currently connects the computer to a hard-copy plotter for analysis and report-quality plots of flight test data.

The standard MINC comes with Basic language software. The KU-FRL package has the RT11-FORTRAN IV software option. This version of FORTRAN allows compiling programs to machine level, which was determined necessary to perform the data reduction task as indicated in Section 4.1.

The MINC computer has been found capable of performing the functions intended. The MMLE process takes approximately 20 minutes for 5 iterations, which is close to the prediction from the floating point benchmark routine.

A DEC LA120 hard-copy terminal was added to the computer system. The LA120 and its specifications are shown in Figure 4.3. This printer is capable of printing at a maximum rate of 180 characters per second. The unit is interfaced to the MINC 11/03 computer system through an RS232 serial interface port.

A Hewlett-Packard 7225B graphics plotter was also added to the computer system. Its specifications are shown in Figure 4.4.



LA120 SPECIFICATIONS

Printer	
Printing technique	Impact dot matrix, smart bidirectional
Print matrix (width by height)	7 by 7
Maximum print speed	180 CPS
Horizontal slew speed	60 IPS
Single linefeed time	33 ms
Vertical slew speed	7.5 IPS
Paper feed	Pin-feed, tractor drive
Paper type	Fanfold, up to six parts (see paper requirements)
Forms length	1 to 168 lines
Vertical pitch (lines per inch)	2,3,4,6,8,12
Horizontal pitch (characters per inch)	
180 CPS	10,12,13.2,16.5
90 CPS	5,6,6.6,8.25
Maximum line length (varies with horizontal pitch)	
5 CPI	66 columns
6 CPI	79 columns
6.6 CPI	87 columns
8 CPI	109 columns
10 CPI	132 columns
12 CPI	158 columns
13.2 CPI	174 columns
16.5 CPI	217 columns
Margins	Left, right, top, bottom
Tabs	217 horizontal, 168 vertical, from keyboard or line
Forms storage	True nonvolatile memory (no batteries)
Positioning commands	Horizontal and vertical, absolute and relative
Character set	ASCII upper/lowercase set
National character sets	
Standard	United States; United Kingdom
Optional	Finland, Sweden, Norway, Denmark, Germany, France
APL character set	Optional
Other printer features	Paper out and cover open interlocks, manual and automatic last character view, selectable auto new line, self-test, status message, 4-digit numeric display used as column counter and to set parameters, factory stored form setup (10 CPI, 6 LPI, 66 lines per form, tab stops every eight columns, etc.)

Keyboard Specifications	
Keyboard	Typewriter style with multi-key rollover
Selectable auto LF	Standard
Optional numeric keypad	18 keys including 4 function keys
Feature selection	Keyboard entry to nonvolatile memory
Other keyboard features	Local form feed key, local line feed key, auto repeat on all alphanumeric keys, and selectable keyclick

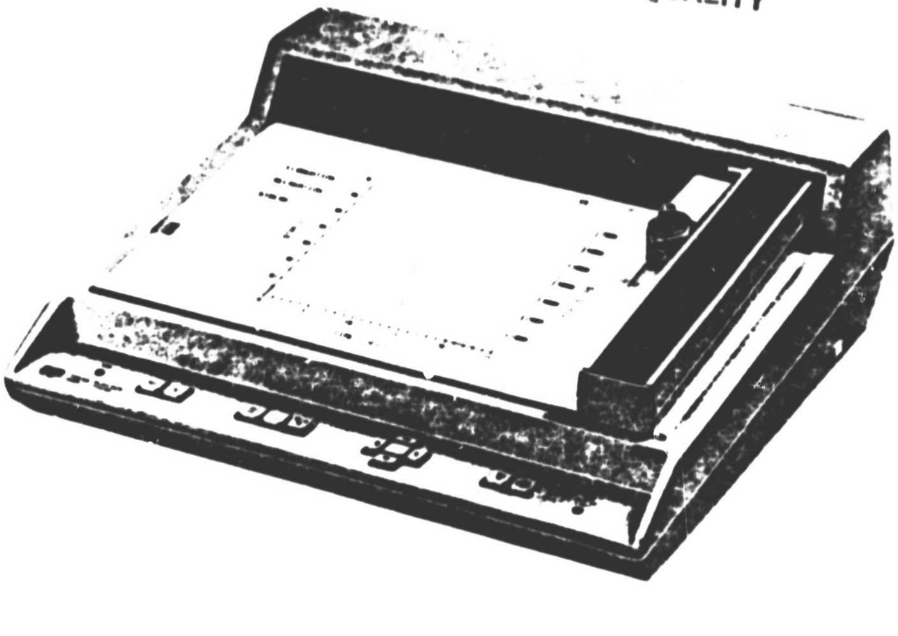
Communication Specifications	
Data transfer	Serial asynchronous
Baud rates (BPS)	50 75 110 134 134.5 150 300 600 1200 1800 2400 4800 7200 9600
Split speeds (BPS)	600 or 1200 receive, with 75 or 150 transmit; 2400 or 4800 receive with 300 or 600 transmit
Parity	Odd, even, or none (8th bit mark or space transmitted, or data bits only)
Input buffer	1024 characters standard; 4096 characters optional
Interface	Full EIA standard (includes auto answer/disconnect)

Physical Specifications	
Dimensions	
Width	69.9 cm (27.5 in)
Height	85.1 cm (33.5 in)
Depth	61.0 cm (24.0 in)
Weight	
Uncrated	46.4 kg (102 lb)
Shipping	63.7 kg (140 lb)

Physical Specifications (Cont)	
Power	
Transformer power supply	
Voltage	87 to 128 V
Frequency	60 Hz ± 1 Hz
Switcher power supply	
Voltage	90–128 V or 180–256 V
Frequency	47–63 Mhz
Input current	4.2 A max at 115 V
Heat dissipation - printing	440 W max
Temperature	
Operating	10° to 40°C (50° to 104°F)
Nonoperating	−40° to 66°C (−40° to 151°F)
Relative Humidity	
Operating	10 to 90 percent with a maximum wet bulb temperature of 28°C (82°F) and a minimum dewpoint of 2°C (36°F), noncondensing
Nonoperating	5 to 95 percent, noncondensing

Paper Requirements	
General	Continuous, fanfold, pin-feed forms
Width	7.6 to 37.8 cm (3 to 14 7/8 in)
Hole spacing	12.7 mm ± 0.25 mm (0.500 in ± 0.010 in) non-accumulative over 5 cm (2 in)
Hole diameter	3.81 to 4.06 mm (0.150 to 0.160 in)
Forms thickness	
Single part	15 lb paper minimum, 0.25 mm (0.010 in) card stock maximum
Multipart	Up to 6 parts (see notes), 0.50 mm (0.020 in) maximum

Figure 4.3 DEC LA120 Printer Specifications



Specifications

Plotting Area
Y axis: 203 mm (8 in.)
X axis: 285 mm (11.2 in.)
Accepts up to ISO A4 or 8½ x 11-in. chart paper

Plotting Accuracy
±0.25 mm (0.01) includes linearity and repeatability, assuming the plotter has been "zeroed" exactly to the lower left (0,0) coordinates

Repeatability
0.1 mm (0.004 in.) from any given point and direction

Addressable Step Size
0.032 mm (0.0013 in.) smallest addressable step

Pen Velocity
250 mm/s (10 in./s) in each axis.
350 mm/s (14 in./s) on 45° angle

Vector Length
No limit — any length vector within the plotter's mechanical limits will be plotted

Character Plotting Speed
Up to 3 characters/s for 2.5-mm (0.1-in.) characters

Power Requirements

Source:	100, 120, 220, 240V, -10%, +5%, internally selectable
Frequency:	48-66 Hz
Consumption:	70 W maximum

Environmental Range

Temperature:	0° C to 55° C
Consumption:	5% to 95% (below 40° C)

Size/Weight

Height:	140 mm (5.5 in.)
Width:	413 mm (16.3 in.)
Depth:	379 mm (14.9 in.)
Net Weight:	8 kg (17.6 lb)
Shipping Weight:	11.4 kg (25 lb) approximately

Figure 4.4 Hewlett Packard 7225B Digital Plotter Specifications

The unit is interfaced to the MINC 11/03 through the GPIB interface. The plotter is capable of plots of up to 8 1/2 x 11 inches in size.

PRECEDING PAGE BLANK NOT FILMED

5. DATA REDUCTION METHOD

This chapter describes the computer analysis procedures used for longitudinal and lateral-directional stability parameter estimation. The overall data analysis procedure is shown in Figure 5.1.

For Phase III the system described in Chapter 3 was used for sensor calibration and airborne data acquisition. The MINC 11/03 computer described in Chapter 4 was used for all data analysis. The computer required much of the analysis procedure to be subdivided into small tasks due to its memory limitations.

The program listings for the computer analysis programs described in this chapter are contained in Appendix A.

5.1 On-Board Computer Programs

The on-board AIM-65 computer is used for three basic purposes: (1) calibration of the instrumentation system, (2) airborne data acquisition, and (3) data transfer to the ground-based computer system (see Appendix A.1 for flowcharts and listing).

5.1.1 Instrument Calibration

The instrument calibration program is used to calibrate the on-board computer and instruments as a complete system. The program is entered by pressing the F2 special function key. It allows any of the 16 analog channels to be accessed through the MDAS-16 at a very high sample rate and displayed on the LED display. The program

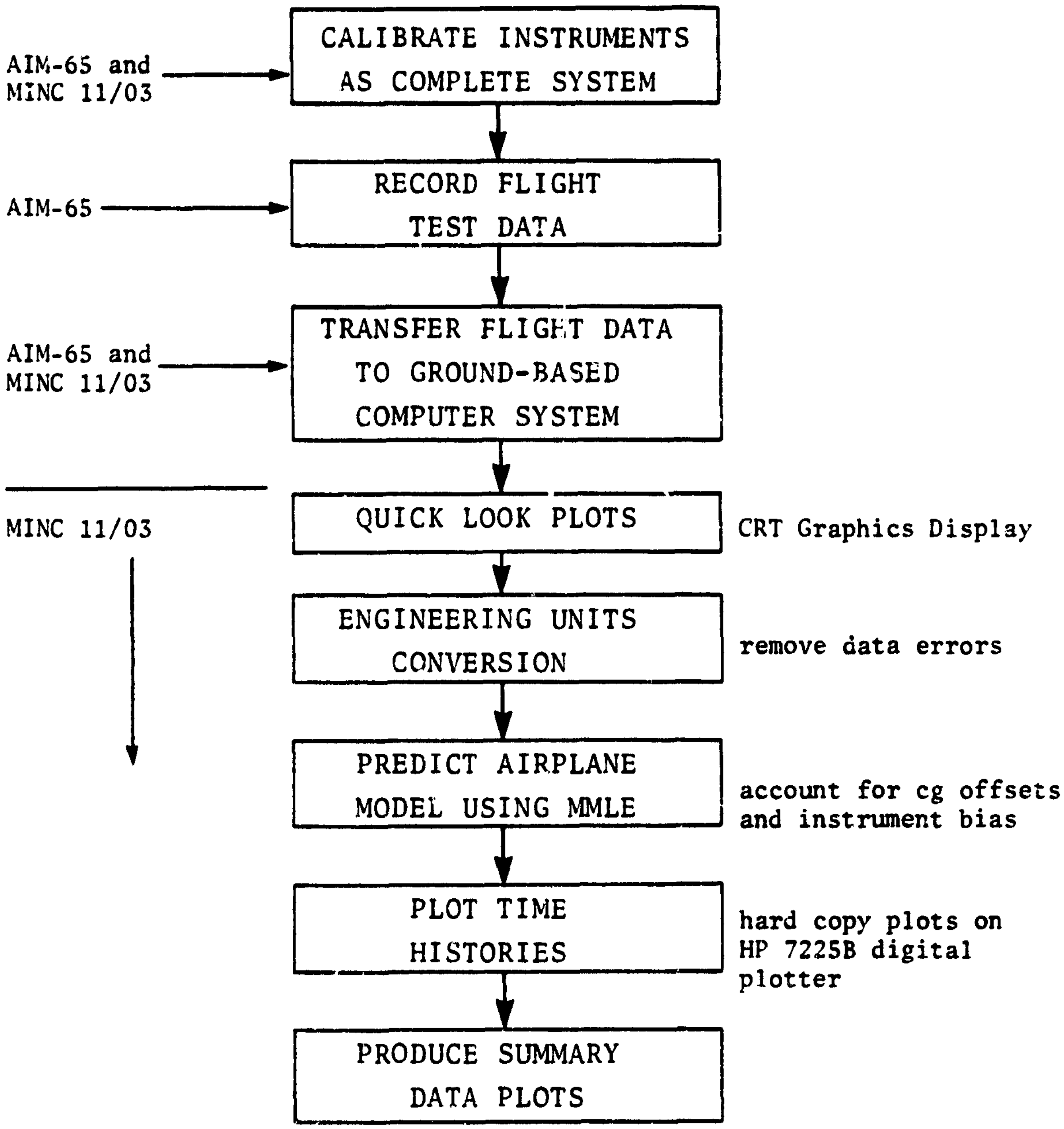


Figure 5.1 Overall Flight Test Data Processing Flowchart

calibrates the instruments without the intermediate steps of calibrating to voltage levels. It is written in assembly language for fast execution. Reference 31 explains in detail the use of the calibration program.

5.1.2 Airborne Data Acquisition

This program is used by the AIM-65 for the control of the MDAS-16 and TEAC digital tape drive. It is written in assembly language and is capable of sampling and recording 16 channels of data continuously at either 10 or 20 SPS.

This program has four control inputs which are located on the operator's control box. Upon power-up, either the 10 or the 20 SPS program can be loaded into the computer. These programs are loaded by a single, double-pole, toggle switch. The switch is toggled to load the program and then returned to its center position to run the program. Once a program is selected, the tape must be initialized. This is accomplished with the "INITIALIZE TAPE" button. Pushing the button causes the computer to rewind the cassette and write a beginning-of-tape mark. This mark is used by the computer for locating the data.

Once the computer has initialized the tape, the "RUN/STANDBY" toggle switch is used to control the data acquisition process. Placing this switch in the "RUN" position causes the computer to begin sampling its 16 channels sequentially and recording their output on the digital cassette. The data are sampled continuously at 10 or 20 SPS. This sampling and recording process continues until the "RUN/STANDBY" switch is placed in the "STANDBY" position. In this mode, the computer remains idle until the "RUN" mode is again selected.

As a method of checking for data recording errors, the four highest bits of the 12-bit digital measurement are recorded twice. These bits are compared when the data is dumped, and differences are flagged as errors.

The final switch on the control box is the "CLOSE & REWIND TAPE" switch. This switch is used to write an end-of-tape mark on the cassette and rewind the tape.

5.1.3 Transfer of Flight Data to the Ground-Based Computer

This data transfer program dumps the flight test data from the AIM-65 computer system to a ground-based computer system for data analysis. The data is dumped from the AIM-65 computer across an RS232 serial interface. The program is entered by pressing the F1 special function key.

A running total of any data errors is kept and printed out by the AIM-65 on its thermal printer. Most errors to date are believed to have been caused by poor quality data cassettes. Using the qualified cassettes (see Table 5.1 and Reference 26), few data errors have been found in the flight data.

The MINC 11/03 program which accepts the data from the AIM-65 is shown in Appendix A.2. This program is used to transfer the flight data from the TEAC cassette tape to the MINC 11/03 disk. In this

Table 5.1 KU-FRL Qualified Data Cassettes

Manufacturer	Type	Part No.
3M	Scotch	834A/1-300
TDK	Data Cassette	HR-850 90C
MAXELL	Data Cassette	M-90
BASF	Digital Power Typing Cassette	52346

(Qualified as per Reference 26)

mode the AIM-65 keyboard is used for controlling the data transfer. The MINC 11/03 program loads the transferred data into its memory and then transfers this data to the data disk.

5.2 Data Analysis Programs

The data analysis programs are that group of programs which take the flight test data from its AIM-65 format to engineering units, perform the MMLE analysis, and present the results.

5.2.1 Quick Look Plots

The Quick Look Plots program is the program which allows rapid examination of the flight test data on the VT 105 graphics CRT. The listing of this program is contained in Appendix A.3. Upon examination, the operator can decide which runs should be subjected to further analysis.

5.2.2 Engineering Units Conversion

This program is used to take the raw AIM-65 formatted data and translate it into engineering units. This program only accounts for sensor calibrations; the instrument position corrections are accounted for in the MMLE program itself. The program listing is contained in Appendix A.4.

5.2.3 Modified Maximum Likelihood Estimation Routine

The flight data were processed through the Modified Maximum Likelihood Estimator (MMLE) developed by NASA (see References 5 and 16-19). This technique has been used by NASA for over 12 years. A simplified MMLE program (NASA Dryden "BONES" version) has been placed on the MINC 11/03 computer and updated. The actual program listings are included in Appendix A.7. Described here is the theory used in this technique, and some of the assumptions made for the KU-FRL version.

The MMLE estimator is an iterative process that determines the coefficients of a given set of linear differential equations describing the motion of the aircraft. It does this by comparing the difference between actual in-flight measured responses of various states, and the predicted responses of these states using an estimate of the coefficients. The actual measured control input is used as the input for the estimating procedure. The estimated coefficients are updated each iteration, using a cost function minimization algorithm. The flow chart below shows the MMLE concept.

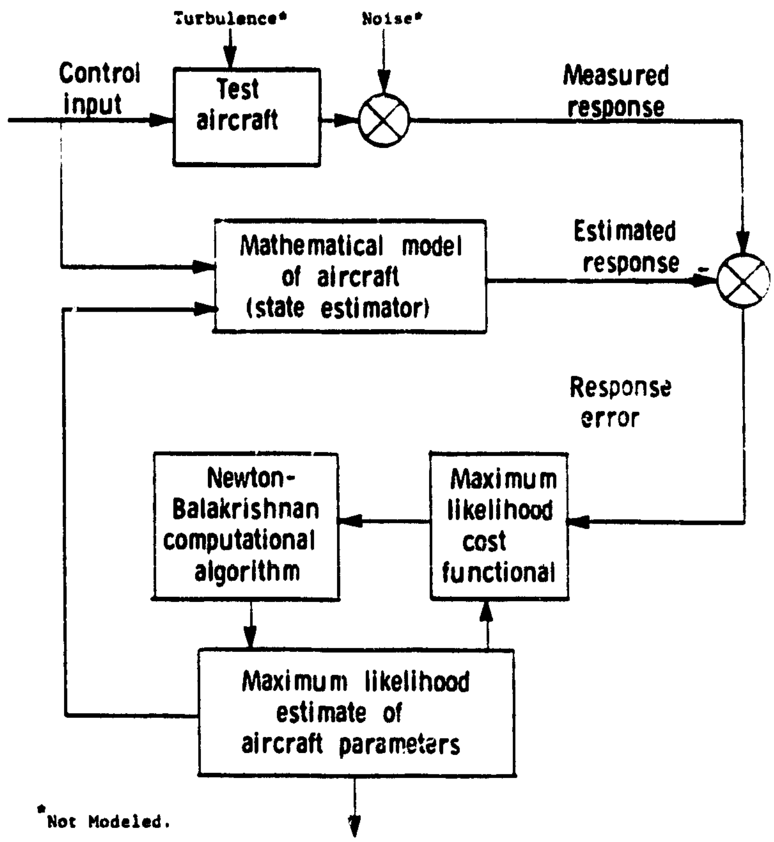


Figure 5.2 Maximum Likelihood Estimation Concept (from Reference 17)

The mathematical model used to describe the airplane is derived from the small perturbation equations of motion (see Reference 25).* These are shown here explicitly, in the nondimensional form.

*The derivatives in Reference 25 are for the stability axes system. See Appendix C for conversion to the body axes used in this report.

- for longitudinal (from Reference 25, Equation 6.1):

$$m\dot{u} = -mg\theta\cos\Theta_1 + \bar{q}_1 S\{-(C_{D_u} + 2C_{D_1})\frac{u}{U_1} + (C_{T_{x_u}} + 2C_{T_{x_1}})\frac{u}{U_1} - (C_{D_\alpha} - C_{L_1})\alpha - C_{D_{\delta_E}}\delta_E\}$$

$$m(\dot{w} - U_1 q) = -mg\theta\sin\Theta_1 + \bar{q}_1 S\{-(C_{L_u} + 2C_{L_1})\frac{u}{U_1} - (C_{L_\alpha} + C_{D_1})\alpha - C_{L_{\dot{\alpha}}}\frac{\dot{\alpha}\bar{c}}{2U_1} - C_{L_q}\frac{q\bar{c}}{2U_1} - C_{L_{\delta_E}}\delta_E\} \quad [5.1]$$

$$I_{yy}\dot{q} = \bar{q}_1 S\bar{c}\{(C_{m_u} + 2C_{m_1})\frac{u}{U_1} + (C_{m_{T_u}} + 2C_{m_{T_1}})\frac{u}{U_1} + C_{m_\alpha}\alpha + C_{m_{T_\alpha}}\alpha + C_{m_{\dot{\alpha}}}\frac{\dot{\alpha}\bar{c}}{2U_1} + C_{m_q}\frac{q\bar{c}}{2U_1} + C_{m_{\delta_E}}\delta_E\}$$

- for lateral (from Reference 25, Equation 6.2):

$$m(\dot{v} + U_1 r) = mg\phi\cos\Theta_1 + \bar{q}_1 S(C_{y_\beta}\beta + C_{y_p}\frac{pb}{2U_1} + C_{y_r}\frac{rb}{2U_1} + C_{y_{\delta_A}}\delta_A + C_{y_{\delta_R}}\delta_R)$$

$$I_{xy}\dot{p} - I_{xz}\dot{r} = \bar{q}_1 Sb(C_{\ell_\beta}\beta + C_{\ell_p}\frac{pb}{2U_1} + C_{\ell_r}\frac{rb}{2U_1} + C_{\ell_{\delta_A}}\delta_A + C_{\ell_{\delta_R}}\delta_R) \quad [5.2]$$

$$I_{zz}\dot{r} - I_{xz}\dot{p} = \bar{q}_1 Sb(C_{n_\beta}\beta + C_{n_{T_\beta}}\beta + C_{n_p}\frac{pb}{2U_1} + C_{n_r}\frac{rb}{2U_1} + C_{n_{\delta_A}}\delta_A + C_{n_{\delta_R}}\delta_R)$$

Using the definitions shown in Table 5.2, Equations [5.1] and [5.2] can be converted to the dimensional form shown below.

- for longitudinal (from Reference 25, Equation 6.72):

$$\dot{u} = -g\theta\cos\theta_1 + X_u u + X_{T_u} u + X_\alpha \alpha + X_{\delta_E}\delta_E$$

$$\dot{w} - U_1 q = -g\theta\sin\theta_1 + Z_u u + Z_\alpha \alpha + Z_{\dot{\alpha}}\dot{\alpha} + Z_q q + Z_{\delta_E}\delta_E \quad [5.3]$$

$$\dot{q} = M_u u + M_{T_u} u + M_\alpha \alpha + M_{T_\alpha}\alpha + M_{\dot{\alpha}}\dot{\alpha} + M_q q + M_{\delta_E}\delta_E$$

- for lateral (from Reference 25, Equation 6.141):

$$\dot{v} + U_1 r = g\phi\cos\theta_1 + Y_\beta \beta + Y_p p + Y_r r + Y_{\delta_A}\delta_A + Y_{\delta_R}\delta_R$$

$$\dot{p} - \frac{I_{xz}}{I_{xx}}\dot{r} = L_\beta \beta + L_p p + L_r r + L_{\delta_A}\delta_A + L_{\delta_R}\delta_R \quad [5.4]$$

$$\dot{r} - \frac{I_{xz}}{I_{zz}}\dot{p} = N_\beta \beta + N_{T_\beta}\beta + N_p p + N_r r + N_{\delta_A}\delta_A + N_{\delta_R}\delta_R$$

Table 5.2(a) **Longitudinal Dimensional Stability Derivatives** *

$$X_u = \frac{-\bar{q}_1 S(C_{D_u} + 2C_{D_1})}{mU_1} \quad (\text{sec}^{-1})$$

$$X_{T_u} = \frac{\bar{q}_1 S(C_{T_{x_u}} + 2C_{T_{x_1}})}{mU_1} \quad (\text{sec}^{-1})$$

$$X_\alpha = \frac{-\bar{q}_1 S(C_{D_\alpha} - C_{L_1})}{m} \quad (\text{ft sec}^{-2})$$

$$X_{\delta_E} = \frac{-\bar{q}_1 S C_{D_{\delta_E}}}{m} \quad (\text{ft sec}^{-2})$$

$$Z_u = -\frac{\bar{q}_1 S(C_{L_u} + 2C_{L_1})}{mU_1} \quad (\text{sec}^{-1})$$

$$Z_\alpha = -\frac{\bar{q}_1 S(C_{L_\alpha} + C_{D_1})}{m} \quad (\text{ft sec}^{-2})$$

$$Z_{\dot{\alpha}} = -\frac{\bar{q}_1 S C_{L_{\dot{\alpha}}} \bar{c}}{2mU_1} \quad (\text{ft sec}^{-1})$$

$$Z_q = -\frac{\bar{q}_1 S C_{L_q} \bar{c}}{2mU_1} \quad (\text{ft sec}^{-1})$$

$$Z_{\delta_E} = -\frac{\bar{q}_1 S C_{L_{\delta_E}}}{m} \quad (\text{ft sec}^{-2})$$

$$M_u = \frac{\bar{q}_1 S\bar{c}(C_{m_u} + 2C_{m_1})}{I_{yy}U_1} \quad (\text{ft}^{-1}\ \text{sec}^{-1})$$

$$M_{T_u} = \frac{\bar{q}_1 S\bar{c}(C_{m_{T_u}} + 2C_{m_{T_1}})}{I_{yy}U_1} \quad (\text{ft}^{-1}\ \text{sec}^{-1})$$

$$M_\alpha = \frac{\bar{q}_1 S\bar{c} C_{m_\alpha}}{I_{yy}} \quad (\text{sec}^{-2})$$

$$M_{T_\alpha} = \frac{\bar{q}_1 S\bar{c} C_{m_{T_\alpha}}}{I_{yy}} \quad (\text{sec}^{-2})$$

$$M_{\dot{\alpha}} = \frac{\bar{q}_1 S\bar{c}^2 C_{m_{\dot{\alpha}}}}{2I_{yy}U_1} \quad (\text{sec}^{-1})$$

$$M_q = \frac{\bar{q}_1 S\bar{c}^2 C_{m_q}}{2I_{yy}U_1} \quad (\text{sec}^{-1})$$

$$M_{\delta_E} = \frac{\bar{q}_1 S\bar{c} C_{m_{\delta_E}}}{I_{yy}} \quad (\text{sec}^{-2})$$

* from Reference 24, Table 6.3, page 413

Table 5.2(b) **Lateral-Directional Dimensional Stability Derivatives** *

$Y_\beta = \dfrac{\bar{q}_1 S C_{y_\beta}}{m}$ (ft sec^{-2})	$L_{\delta_A} = \dfrac{\bar{q}_1 S b C_{\ell_{\delta_A}}}{I_{xx}}$ (sec^{-2})
$Y_p = \dfrac{\bar{q}_1 S b C_{y_p}}{2mU_1}$ (ft sec^{-1})	$L_{\delta_R} = \dfrac{\bar{q}_1 S b C_{\ell_{\delta_R}}}{I_{xx}}$ (sec^{-2})
$Y_r = \dfrac{\bar{q}_1 S b C_{y_r}}{2mU_1}$ (ft sec^{-1})	$N_\beta = \dfrac{\bar{q}_1 S b C_{n_\beta}}{I_{zz}}$ (sec^{-2})
$Y_{\delta_A} = \dfrac{\bar{q}_1 S C_{y_{\delta_A}}}{m}$ (ft sec^{-2})	$N_{T_\beta} = \dfrac{\bar{q}_1 S b C_{n_{T_\beta}}}{I_{zz}}$ (sec^{-2})
$Y_{\delta_R} = \dfrac{\bar{q}_1 S C_{y_{\delta_R}}}{m}$ (ft sec^{-2})	$N_p = \dfrac{\bar{q}_1 S b^2 C_{n_p}}{2I_{zz}U_1}$ (sec^{-1})
$L_\beta = \dfrac{\bar{q}_1 S b C_{\ell_\beta}}{I_{xx}}$ (sec^{-2})	$N_r = \dfrac{\bar{q}_1 S b^2 C_{n_r}}{2I_{zz}U_1}$ (sec^{-1})
$L_p = \dfrac{\bar{q}_1 S b^2 C_{\ell_p}}{2I_{xx}U_1}$ (sec^{-1})	$N_{\delta_A} = \dfrac{\bar{q}_1 S b C_{n_{\delta_A}}}{I_{zz}}$ (sec^{-2})
$L_r = \dfrac{\bar{q}_1 S b^2 C_{\ell_r}}{2I_{xx}U_1}$ (sec^{-1})	$N_{\delta_R} = \dfrac{\bar{q}_1 S b C_{n_{\delta_R}}}{I_{zz}}$ (sec^{-2})

* from Reference 24, Table 6.8, page 445

Using the concept of state variable theory (see Reference 25), Equations [5.3] and [5.4] can be reduced to the following form:

$$[R]\ \{\dot{x}(t)\} = [A]\ \{x(t)\} + [B]\ \{u(t)\} \qquad [5.5]$$

where

$\{x(t)\}$ = state vector

$[R]$ = acceleration transformation matrix

$[A]$ = stability matrix

$[B]$ = control matrix

$\{u(t)\}$ = control vector.

Equation [5.5] can be written more explicitly in the forms which follow:

- for longitudinal (where [R] = identity matrix):

$$\frac{d}{dt}\begin{bmatrix} q \\ U \\ \alpha \\ \theta \end{bmatrix} = \begin{bmatrix} M'_q & M'_u & M'_\alpha & M'_\theta \\ 0 & X'_u & X'_\alpha & -g\cos(\theta_1) \\ \dfrac{Z_q + U_1}{U_1 - Z_{\dot{\alpha}}} & Z'_u & Z'_\alpha & \dfrac{-g}{U_1 - Z_{\dot{\alpha}}}\sin(\theta_1)\cos(\phi_1) \\ \cos(\phi_1) & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} q \\ U \\ \alpha \\ \theta \end{bmatrix} + \begin{bmatrix} M'_{\delta_E} & M'_{\delta_c} & M'_o \\ X'_{\delta_E} & X'_{\delta_c} & X'_o \\ Z'_{\delta_E} & Z'_{\delta_c} & Z'_o \\ 0 & 0 & \dot{\theta}'_o \end{bmatrix}\begin{bmatrix} \delta_E \\ \delta_c \\ 1 \end{bmatrix} \qquad [5.6]$$

(See Table 5.3 for explicit definition of these terms.)

- for lateral:

$$[R]\frac{d}{dt}\begin{bmatrix} p \\ r \\ \beta \\ \phi \end{bmatrix} = \begin{bmatrix} L'_p & L'_r & L'_\beta & 0.0 \\ N'_p & N'_r & N'_\beta & 0.0 \\ \sin(\alpha_1) & -\cos(\alpha_1) & Y'_\beta & \dfrac{g}{U_1}\cos(\theta_1)\cos(\phi_1) \\ 1.0 & \cos(\phi_1)\tan(\theta_1) & 0.0 & 0.0 \end{bmatrix}\begin{bmatrix} p \\ r \\ \beta \\ \phi \end{bmatrix} + \begin{bmatrix} L'_{\delta_A} & L'_{\delta_r} & L'_o \\ N'_{\delta_A} & N'_{\delta_r} & N'_o \\ Y'_{\delta_A} & Y'_{\delta_r} & Y'_o \\ 0.0 & 0.0 & \dot{\phi}'_o \end{bmatrix}\begin{bmatrix} \delta_A \\ \delta_r \\ 1 \end{bmatrix} \qquad [5.7]$$

(See Table 5.3 for explicit definition of these terms.)

Table 5.3(a) Longitudinal, State Vector Coefficients

$$M'_q = M_q + M_{\dot{\alpha}} \frac{Z_q + U_1}{U_1 - Z_{\dot{\alpha}}} \approx M_q + M_{\dot{\alpha}} \quad (\text{sec}^{-1})$$

$$M'_u = M_u + M_{T_u} + \frac{M_{\dot{\alpha}} Z_u}{U_1 - Z_{\dot{\alpha}}} \quad (\text{ft}^{-1}\ \text{sec}^{-1})$$

$$M'_\alpha = M_\alpha + M_{T_\alpha} + \frac{M_{\dot{\alpha}} Z_\alpha}{U_1 - Z_{\dot{\alpha}}} \quad (\text{sec}^{-2})$$

$$M'_\theta = \frac{-M_{\dot{\alpha}} g \sin(\theta_1) \cos(\phi_1)}{U_1 - Z_{\dot{\alpha}}} \approx 0 \quad (\text{sec}^{-2})^{\dagger}$$

$$M'_{\delta_{E,c}} = M_{\delta_{E,c}} + \frac{M_{\dot{\alpha}} Z_{\delta_{E,c}}}{U_1 - Z_{\dot{\alpha}}} \quad (\text{sec}^{-2})$$

M'_o = pitching moment equation bias* (sec^{-2})

$$X'_u = X_u + X_{T_u} \quad (\text{sec}^{-1})$$

$$X'_\alpha = X_\alpha \quad (\text{ft sec}^{-2})$$

$$X'_{\delta_{E,c}} = X_{\delta_{E,c}} \quad (\text{ft sec}^{-2})$$

X'_o = longitudinal acceleration equation bias (ft sec^{-2})*

$$\frac{Z_q + U_1}{U_1 - Z_{\dot{\alpha}}} \approx 1.0^{\dagger}$$

$$Z'_u = \frac{Z_u}{U_1 - Z_{\dot{\alpha}}} \approx \frac{Z_u}{U_1} \quad (\text{ft}^{-1})$$

$$Z'_\alpha = \frac{Z_\alpha}{U_1 - Z_{\dot{\alpha}}} \approx \frac{Z_\alpha}{U_1} \quad (\text{sec}^{-1})$$

$$\frac{-g \sin(\theta_1)\cos(\phi_1)}{U_1 - Z_{\dot{\alpha}}} \approx \frac{-g \sin(\theta_1)\cos(\phi_1)}{U_1} \quad (\text{sec}^{-1})^{\dagger}$$

$$Z'_{\delta_{E,c}} = \frac{Z_{\delta_{E,c}}}{U_1 - Z_{\dot{\alpha}}} \approx \frac{Z_{\delta_{E,c}}}{U_1} \quad (\text{sec}^{-1})$$

Z'_o = normal acceleration equation bias (sec^{-1})*

$\dot{\theta}'_o$ = pitch rate equation bias (sec^{-1})*

* Note: The equation bias terms are used to allow prediction of the complete state which is made up of the steady state and the perturbed state.

† Note: With the approximations above, Equation [5.6] is rewritten as:

$$\frac{d}{dt}\begin{bmatrix} q \\ U \\ \alpha \\ \theta \end{bmatrix} = \begin{bmatrix} M'_q & M'_u & M'_\alpha & 0 \\ 0 & X'_u & X'_\alpha & -\cos(\theta_1)g \\ 1 & Z'_u & Z'_\alpha & -\sin(\theta_1)\cos(\phi_1)\frac{g}{U_1} \\ \cos(\phi_1) & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} q \\ U \\ \alpha \\ \theta \end{bmatrix} + \begin{bmatrix} M'_{\delta_E} & M'_{\delta_c} & M'_o \\ X'_{\delta_E} & X'_{\delta_c} & X'_o \\ Z'_{\delta_E} & Z'_{\delta_c} & Z'_o \\ 0 & 0 & \dot{\theta}_o \end{bmatrix} \begin{bmatrix} \delta_E \\ \delta_c \\ 1 \end{bmatrix}$$

Table 5.3(b) Lateral-Directional State Vector Coefficients

$L'_p = L_p$ (sec^{-1})	$N_{\delta_A}' = N_{\delta_A}$ (sec^{-2})
$L'_r = L_r$ (sec^{-1})	$N_{\delta_r}' = N_{\delta_r}$ (sec^{-2})
$L'_\beta = L_\beta$ (sec^{-1})	$N'_\beta = N_\beta + N_{T_\beta}$ (sec^{-1})
$L_{\delta_A}' = L_{\delta_A}$ (sec^{-2})	$Y'_\beta = \frac{Y_\beta}{U_1}$ (sec^{-1})
$L_{\delta_r}' = L_{\delta_r}$ (sec^{-2})	$Y_{\delta_A}' = \frac{Y_{\delta_A}}{U_1}$ (sec^{-1})
$N'_p = N_p$ (sec^{-1})	$Y_{\delta_r}' = \frac{Y_{\delta_r}}{U_1}$ (sec^{-1})
$N'_r = N_r$ (sec^{-1})	

Y'_o = lateral acceleration equation bias (sec^{-1}) *

$\dot{\phi}'_o$ = roll rate equation bias (sec^{-1}) *

L'_o = rolling moment equation bias (sec^{-2}) *

N'_o = yawing moment equation bias (sec^{-2}) *

*NOTE: The equation bias terms are used to allow prediction of the complete state which is made up of the steady state and the perturbed state.

$$[R] = \begin{bmatrix} 1.0 & -\frac{I_{xz}}{I_{xx}} & 0 & 0 \\ -\frac{I_{xz}}{I_{zz}} & 1.0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

for $I_{xz} \approx 0$; [R] = identity matrix

To allow determination of states other than the ones contained in $\{x(t)\}$, the following expression can be derived:

$$\{y(t)\} = \left[\frac{I}{G}\right] \{x(t)\} + \left[\frac{0}{H}\right] \{u(t)\} + \left\{\frac{0}{v}\right\} ^{*} \qquad [5.8]$$

where

$\{y(t)\}$ = computed observation vector

$[G]$ = observation matrix

$[H]$ = observation matrix

$\{v\}$ = variable bias vector.

(See Table 5.4 for explicit definition of these terms.)

The computed observation vector, $\{y(t)\}$, corresponds to the measured observation vector, shown here:

$$\{z(t)\} = \{y(t)\} + \{\eta(t)\} ^{*} \qquad [5.9]$$

where

$\{z(t)\}$ = measured observation vector = $\{\theta, \phi, p, q, r, A_X, A_Y, A_Z, \delta_E, \delta_A, \delta_R, P_S, P_D, T\}$

$\{\eta(t)\}$ = measured noise vector.

From the terms of Equations [5.5], [5.8], and [5.9], the vector

$$\{c\} = f([A], [B], [G], [H], \{v\}) \qquad [5.10]$$

(where f indicates "a function of") is defined as the vector of unknowns. It is this vector that the MMLE method estimates. MMLE determines the unknowns ($\{c\}$) by minimizing the cost function given by:

$$J = \frac{1}{T} \int_0^T \{z(t) - y(t)\}^{\dagger}[D] \{z(t) - y(t)\} \, dt ^{*} \qquad [5.11]$$

(T, t: indicates time)

*From Reference 5

C-2

Table 5.4 Matrices Used in the Observation Equation

LONGITUDINAL

$$\{y(t)\}^{\dagger} = \{q,\ U,\ \alpha,\ \theta,\ \dot{q},\ A_X,\ A_Z\}$$

$$\left[\frac{0}{v}\right]^{\dagger} = \{0,\ 0,\ 0,\ 0,\ \dot{q}_{bias},\ A_{X_{bias}},\ A_{Z_{bias}}\}$$

$$\left[\frac{O}{H}\right] = \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hdashline M_{\delta_E}' & M_{\delta_c}' & M_o' \\ \frac{X_{\delta_E}'}{g} & \frac{X_{\delta_c}'}{g} & \frac{X_o'}{g} \\ \frac{-U_1}{g} Z_{\delta_E}' & \frac{-U_1}{g} Z_{\delta_c}' & \frac{-U_1}{g} Z_o' \end{array}\right] \qquad \left[\frac{I}{G}\right] = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hdashline M_q' & M_u' & M_\alpha' & 0 \\ 0 & \frac{X_u'}{g} & 0 & 0 \\ 0 & 0 & \frac{-U_1 Z_u'}{g} & 0 \end{array}\right]$$

LATERAL

$$\{y(t)\}^{\dagger} = \{p,\ r,\ \beta,\ \phi,\ \dot{p},\ \dot{r},\ A_Y\}$$

$$\left[\frac{0}{v}\right]^{\dagger} = \{0,\ 0,\ 0,\ 0,\ \dot{p}_{bias},\ \dot{r}_{bias},\ A_{Y_{bias}}\}$$

$$\left[\frac{O}{H}\right] = \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hdashline L_{\delta_A}' & L_{\delta_R}' & L_o' \\ N_{\delta_A}' & N_{\delta_R}' & N_o' \\ \frac{U_1}{g} Y_{\delta_A}' & \frac{U_1}{g} Y_{\delta_R}' & \frac{U_1}{g} Y_o' \end{array}\right] \qquad \left[\frac{I}{G}\right] = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hdashline L_p' & L_r' & L_\beta' & 0 \\ N_p' & N_r' & N_\beta' & 0 \\ 0 & 0 & \frac{Y_\beta' U_1}{g} & 0 \end{array}\right]$$

or approximately in the discrete case:

$$J = \frac{1}{(N-1)} \sum_{i=1}^{N} \{z_i - y_i\}^{+} [D] \{z_i - y_i\} \Delta t \quad ^{*} \qquad [5.12]$$

(where i is the time index, and N the number of time points).

The weighting matrix, [D], is used to provide emphasis on the various measured states; in other words, to allow greater emphasis on the more accurate transducers, or the transducers that are more important to describe the maneuver performed.

The value of the cost functional, J, is minimized using the Newton-Raphson* method. This technique is an iterative procedure, utilizing an estimated value of the vector of unknowns, {c}, and the first and second gradients of the cost functional, J, with respect to the vector of unknowns, {c}. The equation

$$\{c\}_L = \{c\}_{L-1} - \{\nabla_c^2 J\}_L^{-1} \{\nabla_c J\}_L^{+} \quad ^{*} \qquad [5.3]$$

(where L is the iteration number) is used to revise estimates for the vector of unknowns, {c}. The first and second gradients are given by:

$$\{\nabla_c J\} = \frac{2}{N-1} \sum_{i=1}^{N} \{z_i - y_i\}^{+} [D] \nabla_c \{z_i - y_i\} \quad ^{*} \qquad [5.14]$$

$$\{\nabla_c^2 J\} = \frac{2}{N-1} \sum_{i=1}^{N} \nabla_c \{z_i - y_i\}^{+} [D] \nabla_c \{z_i - y_i\} + \frac{2}{N-1} \sum_{i=1}^{N} \{z_i - y_i\}^{+} [D] \nabla_c^2 \{z_i - y_i\} \quad ^{*} \qquad [5.15]$$

*From Reference 5

The Balakrishnan* modification makes use of the fact that the term $\nabla_c^2\{z_i - y_i\}$ approaches zero with convergence and thus can be neglected. The expression for the second gradient becomes:

$$\{\nabla_c^2 J\} = \frac{2}{N-1} \sum_{i=1}^{N} \nabla_c\{z_i - y_i\}^{\dagger}[D]\nabla_c\{z_i - y_i\} \quad * \qquad [5.16]$$

After several iterations the cost function converges near some small value. At this point the parameters of Equations [5.6] and [5.7] have been modified to obtain their most likely value which results in the best fit of the measured states.

The following inputs and modifications were made to the MMLE method, allowing effective use of the technique on the MINC 11/03 computer.

Initial estimates of the derivatives in Equations [5.6] and [5.7] were obtained using the analytical methods of Reference 25. Although the MMLE technique does not require accurate knowledge of these derivatives, this prodecure does speed convergence.

A diagonal multiplying factor allows control over how large a change is made to the derivatives after each iteration. Too large a value of this factor causes sluggishness in the convergence, and too small a value will cause divergence. This factor was set equal to 1.0 for all cases analyzed. This was found to be acceptable.

The weighting matrix, [D], of Equation [5.11], was chosen after analysis of the instrumentation error magnitudes. The first run through the MMLE program, with measurements from this instrumentation

---

*From Reference 5

package, provided a weighted error for each measurement state. As suggested in Reference 5, the values for the weighting matrix were chosen to attempt to equalize the weighted errors. After the values for the weighting matrix were chosen for the instrument package, they were then left at this for further maneuver analysis.

5.2.4 Time History Plotting

The MMLE method not only produces the estimates of the coefficients but also calculates the estimated time histories for the various states. These data are stored on the data disk for plotting by this routine. Appendix A.8 contains the listing for this program. The graphs produced are good visual indications of the goodness of the estimated model coefficients.

5.2.5 Summary Coefficient Plotting

For presentation of the results of many cases which have been run at many flight conditions, the estimated derivatives are plotted as functions of lift coefficient. This is an interactive program that plots lift coefficient, estimated derivative, and confidence level. Once these values are input, the computer plots the value and loops back to input more data.

6. KU-FRL FLIGHT TEST PROGRAM

This section describes the Phase III flight tests conducted on the University of Kansas Cessna 172 airplane. Discussed are instrumentation system installation and calibration, weight and balance calculations, aircraft certification, flight test procedures, a description of the typical flight test maneuver, and the results of the flight tests.

6.1 System Installation and Calibration

Installation of the instrumentation system required approximately eight man-hours. This included the requisite removal of oil and dirt accumulations from the bottom of the aircraft, surface preparations for the external transducers, installation of the external transducers, cockpit modifications, and internal hardware installation. The control position transducers (CPT's) required installed calibrations, which took approximately three man-hours to complete. The CPT's were located such that maximum control surface deflection caused the largest possible potentiometer variation. The aileron and elevator CPT's were calibrated using an inclinometer to measure the angular deflections while observing the outputs through the data acquisition system. Similarly, the rudder was calibrated using a KU-FRL-developed protractor for measuring angular position.

In preparation for mounting the external transducers, the aircraft was thoroughly cleaned, first with commercial degreaser and then with isopropyl alcohol, in the regions of the transducers, and also in the regions of the wiring runs. The transducer wires and the total pressure tubing were routed along the outside of the aircraft to an inspection hole in the belly of the aircraft. They were secured and faired to the aircraft skin with duct tape.

The transducers were mounted with 3M-4265 tape, as discussed in Chapter 3 and Reference 3, and faired with duct tape. Figures 6.1, 6.2, 6.3, and 6.4 show the relative locations and installation details of the elevator and rudder CPT's, the aileron CPT, the pitot pressure tube, and the total temperature sensor, respectively. Figure 6.5 shows the static cone location and installation. Figure 6.6 shows the cable and tube routings into the inspection hole.

Inside the aircraft, the sensor lines entered through a matched inspection hole in the cabin floor (Figure 6.7) and were routed to the transducer package (Figure 6.8). The carpeting from the center cabin floor was removed to simplify the cable routing. The transducer package was mounted solidly to the pilot's and co-pilot's seat tracks, on the aircraft centerline, as shown in Figure 6.9. This pallet was attached to the seat tracks using four C-clamps.

To facilitate the installation of the battery box, computer box, voltage regulator package, and filter package, the rear seat of the airplane was removed. Figure 6.10 shows the location of the above components. The packages were secured using seatbelts and a rope passed through existing cargo hold-downs in the cabin.

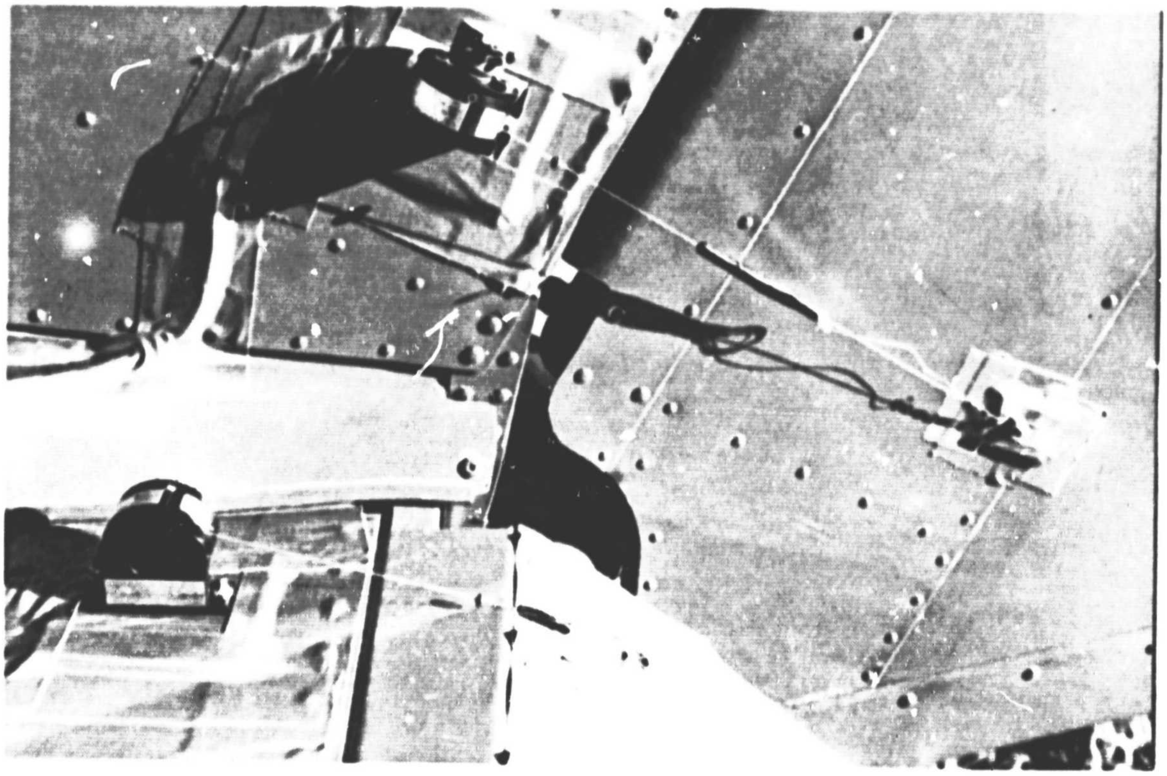


Figure 6.1 Rudder and Elevator Control Position Transducers

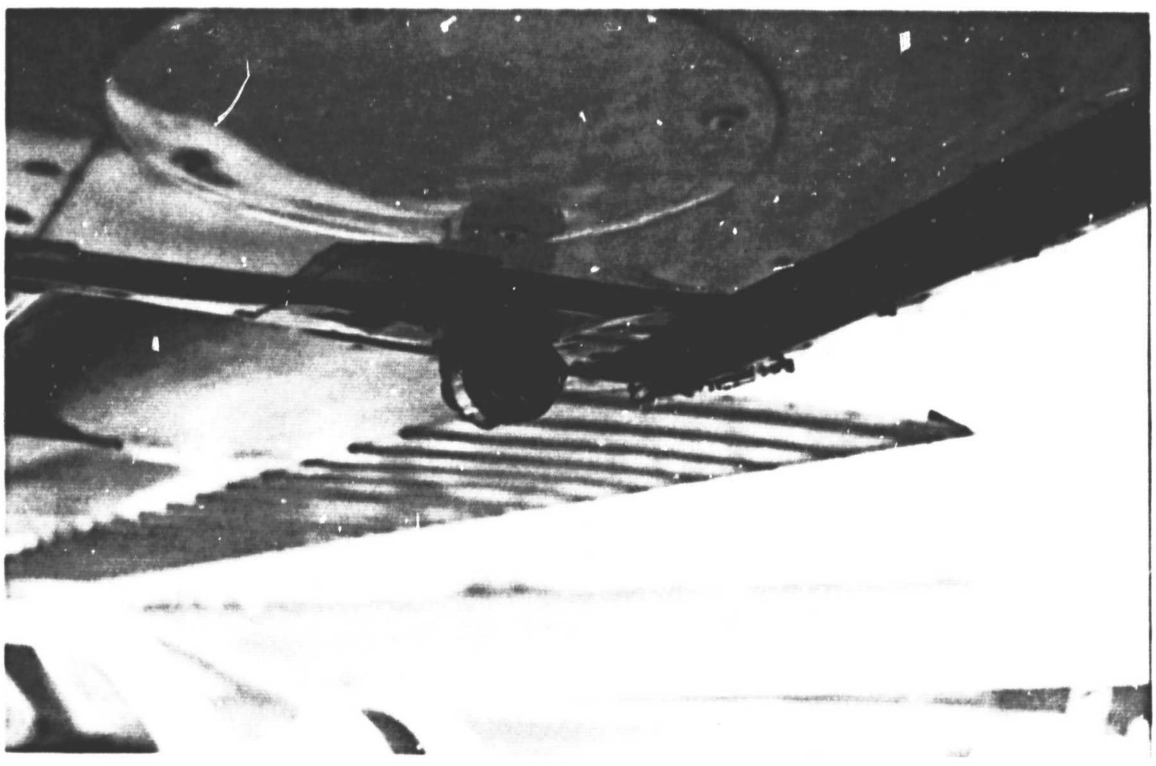


Figure 6.2 Aileron Control Position Transducer

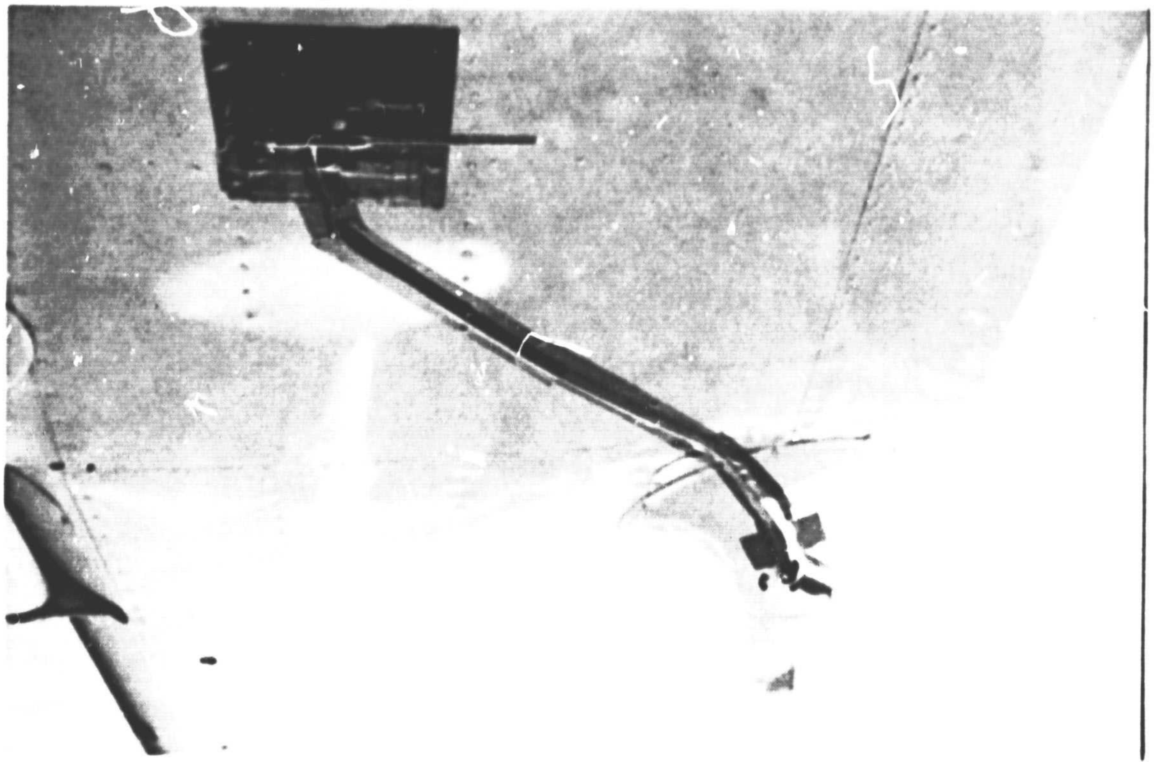


Figure 6.3 Pitot Pressure Probe

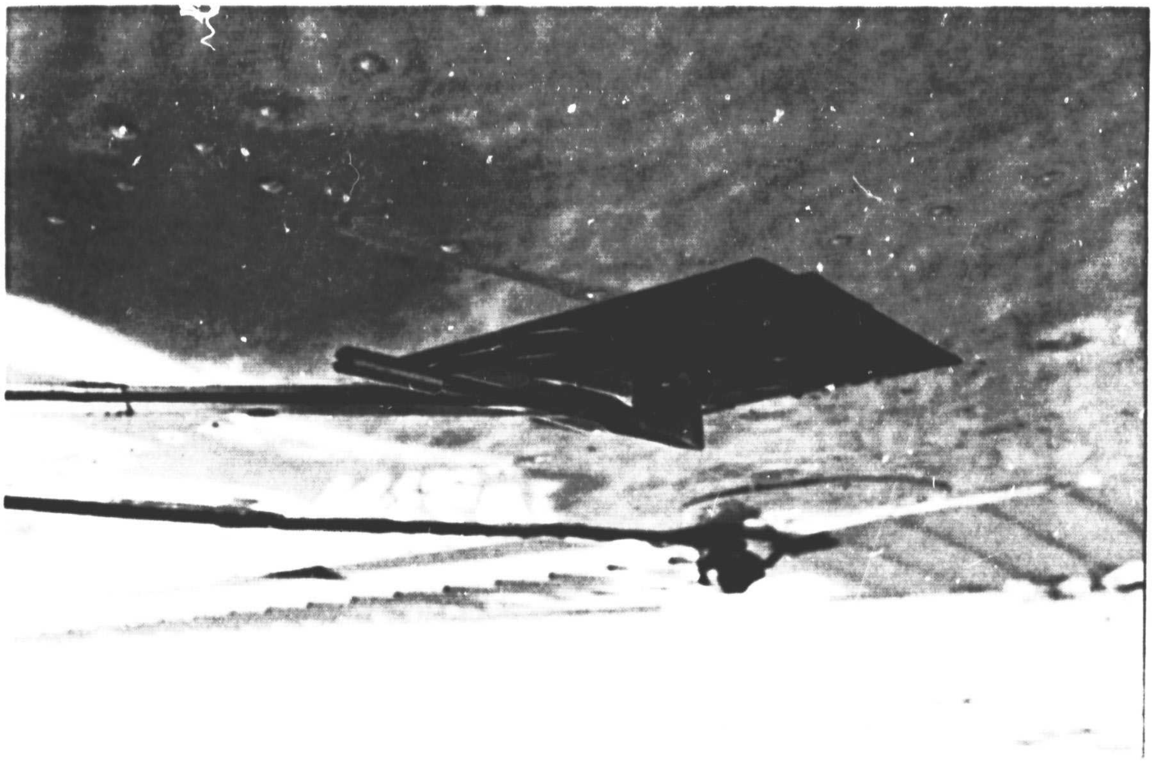


Figure 6.4 Total Temperature Probe

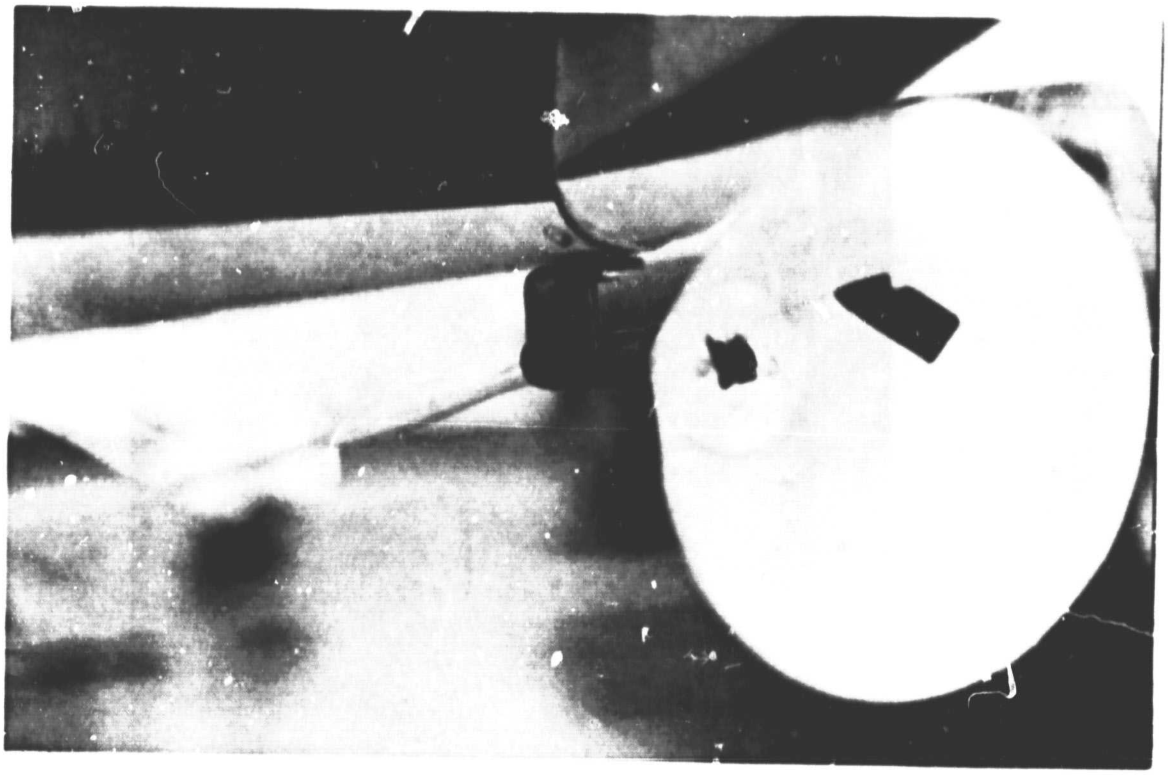


Figure 6.5 Static Pressure Cone

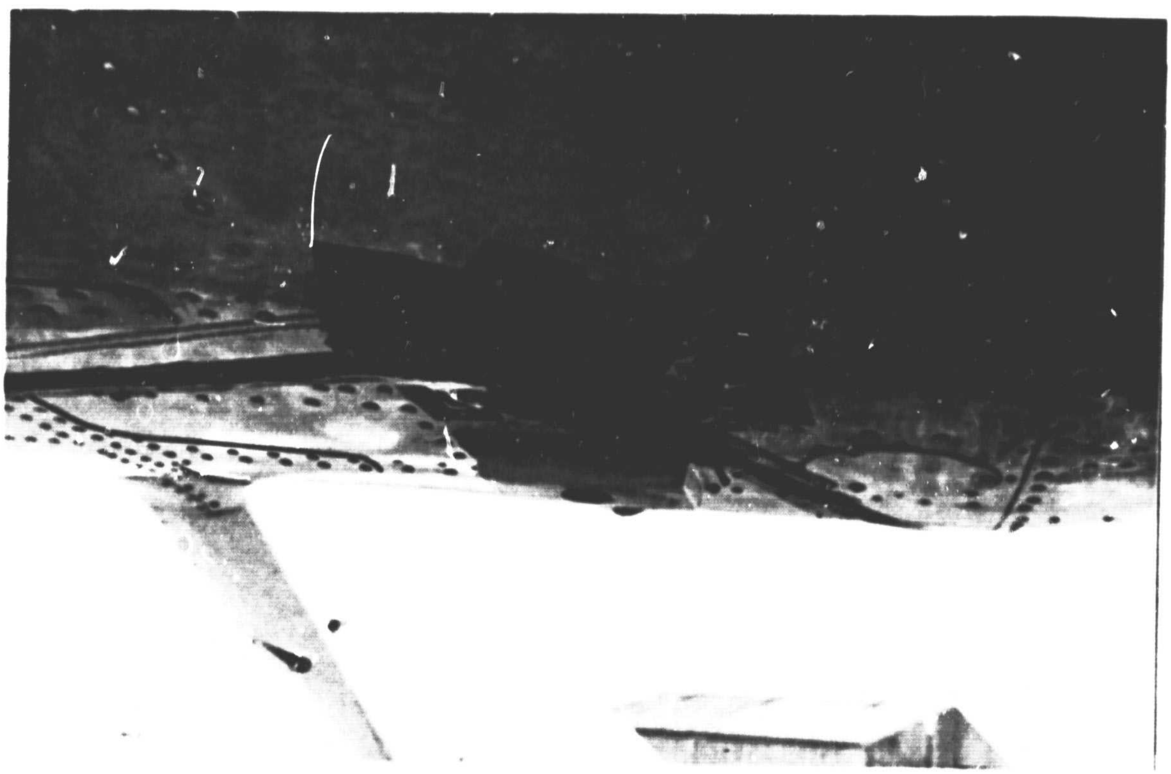


Figure 6.6 Cable and Tube Installation Details (Outside)

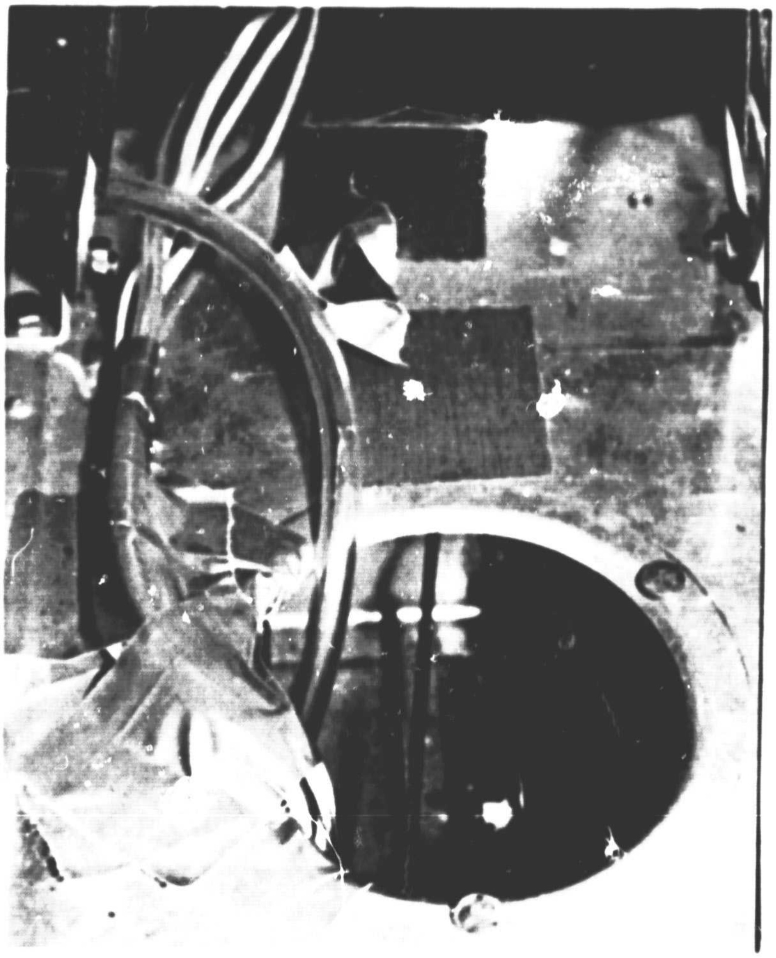


Figure 6.7 Cable and Tube Installation Details (Inside)

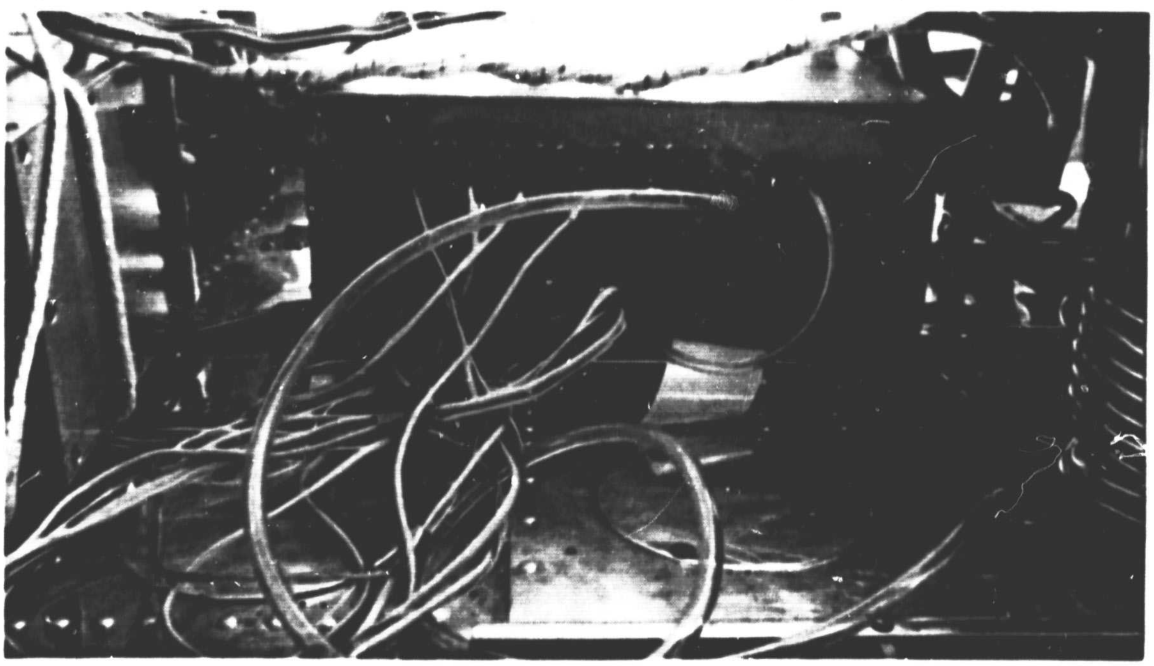


Figure 6.8 Installed Transducer Package (Side View)

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

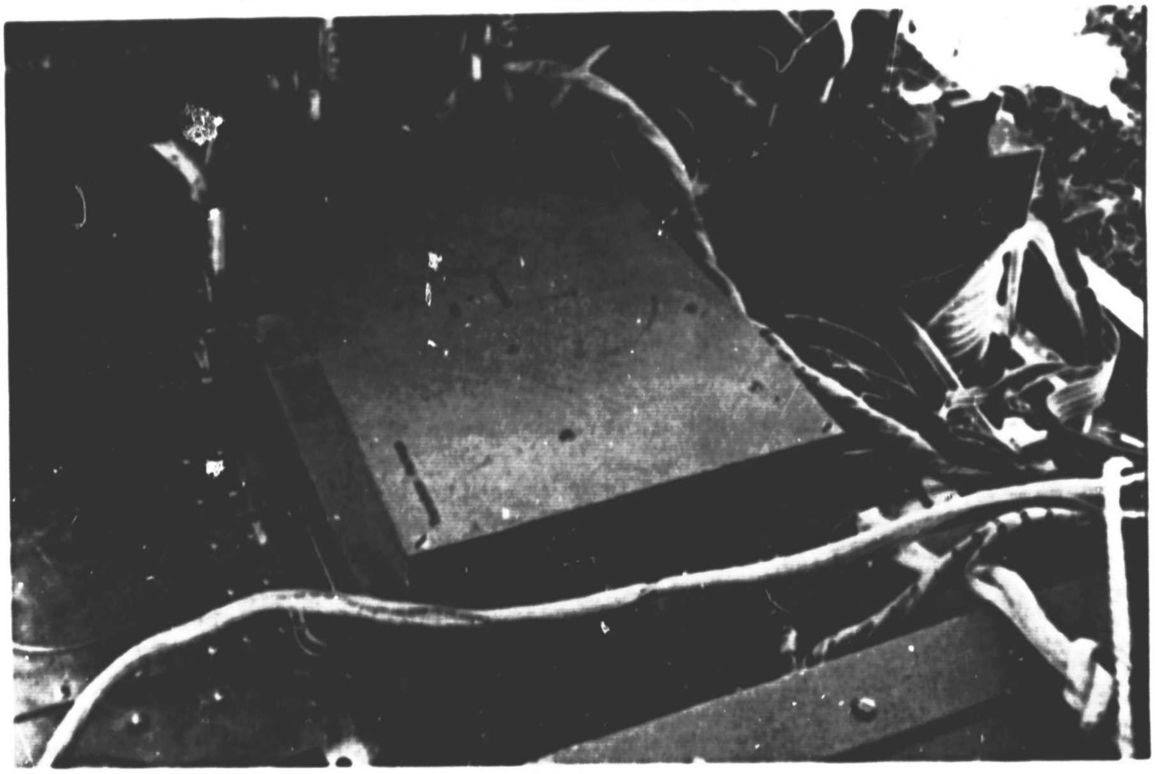


Figure 6.9 Installed Transducer Package (Oblique View)

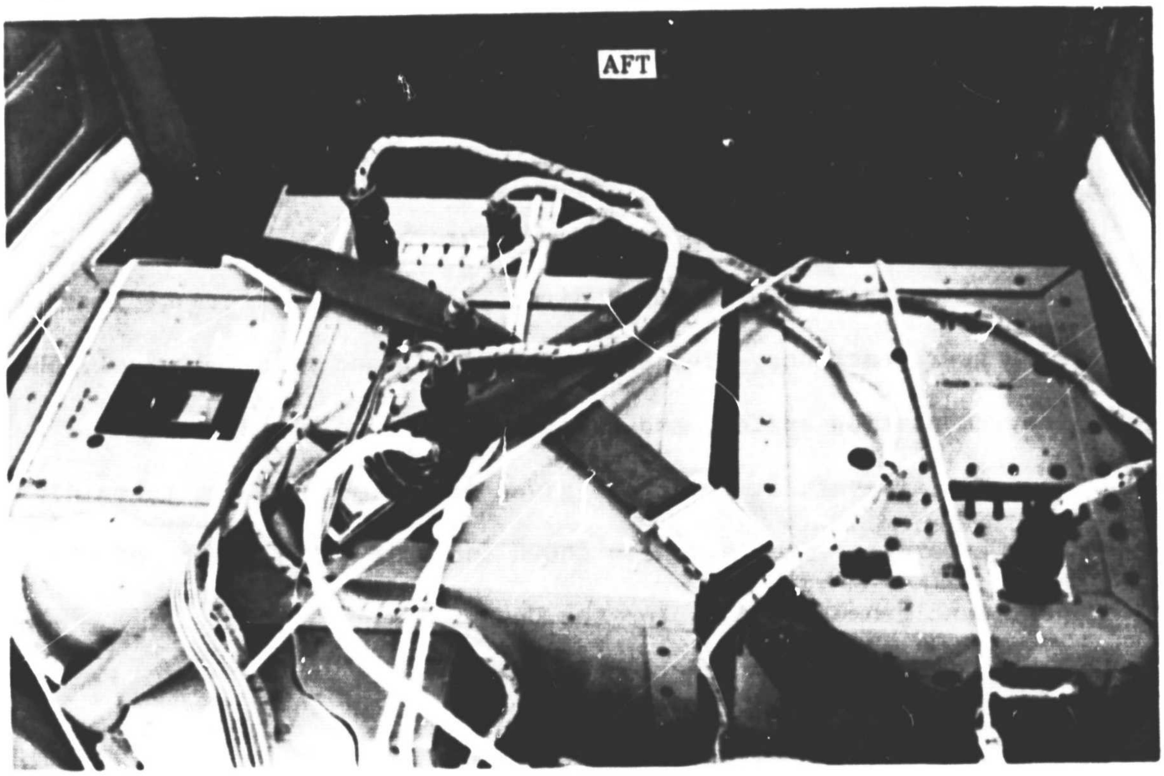


Figure 6.10 Installed Battery and Computer Modules (Oblique View)

6.2 Weight and Balance

The basis for the weight and balance calculations performed for this flight test program was an Aircraft Weight Record for the airplane (a 1974 Cessna 172M, N12800). This Weight Record, made on 3 February 1981, was done at Cessna Aircraft Company, Pawnee Division. The aircraft empty weight and moment are shown in Table 6.1. The component weights of the instrumentation system, pilot, flight engineer, fuel, and oil and their respective moments are also given in Table 6.1. The total aircraft weignt and moment are found to be well within the Standard Category loading envelope, as shown in Figure 6.11.

It was also necessary to locate the vertical and lateral positions of the center of gravity. For the vertical c.g. location, the aircraft was weighed in the level flight attitude and in a nose-up and a nose-down attitude. The intersection of the lines perpendicular to the ground and passing through the longitudinal c.g. located the vertical position of the c.g., as illustrated in Figure 6.12. The lateral position was computed directly from the difference in the main gear reactions. The aircraft was weighed with full fuel, the instrumentation system, and the flight crew on board.

The aircraft inertias, as given in Table 6.2, were taken directly from Reference 3 data. Also shown in Table 6.2 are some of the pertinent geometric data for the Cessna 172.

Table 6.1 Test Airplane Weight and Balance Record

	Weight	Arm	Moment
Empty A/C	1423.6	39.2	55805
Pilot (Shane)	180	34	6120
Engineer (Clarke)	205	34	6970
Fuel (full usable)	228	47.8	10900
Oil (7 qt)	13	-13.3	-173
-Rear seat	-22	79.5	-1749
Battery box	61.3	73	4475
Computer box	37.3	73	2722.9
Voltage reg. box	9.23	61	563
Transducer box	17.0	46	782
Filter box	5.9	46	270.5
Cables & pilot control box	--- negligible ---		

Σ_W = 2158 lb; Σ_M = 86686 in lb

c.g. = 40.2

According to Cessna Loading Charts found in KU-FRL-407-2, this loading falls exactly on top of the point marked "FULL FUEL."

For the empty fuel condition,

Σ_W = 1930.3 lb

Σ_M = 75786.4 in lb.

These fall within Utility Category Loading Envelope.

ORIGINAL PAGE IS
OF POOR QUALITY

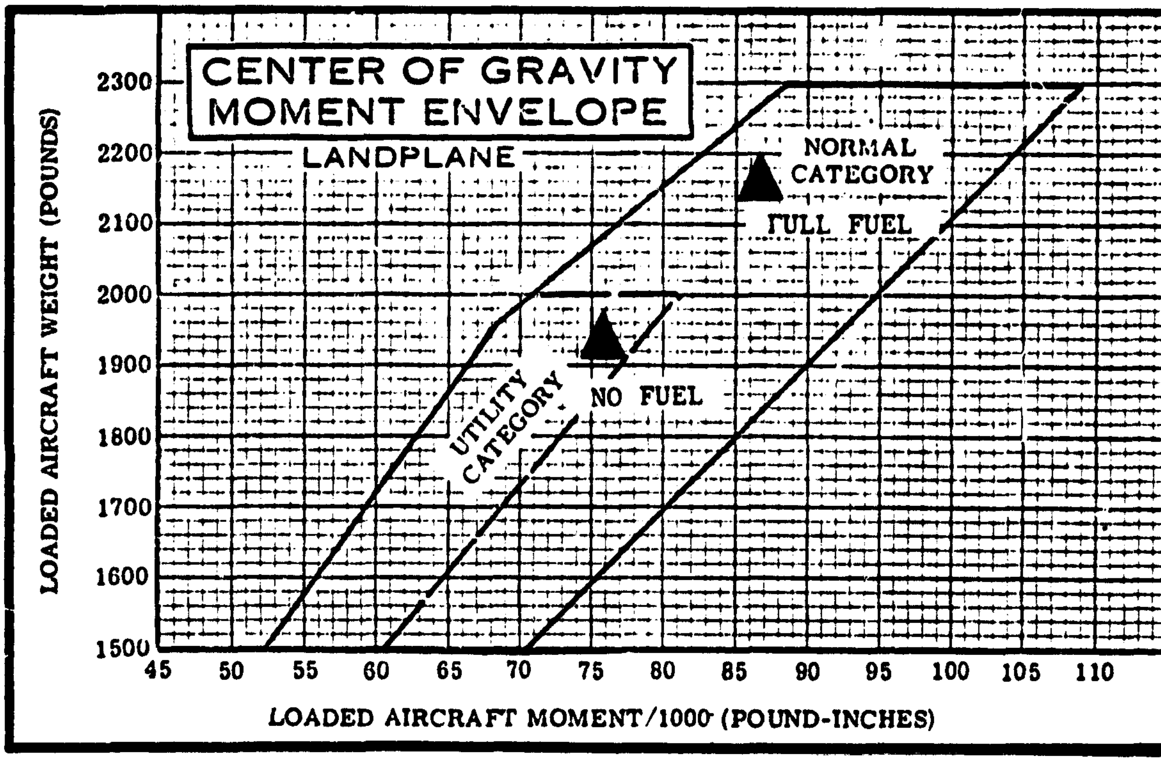


Figure 6.11 Test Airplane Standard Category Loading Envelope

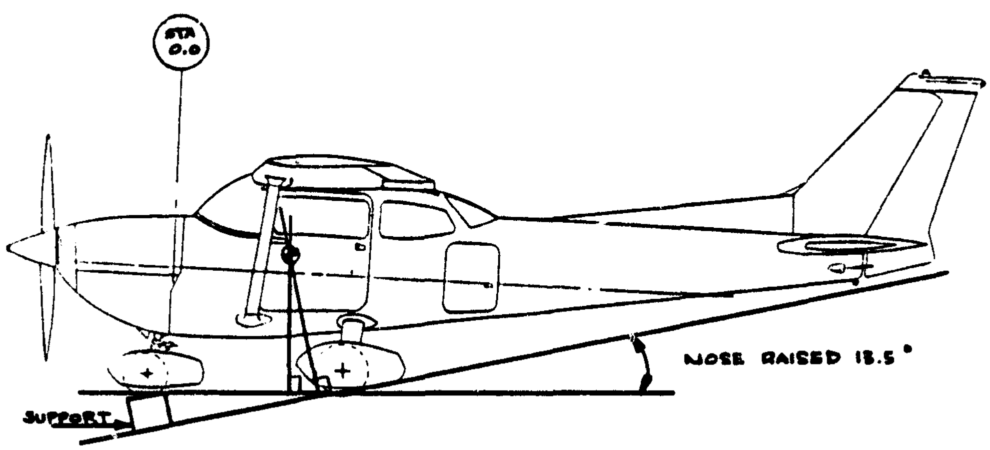


Figure 6.12 Test Airplane Vertical Center-of-Gravity Location Details

Table 6.2 Test Airplane Inertia and Geometric Data

Wing area (S)	174 ft^2
Wing span (b)	35.8 ft
Inertias*	
I_{xx}	1029 slug ft^2
I_{zz}	1891 slug ft^2
I_{xz}	0 slug ft^2
I_{yy}	1092 slug ft^2
Weight	2098 lb
Mass (m)	65.21 slugs
Mean geometric chord ($\bar{c}$)	4.9 ft
Center of gravity (Fuselage Station)	40.0 inch

*Taken from Reference 3

6.3 Aircraft Certification

The attachment of the external transducers, and the aircraft flight test loading (which required the removal of the rear seat), required the airplane to be recertified in the Experimental Category for the duration of the flight test program.

Following the installation of the system, an inspector from the Kansas City FAA Engineering and Maintenance District Office (EMDO) checked the hardware installation, weight and balance statement, and received assurance that no flight test maneuver would be performed outside the manufacturer's flight envelope for the airplane. The inspector was supplied with a copy of the Flight Test Instrumentation Certification Report (Reference 27), which completely detailed the external hardware and the methods of attachment, and the weight and balance statement. The inspector then issued an Experimental Category Airworthiness Certificate and a set of operating limitations for the aircraft.

After the removal of the instrumentation system, an annual (or 100-hour) inspection of the airplane was performed, and the FAA inspector returned, re-examined the airplane, and restored its Standard Category Airworthiness Certificate. The FAA was extremely cooperative in working with the KU-FRL in this and past programs.

6.4 Flight Test Procedures

For the actual flight tests, a set of checklists and flight cards was prepared to ensure consistent test procedures. The checklists are shown in Figure 6.13 and were used in conjunction with the aircraft checklist for each flight.

The flight cards, samples of which are shown in Figure 6.14, were used to provide a standard data-taking format for the pilot or flight engineer. Some of the parameters recorded were:

- Aircraft weight and c.g.
- Ambient temperature
- Voltage outputs of the control position transducers at maximum control deflections both before takeoff and after landing
- Sample rate
- Time of day, pressure altitude, and indicated airspeed before each maneuver
- Type of maneuver, and any comments on the maneuver.

FLIGHT TEST CHECKLIST

BEFORE TAKEOFF:

- TANKS TOPPED
- TAPE IN, WRITE-PROTECT OFF
- COMPUTE REQUIRED TEST POINTS
- ENOUGH DATA SHEETS? TAPES?
- BATTERIES ON, VOLTAGE REGULATORS ON
- VERIFY ALL GYROS COME UP NORMALLY
- INITIALIZE PROGRAM (10/20 SPS)
- INITIALIZE TAPE, VERIFY RUN FUNCTION
- GUARD THE "CLOSE & REWIND TAPE" SWITCH
- CHECK ALL POSITION TRANSDUCERS FOR FUNCTION AND NOTE LIMIT VALUES. VOLTMETER OFF.
- NOTE OAT, LOCAL BARO, ENGINE START TIME, TAKEOFF WEIGHT AND MOMENT, SAMPLE RATE
- RECORD STATIC DATA BEFORE TAKEOFF
- RECORD TIME OF TAKEOFF

PAGE TWO

BEFORE MANEUVER:

- CHECK THAT ALL TRANSDUCERS FUNCTION WITH VOLTMETER. VOLTMETER OFF.
- GUARD THE "CLOSE & REWIND TAPE" SWITCH
- TAKE STEADY-STATE DATA; VERIFY TAPE MOVEMENT
- SET ALTIMETER TO 29.92; CHECK LOCAL ALTIMETER SETTING WITH IIXD
- RECORD ALTITUDE, AIRSPEED, TIME, MANEUVER TYPE AND SEQUENCE, OTHER DATA AS REQUIRED
- CHECK TAPE LENGTH PERIODICALLY

PAGE THREE

BEFORE SHUTDOWN:

- TAKE STATIC DATA
- RECORD OAT
- RECORD TIME
- NOTE ANY ALTIMETER SETTING CHANGE
- NOTE POSITION TRANSDUCER VOLTAGES
- CLOSE OUT TAPE, WRITE-PROTECT
- VOLTAGE REGULATORS OFF, BATTERIES OFF
- VOLTMETER OFF
- AIRCRAFT AND PILOT LOGS
- RECEIPTS
- TOP TANKS, RECORD FUEL USED

Figure 6.13 Flight Test Program Checklists

FLIGHT: 4 DATE: 23 DEC 81 TACH: IN / OUT 1447.7

TAKEOFF WEIGHT: 2158 LB Mom. = 86.915 OAT: 44°F

TAKEOFF TIME: 11:28 CST LOCAL ALTIMETER: 30.32

TAKEOFF FUEL: 38 GAL ENGNR WGT: 205 LB

POSITION TRANSDUCER VOLTAGES:

		OUT	IN
ELEVATOR:	UP:	- 4.915	- 4.920
	DOWN:	+ 4.090	+ 4.102
AILERON:	LEFT:	+ 2.073	2.292
	RIGHT:	-4.789	-4.743
RUDDER:	LEFT:	-3.913	-4.253
	RIGHT:	+ 2.771	2.781

SAMPLE RATE: (10) 20 SPS

LANDING TIME: 12:20 CST LANDING LOCATION: OSC

FUEL USED: 8.3 GAL

REMARKS:

OAT = 15°F

ENTRY SPEED & ALT	TIME (CST)	L	L-D	2/3/1	DESCR
121 (MPH) 3250 (ft)	11:44	✓		✓	
121 3450	11:45	✓			Phugoid Exc.
125 3580	11:47		✓	✓	δa δr LRL - LRL
125 3600	11:48		✓		Spiral Vac δa left only
70 4250	11:55	✓		✓	1950 RPM
70 4390	11:56	✓			Phugoid δe pulse up
70 4550	11:57		✓	✓	LRL LRL
70 4650	11:59		✓	✓	Spiral - Assym. input
70 4700	12:00		✓		"New" Maneuver
70 4660	12:02		✓		
90 4960	12:05	✓		✓	
90 5000	12:06		✓	✓	
90 5050	12:07	✓		✓	
90 5100	12:08		✓	✓	

Figure 6.14 Flight Test Program Flight Cards

A typical flight began with installing the onboard packages (batteries, transducer package, voltage regulators, filters, and computer) and starting the system (before engine start) to verify that the gyros erected properly. Then, before takeoff, steady-state data were taken and the tape drive cycled. Once airborne, all transducers were excited and output voltages checked for proper trends.

The aircraft was flown to an altitude with little or no turbulence (to minimize turbulence noise in the data) and trimmed to the desired flight condition. Attempts were made to duplicate the flight conditions (airspeeds and lift coefficients) of the Reference 4 flight tests. Once trimmed and stabilized, steady-state data were taken for approximately 3 seconds and then the maneuver was initiated.

When completed, the controls were held fixed; and data were recorded for 5 to 10 seconds more. Then the next flight condition was set up and the next maneuver flown.

6.5 Flight Test Maneuver

The analysis of flight test data with the MMLE process utilizes dynamic maneuvers. The important factor with the MMLE process is to excite all aircraft modes if all parameters are to be estimated with high accuracy.

The standard maneuver chosen for this series of tests was a 2-3-1 input suggested by NASA as being a good equal-energy input that could be easily flown. Further consultation with NASA-DFRF indicated that experience had shown that the maneuver itself was fairly noncritical; the maneuver was very forgiving and allowed much deviation from exact step inputs. The KU-FRL flight tests also indicated that to be essentially true. This is further discussed in the next chapter.

The longitudinal input was nominally a 2 seconds back, 3 seconds forward, and 1 second back yoke input of about 10 degrees maximum control surface deflection.

The lateral-directional input was a nominal 2-3-1 aileron input with a symmetrical 2-3-1 rudder input, which had its 2-second segment superimposed on the 1-second aileron input.

Variations on the standard maneuvers included an antisymmetrical lateral input and long sample times to record the entire phugoid and spiral modes. The antisymmetrical input was a left-right-left aileron input coupled with a right-left-right rudder input.

6.6 Flight Test Results

Presented here are the results of 35 separate maneuvers which were analyzed with the MMLE process. Of these maneuvers 18 were longitudinal and 17 were lateral-directional. Tables 6.3 and 6.4 contain the estimated parameters of the longitudinal and lateral-directional analyses, respectively. Figures 6.15 and 6.16 show two typical flight time-history comparisons. On these figures the dashed line is the computed time history, while the solid line is the measured time history.

The estimated parameters have been compared with flight test results obtained by NASA Langley on a Cessna 172 (Reference 4). Figures 6.17 and 6.18 show these comparisons of the derivative estimates. Confidence levels are shown for the KU-FRL results. The Langley results are shown as dashed lines. For the acceleration derivatives ($C_{Z_{\delta_E}}$ and $C_{Y_{\delta_R}}$) two methods were used for estimating these values. The first method was just to allow the MMLE process to estimate these derivatives. The second method was to take the moment derivatives ($C_{m_{\delta_E}}$ and $C_{n_{\delta_R}}$) and divide by the nondimensional moment arms. (These second estimates are shown as crossed circles on Figures 6.17 and 6.18.)

Table 6.3 Longitudinal Stability and Control Derivative Summary

Case	C_L	C_{m_q}	C_{m_u}	C_{m_α}	$C_{m_{\delta_e}}$	C_{X_u}	C_{X_α}	$C_{X_{\delta_e}}$	C_{Z_u}	C_{Z_α}	$C_{Z_{\dot\alpha}}$	$C_{Z'_{\delta_e}}$ *
2A-2	0.35	-17.95 -4.2	0.1829 5.9	-0.4790 2.5	-0.8945 2.9	-0.07227 25.9	0.08673 87.4	-0.3431 6.5	0.00244 1275.8	-3.770 2.5	0.5039 15.8	-0.30
2A-3**	0.37	-15.14 4.2	0.1699 4.8	-0.5147 2.3	-0.8537 2.7	-0.1336 12.4	0.1903 34.9	-0.3638 5.1	-0.1307 26.1	-4.075 2.4	0.5727 13.6	-0.29
2A-7	1.03	-16.63 7.1	0.4297 6.6	-0.9965 3.3	-1.101 4.3	-0.0696 40.4	-0.2619 44.8	-0.9179 4.8	-0.1765 34.4	-4.669 5.2	0.0965 201.5	-0.37
2A-8	1.03	-15.19 4.6	0.3610 6.3	-0.9927 2.1	-1.110 2.2	-0.0170 217.6	-0.7521 17.3	-0.8702 5.7	-0.4461 19.0	-5.574 4.3	-0.6413 29.4	-0.38
2A-11	0.60	-14.82 4.3	0.3665 14.1	-0.6540 2.4	-0.9036 2.9	0.1522 50.1	0.5851 23.9	-0.4553 8.9	-0.2103 49.1	-3.799 3.1	0.3571 28.0	-0.31
2A-12	0.60	-17.16 3.6	0.2723 10.1	-0.6016 2.6	-0.9492 2.4	0.1126 46.9	0.5965 17.1	-0.4462 8.2	-0.2935 34.7	-3.480 4.1	0.4997 25.8	-0.32
2A-15**	0.90	-17.51 5.0	0.3845 7.7	-0.8530 2.7	-1.144 3.2	0.1806 43.0	0.07329 191.3	-0.6606 7.7	-0.2671 47.7	-4.473 4.0	0.2211 65.8	-0.39
2A-16	0.91	-18.41 3.9	0.3976 7.9	-0.8313 2.2	-1.155 2.4	0.0710 116.5	0.05211 263.9	-0.6843 7.8	-0.1968 71.4	-4.444 3.6	0.1794 72.4	-0.39
2A-19	0.30	-19.93 4.2	0.2151 8.6	-0.5163 2.4	-1.063 2.9	-0.1812 39.0	0.03508 23'.0	-0.2719 10.5	-0.2401 42.1	-4.478 2.4	0.1603 54.3	-0.36
2A-20	0.31	-18.14 4.3	0.1167 11.5	-0.5071 2.6	-1.0142 2.8	-0.1259 26.8	-0.0857 150.8	-0.2649 10.9	-0.0186 269.8	-4.741 2.5	0.1261 75.3	-0.34
2B-2**	0.33	-19.57 2.6	0.1263 13.8	-0.4881 1.5	-1.000 1.8	-0.0593 86.4	-0.3135 39.4	-0.2579 8.2	-0.2309 57.4	-4.475 1.6	0.09555 54.4	-0.34
3A-3	0.36	-20.31 4.1	0.1830 5.3	-0.4896 2.7	-1.002 2.9	-0.1496 14.4	-0.2577 48.1	-0.3763 7.4	0.0194 170.9	-4.213 2.5	0.3060 27.6	-0.34
3A-8	0.61	-16.18 4.3	0.3410 24.0	-0.7236 3.0	-1.040 2.9	0.8987 34.5	0.6966 38.8	-0.3641 16.4	-0.3170 109.5	-4.598 3.3	0.1458 91.4	-0.35
4A-12	1.11	-15.16 4.9	0.2284 6.4	-1.206 2.0	-1.428 2.2	-0.1381 12.6	0.6329 23.0	-0.9334 6.4	-0.0130 501.9	-7.329 3.1	-0.9574 19.6	-0.49
4A-14	0.86	-16.88 2.4	0.4504 5.9	-0.8655 1.5	-1.120 1.4	-0.07767 52.4	0.4096 19.9	-0.7274 5.2	-0.00098 10679.0	-4.651 2.3	-.0842 107.0	-0.38
4A-19	0.67	-17.99 3.2	0.0846 10.8	-0.7476 1.6	-1.175 1.9	-0.01388 59.8	-1.0863 23.2	-0.5572 6.5	-0.00104 1230.8	-5.442 2.3	-0.3892 27.4	-0.37
4C-2**	0.37	-19.85 2.5	0.1327 1.8	-0.5591 1.4	-1.085 1.7	-0.1067 4.7	0.0696 65.7	-0.3390 5.0	0.0164 50.2	-5.084 1.4	-0.0273 207.3	-0.37
4C-3	0.36	-17.071 2.6	0.1267 3.2	-0.5279 1.5	-0.9691 1.6	-0.09434 7.8	0.04277 164.5	-0.3257 5.1	0.08176 19.3	-5.002 1.5	-0.0597 105.7	-0.33

* $C_{Z'_{\delta_e}} = C_{m_{\delta_e}}$ /nondimensional moment arm. (Nondimensional moment arm = 2.94.)

** Data sample rate is 20 SPS.

top value is the derivative
lower value is the Cramèr Rao bound

ORIGINAL PAGE IS
OF POOR QUALITY

Table 6.4 Lateral-Directional Stability and Control Derivative Summary

Case	C_L	C_{l_p}	C_{l_r}	C_{l_β}	$C_{l_{\delta_A}}$	$C_{l_{\delta_R}}$	C_{n_p}	C_{n_r}	C_{n_β}	$C_{n_{\delta_A}}$	$C_{n_{\delta_R}}$	$\sin \phi_1$	ϕ_1	C_{Y_β}	$C_{Y_{\delta_A}}$	$C_{Y_{\delta_R}}$	$C_{Y_{\delta_R}}'$ *
2A-4	0.35	-0.6279 5.7	0.00674 156.9	-0.07658 5.3	0.3472 5.4	-0.01325 29.9	-0.06450 25.0	-0.1232 2.9	0.03843 5.2	0.04033 25.0	-0.06800 3.1	0.0134 98.9	0.77°	-0.4289 2.7	-0.00890 98.9	-0.00411 416.6	0.19
2A-5	0.34	-0.6288 6.2	-0.06350 23.6	-0.07913 5.9	0.3561 5.6	-0.01769 31.0	-0.05820 37.5	-0.1562 3.5	0.04385 5.5	0.03944 31.1	-0.07451 4.2	-0.0250 53.8	-1.43°	-0.4814 3.4	-0.00324 642.5	0.07039 30.0	0.21
2A-6	0.34	-0.7353 5.1	-0.06304 21.6	-0.08314 5.0	0.3893 4.8	-0.02534 18.9	-0.01114 154.4	-0.1344 3.1	0.04245 4.2	0.01120 79.8	-0.06407 3.7	-0.0179 54.8	-1.03°	-0.4419 2.7	-0.02898 49.1	0.03536 48.1	0.18
2A-9	1.03	-0.5775 3.4	0.11580 6.6	-0.05613 3.4	0.2619 3.0	0.00248 123.6	-0.03722 36.4	-0.1569 2.8	0.03257 3.8	0.01073 54.2	-0.07922 2.2	0.1120 15.8	6.43°	-0.5100 2.1	-0.00387 419.0	0.08694 21.7	0.23
2A-13	0.60	-0.5742 3.4	0.06380 9.8	-0.06658 3.2	0.2868 3.1	0.00234 99.4	-0.07413 15.3	-0.1276 2.2	0.02844 4.2	0.03869 14.3	-0.06479 1.7	0.0812 14.8	4.66°	-0.4241 1.6	-0.02500 49.1	0.02538 50.5	0.19
2A-14	0.60	-0.5493 5.0	-0.03952 22.9	-0.06519 4.5	0.2433 4.3	-0.02552 15.4	-0.13510 13.2	-0.1922 2.3	0.02384 7.2	0.04779 15.5	-0.08858 3.0	0.0856 19.7	4.91°	-0.4319 2.3	-0.04820 32.3	0.08243 24.2	0.25
2A-17	0.91	-0.6422 5.4	0.00796 119.1	-0.07881 4.9	0.2885 4.7	-0.00564 61.4	-0.17940 12.8	-0.2150 1.93	0.02427 9.4	0.06496 15.0	-0.09630 2.5	0.0975 19.9	5.60°	-0.5480 2.6	-0.03651 57.3	0.17260 13.6	0.27
2A-18	0.91	-0.7466 5.0	0.04305 23.2	-0.09981 4.8	0.3378 4.7	0.00431 64.9	-0.15570 13.6	-0.1550 2.7	0.02806 9.0	0.07035 13.0	-0.07325 1.9	0.0241 71.2	1.38°	-0.5921 2.6	-0.11150 19.0	0.06632 29.9	0.21
3A-4	0.36	-0.6094 4.4	-0.00634 163.4	-0.08362 4.2	0.3696 4.1	-0.01000 37.6	-0.01626 89.3	-0.1334 2.6	0.04729 3.9	0.02088 42.4	-0.07364 2.6	-0.0049 241.7	-2.28°	-0.4958 2.0	-0.01054 178.2	0.05620 27.2	0.20
3A-5	0.36	-0.5265 4.4	0.08359 12.8	-0.07357 4.6	0.3114 4.1	0.00881 40.2	-0.01389 102.3	-0.1179 3.6	0.04365 4.4	0.01514 43.4	-0.06911 2.4	0.0065 200.8	0.37°	-0.4632 2.8	-0.01940 84.1	0.01786 94.6	0.20
3A-9	0.64	-0.5295 4.2	0.06451 13.6	-0.07193 4.2	0.2611 3.8	-0.01263 37.1	0.01406 103.1	-0.1052 4.0	0.04668 4.2	0.00569 120.8	-0.06690 3.7	0.0031 555.64	0.18°	-0.5294 2.8	-0.01040 178.0	-0.00674 479.2	0.19
3A-10**	0.61	-0.5623 3.5	0.03624 20.3	-0.07223 3.3	0.2955 3.2	0.00096 285.2	0.01688 76.2	-0.1208 2.4	0.04631 3.5	0.00390 169.7	-0.07440 1.9	0.0182 72.8	1.04°	-0.5388 2.1	-0.03570 43.7	0.09151 20.7	0.21
4A-8**	1.10	-0.7145 3.3	0.02607 43.3	-0.10110 3.9	0.3322 2.9	0.01584 18.7	-0.13710 12.8	-0.1968 2.2	0.03350 6.6	0.06427 12.1	-0.08131 2.5	0.0204 89.4	1.17°	-0.7188 3.0	-0.10000 20.4	0.1630 14.3	0.23
4A-15	0.86	-0.6167 6.2	-0.00602 216.8	-0.07175 5.7	0.2940 5.3	-0.01019 43.2	-0.18370 14.7	-0.2126 3.1	0.02871 8.8	0.07998 15.0	-0.09840 3.5	0.1245 19.7	7.15°	-0.5428 3.7	-0.00548 483.2	0.16370 15.8	0.28
4A-18	0.65	-0.6865 5.1	0.02036 46.8	-0.09428 [illegible]	0.3653 4.7	-0.00062 548.0	-0.06335 30.9	-0.1381 2.6	0.04000 6.3	0.04628 22.2	-0.07998 2.1	0.0509 30.4	2.92°	-0.5635 2.5	-0.04670 49.1	0.09226 23.2	0.23
4A-20	0.69	-0.6038 4.1	0.03108 25.3	-0.08086 4.0	0.3060 3.6	-0.00667 42.9	-0.11260 14.4	-0.1549 2.3	0.03146 6.2	0.06162 13.0	-0.08167 1.8	0.0806 19.9	4.62°	-0.5220 2.3	-0.03497 58.1	0.10730 19.8	0.23
4B-5	0.38	-0.5422 3.6	0.01051 74.2	-0.06686 3.8	0.2977 3.4	-0.0246 11.7	0.02770 45.9	-0.1524 2.5	0.04917 3.2	0.02228 31.5	-0.09204 2.1	0.0379 22.5	2.17°	-0.4436 2.1	0.06914 18.3	0.02243 51.7	0.36

$^{*}C_{Y_{\delta_R}}' = C_{n_{\delta_R}}$/nondimensional moment arm. (Nondimensional moment arm = 0.35.)

**Data sample rate is 20 SPS.

top value is the derivative
lower value is the Cramèr Rao bound

ORIGINAL PAGE IS
OF POOR QUALITY

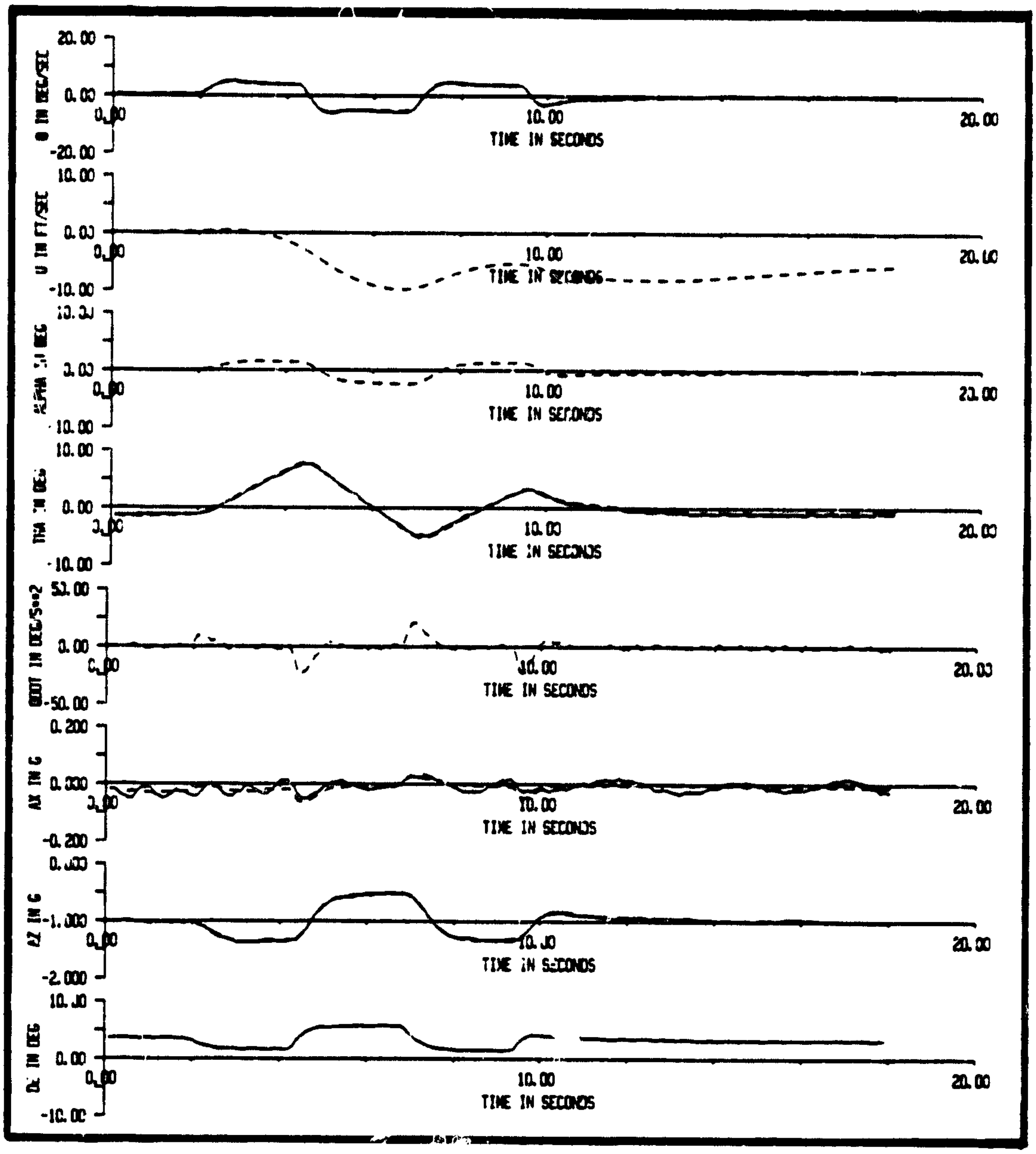


Figure 6.15 Typical Longitudinal Flight Time History Comparison (Run 2A-3)

ORIGINAL PAGE IS
OF POOR QUALITY

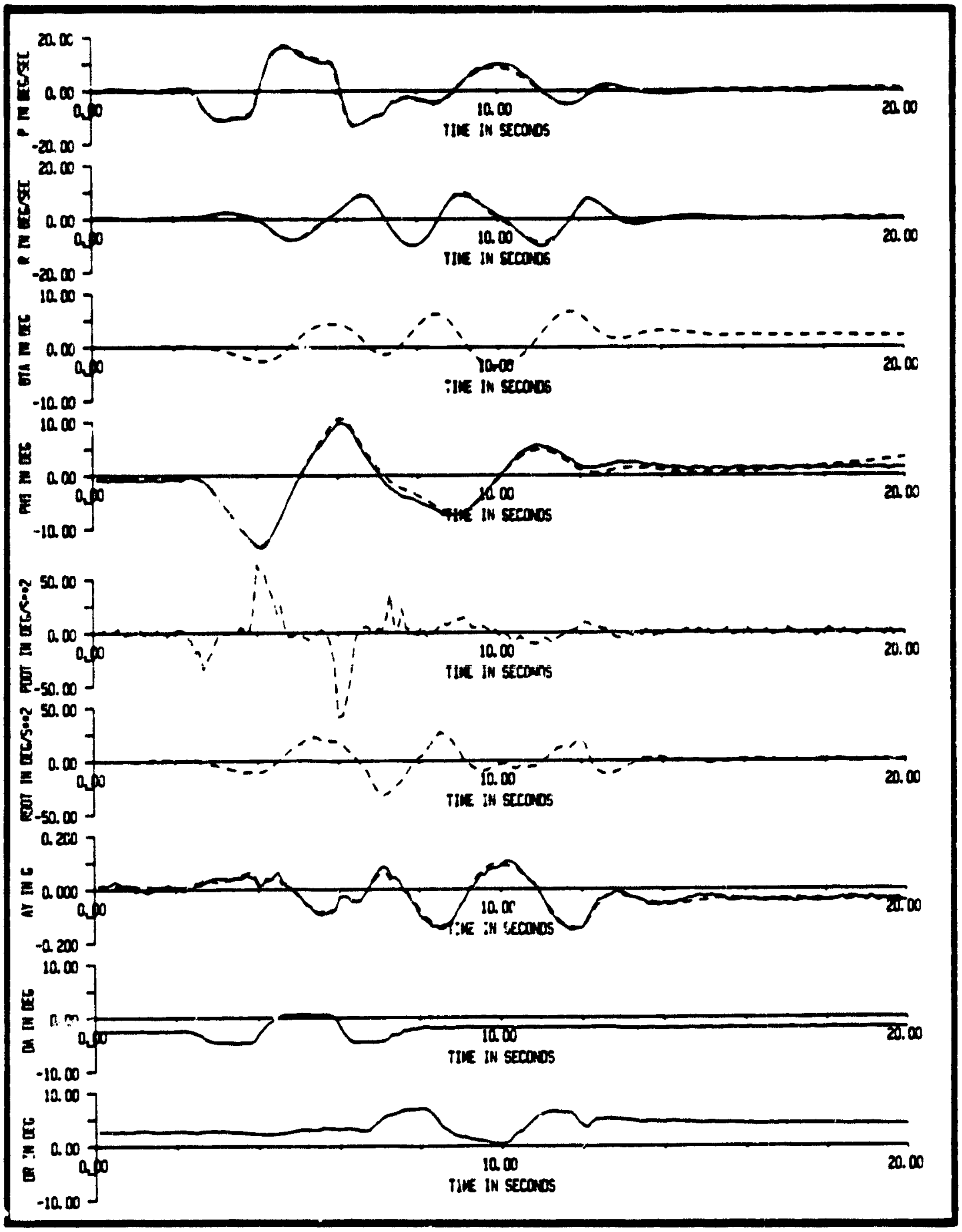


Figure 6.16 Typical Lateral-Directional Flight Time History Comparison (Run 2A-4)

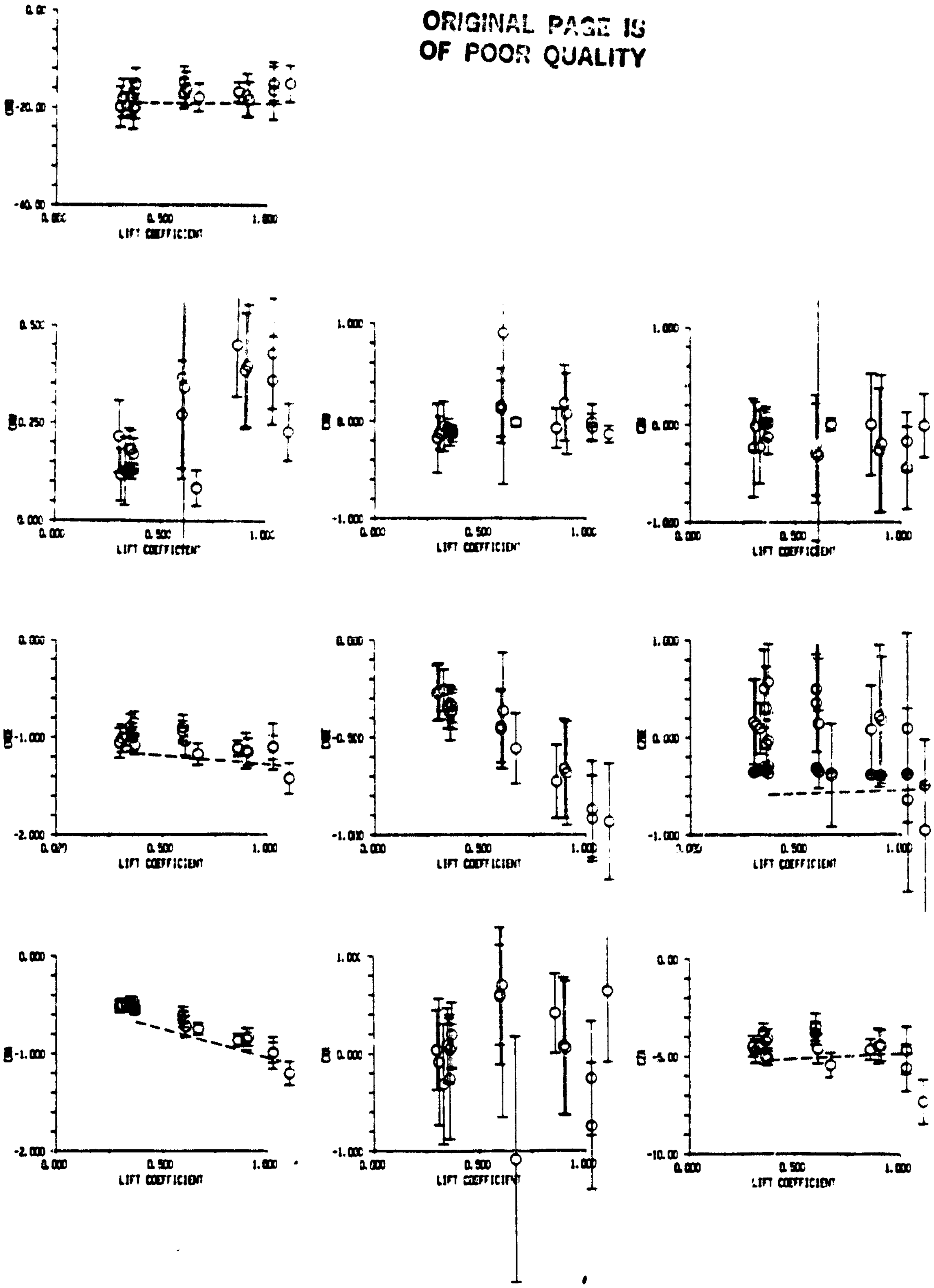


Figure 6.17 Longitudinal Stability and Control Derivative Comparisons

ORIGINAL PAGE IS
OF POOR QUALITY

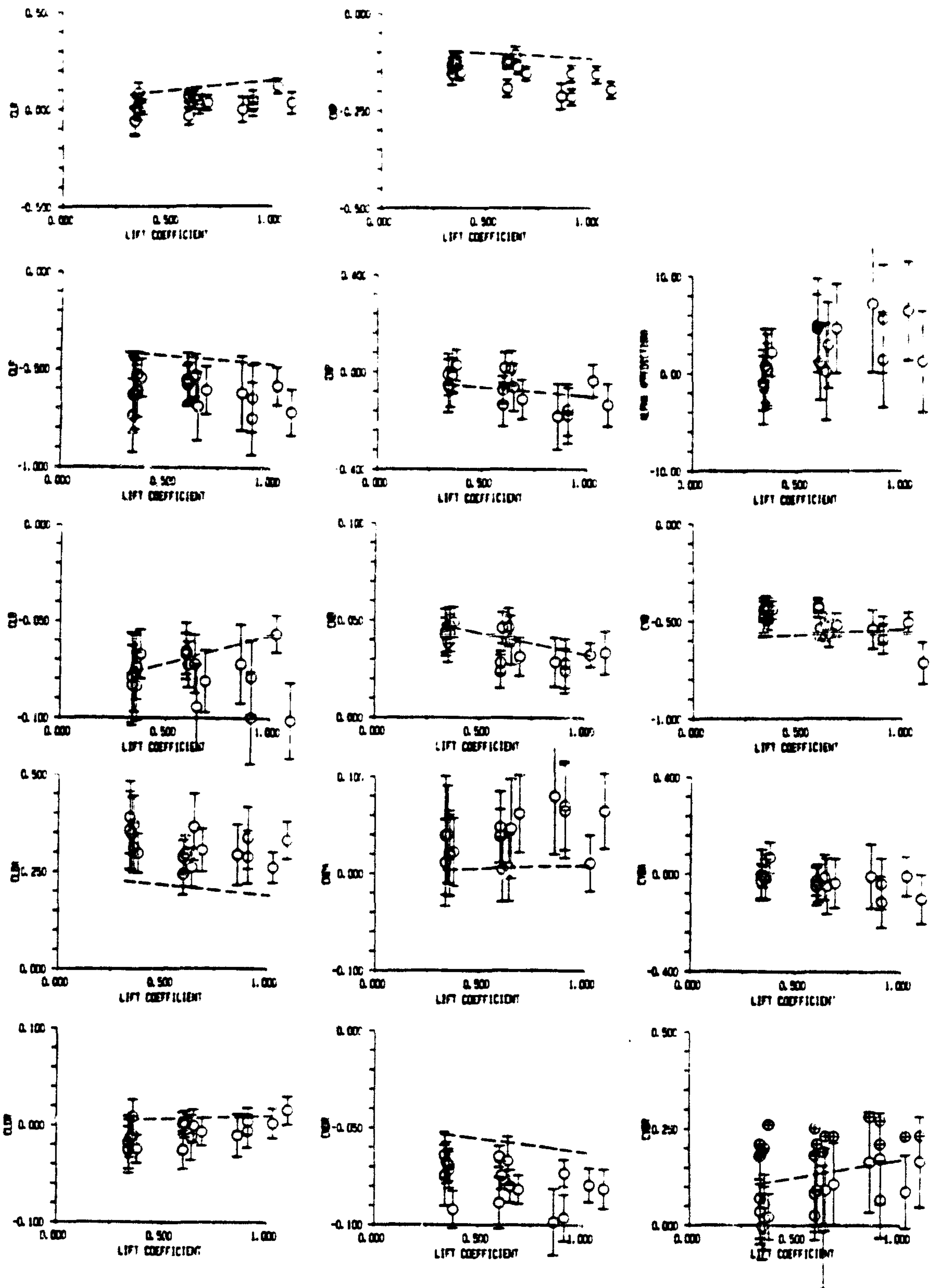


Figure 6.18 Lateral-Directional Stability and Control Derivative Comparisons

7. CONCLUSIONS

The flight test system that has been designed, developed, and evaluated in three separate flight test programs has consistently met the following objectives:

- It is easy to install.
- It is self-contained.
- It is simple to operate.
- The flight test program requires no complex flight maneuvers.
- The system has shown that it is applicable to general-aviation-class airplanes.
- The system is capable of longitudinal and lateral-directional stability and control parameter estimation.
- The system has proven to be low in cost when compared to other systems of this sophistication and capability.

In the analysis of flight test data, the MMLE method is capable of estimating all parameters in the vector differential equation [5.5]. These estimated parameters are body axes dimensional stability and control derivatives which can be converted to nondimensional format using Tables 5.2 and 5.3. They can be expressed in stability axes using the equations of Appendix C.

Some of the flight test work from Phase III has shown that relative to the earlier phases the acceleration derivatives--i.e., $C_{Z_{\delta_E}}$ and $C_{Y_{\delta_R}}$ --were better predicted in the earlier tests. The basic differences in data were insignificant with the exception of the sharpness of the step control inputs. This is evidenced by the angular accelerations which in the earlier work were twice the value of those in Phase III. This has led to some recommendations about the flight maneuver for these types of flight testing.

The basic 2-3-1 step input has proven to be an easy maneuver to fly and a relatively good maneuver for getting reasonable estimates of the stability and control derivatives. For good estimation of all control derivatives, the step inputs should be sharp and well-defined within the operational flight load envelope of the airplane. To assure accurate estimation of the long period dynamics of the airplane math model, the time histories need to include at least one full oscillation of these dynamic modes.

The 20-sample-per-second data rate, which was tested in Phase III, was not found to give significantly better estimates than the 10-sample-per-second rate. It may be found that with the sharper step inputs an improvement due to the higher data rate may be noticed.

Recommendations for modifications and uses of this flight test package are made in Chapter 8.

The total system component cost can be broken into three costs:

(1) cost of ground-based computer system, $25,000;

(2) cost of flight test instruments, $12,000; and

(3) cost of data management computer, $2,000.

These costs do not include the money spent for system construction. These costs are believed to be lower than similar system costs.

The cost of one commercially available system with similar (or greater) capabilities was $24,000. This system consisted of the following components:

(1) Sealed Cartridge Recorder with controller

(2) 8-bit computer with an RS232 interface

(3) 32-channel, 12-bit analog-to-digital conversion module.

The system met the following military specifications: MIL-E-4158, MIL-E-5400, and MIL-E-16400.

8. RECOMMENDATIONS

The recommendations made in the Phase II report concerning equipment, calibration procedures, and data reduction techniques have largely been incorporated into the present system. Some further refinements in the size of the computer package are still possible. The use of a pendulum for calibration of the entire system has increased the system accuracy as well as reduced the Cramèr-Rao bounds of the estimated parameters. Some further recommendations about the system are:

- A new, more accurate airspeed and altitude sensor should be acquired.
- Maneuver inputs should be examined in more detail to see the effects upon estimated parameters.
- Limits for the applicability of the estimated math model should be tested. In particular the applicability could be tested in level turns or other high "g" flight conditions.
- Some form of quick look capability should be incorporated into the system to improve the efficiency of remote site operations.

PRECEDING PAGE BLANK NOT FILMED

9. REFERENCES

1. Renz, R. R. L.; Clarke, R.; Roskam, J.: "A Literature Survey of Performance and Stability Flight Testing." KU-FRL-407-1, University of Kansas Flight Research Laboratory; Lawrence, Kansas; 1979.

2. Renz, R. R. L.; Clarke, R.; Hill, J. R.; Mosser, M.; Roskam, J.; Rummer, D. I.: "Progress Report on Phase I: Development of a Simple, Self-Contained Flight Test Data Acquisition System." KU-FRL-407-3; University of Kansas Flight Research Laboratory; Lawrence, Kansas; 1980.

3. Renz, R. R. L.; Roskam, J.; Rummer, D. I.: "Progress Report on Phase II: Development of a Simple, Self-Contained Flight Test Data Acquisition System." KU-FRL-407-6, University of Kansas Flight Research Laboratory; Lawrence, Kansas; 1981.

4. Suit, W. T.; Cannaday, R. L.: "Comparison of Stability and Control Parameters for a Light, Single Engine, High-Winged Aircraft Using Different Flight Test and Parameter Estimation Techniques." NASA Technical Memorandum 80163, 1979.

5. Main, R. E.; Iliff, K. W.: "A Fortran Program for Determining Stability and Control Derivatives from Flight Data." NASA TN D-7831, 1975.

6. Gainer, T. C.; Hoffman, S.: "Summary of Transformation Equations and Equations of Motion Used in Free-Flight and Wind-Tunnel Data Reduction and Analysis." NASA SP-3070, 1972.

7. Gerlach, O. H.: "High Accuracy Instrumentation Techniques for Non-Steady Flight Measurements." In Perry, M. A., ed.: "Flight Test Instrumentation," Vol. 3, Pergamon Press; 1964.

8. Gerlach, O. H.: "Determination of Performance, Stability, and Control Characteristics from Measurements in Non-Steady Maneuvers." Stability and Control, AGARD Conf. Proc. No. 17, pgs. 497-523; 1966.

9. Gerlach, O. H.: "Measurements of Performance, Stability and Control Characteristics in Non-Steady Flight with a High Accuracy Instrumentation System." In Perry, M. A., ed.: "Aerospace Instrumentation," Vol. 4, Pergamon Press; 1966.

10. Gerlach, O. H.: "The Application of Regression Analysis to the Evaluation of Instrument Calibration." Flight Test Instrumentation, AGARD Conf. Proc. No. 32; 1967.

11. Gerlach, O. H.: "Determination of Performance and Stability Parameters from Non-Steady Flight Test Maneuvers." SAE Paper #700236, 1970.

12. Gerlach, O. H.: "The Determination of Stability Derivatives and Performance Characteristics from Dynamic Maneuvers." Report VTH-163; Delft University of Technology; Delft, The Netherlands; 1971.

13. Hosman, R. J. A. W.: "A Method to Derive Angle of Pitch, Flight Path Angle and Angle of Attack from Measurements in Non-Steady Flight." Report VTH-156; Delft University of Technology; Delft, The Netherlands; 1971.

14. Hosman, R. J. A. W.: "Advanced Flight Test Instrumentation: Design and Calibration." Memorandum M-222; Delft University of Technology; Delft, The Netherlands; October 1974.

15. van Woerkom, K.: "Design and Evaluation of an Instrumentation System for Measurements in Non-Steady Symmetrical Flight Conditions with the Hawker Hunter MK VII." Report LR-308; Delft University of Technology; Delft, The Netherlands; 1981.

16. Iliff, K. W.; Main, R. E.: "Practical Aspects of Using a Maximum Likelihood Estimation Method to Extract Stability and Control Derivatives from Flight Data." NASA TN D-8209; 1976.

17. Iliff, K. W.: "Estimation of Aerodynamic Characteristics from Dynamic Flight Test Data." Dymamic Stability Parameters, AGARD Conf. Proc. #235; 1978.

18. Iliff, K. W.; Maine, R. E.; Montogomery, T. D.: "Important Factors in the Maximum Likelihood Analysis of Flight Test Maneuvers." NASA TP 1459, 1979.

19. Iliff, K. W.; Maine, R. E.: "NASA BONES." NASA Dryden, unpublished computer program.

20. Klein, V.; Gregory, R.: "A Proposal for Self-Contained Instrumentation System for Flight Research on Stability and Control." Report CIT-FI-72-013, Cranfield Institute of Technology, March 1972.

21. Smetana, F. O.; Fox, S. R.: "Flight Test Evaluation of Predicted Light Aircraft Drag, Performance, and Stability." NASA CR-158076, 1979.

22. Sorensen, J. A.; Tyler, J. S.; Powell, D. J.: "Evaluation of Flight Instrumentation for the Identification of Stability and Control Derivatives." AIAA Paper 72-963, September 1972.

23. Eckholdt, D. C.; Wells, W. R.: "A Survey of AFFDL Parameter Estimation Efforts and Future Plans." Parameter Estimation Techniques and Applications in Aircraft Flight Testing, NASA TN D-7647, 1973.

24. Mace, W. D.; Pool, A. (editors): "Flight Test Instrumentation Series." AGARD No. 160, Vols. 1-8; 1973 to 1977.

25. Roskam, J.: "Airplane Flight Dynamics and Automatic Flight Controls." Roskam Aviation and Engineering Corporation, Route 4, Box 274, Ottawa, Kansas; 1979.

26. Renz, R. R. L.; Mosser, M. A.: "Digital Tape Qualifying Procedure for KU-FRL Instrumentation Package." KU-FRL-407-5, University of Kansas Flight Research Laboratory; Lawrence, Kansas; 1981.

27. Renz, R. R. L.: "Flight Test Instrumentation Certification Report." KU-FRL-407-2, University of Kansas Flight Research Laboratory; Lawrence, Kansas; 1980.

28. Wuest, W.: "Measurements of Flow Speed and Flow Direction by Aerodynamic Probes and Vanes." AGARD Conf. Proc. 32, 1967.

29. Ikhtiari, P. A.; Marth, V. G.: "Trailing Cone Static Pressure Measurement Device." Journal of Aircraft, Vol. 1, No. 2, pgs. 93-94; March 1969.

30. Lewis, J. R.: "TRS-80 Performance, Evaluation by Program Timing." Byte Magazine, pgs. 84-94; McGraw-Hill; New York; March 1980.

31. Renz, R. R. L.: "Calibration of MDAS-16 Analog-to-Digital Converter." KU-FRL-407-4, University of Kansas Flight Research Laboratory; Lawrence, Kansas; 1981.

10. KU-FRL INSTRUMENTATION SYSTEM REPORTS

KU-FRL Number	Title	Date
KU-FRL-407-1	A Literature Survey of Performance and Stability Flight Testing	1979
KU-FRL-407-2	Flight Test Instrumentation Certification Report	1980
KU-FRL-407-3	Progress Report on Phase I: Development of a Simple, Self-Contained Flight Test Data Acquisition System	1980
KU-FRL-407-4	Calibration of MDAS-16 Analog-to-Digital Converter	1981
KU-FRL-407-5	Digital Tape Qualifying Procedure for KU-FRL Instrumentation Package	1981
KU-FRL-407-6	Progress Report on Phase II: Development of a Simple, Self-Contained Flight Test Data Acquisition System	1981
KU-FRL-407-7 (present report)	Progress Report on Phase III: Development of a Simple, Self-Contained Flight Test Data Acquisition System	1982
KU-FRL-407-P1	Development of a Simple, Self-Contained Flight Test Data Acquisition System (Paper presented at Society of Flight Test Engineers, Atlanta, Georgia)	1980
KU-FRL-407-P2	A Microcomputer Based Data Acquisition System for Use in Flight Testing of General Aviation Airplanes (Paper presented at IEEE Mid America Electronics Conference, Kansas City)	1980
KU-FRL-407-P3	Development of a Simple, Self-Contained Flight Test Data Acquisition System (SAE Paper 810596, presented at Business Aircraft Meeting & Exposition, Wichita, Kansas, April 7-10)	1981
KU-FRL-407-P4	Data Acquisition/Reduction System for Flight Testing General Aviation Aircraft (Paper presented at ISMM conference, San Francisco, May 20-22; in International Society for Mini and Microcomputers publication ISBN 0-88986-026-2)	1981

APPENDIX A

COMPUTER PROGRAMS

This appendix contains listings of the programs and subroutines used in the flight test system. Each listing is preceded by a brief description, a flow chart (when needed for clarification), and programming notes that explain conventions used and point out items needed for a better understanding of the operation of the programs.

A.1 AIM-65 EPROM PROGRAMS
A.2 MINC DATA TRANSFER (AIM-65 TO MINC)
A.3 MINC QUICK LOOK DATA PLOT (CRT GRAPHICS)
A.4 MINC ENGINEERING CONVERSION
A.5 MINC INSTRUMENTATION CALIBRATION
A.6 MINC MMLE SET-UP
A.7 MINC MMLE PROGRAM (NEWTON)
A.8 MINC TIME HISTORY PLOTTING (HARD COPY OUTPUT)
A.9 MINC PLOTTER LIBRARY ROUTINES (FOR HP 7225B PLOTTER)
A.10 MINC TIME HISTORY PLOTTING (CRT GRAPHICS)
A.11 MINC DATA TRANSFER (MINC TO MAINFRAME COMPUTER)
A.12 DATA ERROR CORRECTION
A.13 MINC SUMMARY DERIVATIVE OUTPUT

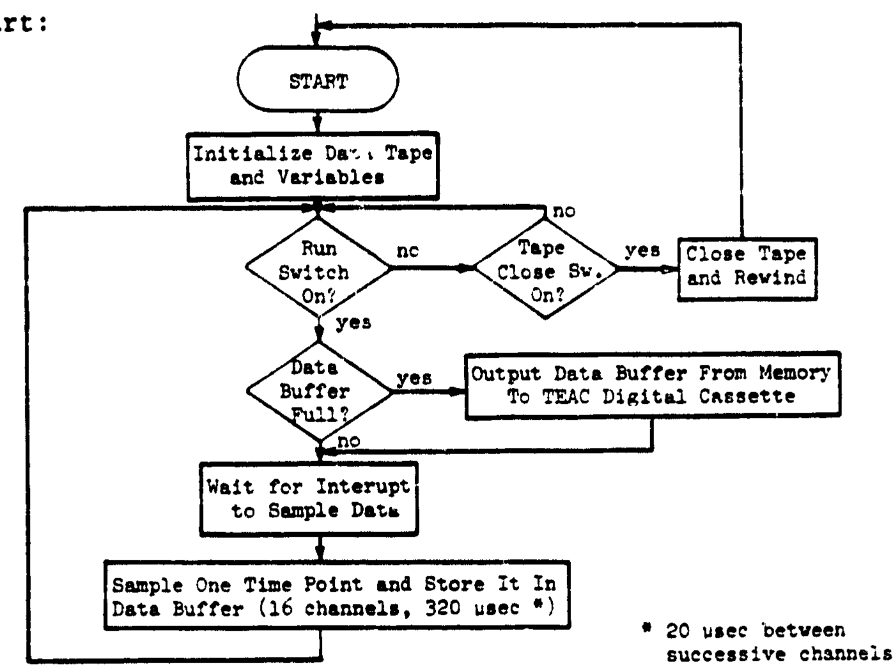
A.1) AIM ROM PROGRAMS

Description: This program consists of an executive that performs variable initialization and activates one of the four modes of the program. The EPROM is placed in the AIM-65 in the location normally reserved for the BASIC interpreter ROM. The monitor routines provide two entry locations used to start initialization of the variable and either the 10 samples per second (SPS) data acquisition program or the 20 SPS program. Once the initialization is complete, the 3 special function keys provide linkages to the data recovery program, the MDAS-16 direct readout program, and the 100 SPS data acquisition program.

Program 10 OR 20 SPS DATA ACQUISITION

Description: This AIM-65 program collects and saves the measured state time histories. Information is collected and stored on the cassette tape in blocks of 5 timepoints (one-half or one-fourth seconds of data). The data for each channel are coded as two binary eight-bit words totaling sixteen bits. The first word contains the eight most significant bits. The second word holds the four most significant and the four least significant bits. This provides a redundancy check for the four highest bits of each measurement.

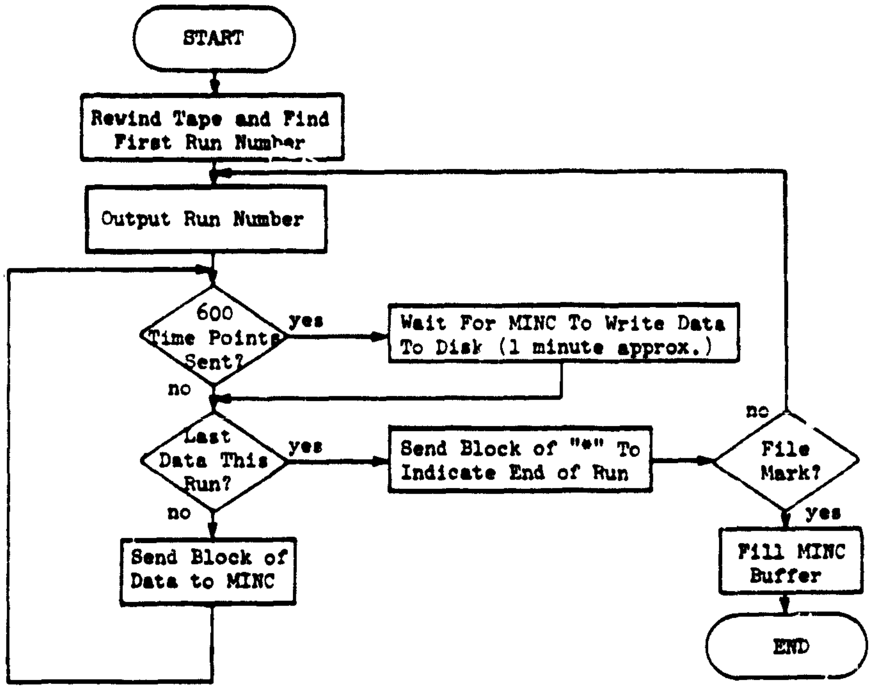
Flowchart:



Program DATA RECOVERY

Description: This program permits the AIM-65 to transfer the information recorded by the DATA ACQUISITION programs. The information is passed directly to the MINC through an RS-232 port. The data format is eight data bits with even parity. The data are dumped automatically at 9600 baud.

Flowchart:



Programming Note: This routine makes extensive use of thc AIM-65 monitor. The RS-232 data transfer is made to the ground based computer at 9600 buad with even parity. The baud rate is stored in memory locations $A417 and $A418. Rockwell Application Note R6500 N08 lists values which change the transfer baud rate to other standard values. For computers that are unable to use this high transfer rate the baud rate can be lowered to an acceptable value.

ORIGINAL PAGE IS
OF POOR QUALITY

Program MDAS-16 CALIBRATION

Description: This program provides a means of looking directly at the digitization of analog inputs. Linearity and offsets can be determined for calibration of the MDAS module. The program also permits the calibration of steady state instrument outputs in direct digital fashion.

Program 100 SPS DATA ACQUISITION

Description: This program performs like the 10 or 20 SPS DATA ACQUISITION program. The data are collected much faster than they can be recorded on tape. To overcome this problem, the data buffer has been expanded to almost 12K RAM. This provides about 3.7 seconds of buffer space for data.

Listing:

```
==0000
;*****     THE UNIVERSITY OF KANSAS FLIGHT RESEARCH LAB
;*****     PROJECT 4070    BOB CLARKE
;*****     AIM-65 CLOCK SPEED MEASURED AT 999657 HZ
;*****     TIMING LOOPS REFLECT THIS SPEED FOR INCREASED
;*****     ACCURACY     TEST RUN ON 3-NOV-81 BY BOB CLARKE
;*****     PROGRAMS IN EPROM:
;*****     1) INITIALIZATION
;*****     2) 10 OR 20 SPS DATA ACQUISITION
;*****     3) DATA RECOVERY (AUTO-TRANSMISSION TO MINC)
;*****     4) MDAS-16 VOLTAGE OFFSET/LINEARITY CALIBRATION
;*****     5) 100 SPS DATA ACQUISITION FOR TRANSDUCER CALIBRAT

;PAGE ZERO VAR
==0000 RNCNT=0
==0000 BLKCNT=2
==0000 BUFCNT=4
==0000 IBUF=6
==0000 OBUF=8
==0000 CNT=10
==0000 OCNT=12
==0000 TEMPO=14
==0000 TMP=16
==0000 MEM=18
==0000 MEM1=19
==0000 MEM2=20
==0000 TIMEL=21
==0000 TIMEH=22

;MDAS-16 REG
==0000 CLEAR=$8000
==0000 SEQ=$8001
==0000 MED=$8002
==0000 LOW=$8003

;TEAC BUF
==0000 BUF1=$200
==0000 BUF2=$300
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
;TEAC REG
==0000 DBR=$9008
==0000 WDC=$9009
==0000 CDR=$900A
==0000 MDR0=$900B
==0000 CSR=$900C

==0000 ESR=$900D
==0000 ISR=$900E
==0000 MDR1=$900F

;TEAC COMDS
==0000 SRST=$92
==0000 WRT=$C1
==0000 WTM=$C2
==0000 ERA=$C3
==0000 RDL=$C4
==0000 SLF=$C8
==0000 SLE=$C9
==0000 REW=$CA

;TEAC FLAGS
==0000 EOIFLG=$02
==0000 FPT=$04
==0000 NRDY=$10
==0000 DBRE=$40

;KEYBOARD REG
==0000 KDRA2=$A480
==0000 KDDRA2=$A481
==0000 KDRB2=$A482
==0000 KDDRB2=$A483

;INT VECT ADDRESS
==0000 VECTL=$A404
==0000 VECTH=$A405

;TIMER
==0000 UDRB=$A000
==0000 UT1L=$A004
==0000 UT1CH=$A005
==0000 UT2L=$A008
==0000 UT2H=$A009
==0000 UACR=$A00B
==0000 UIFR=$A00D
==0000 UIER=$A00E

;MONITOR LINKAGES
==0000 BAUDL=$A417
==0000 BAUDH=$A418
==0000 STIY=$A427
==0000 DRB=$A800
==0000 READM=$E93C
==0000 OUTPUT=$E97A
==0000 INALL=$E993
==0000 OUTALL=$E9BC
==0000 CRLF=$E9F0
==0000 NUMA=$EA46
==0000 NOUT=$EA51
==0000 HEX=$EA7D
==0000 PHXY=$EB9E
==0000 PLXY=$EBAC
==0000 DELAY=$EC0F
==0000 RED1=$FE96

;MISC CONSTANTS
==0000 BIT5=$20

==0000 LOADK=$EF
==0000 RECK=$BF
==0000 CLOSEK=$DF
==0000 TIME1H=$27
==0000 TIME1L=$10
==0000 CR=$0D
==0000 LOADC=$4C
==0000 READC=$52
==0000 CLOSEC=$43
------------------

==0000
        *=$B000
==B000
;INITIALIZATION OF DATA SAMPLE RATE AND FUNCTION KEYS
;ENTRY POINT $B000 ('5' KEY) IS 10 SPS DATA AQ
;ENTRY POINT $B003 ('6' KEY) IS 20 SPS DATA AQ
==B000 STARTB
4C13B0 JMP INIT10
==B003 INIT20
A06A   LDY #M20SPS-M0
20C4B2 JSR MESS
A99F   LDA #<$619F
8515   STA TIMEL
A961   LDA #>$619F
8516   STA TIMEH
4C20B0 JMP INIT
==B013 INIT10
A07A   LDY #M10SPS-M0
20C4B2 JSR MESS
A93F   LDA #<$C33F
8515   STA TIMEL
A9C3   LDA #>$C33F
8516   STA TIMEH
;         LOAD ACCUM WITH JMP INSTRUCTION
==B020 INIT
A94C   LDA #$4C
8D0C01 STA $10C
8D0F01 STA $10F
8D1201 STA $112
A9EC   LDA #<DATREC
8D0D01 STA $10D
==B030
A9B1   LDA #>DATREC
8D0E01 STA $10E
A947   LDA #<MDABC
8D1001 STA $110
A9B4   LDA #>MDABC
8D1101 STA $111
A9A6   LDA #<TRANBC
==B041
8D1301 STA $113
A9B4   LDA #>TRANBC
8D1401 STA $114
;       INITIALIZE BUAD RATE FOR TRANSFER
A900   LDA #$00
8D17A4 STA BAUDL
A925   LDA #$25
8D18A4 STA BAUDH
------------------

==B053
;VARIABLE SAMPLE RATE DATA ACQUISITION
AD0080 LDA CLEAR
A992   LDA #SRST
8D0B90 STA MDR0
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
A901   LDA #1
8500   STA RNCNT
A900   LDA #0
8501   STA RNCNT+1
==B063
A9FF   LDA #$FF
8D81A4 STA KDDRA2
A900   LDA #0
8D83A4 STA KDDRB2
8D80A4 STA KDRA2
A9C0   LDA #$C0
8D0BA0 STA UACK
==B075 START
A912   LDA #$12
8D0B90 STA MDRO
A9CA   LDA #REW
2041B1 JSR COMD
A9CA   LDA #REW
2041B1 JSR COMD
==B084 MAIN
A900   LDA #0
8504   STA BUFCNT
2025B1 JSR GKEY
C9EF   CMP #LOADK
D0F3   BNE MAIN
A9CA   LDA #REW
2041B1 JSR COMD
==B094
A9C9   LDA #SLE
2041B1 JSR COMD
==B099 MAIN2
2025B1 JSR GKEY
C9BF   CMP #RECK
F01B   BEQ RECORD
C9DF   CMP #CLOSEK
D0F5   BNE MAIN2
==B0A4 CLOSE
A9C2   LDA #WTM
2041B1 JSR COMD
A9C2   LDA #WTM
2041B1 JSR COMD
A20C   LDX #12
==B0B0 CLOSE1
A9C3   LDA #ERA
2041B1 JSR COMD
CA     DEX
D0F8   BNE CLOSE1
4C75B0 JMP START
-------------------

==B0BB RECORD
A900   LDA #0
8506   STA IBUF
8508   STA OBUF
A902   LDA #>BUF1
8507   STA IBUF+1
A903   LDA #>BUF2
8509   STA OBUF+1
A9B1   LDA #>INT
==B0CB
8D05A4 STA VECTH
A9B0   LDA #<INT
8D04A4 STA VECTL
A9C0   LDA #$C0
8D0EA0 STA UIER
A515   LDA TIMEL
8D04A0 STA UT1L
==B0DD
A516   LDA TIMEH
8D05A0 STA UT1CH
58     CLI
!  MAKE (BLKCNT,BLKCNT+1) = -1
A9FF   LDA #$FF
8502   STA BLKCNT
8503   STA BLKCNT+1
==B0E9 REC2
2025B1 JSR GKEY
C9BF   CMP #RECK
D015   BNE RECX
A504   LDA BUFCNT
C9A0   CMP #160
D0F3   BNE REC2
E602   INC BLKCNT
D002   BNE REC1
==B0FA
E603   INC BLKCNT+1
==B0FC REC1
2094B1 JSR SWAP
2064B1 JSR WRITE
4CE9B0 JMP REC2
==B105 RECX
A504   LDA BUFCNT
C9A0   CMP #160
D0FA   BNE RECX
A940   LDA #$40
8D0EA0 STA UIER
2094B1 JSR SWAP
A9FF   LDA #$FF
==B115
8502   STA BLKCNT
8503   STA BLKCNT+1
2064B1 JSR WRITE
E600   INC RNCNT
D002   BNE RECX2
E601   INC RNCNT+1
==B122 RECX2
4C99B0 JMP MAIN2
-------------------

==B125 GKEY
AD82A4 LDA KDRB2
48     PHA
A910   LDA #TIME1L
8D08A0 STA UT2L
A927   LDA #TIME1H
8D09A0 STA UT2H
==B133 GKEY1
AD0DA0 LDA UIFR
2920   AND #BIT5
F0F9   BEQ GKEY1
68     PLA
CD82A4 CMP KDRB2
D0E5   BNE GKEY
60     RTS
-------------------

==B141 COMD
48     PHA
AD0D90 LDA ESR
==B145 COMD1
AD0C90 LDA CSR
2910   AND #NRDY
D0F9   BNE COMD1
AD0C90 LDA CSR
2904   AND #FPT
D0F2   BNE COMD1
68     PLA
8D0A90 STA CDR
==B157 COMD2
AD0E90 LDA ISR
2D00BF AND CCE
F0FB   BEQ COMD2
60     RTS
-------------------

==B160 WAIT
2063B1 JSR WAITX
==B163 WAITX
60     RTS
-------------------

==B164 WRITE
AD0D90 LDA ESR
A9A4   LDA #164
8D0990 STA WDC
A9C1   LDA #WRT
8D0A90 STA CDR
A500   LDA RNCNT
20A3B1 JSR WWORD
==B176
A501   LDA RNCNT+1
20A3B1 JSR WWORD
A502   LDA BLKCNT
20A3B1 JSR WWORD
A503   LDA BLKCNT+1
20A3B1 JSR WWORD
A000   LDY #0
==B187 WRITE1
B108   LDA (OBUF),Y
20A3B1 JSR WWORD
C8     INY
C0A0   CPY #160
D0F6   BNE WRITE1
==B191 WRITE2
4C57B1 JMP COMD2
-------------------

==B194 SWAP
A509   LDA OBUF+1
48     PHA
A507   LDA IBUF+1
8509   STA OBUF+1
68     PLA
8507   STA IBUF+1
A900   LDA #0
8504   STA BUFCNT
60     RTS
-------------------

==B1A3 WWORD
48     PHA
==B1A4 WWORD1
AD0E90 LDA ISR
2940   AND #DBRE
F0F9   BEQ WWORD1
68     PLA
8D0890 STA DBR
60     RTS
-------------------
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
==B1B0 INT
48      PHA
AD00A0 LDA UDRB
1031    BPL INTEX
98      TYA
48      PHA
A404    LDY BUFCNT
A90F    LDA #15
850A    STA CNT
AD0180 LDA SEQ
==B1C1
2060B1 JSR WAIT
==B1C4 ILOOP
AD0280 LDA MED
9106    STA (IBUF),Y
AD0180 LDA SEQ
C8      INY
AD0380 LDA LOW
9106    STA (IBUF),Y
C8      INY
C60A    DEC CNT
==B1D5
D0ED    BNE ILOOP
AD0280 LDA MED
9106    STA (IBUF),Y
C8      INY
AD0380 LDA LOW
9106    STA (IBUF),Y
C8      INY
8404    STY BUFCNT
==B1E5
68      PLA
A8      TAY
==B1E7 INTEX
AD04A0 LDA UTIL
68      PLA
40      RTI

--------- ----------

;DATA RECOVERY (AUTO-TRANSMISSION TO MINC)
;ENTRY POINT FROM 'F1' KEY
==B1EC DATREC
A0BA    LDY #MTRANS-MO
20C4B2 JSR MESS
A992    LDA #??ST
8D0B90 STA MDRO
A900    LDA #$00
8D0BA0 STA UACR
==B1FB MAINR
20D2B2 JSR GCOM
C94C    CMP #LOADC
F006    BEQ MAIN2R
20DEB2 JSR INVAL
4CFBB1 JMP MAINR
==B208 MAIN2R
A912    LDA #$12
8D0B90 STA MDRO
A900    LDA #0
850C    STA OCNT
850E    STA TEMPO
A9CA    LDA #REW
2088B3 JSR COMDA
==B218
A9C8    LDA #SLP
2088B3 JSR COMDA
==B21D MAIN3
20D2B2 JSR GCOM
C952    CMP #READC
F017    BEQ READ
C943    CMP #CLOSEC
F006    BEQ CLOSER
20DEB2 JSR INVAL
4C1DB2 JMP MAIN3
-------------------

==B22E CLOSER
A9CA    LDA #REW
2088B3 JSR COMDA
A9CA    LDA #REW
2088B3 JSR COMDA
4CFBB1 JMP MAINR
-------------------

==B23B READ
2046B3 JSR RBLK
B0EE    BCS CLOSER
A502    LDA BLKCNT
0503    ORA BLKCNT+1
D00A    BNE SENDB
A04B    LDY #MRNCNT-MO
20C4B2 JSR MESS
==B24B
A500    LDA RNCNT
2046EA JSR NUMA
==B250 SENDB
;RNCNT AND BLKCNT CAN BE SENT HERE
==B250 SENDB1
A200    LDX #0
==B252 CNVT
BD0002 LDA BUF1,X
29F0    AND #$F0
850E    STA TEMPO
E8      INX
BD0002 LDA BUF1,X
29F0    AND #$F0
C50E    CMP TEMPO
F003    BEQ CNVT1
==B263
20E4B3 JSR FIX
==B266 CNVT1
CA      DEX
BD0002 LDA BUF1,X
38      SEC
7E0002 ROR BUF1,X
5E0002 LSR BUF1,X
2903    AND #$3
18      CLC
2A      ROL A
2A      ROL A
==B276
2A      ROL A
2A      ROL A
850E    STA TEMPO
E8      INX
BD0002 LDA BUF1,X
290F    AND #$0F
050E    ORA TEMPO
0940    ORA #$40
9D0002 STA BUF1,X
==B287
E8      INX
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
E0A0   CPX #160
D0C6   BNE CNVT
A000   LDY #0
==B28E SENDB2
B90002 LDA BUF1,Y
20E4B2 JSR SEND
C8     INY
C0A0   CPY #160
D0F5   BNE SENDB2
E60C   INC OCNT
A60C   LDX OCNT
E078   CPX #120
==B29F
D003   BNE LAB1
201FB4 JSR MWAIT
==B2A4 LAB1
A502   LDA BLKCNT
C9FF   CMP #$FF
D004   BNE SENDB3
C503   CMP BLKCNT+1
F00B   BEQ END
==B2AE SENDB3
2046B3 JSR RBLK
B003   BCS CLOS1R
4C50B2 JMP SENDB
==B2B6 CLOS1R
4C2EB2 JMP CLOSER
-------------------
==B2B9 END
2007B4 JSR EOF
A00D   LDY #MEND-MO
20C4B2 JSR MESS
4C3BB2 JMP READ
-------------------
==B2C4 MESS
B902BF LDA MO,Y
48     PHA
297F   AND #$7F
207AE9 JSR OUTPUT
C8     INY
68     PLA
10F3   BPL MESS
60     RTS
-------------------
==B2D2 GCOM
A0D6   LDY #MINPUT-MO
20C4B2 JSR MESS
203CE9 JSR READM
20BCE9 JSR OUTALL
60     RTS
-------------------
==B2DE INVAL
A022   LDY #MINV-MO
20C4B2 JSR MESS
60     RTS
-------------------
==B2E4 SEND
;GENERATE EVEN PARITY
;SAVE X,Y ON STACK
209EEB JSR PHXY
==B2E7 PARITY
A209   LDX #09
```

```
8610   STX TMP
CA     DEX
==B2EC AGAIN
6A     ROR A
9002   BCC NOPR
E610   INC TMP
==B2F1 NOPR
CA     DEX
D0F8   BNE AGAIN
6610   ROR TMP
6A     ROR A
20ACEB JSR PLXY
;OUTPUT ACC TO TTY SUBROUTINE
;X,Y ARE PRESERVED
48     PHA
209EEB JSR PHXY
8D27A4 STA STIY
==B301
200FEC JSR DELAY
AD00A8 LDA DRB
29FB   AND #$FB
8D00A8 STA DRB
8D28A4 STA STIY+1
200FEC JSR DELAY
==B312
A208   LDX #$08
2E27A4 ROL STIY
2E27A4 ROL STIY
2E27A4 ROL STIY
==B31D OUTT1
6E27A4 ROR STIY
AD27A4 LDA STIY
2904   AND #$04
0D28A4 ORA STIY+1
8D00A8 STA DRB
08     PHP
200FEC JSR DELAY
==B32F
28     PLP
CA     DEX
D0EA   BNE OUTT1
A904   LDA #$04
0D28A4 ORA STIY+1
8D00A8 STA DRB
200FEC JSR DELAY
;EXTRA DELAY TO ALLOW FOR MINC RESPONSE
200FEC JSR DELAY
==B341
20ACEB JSR PLXY
68     PLA
60     RTS
-------------------
==B346 RBLK
AD0C90 LDA CSR
2910   AND #NRDY
D0F9   BNE RBLK
A9A4   LDA #164
8D0990 STA WDC
A9C4   LDA #RDL
8D0A70 STA CDR
==B357
20D0B3 JSR RWORD
B029   BCS RBLK2
8500   STA RNCNT
20D0B3 JSR RWORD
B022   BCS RBLK2
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
8501    STA RNCNT+1
20D0B3  JSR RWORD
==B368
B011    BCS RBLK2
8502    STA BLKCNT
20D0B3  JSR RWORD
B014    BCS RBLK2
8503    STA BLKCNT+1
A000    LDY #0
==B375  RBLK1
20D0B3  JSR RWORD
B00B    BCS RBLK2
990002  STA BUF1,Y
C8      INY
C0A0    CPY #160
D0F3    BNE RBLK1
4C97B3  JMP COMDA2
==B385  RBLK2
4C9FB3  JMP COMDA4
--------------------

==B388  COMDA
48      PHA
AD0D90  LDA ESR
==B38C  COMDA1
AD0C90  LDA CSR
2910    AND #NRDY
D0F9    BNE COMDA1
68      PLA
8D0A90  STA CDR
==B397  COMDA2
AD0E90  LDA ISR
2D00BF  AND CCE
F0F8    BEQ COMDA2
==B39F  COMDA4
AD0C90  LDA CSR
48      PHA
2902    AND #EOIFLG
F010    BEQ COMDA5
A062    LDY #WORKIN-M0
20C4B2  JSR MESS
20EAB3  JSR EOT
==B3AF
A036    LDY #MERR1-M0
20C4B2  JSR MESS
68      PLA
38      SEC
60      RTS
==B3B7  COMDA5
68      PLA
2981    AND #$81
D002    BNE COMDA3
18      CLC
60      RTS
==B3BE  COMDA3
A000    LDY #M0-M0
48      PHA
20C4B2  JSR MESS
68      PLA
2046EA  JSR NUMA
AD0D90  LDA ESR
2046EA  JSR NUMA
==B3CE
18      CLC
60      RTS
--------------------

==B3D0  RWORD
AD0E90  LDA ISR
2C00BF  BIT CCE
D00A    BNE RWORD2
2C01BF  BIT DA
F0F3    BEQ RWORD
AD0890  LDA DBR
==B3E0
18      CLC
60      RTS
==B3E2  RWORD2
38      SEC
60      RTS
--------------------

==B3E4  FIX
A056    LDY #MERROR-M0
20C4B2  JSR MESS
60      RTS
--------------------

==B3EA  EOT
A000    LDY #0
==B3EC  EOT1
A924    LDA #$24
;$
20E4B2  JSR SEND
;WAIT FOR MINC TO RESPOND (10 CALLS TO DELAY ROUTINE)
A20A    LDX #10
==B3F3  HOLD
200FEC  JSR DELAY
CA      DEX
D0FA    BNE HOLD
C8      INY
C0A0    CPY #160
D0EE    BNE EOT1
E60C    INC OCNT
A40C    LDY OCNT
C078    CPY #120
==B404
D0E4    BNE EOT
60      RTS
--------------------

==B407  EOF
A000    LDY #0
==B409  EOF1
A92A    LDA #$2A
20E4B2  JSR SEND
C8      INY
C0A0    CPY #160
D0F6    BNE EOF1
E60C    INC OCNT
A60C    LDX OCNT
E078    CPX #120
==B419
D003    BNE EOF2
201FB4  JSR MWAIT
==B41E  EOF2
60      RTS
```

```
--------------------

==B41F MWAIT
A062    LDY #WORKIN-MO
20C4B2  JSR MESS
A200    LDX #0
A000    LDY #0
==B428  MWT
A900    LDA #0
;FROM PAGE 8-37 OF AIM-65 USER'S GUIDE
850C    STA OCNT
8D04A0  STA UT1L
A9FF    LDA #$FF
8D05A0  STA UT1CH
A940    LDA #$40
==B436  MW1
2C0DA0  BIT UIFR
F0FB    BEQ MW1
AD04A0  LDA UT1L
E8      INX
D0E7    BNE MWT
C8      INY
C004    CPY #4
D0E2    BNE MWT
==B446
60      RTS

--------------------

;MDAS-16 ANALOG/DIGITAL CONVERSION CALIBRATION
;ENTRY POINT FROM 'F2' KEY
==B447  MDASC
A09D    LDY #MCALIB-MO
20C4B2  JSR MESS
A0C3    LDY #MCHANL-MO
20C4B2  JSR MESS
2096FE  JSR RED1
207DEA  JSR HEX
==B457
A8      TAY
--------------------

==B458  LOOP
AD0080  LDA CLEAR
2060B1  JSR WAIT
A900    LDA #$00
8512    STA MEM
8513    STA MEM1
8514    STA MEM2
98      TYA
AA      TAX
==B468
E8      INX
CA      DEX
==B46A  ISEQ
F00A    BEQ SAMPL
AD0180  LDA SEQ
2060B1  JSR WAIT
CA      DEX
4C6AB4  JMP ISEQ
==B476  SAMPL
2060B1  JSR WAIT
AD0280  LDA MED
AA      TAX
290F    AND #$0F
8512    STA MEM
8A      TXA
```

```
6A      ROR A
6A      ROR A
6A      ROR A
6A      ROR A
==B486
290F    AND #$0F
8513    STA MEM1
AD0380  LDA LOW
290F    AND #$0F
8514    STA MEM2
20F0E9  JSR CRLF
A513    LDA MEM1
==B496
2051EA  JSR NOUT
A512    LDA MEM
2051EA  JSR NOUT
A514    LDA MEM2
2051EA  JSR NOUT
4C58B4  JMP LOOP

--------------------

==B4A6
;100 SPS DATA ACQUISITION FOR TRANSDUCER CALIBRATION
;ENTRY POINT FROM 'F3' KEY
==B4A6 TRANSC
A0B0    LDY #MFAST-MO
20C4B2  JSR MESS
AD0080  LDA CLEAR
A992    LDA #SRST
8D0B90  STA MDRO
A901    LDA #1
8500    STA RNCNT
==B4B7
A900    LDA #0
8501    STA RNCNT+1
A9FF    LDA #$FF
8D81A4  STA KDDRA2
A900    LDA #0
8D83A4  STA KDDRB2
8D80A4  STA KDRA2
==B4C8
A9C0    LDA #$C0
8D0BA0  STA UACR
==B4CD STARTT
A912    LDA #$12
8D0B90  STA MDRO
A9CA    LDA #REW
2041B1  JSR COMD
A9CA    LDA #REW
2041B1  JSR COMD
==B4DC MAINT
2025B1  JSR GKEY
C9EF    CMP #LOADK
D0F9    BNE MAINT
A9CA    LDA #REW
2041B1  JSR COMD
A9C9    LDA #SLE
2041B1  JSR COMD
==B4ED MAIN2T
2025B1  JSR GKEY
C9BF    CMP #RECK
F01B    BEQ RECRDT
C9DF    CMP #CLOSEK
D0F5    BNE MAIN2T
==B4F8 CLOSET
A9C2    LDA #WTM
2041B1  JSR COMD
A9C2    LDA #WTM
2041B1  JSR COMD
A20C    LDX #12
==B504 CLOSIT
A9C3    LDA #EDA
2041B1  JSR COMD
CA      DEX
D0F8    BNE CLOSIT
4CCDB4  JMP STARTT
-------------------

==B50F RECRDT
A900    LDA #0
8504    STA BUFCNT
8506    STA IBUF
8508    STA OBUF
A901    LDA #1
850E    STA TEMPO
A902    LDA #>BUF1
8507    STA IBUF+1
==B51F
8509    STA OBUF+1
A9B5    LDA #>INTT
8D05A4  STA VECTH
A989    LDA #<INTT
8D04A4  STA VECTL
A9C0    LDA #$C0
8D0EA0  STA UIER
==B530
A986    LDA #<$1386
8D04A0  STA UT1L
A913    LDA #>$1386
8D05A0  STA UT1CH
58      CLI
A9FF    LDA #$FF
8502    STA BLKCNT
8503    STA BLKCNT+1
==B541 REC2T
A50E    LDA TEMPO
D0FC    BNE REC2T
E607    INC IBUF+1
A901    LDA #1
850E    STA TEMPO
A507    LDA IBUF+1
C930    CMP #$30
D0F0    BNE REC2T
==B551 RECXT
A940    LDA #$40
8D0EA0  STA UIER
==B556 LOOPT
E602    INC BLKCNT
A502    LDA BLKCNT
D002    BNE LOOP1T
E603    INC BLKCNT+1
==B55E LOOP1T
A509    LDA OBUF+1
C92F    CMP #$2F
F013    BEQ LAST
2064B1  JSR WRITE
A508    LDA OBUF
18      CLC
69A0    ADC #160
8508    STA OBUF
==B56E
```

```
A509     LDA OBUF+1
6900     ADC #0
8509     STA OBUF+1
4C56B5 JMP LOOPT
==B577 LAST
A9FF     LDA #$FF
8502     STA BLKCNT
8503     STA BLKCNT+1
2064B1 JSR WRITE
E600     INC RNCNT
D002     BNE RECX2T
E601     INC RNCNT+1
==B586 RECX2T
4CEDB4 JMP MAIN2T
--------------------

==B589 INTT
48       PHA
AD0CA0 LDA UDRB
1037     BPL INTEXT
98       TYA
48       PHA
A404     LDY BUFCNT
A90F     LDA #15
850A     STA CNT
AD0180 LDA SEQ
==B59A
2060B1 JSR WAIT
==B59D ILOOPT
AD0280 LDA MED
9106     STA (IBUF),Y
AD0180 LDA SEQ
C8       INY
AD0380 LDA LOW
9106     STA (IBUF),Y
C8       INY
C60A     DEC CNT
==B5AE
D0ED     BNE ILOOPT
AD0280 LDA MED
9106     STA (IBUF),Y
C8       INY
AD0380 LDA LOW
9106     STA (IBUF),Y
C8       INY
8404     STY BUFCNT
==B5BE
D004     BNE BTMT
A900     LDA #0
850E     STA TEMPO
==B5C4 BTMT
68       PLA
A8       TAY
==B5C6 INTEXT
AD04A0 LDA UTIL
68       PLA
40       RTI
==B5CB
         *=$BF00
==BF00 CCE
80       .BYTE $80
==BF01 DA
20       .BYTE $20
==BF02 MO
0D       .BYTE CR,'TAPE ERROR ',$A0
5441
A0
==BF0F MEND
0D       .BYTE CR,' LAST BLOCK THIS RU',$CE
204C
==BF23
CE
==BF24 MINV
0D       .BYTE CR,'    INVALID COMMAN',$C4
2020
==BF37
C4
==BF38 MERR1
0D       .BYTE CR,'  FILE MARK FOUN',$C4
2020
==BF49
C4
==BF4A MRNCNT
0D       .BYTE CR,'RUN NUMBER  ',$A0
5255
A0
==BF58 MERROR
0D       .BYTE CR,'DATA ERROR',$A0
4441
A0
==BF64 WORKIN
0D       .BYTE CR,'WORKIN',$C7
574F
C7
==BF6C M20SPS
0D       .BYTE CR,'20 SPS DATA AQ',$A0
3230
A0
==BF7C M10SPS
0D     . .BYTE CR,'10 SPS DATA AQ',$A0
3130
A0
==BF8C MTRANS
0D       .BYTE CR,'RS-232 DATA RECOV',$A0
5253
==BF9E
A0
==BF9F MCALIB
0D       .BYTE CR,'    MDAS-16 CALIB',$A0
2020
==BFB1
A0
==BFB2 MFAST
0D       .BYTE CR,' TRANSDUCER CALIB',$A0
2054
==BFC4
A0
==BFC5 MCHANL
0D       .BYTE CR,' CHANNEL #  (0-F)',$BF
2043
==BFD7
BF
==BFD8 MINPUT
0D       .BYTE CR,'COMMAND?',$A0
434F
A0
         .END
 ERRORS= 0000
```

A.2) MINC DATA TRANSFER (AIM-65 TO MINC)

Description: This program accepts raw data from the AIM-65. This raw data is collected and stored on floppy disks for use in the ENGINEERING CONVERSION and QUICK LOOK DATA PLOT routines.

Listing:

```
FORTRAN IV      V02.5-2    Sun 27-Dec-81 04:09:22                PAGE 001

0001          PROGRAM AIMIN
        C
        C.... AIM  TO MINC PROGRAM  FOR 16  CHANNEL
        C.... INPUT USING ASYNCHRONOUS TRANSMISSION
        C.... WRITTEN  BY MARK A MOSSER
        C.... MODIFIED BY ROBERT CLARKE
        C
        C     This program inputs data from the  AIM - 65  through SLU-1
        C     as characters (32 at a time ) to fill a 600 X 32 character
        C     array.   When full, this array is output to the sequential
        C     specified files.  UNFORMATTED RECORDS
        C
0002          DIMENSION IADDR (4)
0003          BYTE IDATA(32,600),NAME(15)
0004          LOGICAL*1 STAR
        C.... DEFINE FILE NAME FOR AIM-65 TRANSFERED DATA FILES
0005          DATA NAME /'D','Y','1',':','A','I','M','0','0','0',
            1            '.','D','A','T',0/
        C.... DEFINE FLAG FOR RUN DATA SEPARATOR
0006          DATA STAR /.TRUE./
        C.... INITIALIZE DATA FILE TO BLANKS
0007          DATA IDATA /19200*' '/
        C.... START PROGRAM
0008          TYPE *, 'THIS PROGRAM READS DATA FROM THE AIM-65 '
0009          TYPE *, 'AT 9600 BAUD USING INPUT PORT SLU1 '
        C.... INITIALIZE ISET FOR NUMBER OF MINC DATA FILE
0010          ISET     = 1
        C.... OPEN FIRST OUTPUT FILE FOR DATA
0011          ENCODE (3,999,NAME(8)) ISET
0012      999 FORMAT(I3)
        C.... CHANGE ENCODED BLANKS BACK TO ZEROS
0013          DO 1000 JJ=8,10
0014          IF (NAME(JJ).EQ.' ') NAME(JJ) = '0'
0016     1000 CONTINUE
        C.... OPEN OUTPUT FILE
0017          OPEN (UNIT=1,NAME=NAME,TYPE='NEW',ACCESS='SEQUENTIAL',
            1        FORM='UNFORMATTED',RECORDSIZE=32,BUFFERCOUNT=2)
0018     2002 CONTINUE
        C.... TOP OF LOOP FOR DATA INPUT FROM AIM-65
        C.... ATTACH THE INPUT PORT (OR REATTACH AS THE CASE MAY BE)
0019          TYPE *, 'ATTACH SLU 1.'
        C.... CHECK FOR ERRORS
0020          IERR=MTATCH(2)
0021          TYPE 998, IERR
0022      998 FORMAT ('IERR = ',I2)
0023          TYPE *, 'SET UP FOR READING.'
0024          IADDR(1) = "50010
0025          IADDR(2) = 0
0026          IADDR(3) = 0
0027          IADDR(4) = 0
0028          IERR     = MTSET(2,IADDR(1))
0029          TYPE 998, IERR
        C.... READ DATA FROM INPUT PORT
0030          TYPE *, '                          READ FUNCTION:'
```

```
      DO 101 I=1,600
      DO 102 J=1,32
      IERR = MTIN(2,IDATA(J,I))
  102 CONTINUE
  101 CONTINUE
      TYPE 103,I-1
  103 FORMAT(10X,'BUFFER FULL,  DUMP IT TO DISK ',/,
     .       10X,'TOTAL SIZE = ',I5,' (AIM-65 DATA BLOCKS)')
C.... OUTPUT DATA TO DISK FILE
C.... WRITE OUT DATA FILE NAME
      TYPE 700, (NAME(JJ),JJ=1,14)
  700 FORMAT(10X,'MINC FILE NAME FOR OUTPUT = ',14A1)
C.... DUMP DATA TO DISK
      DO 135 K=1,600
C.... CHECK FOR TAPE MARK FLAG (FLAG IS TRANSFER OF DOLLAR SIGN)
      IF (IDATA(1,K).EQ.'$') GO TO 135
C.... CHECK FOR END OF RUN (FLAG IS TRANSFER OF STAR)
C.... SEE IF FIRST CHARACTER ON THIS LINE IS A STAR
      IF (IDATA(1,K).NE.'*') GO TO 134
C.... FIRST CHARACTER WAS A STAR
C.... SEE IF NEXT CHARACTER IS A STAR
      IF (IDATA(2,K).EQ.'*') GO TO 133
C.... ERROR: ONLY IF NOT A STAR
      TYPE *,' **** ERROR ****  DATA HAS ERROR IN FLAG FOR END OF RUN'
C.... CLOSE FILE AND STOP
      CLOSE (UNIT=1,DISPOSE='SAVE')
      STOP
  133 CONTINUE
C.... NO ERROR, SO CLOSE FILE IF THIS IS THE FIRST LINE OF STARS
C.... SINCE 5 LINES OF STARS WILL BE TRANSFERED IN ALL
      IF (.NOT.STAR) GO TO 135
C.... MUST BE FIRST LINE OF STARS
      CLOSE (UNIT=1,DISPOSE='SAVE')
C.... OPEN NEW FILE AND SET STAR FLAG TO FALSE
C.... OPEN OUTPUT FILE FOR DATA
      ISET  = ISET+1
      ENCODE (3,999,NAME(8)) ISET
      DO 2001 JJ=8,10
      IF (NAME(JJ).EQ.' ') NAME(JJ) = '0'
 2001 CONTINUE
      TYPE 700, (NAME(JJ),JJ=1,14)
C.... OPEN OUTPUT FILE
      OPEN (UNIT=1,NAME=NAME,TYPE='NEW',ACCESS='SEQUENTIAL',
     1      FORM='FORMATTED',RECORDSIZE=32,BUFFERCOUNT=2)
C.... SET STAR FLAG TO FALSE
      STAR = .FALSE.
C.... CONTINUE THROUGH LOOP
      GO TO 135
  134 CONTINUE
C.... SET STAR FLAG TO TRUE
      STAR = .TRUE.
      WRITE (1) (IDATA(L,K),L=1,32)
  135 CONTINUE
C.... DETACH INPUT PORT TO ELIMINATE ERRORS
      TYPE *, 'DETATCH INPUT PORT.'
      IERR = MTDTCH(2)
      TYPE 998, IERR
      GO TO 2002
      END
```

A.3) MINC QUICK LOOK DATA PLOT

Description: The QUICK LOOK DATA PLOT program is used to aid in choosing maneuvers to be analyzed by the MMLE program. The data are plotted using the stripchart graphics mode of the VT-105 Graphics Terminal. This routine uses the AIM-65 data files created by the MINC DATA TRANSFER routine.

Programming Note: This routine bypasses the MINC graphics routine PLOT55 to use the stripchart graphics mode. It uses the CONVERT subroutine which is explained in the ENGINEERING CONVERSION program.

Listing:

```
FORTRAN IV      V02.5-2     Sun 27-Dec-81 02:55:30              PAGE 001

0001          PROGRAM GRAPH
        C.... THIS PROGRAM USES THE GRAPHICS CAPABILITIES
        C.... OF THE VT105 TERMINAL TO DISPLAY QUICK LOOK
        C.... PLOTS OF THE AIM-65 RECORDED DATA FILES (UNFORMATTED RECORDS)
0002          BYTE NAME(15),DATA(32,500),NAMES(15,16)
0003          LOGICAL*1 MORE,FIRST
        C.... INITIALIZE NAME ARRAY FOR FILES
0004          DATA NAME /'D','Y','1',':','A','I','M','0','0','0',
             .           '.','D','A','T',0/
        C.... INITIALIZE DATA ARRAY
0005          DATA DATA /16000*' '/
        C.... SET NAMES OF CHANNELS
0006          DATA NAMES /
             . 'R','O','L','L',' ','R','A','T','E',' ',' ',' ',' ',' ',' ',
             . 'P','I','T','C','H',' ','R','A','T','E',' ',' ',' ',' ',' ',
             . 'Y','A','W',' ','R','A','T','E',' ',' ',' ',' ',' ',' ',' ',
             . 'B','A','N','K',' ','A','N','G','L','E',' ',' ',' ',' ',' ',
             . 'P','I','T','C','H',' ','A','T','T','I','T','U','D','E',' ',
             . 'Z',' ','A','C','C','E','L','E','R','A','T','I','O','N',' ',
             . 'Y',' ','A','C','C','E','L','E','R','A','T','I','O','N',' ',
             . 'X',' ','A','C','C','E','L','E','R','A','T','I','O','N',' ',
             . 'T','O','T','A','L',' ','P','R','E','S','S','U','R','E',' ',
             . 'S','T','A','T','I','C',' ','P','R','E','S','S','U','R','E',
             . 'D','E','L','T','A',' ','R','U','D','D','E','R',' ',' ',' ',
             . 'D','E','L','T','A',' ','A','I','L','E','R','O','N','S',' ',
             . 'D','E','L','T','A',' ','E','L','E','V','A','T','O','R',' ',
             . 'T','O','T','A','L',' ','T','E','M','P',' ',' ',' ',' ',' ',
             . 'C','H','A','N','N','E','L',' ','1','5',' ',' ',' ',' ',' ',
             . 'C','H','A','N','N','E','L',' ',' ','0',' ',' ',' ',' ',' '/
        C.... INITIALIZE MORE DATA IN FILE FLAG AND FIRST TIME FLAG
0007          DATA MORE,FIRST /.FALSE.,.TRUE./
        C.... GET READY TO PLOT DATA
        C.... ERASE TEXT FROM SCREEN TOP TO BOTTOM
0008          IERR   = ITTOUR("033)
0009          IERR   = ITTOUR("133)
0010          IERR   = ITTOUR("062)
0011          IERR   = ITTOUR("112)
        C.... SET TEXT SCROLL REGION TO BOTTOM 4 LINES
0012          IERR   = ITTOUR("033)
0013          IERR   = ITTOUR("133)
0014          IERR   = ITTOUR("062)
```

```
      IERR   = ITTOUR("061)
      IERR   = ITTOUR("073)
      IERR   = ITTOUR("062)
      IERR   = ITTOUR("064)
      IERR   = ITTOUR("162)
C.... TOP OF NEW FILE LOOP
 1000 CONTINUE
C.... ENTER GRAPHICS MODE
      IERR   = ITTOUR("033)
      IERR   = ITTOUR("061)
C.... CLEAR GRAPHICS MEMORY
      IERR   = ITTOUR("111)
      IERR   = ITTOUR("060)
C.... ENABLE SQUARE GRAPHICS DISPLAY FORMAT (512X240)
      IERR   = ITTOUR("111)
      IERR   = ITTOUR("040)
      IERR   = ITTOUR("041)
C.... ENABLE VERTICAL AND HORIZONTAL LINES
      IERR   = ITTOUR("111)
      IERR   = ITTOUR("043)
      IERR   = ITTOUR("041)
C.... DEFINE VERTICAL LINES
C.... LEFT SIDE
      IERR   = ITTOUR("114)
      IERR   = ITTOUR("040)
      IERR   = ITTOUR("060)
C.... RIGHT SIDE
      IERR   = ITTOUR("114)
      IERR   = ITTOUR("077)
      IERR   = ITTOUR("077)
C.... DEFINE HORIZONTAL LINES
C.... BOTTOM LINE
      IERR   = ITTOUR("104)
      IERR   = ITTOUR("047)
      IERR   = ITTOUR("061)
C.... NEXT LINE UP
      IERR   = ITTOUR("104)
      IERR   = ITTOUR("071)
      IERR   = ITTOUR("062)
C.... NEXT LINE UP
      IERR   = ITTOUR("104)
      IERR   = ITTOUR("053)
      IERR   = ITTOUR("064)
C.... NEXT LINE UP
      IERR   = ITTOUR("104)
      IERR   = ITTOUR("075)
      IERR   = ITTOUR("065)
C.... TOP LINE
      IERR   = ITTOUR("104)
      IERR   = ITTOUR("057)
      IERR   = ITTOUR("067)
C.... DISABLE GRAPHS
      IERR   = ITTOUR("101)
      IERR   = ITTOUR("040)
      IERR   = ITTOUR("040)
C.... ENABLE STRIPCHART GRAPHICS DISPLAY
      IERR   = ITTOUR("101)
      IERR   = ITTOUR("047)
      IERR   = ITTOUR("050)
C.... WRITE INITIAL POINT TO SCREEN
      IERR   = ITTOUR("110)
      IERR   = ITTOUR("040)
      IERR   = ITTOUR("040)
C.... LEAVE GRAPHICS MODE
      IERR   = ITTOUR("033)
      IERR   = ITTOUR("062)
C.... MOVE CURSOR TO LOCATIONS FOR LABELS
      IERR   = ITTOUR("033)
      IERR   = ITTOUR("133)
```

```
      IERR    = ITTOUR("061)
      IERR    = ITTOUR("073)
      IERR    = ITTOUR("061)
      IERR    = ITTOUR("146)
C.... TYPE LABELS
      TYPE 199
  199 FORMAT(' +5.00 V',////,' +2.50 V',/////,'  0.00 V',/////,
     .          ' -2.50 V',/////,' -5.00 V')
C.... MOVE CURSOR TO SCROLLING REGION
      IERR    = ITTOUR("033)
      IERR    = ITTOUR("133)
      IERR    = ITTOUR("062)
      IERR    = ITTOUR("061)
      IERR    = ITTOUR("073)
      IERR    = ITTOUR("061)
      IERR    = ITTOUR("146)
C.... GET FILE NAME OF AIM-65 INPUT DATA FILE
      TYPE 1
    1 FORMAT(///////,
     .        '$',9X,'Input the number of the AIM-65 data file : ')
      ACCEPT 2,IAIM
    2 FORMAT(1I4)
      ENCODE (3,999,NAME(8)) IAIM
  999 FORMAT(I3)
C.... CHANGE ENCODED BLANKS BACK TO ZEROS
      DO 998 I=8,10
      IF (NAME(I).EQ.' ') NAME(I)='0'
  998 CONTINUE
      TYPE 997, (NAME(I),I=1,14)
  997 FORMAT(' ',15X,'FILE NAME: ',14A1,/)
C.... OPEN FILE FOR READING AND CONVERSION
      OPEN (UNIT=1,NAME=NAME,TYPE='OLD',ACCESS='SEQUENTIAL',
     .      FORM='UNFORMATTED',RECORDSIZE=32,BUFFERCOUNT=2,
     .      READONLY)
C.... GET VALUE OF OF PARAMETER TO DISPLAY (IF 0 THEN QUIT)
      TYPE 6
    6 FORMAT('$',9X,'Input the parameter channel number:',
     .                ' (1 - 16; 0 to quit) ')
      ACCEPT 7,INUM
    7 FORMAT(I3)
C.... SAVE CURSOR POSITION AND ATTRIBUTES
      IERR    = ITTOUR("033)
      IERR    = ITTOUR("067)
C.... SELECT BOLD CHAR ATTRIBUTES
      IERR    = ITTOUR("033)
      IERR    = ITTOUR("133)
      IERR    = ITTOUR("061)
      IERR    = ITTOUR("155)
C.... MOVE CURSOR TO LOCATIONS FOR NAMES LABEL
      IERR    = ITTOUR("033)
      IERR    = ITTOUR("133)
      IERR    = ITTOUR("061)
      IERR    = ITTOUR("073)
      IERR    = ITTOUR("063)
      IERR    = ITTOUR("060)
      IERR    = ITTOUR("146)
      TYPE 34,(NAMES(IAC,INUM),IAC=1,15)
   34 FORMAT('$',15A1)
C.... RESTORE CURSOR
      IERR    = ITTOUR("033)
      IERR    = ITTOUR("070)
      IF ((INUM.GT.16.).OR.(INUM.LT.1)) GO TO 4000
C.... TOP OF NEW CHANNEL NUMBER LOOP
 3000 CONTINUE
C.... SAVE CURSOR POSITION AND ATTRIBUTES
      IERR    = ITTOUR("033)
      IERR    = ITTOUR("067)
C.... SELECT BOLD CHAR ATTRIBUTES
      IERR    = ITTOUR("033)
```

```
              IERR    = ITTOUR('133)
              IERR    = ITTOUR('061)
              IERR    = ITTOUR('155)
      C....   MOVE CURSOR TO LOCATIONS FOR NAMES LABEL
              IERR    = ITTOUR('033)
              IERR    = ITTOUR('133)
              IERR    = ITTOUR('061)
              IERR    = ITTOUR('073)
              IERR    = ITTOUR('063)
              IERR    = ITTOUR('060)
              IERR    = ITTOUR('146)
              TYPE 34,(NAMES(IAC,INUM),IAC=1,15)
      C....   RESTORE CURSOR
              IERR    = ITTOUR('033)
              IERR    = ITTOUR('070)
      C....   REWIND FILE READ DATA AGAIN
              REWIND 1
              FIRST   = .TRUE.
      C....   TOP OF READING DATA FROM FILE LOOP
         3001 CONTINUE
      C....   INITIALIZE JLAST
              JLAST   = 1
      C....   RESET MORE DATA ON FILE FLAG
              MORE    = .FALSE.
      C....   READ DATA FROM FILE (UP TO 500 TIME POINTS)
              DO 4 J=1,500
              READ (1,END=5) (DATA(I,J),I=1,32)
              JLAST   = J
            4 CONTINUE
      C....   SET MORE DATA IN FILE FLAG TO .TRUE.
              MORE    = .TRUE.
            5 CONTINUE
      C....   TOP OF GRAPH DATA LOOP
         3002 CONTINUE
      C....   CONVERT THE DATA AND PUT IT INTO
      C....   IDATA AFTER SCALING FOR PLOTTING
              IPOS1      = INUM*2-1
              IPOS2      = INUM*2
              DO 8 K=1,JLAST
      C....   GET DATA FROM ARRAY
              IDATAH     = DATA(IPOS1,K)
              IDATAL     = DATA(IPOS2,K)
              CALL CONVRT(IDATAH,IDATAL,IDATA)
              VOLT       = (FLOAT(IDATA)/16.)*0.0024356+0.0633
              IPLT       = IDATA/327.67+139
              IUPPER     = IPLT/32
              ILOWER     = IPLT-IUPPER*32
      C....   ENTER GRAPHICS MODE
              IERR    = ITTOUR('033)
              IERR    = ITTOUR('061)
      C....   WRITE PLOT DATA TO TERMINAL
              IERR    = ITTOUR('102)
              IERR    = ITTOUR(32+ILOWER)
              IERR    = ITTOUR(32+IUPPER)
      C....   LEAVE GRAPHICS MODE
      C        IERR    = ITTOUR('033)
      C        IERR    = ITTOUR('062)
      C        TYPE 700,K,VOLT
          700 FORMAT(' ',20X,'TIME POINT = ',I4,' VOLTAGE = ',F10.3)
            8 CONTINUE
      C....   LEAVE GRAPHICS MODE
              IERR    = ITTOUR('033)
              IERR    = ITTOUR('062)
      C....   IF MORE DATA ON FILE FLAG IS .TRUE.! THEN READ MORE AND PLOT IT
      C....   SET FIRST FLAG TO FALSE
              IF (MORE) FIRST = .FALSE.
              IF (MORE) GO TO 3001
      C....   CHANNEL HAS BEEN COMPLETELY PLOTTED
      C....   GET NEW CHANNEL NUMBER
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
C.... GET VALUE OF OF PARAMETER TO DISPLAY (IF 0 THEN QUIT)
          TYPE 66
       66 FORMAT(' ')
          TYPE 6
          ACCEPT 7,INUM
          IF ((INUM.GT.16.).OR.(INUM.LT.1)) GO TO 4000
C.... THEREFORE VERTICAL SIDES ARE RESET AND STARTING POINT IS SET
C.... ENTER GRAPHICS MODE
          IERR   = ITTOUR('033)
          IERR   = ITTOUR('061)
C.... DISABLE GRAPHS
          IERR   = ITTOUR('101)
          IERR   = ITTOUR('040)
          IERR   = ITTOUR('040)
C.... ENABLE SQUARE GRAPHICS DISPLAY FORMAT (512X240)
          IERR   = ITTOUR('111)
          IERR   = ITTOUR('040)
          IERR   = ITTOUR('041)
C.... ENABLE VERTICAL AND HORIZONTAL LINES
          IERR   = ITTOUR('111)
          IERR   = ITTOUR('043)
          IERR   = ITTOUR('041)
C.... DEFINE VERTICAL LINES
C.... LEFT SIDE
          IERR   = ITTOUR('114)
          IERR   = ITTOUR('040)
          IERR   = ITTOUR('060)
C.... RIGHT SIDE
          IERR   = ITTOUR('114)
          IERR   = ITTOUR('077)
          IERR   = ITTOUR('077)
C.... DEFINE HORIZONTAL LINES
C.... BOTTOM LINE
          IERR   = ITTOUR('104)
          IERR   = ITTOUR('047)
          IERR   = ITTOUR('061)
C.... NEXT LINE UP
          IERR   = ITTOUR('104)
          IERR   = ITTOUR('071)
          IERR   = ITTOUR('062)
C.... NEXT LINE UP
          IERR   = ITTOUR('104)
          IERR   = ITTOUR('053)
          IERR   = ITTOUR('064)
C.... NEXT LINE UP
          IERR   = ITTOUR('104)
          IERR   = ITTOUR('075)
          IERR   = ITTOUR('065)
C.... TOP LINE
          IERR   = ITTOUR('104)
          IERR   = ITTOUR('057)
          IERR   = ITTOUR('067)
C.... ENABLE STRIPCHART GRAPHICS DISPLAY
          IERR   = ITTOUR('101)
          IERR   = ITTOUR('047)
          IERR   = ITTOUR('050)
C.... WRITE INITIAL POINT TO SCREEN
          IERR   = ITTOUR('110)
          IERR   = ITTOUR('040)
          IERR   = ITTOUR('040)
C.... LEAVE GRAPHICS MODE
          IERR   = ITTOUR('033)
          IERR   = ITTOUR('062)
C.... SAVE CURSOR POSITION AND ATTRIBUTES
          IERR   = ITTOUR('033)
          IERR   = ITTOUR('067)
C.... SELECT BOLD CHAR ATTRIBUTES
          IERR   = ITTOUR('033)
          IERR   = ITTOUR('133)
```

```
      IERR   = ITTOUR('061)
      IERR   = ITTOUR('155)
C.... MOVE CURSOR TO LOCATIONS FOR NAMES LABEL
      IERR   = ITTOUR('033)
      IERR   = ITTOUR('133)
      IERR   = ITTOUR('061)
      IERR   = ITTOUR('073)
      IERR   = ITTOUR('063)
      IERR   = ITTOUR('060)
      IERR   = ITTOUR('146)
      TYPE 34,(NAMES(IAC,INUM),IAC=1,15)
C.... RESTORE CURSOR
      IERR   = ITTOUR('033)
      IERR   = ITTOUR('070)
C.... IF FILE IS SMALL AND ALL DATA FROM THIS CHANNEL
C.... HAS BEEN PLOTTED THEN THE BUFFER DOES NOT GET REFILLED
      IF ((.NOT.MORE).AND.FIRST) GO TO 3002
C.... IF FILE IS NOT SMALL AND IS LARGER THAN THE BUFFER
C.... BUFFER MUST BE REFILLED FROM BEGINNING OF FILE
      IF ((.NOT.MORE).AND.(.NOT.FIRST)) GO TO 3000
 4000 CONTINUE
      CLOSE (UNIT=1)
C.... GO TO TOP OF LOOP
      GO TO 1000
      STOP
      END
```

A.4) MINC ENGINEERING CONVERSION

Description: This program performs the conversion from AIM-65 voltage format to engineering units. Instrumentation biases are not removed to engineering units. Instrumentation biases are not removed from accelerometer data; nor are the instrumentation offsets from the center-of-gravity accounted for by this program. These functions are now incorporated into the MMLE analysis program.

Programming Note: The actual conversion is handled in the macro subroutine CONVERT which is written in assembly language.

Listing:

```
FORTRAN IV      V02.5-2    Sun 07-Feb-82 01:54:20              PAGE 001

0001          PROGRAM ENGR
        C.... THIS PROGRAM CONVERTS THE AIM-65 RAW DATA TO
        C.... ENGINEERING UNITS FOR USE BY THE MMLE PROGRAM
        C.... (NOTE: ALL ANGULAR UNITS ARE RADIANS)
        C.... THE AIM-65 DATA FILES ARE UNFORMATTED RECORDS
        C.... THE OUTPUT DATA RECORD ARE ALSO UNFORMATTED RECORDS
        C........ THE DATA RECORD FOR INPUT TO THE MMLE PROGRAM IS:
        C........ [THA,Q,AZ,AX,DE,PHI,P,AY,R,DA,DR]
        C........ IF EXTRA DATA IS INCLUDED THIS DATA RECORD IS ALSO ADDED:
        C........ [U,ALP,QDOT,BTA,PDOT,RDOT]
0002          BYTE DATA(32,500),NAME(15),NNAME(15)
0003          INTEGER ILINE(16),OFFSET
0004          LOGICAL*1 MORE
0005          REAL AOUT(17),LINE(16),RUDBP(9),RUDANG(9),
             .     AILBP(8),AILANG(8),ELVBP(8),ELVANG(8)
        C.... INITIALIZATION OF CONSTANTS
        C.... INITIALIZE RUDDER ARRAYS
0006          DATA RUDBP  /-1456.,-1236., -726., -242., 176.,
             .               566.,  900., 1209., 1293./
0007          DATA RUDANG /  17.0,  15.0,  10.0,   5.0,  0.0,
             .               -5.0, -10.0, -15.0, -16.8/
        C.... INITIALIZE AILERON ARRAYS
0008          DATA AILBP  /-2020.,-1767.,-1443.,-1054.,
             .              -565., -199.,  299.,  820./
0009          DATA AILANG /  17.6,  12.1,   6.1,   0.0,
             .               -5.1,  -8.5, -12.8, -17.6/
        C.... INITIALIZE ELEVATOR ARRAYS
0010          DATA ELVBP  /-2040.,-1064., -665., -291.,
             .                97.,  403., 1025., 1699./
0011          DATA ELVANG / -27.5, -15.2, -10.2,  -5.2,
             .                0.0,   4.8,  14.8,  26.6/
        C.... INITIALIZE NAME ARRAY FOR FILES
0012          DATA NAME  /'D','Y','1',':','A','I','M','0','0','0',
             .            '.','D','A','T',0/
0013          DATA NNAME /'D','Y','1',':','M','L','E','0','0','0',
             .            '.','D','A','T',0/
        C.... INITIALIZE DATA ARRAY
0014          DATA DATA /16000*' '/
        C.... INITIALIZE MORE DATA FLAG
0015          DATA MORE /.FALSE./
        C.... INITIALIZE OFFSET COUNTER
0016          DATA OFFSET /0/
```

```
      C.... DEFINE DEGRFES TO RADIAN CONVERSION FACTOR
0017        DATA DGR /5,.'/
      C
      C.... DEFINE CHANNELS
      C....  ( 1) ROLL RATE
      C....  ( 2) PITCH RATE
      C....  ( 3) YAW RATE
      C....  ( 4) BANK ANGLE
      C....  ( 5) PITCH ATTITUDE
      C....  ( 6) Z ACCELERATION
      C....  ( 7) Y ACCELERATION
      C....  ( 8) X ACCELERATION
      C....  ( 9) TOTAL PRESSURE
      C....  (10) STATIC PRESSURE
      C....  (11) RUDDER DEFLECTION
      C....  (12) AILERON DEFLECTION
      C....  (13) ELEVATOR DEFLECTION
      C....  (14) TOTAL TEMPERATURE
      C....  (15) CHANNEL 15
      C....  ( 0) CHANNEL 0
      C
      C.... DEFINE TOP OF LOOP
0018   1000 CONTINUE
      C
      C.... GET FILE NAME OF AIM-65 INPUT DATA FILE
0019        TYPE 1
0020      1 FORMAT(///////,
         .         '$',9X,'Input the number of the AIM-65 data file : ')
0021        ACCEPT 2,IAIM
0022      2 FORMAT(1I4)
0023        ENCODE (3,999,NAME(8)) IAIM
0024    999 FORMAT(I3)
      C.... CHANGE ENCODED BLANKS BACK TO ZEROS
0025        DO 998 I=8,10
0026        IF (NAME(I).EQ.' ') NAME(I)='0'
      C.... ALSO CHANGE THE NAME FOR THE OUTPUT FILE
0028        NNAME(I)= NAME(I)
0029    998 CONTINUE
0030        TYPE 997, (NAME(I),I=1,14),(NNAME(J),J=1,14)
0031    997 FORMAT(' ',5X,'INPUT FILE NAME: ',14A1,
         .             5X,'OUTPUT FILE NAME: ',14A1,/)
      C.... OPEN FILE FOR READING AND CONVERSION
0032        OPEN (UNIT=1,NAME=NAME,TYPE='OLD',ACCESS='SEQUENTIAL',
         .        FORM='UNFORMATTED',RECORDSIZE=32,BUFFERCOUNT=2,
         .        READONLY)
      C.... OPEN FILE FOR OUTPUT OF CONVERTED ENGINEERING UNITS DATA
0033        OPEN (UNIT=2,NAME=NNAME,TYPE='NEW',ACCESS='SEQUENTIAL',
         .        FORM='UNFORMATTED',BUFFERCOUNT=2,DISPOSE='SAVE')
      C
      C.... INITIALIZE TIME POINT COUNTER
0034        ICOUNT  = 0
      C.... TOP OF READING DATA FROM FILE LOOP
0035   3001 CONTINUE
      C.... INITIALIZE JLAST
0036        JLAST   = 1
      C.... RESET MORE DATA ON FILE FLAG
0037        MORE    = .FALSE.
      C.... READ DATA FROM FILE (UP TO 500 TIME POINTS)
0038        DO 4 J=1,500
0039        READ (1,END=5) (DATA(I,J),I=1,32)
0040        JLAST   = J
0041      4 CONTINUE
      C.... SET MORE DATA IN FILE FLAG TO .TRUE.
0042        MORE    = .TRUE.
0043      5 CONTINUE
      C.... ADD NUMBER OF TIME POINTS TO TIME POINT COUNTER
```

```
      ICOUNT   = ICOUNT+JLAST
C
C.... TOP OF CONVERT DATA LOOP
C.... GET DATA FROM ARRAY
      DO 8 K=1,JLAST
C.... CONVERT THE DATA AND PUT IT INTO ILINE
      DO 19 INUM=1,16
      IDATAH   = DATA(INUM*2-1,K)
      IDATAL   = DATA(INUM*2,K)
      CALL CONVRT(IDATAH,IDATAL,IDATA)
      ILINE(INUM) = IDATA/16
   19 CONTINUE
C.... CHECK FOR DATA COMPATABILITY
      IF ((ILINE(9).GT.1800).AND.(ILINE(9).LT.1900)) GO TO 500
C.... DATA IS NOT COMPATABLE WITHOUT OFFSET
C.... TRY OLD OFFSET
      INUM     = 9+OFFSET
      IF (INUM.GT.16) INUM=INUM-16
      IF ((ILINE(INUM).GT.1800).AND.(ILINE(INUM).LT.1900)) GO TO 490
C.... OLD OFFSET IS NO GOOD; FIND NEW OFFSET
      DO 480 I=1,16
      IF ((ILINE(I).GT.1800).AND.(ILINE(I).LT.1900)) IPOS = I
  480 CONTINUE
      IF (IPOS.LT.9) IPOS = IPOS+16
      OFFSET   = IPOS-9
  490 CONTINUE
C.... OFFSET IS GOOD; RECOVER DATA
      TYPE 499,OFFSET
  499 FORMAT(40X,'  DATA IS BEING RECOVERED; OFFSET = ',I3)
      DO 491 I=1,16
      INUM     = I+OFFSET
      IF (INUM.GT.16) INUM=INUM-16
      LINE(I) = FLOAT(ILINE(INUM))
  491 CONTINUE
      GO TO 502
  500 CONTINUE
C.... DATA IS COMPATABLE WITH NO OFFSET
      DO 501 I=1,16
      LINE(I) = FLOAT(ILINE(I))
  501 CONTINUE
  502 CONTINUE
C.... CHECK FOR GOOD DATA
      IF ((LINE(10).GT.500.).AND.(LINE(14).GT.900.)) GO TO 505
C.... DATA MAY BE BAD; PRINT MESSAGE TO INFORM USER
      TYPE 506,LINE(10),LINE(14)
  506 FORMAT(40X,' DATA MAY BE BAD ',2F10.0)
  505 CONTINUE
C
C...... CONVERT TO ENGINEERING UNITS FOR OUTPUT
C...... CONVERT CHANNEL # 1 (ROLL RATE)
        AOUT( 1) = (LINE( 1)+35.)*0.0004266
C...... CONVERT CHANNEL # 2 (PITCH RATE)
        AOUT( 2) = (LINE( 2)+37.)*0.0004271
C...... CONVERT CHANNEL # 3 (YAW RATE)
        AOUT( 3) = (LINE( 3)+40.)*0.0004278
C...... CONVERT CHANNEL # 4 (BANK ANGLE)
        AOUT( 4) = LINE( 4)*0.0007642+0.02870
C...... CONVERT CHANNEL # 5 (PITCH ATTITUDE)
        AOUT( 5) =-(LINE( 5)*0.0005078+0.015460)
C...... CONVERT CHANNEL # 6 (NORMAL ACCELERATION)
        AOUT( 6) = LINE( 6)*0.0024498+0.076843
C...... CONVERT CHANNEL # 7 (LATERAL ACCELERATION)
        AOUT( 7) = LINE( 7)*0.0002447+0.002430
C...... CONVERT CHANNEL # 8 (LONGITUDINAL ACCELERATION)
        AOUT( 8) = LINE( 8)*0.0009765+0.02432
C...... CONVERT CHANNEL # 9 (TOTAL PRESSURE)
        AOUT( 9) =-LINE( 9)*0.005197+9.8520
C...... CONVERT CHANNEL #10 (STATIC PRESSURE)
        AOUT(10) =-LINE(10)*0.010441+19.886
```

```
C...... CONVERT CHANNEL #11 (RUDDER DEFLECTION)
C...... RANGE LIMIT RUDDER DEFLECTION
        IF (LINE(11).GT. 1293.) LINE(11)= 1293.
        IF (LINE(11).LT.-1456.) LINE(11)=-1456.
C...... FIND RUDDER INDEX INTO DEFLECTION TABLE
        IRUDP1    = 2
        DO 600 I=2,8
        IF (LINE(11).GT.RUDBP(I)) IRUDP1 = I+1
        IRUD      = IRUDP1-1
  600   CONTINUE
        RATRUD    = (LINE(11)-RUDBP(IRUD))/(RUDBP(IRUDP1)-RUDBP(IRUD))
        AOUT(11) = RUDANG(IRUD)+RATRUD*(RUDANG(IRUDP1)-RUDANG(IRUD))
C...... CONVERT TO RADIAN UNITS
        AOUT(11) = AOUT(11)/DGR
C...... CONVERT CHANNEL #12 (AILERON DEFLECTION)
C...... RANGE LIMIT AILERON DEFLECTION
        IF (LINE(11).GT.  820.) LINE(12)= 820.
        IF (LINE(11).LT.-2020.) LINE(12)=-2020.
C...... FIND AILERON INDEX INTO DEFLECTION TABLE
        IAILP1    = 2
        DO 601 I=2,7
        IF (LINE(12).GT.AILBP(I)) IAILP1 = I+1
        IAIL      = IAILP1-1
  601   CONTINUE
        RATAIL    = (LINE(12)-AILBP(IAIL))/(AILBP(IAILP1)-AILBP(IAIL))
        AOUT(12) = AILANG(IAIL)+RATAIL*(AILANG(IAILP1)-AILANG(IAIL))
C...... CONVERT TO RADIAN UNITS
        AOUT(12) = AOUT(12)/DGR
C...... CONVERT CHANNEL #13 (ELEVATOR DELFECTION)
C...... RANGE LIMIT ELEVATOR DEFLECTION
        IF (LINE(13).GT. 1699.) LINE(13)= 1699.
        IF (LINE(13).LT.-2040.) LINE(13)=-2040.
C...... FIND ELEVATOR INDEX INTO DEFLECTION TABLE
        IELVP1    = 2
        DO 602 I=2,7
        IF (LINE(13).GT.ELVBP(I)) IELVP1 = I+1
        IELV      = IELVP1-1
  602   CONTINUE
        RATELV    = (LINE(13)-ELVBP(IELV))/(ELVBP(IELVP1)-ELVBP(IELV))
        AOUT(13) = ELVANG(IELV)+RATELV*(ELVANG(IELVP1)-ELVANG(IELV))
C...... CONVERT TO RADIAN UNITS
        AOUT(13) = AOUT(13)/DGR
C...... CONVERT CHANNEL #14 (TEMPERATURE)
        AOUT(14) = LINE(14)*1.02072-550.33
C.... DETERMINE ESTIMATE OF ALTITUDE
      IF (AOUT(10).LT.0) AOUT(10)=0.
      H        = 145448.*(1.-((AOUT(10)/14.696)**0.19026))
C.... WRITE DATA TO OUTPUT
      WRITE(2) AOUT(5),AOUT(2),AOUT(6),AOUT(8),AOUT(13),
     .         AOUT(4),AOUT(1),AOUT(7),AOUT(3),AOUT(12),AOUT(11)
C.... TYPE OUT ALTIDUE, QBAR, AND TEMPERATURE
      PHI      = AOUT(4)*DGR
      THA      = AOUT(5)*DGR
      TYPE 700,H,AOUT(9),AOUT(14),PHI,THA
  700 FORMAT(' ',1F10.1,1F9.4,1F8.1,2F7.3)
    8 CONTINUE
      IF (MORE) GO TO 3001
      CLOSE (UNIT=1)
      CLOSE (UNIT=2)
C.... TYPE OUT TIME POINT COUNTER
      TYPE 806,ICOUNT
  806 FORMAT(40X,'TOTAL TIME POINTS = ',I10)
C.... GO TO TOP OF LOOP
      GO TO 1000
      STOP
      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

Subroutine CONVRT

Description: This subroutine converts the AIM-65 data to integers for calculation of engineering units.

Listing:

```
CONVRT  MACRO V04.00  28-JAN-82 02:05:03 PAGE 1

                .TITLE  CONVRT
                .SBTTL  CONVERSION ROUTINE
                .GLOBL  CONVRT
        CONVRT: CLC
                CLR     R1
                BIC     #177700,@2(R5)  ;PEEL OFF 1ST 2 BITS
                BIC     #177700,@4(R5)  ;OF THE ARGS
                MOVB    @2(R5),R1       ;PUT HIGH 6 BITS INTO R1
                ASH     #6,R1           ;SHIFT LEFT 6
                ADD     @4(R5),R1       ;ADD IN LOW 6 BITS
                MOV     R1,@6(R5)       ;PLACE IN RESULT ARG
                ROL     @6(R5)
                ROL     @6(R5)          ;ROTATE LEFT 4
                ROL     @6(R5)
                ROL     @6(R5)
        TST:    CMP     @6(R5),#100000  ;SEE IF POSITIVE OR NEGATIVE
                BMI     NEGA
        POS:    COM     @6(R5)          ;POSITIVE NUMBERS ARE 1'S COMPLIMENTED
                ADD     @6(R5),#1       ;THEN ADD 1
                RTS     PC              ;DONE
        NEGA:   NEG     @6(R5)          ;NEGATIVE #'S ARE 2'S COMPLIMENTED
                RTS     PC
                .END
```

A.5) MINC INSTRUMENTATION CALIBRATION

Description: This program is used to analyze the calibration data collected by the AIM-65. Gains and offsets can be adjusted by the user to minimize an error squared cost function. This allows the calibration of the rate gyros in a dynamic mode without assumptions made about response of the calibration pendulum.

Listing:

```
FORTRAN IV      V02.5-2    Sat 30-Jan-82 02:54:08                PAGE 001

            PROGRAM CALIB
      C.... THIS PROGRAM PROVIDES A MEANS TO CALIBRATE THE
      C.... THE VARIOUS TRANSDUCERS IN THE SYSTEM
      C.... USING THE AIM-65 RECORDED DATA FILES (UNFORMATTED RECORDS)
            BYTE NAME(15),DATA(32,500)
            INTEGER MDASBP(32),ILINE(16)
            LOGICAL*1 MORE
      C.... INITIALIZE MDAS ANGLE ARRAY
            DATA MDASBP /-1974, -1848, -1724, -1599,
           .             -1472, -1343, -1218, -1095,
           .             - 968, - 842, - 719, - 594,
           .             - 469, - 350, - 228, - 107,
           .                19,   139,   257,   377,
           .               496,   616,   736,   854,
           .               975,  1094,  1217,  1336,
           .              1455,  1574,  1691,  1810/
      C.... INITIALIZE NAME ARRAY FOR FILES
            DATA NAME /'D','Y','1',':','A','I','M','0','0','0',
           .           '.','D','A','T',0/
      C.... INITIALIZE DATA ARRAY
            DATA DATA /16000*' '/
      C.... DEFINE CHANNELS
      C....  ( 1) ROLL RATE
      C....  ( 2) PITCH RATE
      C....  ( 3) YAW RATE
      C....  ( 4) BANK ANGLE
      C....  ( 5) PITCH ATTITUDE
      C....  ( 6) Z ACCELERATION
      C....  ( 7) Y ACCELERATION
      C....  ( 8) X ACCELERATION
      C....  ( 9) TOTAL PRESSURE
      C....  (10) STATIC PRESSURE
      C....  (11) CHANNEL 11
      C....  (12) CHANNEL 12
      C....  (13) PENDULUM ANGLE
      C....  (14) TOTAL TEMPERATURE
      C....  (15) CHANNEL 15
      C....  ( 0) CHANNEL 0
      C.... INITIALIZE MORE DATA IN FILE FLAG
            DATA MORE /.FALSE./
       1000 CONTINUE
      C.... GET FILE NAME OF AIM-65 INPUT DATA FILE
            TYPE 1
          1 FORMAT(///////,
           .      '$',9X,'Input the number of the AIM-65 data file : ')
            ACCEPT 2,IAIM
          2 FORMAT(1I4)
            ENCODE (3,999,NAME(8)) IAIM
        999 FORMAT(I3)
```

```
C.... CHANGE ENCODED BLANKS BACK TO ZEROS
      DO 998 I=8,10
      IF (NAME(I).EQ.' ') NAME(I)='0'
  998 CONTINUE
      TYPE 997, (NAME(I),I=1,14)
  997 FORMAT(' ',15X,'FILE NAME: ',14A1,/)
C.... OPEN FILE FOR READING AND CONVERSION
      OPEN (UNIT=1,NAME=NAME,TYPE='OLD',ACCESS='SEQUENTIAL',
     .       FORM='UNFORMATTED',RECORDSIZE=32,BUFFERCOUNT=2,
     .       READONLY)
C.... READ IN THE ANGULAR OFFSET AND TIME INCREMENT
      TYPE 401
  401 FORMAT('$',5X,'Input the channel number for output: ')
      ACCEPT *,ISAVE
      DELT    = 0.05
C.... OUTPUT THE PENDULUM, DERIVED PENDULUM RATE,
C.... RATE GYRO AND ACCELEROMETER OUTPUTS
C.... SET INITIAL CONDITION ON PENDA
      ANGLEO  = 0.
C.... REWIND FILE READ DATA AGAIN
      REWIND 1
C.... TOP OF READING DATA FROM FILE LOOP
 3001 CONTINUE
C.... INITIALIZE JLAST
      JLAST   = 1
C.... RESET MORE DATA ON FILE FLAG
      MORE    = .FALSE.
C.... READ DATA FROM FILE (UP TO 500 TIME POINTS)
      DO 4 J=1,500
      READ (1,END=5) (DATA(I,J),I=1,32)
      JLAST   = J
    4 CONTINUE
C.... SET MORE DATA IN FILE FLAG TO .TRUE.
      MORE    = .TRUE.
    5 CONTINUE
C.... TOP OF CONVERT DATA LOOP
 3002 CONTINUE
      TYPE 402
  402 FORMAT('$',5X,'Input A0, A1, and angular initial condition: ')
      ACCEPT *,A0,A1,AINIT
C.... SET INITIAL CONDITIONS
      APRED   = AINIT
      SUM     = 0.
      TYPE 54
   54 FORMAT(10X,' PEND ANGLE   PEND RATE  ANGULAR RATE',
     .           '        PRED      ERROR SQ')
C.... GET DATA FROM ARRAY
      DO 8 K=1,JLAST
C.... CONVERT THE DATA AND PUT IT INTO ILINE
      DO 19 INUM=1,16
      IDATAH    = DATA(INUM*2-1,K)
      IDATAL    = DATA(INUM*2,K)
      CALL CONVRT(IDATAH,IDATAL,IDATA)
      ILINE(INUM) = IDATA/16
   19 CONTINUE
C.... CONVERT TO ENGINEERING UNITS FOR OUTPUT
C.... GET PENDULUM ANGLE CONVERSION
C.... RANGE LIMIT MDAS VALUE
      IF (ILINE(13).LT.MDASBP(1)) ILINE(13) = MDASBP(1)
      IF (ILINE(13).GT.MDASBP(32)) ILINE(13) = MDASBP(32)
      IL      = 1
      DO 32 IMDAS=1,31
      IF (ILINE(13).GT.MDASBP(IMDAS)) IL = IMDAS
   32 CONTINUE
      ILP1    = IL+1
      RATMD   = FLOAT((ILINE(13)-MDASBP(IL)))/
     .          FLOAT((MDASBP(ILP1)-MDASBP(IL)))
      ANGLE   = 10.*FLOAT(IL)+RATMD*10.+CONST
```

```
      RATE    = (ANGLE-ANGLEO)/DELT
      APRED   = APRED+(FLOAT(ILINE(ISAVE)+A0)*A1)*DELT
      SUM     = SUM+(ANGLE-APRED)**2
C.... PRINT OUT LINE
      TYPE 55, ANGLE,RATE,ILINE(ISAVE),APRED,SUM
   55 FORMAT(8X,2F12.4,1I10,6X,2G14.4)
      ANGLEO  = ANGLE
    8 CONTINUE
      GO TO 3002
      CLOSE (UNIT=1)
C.... GO TO TOP OF LOOP
      GO TO 1000
      STOP
      END
```

A.6) MINC MMLE SET-UP

Description: This program is an interactive program which is used to set up the input data for the MMLE program. Nondimensional derivatives, geometric, and inertial data for a given airplane are input and used to form the initial estimate for the MMLE program.

Listing:

```
FORTRAN IV      V02.5-2    Sun 07-Feb-82 01:59:18               PAGE 001

            PROGRAM SET
      C.... THIS PROGRAM SETS UP THE DATA USED IN MINI-MMLE
      C.... DEFAULT VALUES  (IF THEY EXIST)  ARE SHOWN AFTER EACH QUESTION.
            REAL A(5,4),B(5,4),AA(5,4),BB(5,4)
            REAL ZERO(4),BIAS(4),D1(7,7)
      C.... VARIABLES TO HOLD CHARACTER INPUT
            BYTE NAME(15),BANNER(80,4)
            DOUBLE PRECISION CASE,TEMP
      C.... INITIALIZE INPUT AND ARRAY VARIABLES
            DATA VALUE,IVALUE,AA,BB/0.,0,20*0.,20*0./
            DATA D1,BIAS,ZERO/49*0.,4*0.,4*0./
            DATA CASE,TEMP/'        ','        '/
      C.... INITIALIZE NAME ARRAY FOR FILE
            DATA NAME /'D','Y','1',':','S','E','T','0','0','0',
           .           '.','D','A','T',0/
      C.... INITIALIZE BANNER
            DATA BANNER /320*' '/
      C.... SET DEFAULT VALUES
            NN     = 200
            ITR    = 10
            MZ     = 7
            MAPR   = 0
            ISUB   = 0
            HH     = 0.10
            XLA    = 1.0
      C.... GET FILE NAME FOR SET-UP DATA
            TYPE 11
         11 FORMAT(/////,'$',9X,'Input the number of the MLE data file : ')
            ACCEPT 12,INUM
         12 FORMAT(I4)
            ENCODE (3,999,NAME(8)) INUM
        999 FORMAT(I3)
      C.... CHANGE ENCODED BLANK BACK TO ZEROS
            DO 998 I=8,10
            IF (NAME(I).EQ.' ') NAME(I)='0'
        998 CONTINUE
            TYPE 997,(NAME(I),I=1,14)
        997 FORMAT(' ',5X,'OUTPUT FILE NAME: ',14A1)
      C.... OPEN UNIT 1 FOR OUTPUT OF SET-UP DATA
         10 CONTINUE
            TYPE 30
         30 FORMAT(   //,' ',10X,'Indicate type of run:',
           .           /,' ',10X,'If Longitudinal         type "L"',
           .           /,' ',10X,'If Lateral-Directional type "D"',
           .           /,'$',10X,'Select run: ')
            ACCEPT 40,CASE
         40 FORMAT(1A8)
            IF (CASE.EQ.'L') OPEN (UNIT=1,NAME=NAME,TYPE='NEW',
           .                  RECORDSIZE=96,INITIALSIZE=50,DISPOSE='SAVE')
            IF (CASE.EQ.'D') OPEN (UNIT=1,NAME=NAME,TYPE='NEW',
           .                  RECORDSIZE=96,INITIALSIZE=50,DISPOSE='SAVE')
            IF ((CASE.NE.'L').AND.(CASE.NE.'D')) GO TO 10
      C.... START INPUT OF DATA FOR LATER OUTPUT
      C.... BASIC DATA FOR EITHER LONGITUDINAL OR LATERAL-DIRECTIONAL CASE
```

```
C.... GET BANNER
      TYPE 41
   41 FORMAT(' ',10X,'Input banner line #1: (up to 80 char) ')
      ACCEPT 42,(BANNER(I,1),I=1,80)
   42 FORMAT(40A1)
      TYPE 43
   43 FORMAT(' ',10X,'Input banner line #2: (up to 80 char) ')
      ACCEPT 42,(BANNER(I,2),I=1,80)
      TYPE 44
   44 FORMAT(' ',10X,'Input banner line #3: (up to 80 char) ')
      ACCEPT 42,(BANNER(I,3),I=1,80)
      TYPE 45
   45 FORMAT(' ',10X,'Input banner line #4: (up to 80 char) ')
      ACCEPT 42,(BANNER(I,4),I=1,80)
C.... GET NUMBER OF DATA POINTS
      TYPE 50
   50 FORMAT('$',10X'Input the number of data points: (default 200) ')
   60 FORMAT(2F10.0)
      ACCEPT 61,IVALUE
   61 FORMAT(1I5)
      IF (IVALUE.GT.0) NN=IVALUE
      IVALUE = 0
C.... GET NUMBER OF ITERATIONS
      TYPE 70
   70 FORMAT('$',10X,'Input the number of iterations: (default 10) ')
      ACCEPT 61,IVALUE
      IF (IVALUE.GT.0) ITR=IVALUE
      IVALUE = 0
C.... GET NUMBER OF STATE OBSERVATIONS
      TYPE 80
   80 FORMAT('$',10X,'Input the number of states: (default 7) ')
      ACCEPT 61,IVALUE
      IF (IVALUE.GT.0) MZ=IVALUE
      IVALUE = 0
C.... GET APRORI CONTROL NUMBER
      TYPE 90
   90 FORMAT(' ',10X,'Input the control number for the aprori option',/
     .        '$',10x,'(default 0: no aprori input): ')
      ACCEPT 61,IVALUE
      IF (IVALUE.GT.0) MAPR=IVALUE
      IVALUE = 0
C.... GET SUBROUTINE TRACE FLAG
      TYPE 91
   91 FORMAT('$',10X,'Input the subroutine trace flag: (default 0) ')
      ACCEPT 61,IVALUE
      IF (IVALUE.GT.0) ISUB=IVALUE
      IVALUE = 0
C.... GET TIME INCREMENT
      TYPE 100
  100 FORMAT('$',10X,'Input the time increment: (default 0.10) ')
      ACCEPT 60,VALUE
      IF (VALUE.GT.0.) HH=VALUE
      VALUE  = 0.
C.... GET DIAGONAL MULTIPLYING FACTOR
      TYPE 110
  110 FORMAT('$',10X,'Input the diagonal mult factor: (default 1.0) ')
      ACCEPT 60,VALUE
      IF (VALUE.GT.0.) XLA=VALUE
C.... WRITE OUTPUT TO FILE
      WRITE(1,140)((BANNER(I,J),I=1,80),J=1,4)
      PRINT 141,((BANNER(I,J),I=1,80),J=1,4)
      WRITE(1,150)NN,ITR,MZ,MAPR,ISUB
      WRITE(1,160)HH,XLA
  140 FORMAT(80A1)
  141 FORMAT(5X,80A1)
  150 FORMAT(7I10)
  160 FORMAT(8F10.4)
C.... ENTER THE MASS AND GEOMETRIC DATA
C.... GET AIRPLANE WEIGHT
      TYPE 170
```

```
  170 FORMAT('$',10X,'Input the airplane weight (in lb): ')
      ACCEPT 60,WEIGHT
      AMSS   = WEIGHT/32.174
C.... GET AIRPLANE WING AREA
      TYPE 180
  180 FORMAT('$',10X,'Input the airplane wing area (in ft**2): ')
      ACCEPT 60,S
C.... GET AIRPLANE CBAR
      TYPE 190
  190 FORMAT('$',10X,'Input the airplane cbar (in ft): ')
      ACCEPT 60,CBAR
C.... GET AIRPLANE WING SPAN
      TYPE 195
  195 FORMAT('$',10X,'Input the airplane wing span (in ft): ')
      ACCEPT 60,SPAN
C.... GET ALTITUDE FOR RUN
      TYPE 200
  200 FORMAT('$',10X,'Input altitude of run (in ft): ')
      ACCEPT 60,H
C.... COMPUTE ATMOSPHERIC CONDITIONS FROM APPROXIMATE RELATIONS
      TA     = 518.7-H*0.00358
      IF (TA.LT.390.) TA=390.
      PA     = 2116.22*(1.-0.0000068784*H)**5.2532
      RHO    = PA/(1716.56*TA)
      AVEL   = 49.02*SQRT(TA)
      PRINT 205,PA,RHO,TA,AVEL
  205 FORMAT('  PA = ',F10.4,'  RHO = ',F10.6,
     .       '  TA = ',F10.4,'  ASONIC = ',F10.4)
C.... ENTER THE STEADY-STATE FLIGHT CONDITIONS
C.... GET THE VELOCITY
      TYPE 210
  210 FORMAT('$',10X,'Input the steady state velocity (in ft/sec): ')
      ACCEPT 60,U1
      CL1    =2.*WEIGHT/(RHO*U1*U1*S)
C.... GET THETA
      TYPE 220
  220 FORMAT('$',10X,'Input the steady state theta (in deg): ')
      ACCEPT 60,THA
      THA    = THA/DGR
C.... GET PHI
      TYPE 230
  230 FORMAT('$',10X,'Input the steady state phi (in deg): ')
      ACCEPT 60,PHI
      PHI    = PHI/DGR
C.... ASSUME STEADY STATE ALPHA IS STEADY STATE THETA
      ALP    = THA
      SINALP = SIN(ALP)
      COSALP = COS(ALP)
      SINTHA = SIN(THA)
      COSTHA = COS(THA)
      SINPHI = SIN(PHI)
      COSPHI = COS(PHI)
      TANTHA = SINTHA/COSTHA
C.... ENTER THE INERTIAL DATA
C.... GET IYYB
      TYPE 260
  260 FORMAT('$',10X,'Input IYYB (in slug*ft**2): ')
      ACCEPT 60,AIY
C.... GET IXXB
      TYPE 270
  270 FORMAT('$',10X,'Input IXXB (in slug*ft**2): ')
      ACCEPT 60,AIX
C.... GET IZZB
      TYPE 280
  280 FORMAT('$',10X,'Input IZZB (in slug*ft**2): ')
      ACCEPT 60,AIZ
C.... GET IXZB
      TYPE 290
  290 FORMAT('$',10X,'Input IXZB (in slug*ft**2): ')
```

```
              ACCEPT 60,AIXZ
       C....  SPLIT FOR CASES
              IF (CASE.EQ.'L') GO TO 300
              IF (CASE.EQ.'D') GO TO 500
       C....  IF THIS STATEMENT IS REACHED THEN AN ERROR MUST HAVE OCCURED
              STOP
       C....  LONGITUDINAL CASE
       C
          300 CONTINUE
              TYPE 310
          310 FORMAT(' ',10X,'Input CXU,  0 or 1: ')
              TYPE 311
          311 FORMAT('$',10X,'(1 if this is a variable and 0 otherwise) ')
              ACCEPT 60,A(2,2),AA(2,2)
       C
              TYPE 330
          330 FORMAT('$',10X,'Input CXA,  0 or 1: ')
              ACCEPT 60,A(2,3),AA(2,3)
       C
              TYPE 340
          340 FORMAT('$',10X,'Input CXDE, 0 or 1: ')
              ACCEPT 60,B(2,1),BB(2,1)
       C
              TYPE 350
          350 FORMAT('$',10X,'Input CZU,  0 or 1: ')
              ACCEPT 60,A(3,2),AA(3,2)
       C
              TYPE 360
          360 FORMAT('$',10X,'Input CZA,  0 or 1: ')
              ACCEPT 60,A(3,3),AA(3,3)
       C
              TYPE 370
          370 FORMAT('$',10X,'Input CZDE, 0 or 1: ')
              ACCEPT 60,B(3,1),BB(3,1)
       C
              TYPE 390
          390 FORMAT('$',10X,'Input CMO,  0 or 1: ')
              ACCEPT 60,A(1,1),AA(1,1)
       C
              TYPE 400
          400 FORMAT('$',10X,'Input CMU,  0 or 1: ')
              ACCEPT 60,A(1,2),AA(1,2)
       C
              TYPE 420
          420 FORMAT('$',10X,'Input CMA,  0 or 1: ')
              ACCEPT 60,A(1,3),AA(1,3)
       C
              TYPE 440
          440 FORMAT('$',10X,'Input CMDE, 0 or 1: ')
              ACCEPT 60,B(1,1),BB(1,1)
       C
       C....  DEFINE OTHER [A] MATRIX ELEMENTS
              A(1,4)  = 0.0
              A(2,1)  = 0.0
              A(2,4)  =-COSTHA*32.174
              A(3,1)  = 1.0
              A(3,4)  =-SINTHA*COSPHI*32.174/U1
              A(4,1)  = COSPHI
              A(4,2)  = 0.0
              A(4,3)  = 0.0
              A(4,4)  = 0.0
       C....  ALL OTHER [B] MATRIX ELEMENTS ARE ZERO
       C....  ALL OTHER [AA] MATRIX ELEMENTS ARE ZERO
       C....  DEFINE ADDITIONAL ELEMENTS OF THE [BB] MATRIX
              BB(1,3) = 1.0
              BB(2,3) = 1.0
              BB(3,3) = 1.0
              BB(4,3) = 1.0
       C....  SKIP LATERAL DIRECTIONAL INPUT CASE
```

```
            GO TO 700
C.... LATERAL DIRECTIONAL CASE
C
      500 CONTINUE
            TYPE 510
      510 FORMAT('$',10X,'Input CLP,  0 or 1: ')
            TYPE 511
      511 FORMAT('$',10X,' (1 if this is a variable and 0 otherwise) ')
            ACCEPT 60,A(1,1),AA(1,1)
C
            TYPE 520
      520 FORMAT('$',10X,'Input CLR,  0 or 1: ')
            ACCEPT 60,A(1,2),AA(1,2)
C
            TYPE 530
      530 FORMAT('$',10X,'Input CLB,  0 or 1: ')
            ACCEPT 60,A(1,3),AA(1,3)
C
            TYPE 540
      540 FORMAT('$',10X,'Input CLDA, 0 or 1: ')
            ACCEPT 60,B(1,1),BB(1,1)
C
            TYPE 550
      550 FORMAT('$',10X,'Input CLDR, 0 or 1: ')
            ACCEPT 60,B(1,2),BB(1,2)
C
            TYPE 560
      560 FORMAT('$',10X,'Input CNP,  0 or 1: ')
            ACCEPT 60,A(2,1),AA(2,1)
C
            TYPE 570
      570 FORMAT('$',10X,'Input CNR,  0 or 1: ')
            ACCEPT 60,A(2,2),AA(2,2)
C
            TYPE 580
      580 FORMAT('$',10X,'Input CNB,  0 or 1: ')
            ACCEPT 60,A(2,3),AA(2,3)
C
            TYPE 590
      590 FORMAT('$',10X,'Input CNDA, 0 or 1: ')
            ACCEPT 60,B(2,1),BB(2,1)
C
            TYPE 600
      600 FORMAT('$',10X,'Input CNDR, 0 or 1: ')
            ACCEPT 60,B(2,2),BB(2,2)
C
            TYPE 610
      610 FORMAT('$',10X,'Input CYB,  0 or 1: ')
            ACCEPT 60,A(3,3),AA(3,3)
C
            TYPE 620
      620 FORMAT('$',10X,'Input CYDA, 0 or 1: ')
            ACCEPT 60,B(3,1),BB(3,1)
C
            TYPE 630
      630 FORMAT('$',10X,'Input CYDR, 0 or 1: ')
            ACCEPT 60,B(3,2),BB(3,2)
C.... DEFINE OTHER [A] MATRIX ELEMENTS
            A(1,4)=0.0
            A(2,4)=0.0
            A(3,1)=SINALP
            A(3,2)=-COSALP
            A(3,4)=32.174*COSTHA*COSPHI/U1
            A(4,1)=1.0
            A(4,2)=COSPHI*TANTHA
            A(4,3)=0.0
            A(4,4)=0.0
C.... ALL OTHER [B] MATRIX ELEMENTS ARE ZERO
C.... ALL OTHER [AA] MATRIX ELEMENTS ARE ZERO
```

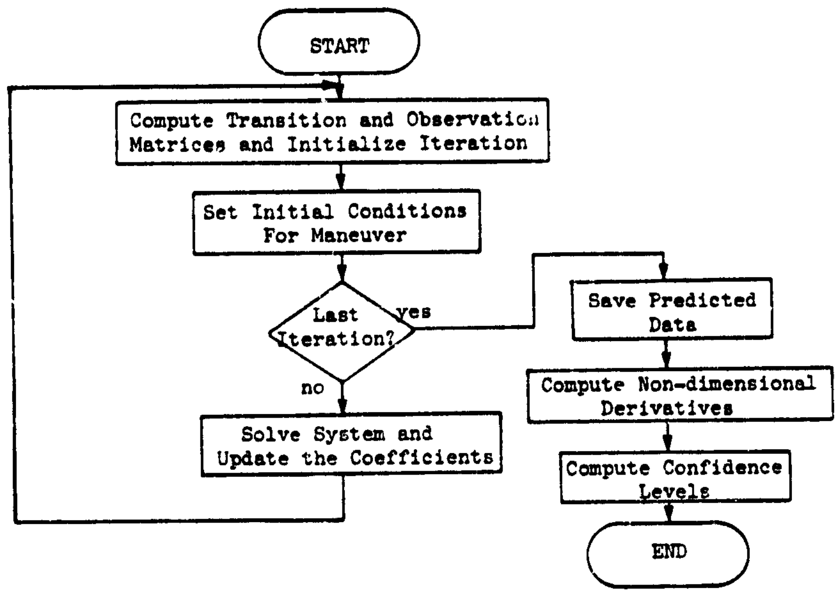
```
C.... DEFINE ADDITIONAL ELEMENTS OF THE [BB] MATRIX
      BB(1,3)=1.0
      BB(2,3)=1.0
      BB(3,3)=1.0
      BB(4,3)=1.0
C.... OUTPUT INFORMATION
  700 CONTINUE
      MU     = 3
      MZ     = 4
      MY     = 7
C.... ECHO DATA BACK
      PRINT 710,S,WEIGHT,SPAN,CBAR,U1,RHO,ALP,THA,PHI,AIY,AIX,
     .         AIXZ,AIZ,CL1
  710 FORMAT('  AIRPLANE INPUT DATA',/,
     .'   WING AREA            (IN FT**2)         = ',F12.4,/,
     .'   WEIGHT               (IN LBS)           = ',F12.4,/,
     .'   WING SPAN            (IN FT)            = ',F12.4,/,
     .'   CBAR                 (IN FT)            = ',F12.4,/,
     .'   AIRSPEED             (IN FT/SEC)        = ',F12.4,/,
     .'   DENSITY              (IN SLUG/FT**3)    = ',F12.4,/,
     .'   ALPHA1               (IN RAD)           = ',F12.4,/,
     .'   THETA1               (IN RAD)           = ',F12.4,/,
     .'   PHI1                 (IN RAD)           = ',F12.4,/,
     .'   IYY                  (IN SLUG*FT**2)    = ',F12.4,/,
     .'   IXX                  (IN SLUG*FT**2)    = ',F12.4,/,
     .'   IXZ                  (IN SLUG*FT**2)    = ',F12.4,/,
     .'   IZZ                  (IN SLUG*FT**2)    = ',F12.4,/,
     .'   CL1                                     = ',F12.4,//)
C.... SPLIT FOR CASES
      IF (CASE.EQ.'L') GO TO 750
      IF (CASE.EQ.'D') GO TO 850
      STOP
C.... LONGITUDINAL CASE
  750 CONTINUE
C.... GET THE WEIGHTING MATRIX DIAGONAL VALUES
      TYPE 770
  770 FORMAT('$','Input the weighting factor for q: ')
      ACCEPT 60,D1(1,1)
C
      TYPE 780
  780 FORMAT('$','Input the weighting factor for velocity: ')
      ACCEPT 60,D1(2,2)
C
      TYPE 790
  790 FORMAT('$','Input the weighting factor for alpha: ')
      ACCEPT 60,D1(3,3)
C
      TYPE 800
  800 FORMAT('$','Input the weighting factor for theta: ')
      ACCEPT 60,D1(4,4)
C
      TYPE 810
  810 FORMAT('$','Input the weighting factor for adot: ',
      ACCEPT 60,D1(5,5)
C
      TYPE 820
  820 FORMAT('$','Input the weighting factor for ax: ')
      ACCEPT 60,D1(6,6)
C
      TYPE 830
  830 FORMAT('$','Input the weighting factor for az: ')
      ACCEPT 60,D1(7,7)
C.... SKIP PAST LATERAL DIRECTIONAL CASE
      GO TO 950
C.... LATERAL DIRECTIONAL CASE
  850 CONTINUE
C.... GET THE WEIGHTING MATRIX DIAGONAL VALUES
      TYPE 870
  870 FORMAT('$','Input the weighting factor for p: ')
      ACCEPT 60,D1(1,1)
```

```
C
      TYPE 880
  880 FORMAT('$','Input the weighting factor for r: ')
      ACCEPT 60,D1(2,2)
C
      TYPE 890
  890 FORMAT('$','Input the weighting factor for beta: ')
      ACCEPT 60,D1(3,3)
C
      TYPE 900
  900 FORMAT('$','Input the weighting factor for phi: ')
       ACCEPT 60,D1(4,4)
C
      TYPE 910
  910 FORMAT('$','Input the weighting factor for pdot: ')
      ACCEPT 60,D1(5,5)
C
      TYPE 920
  920 FORMAT('$','Input the weighting factor for rdot: ')
      ACCEPT 60,D1(6,6)
C
      TYPE 930
  930 FORMAT('$','Input the weighting factor for ay: ')
      ACCEPT 60,D1(7,7)
  950 CONTINUE
C.... WRITE MATRICES TO FILE
  960 FORMAT(2I10)
 1160 FORMAT(8F12.6)
      WRITE(1,960)4,4
      DO 970 I=1,4
      WRITE(1,1160)(A(I,J),J=1,4)
  970 CONTINUE
      WRITE(1,960)4,MU
      DO 980 I=1,4
      WRITE(1,1160)(B(I,J),J=1,MU)
  980 CONTINUE
      WRITE(1,960)4,4
      DO 1000 I=1,4
      WRITE(1,1160)(AA(I,J),J=1,4)
 1000 CONTINUE
      WRITE(1,960)4,MU
      DO 1010 I=1,4
      WRITE(1,1160)(BB(I,J),J=1,MU)
 1010 CONTINUE
      WRITE(1,960)7,7
      DO 1040 I=1,7
      WRITE(1,1160)(D1(I,J),J=1,7)
 1040 CONTINUE
C.... WRITE TRANSDUCER POSITION DATA
C.... XAX,ZAX,ZAY,YAY,ZAY,XAZ,ZAZ,THAI
      WRITE(1,1200)-1.45,+1.54,-1.33,+0.27,+1.55,-1.23,+1.54,-0.079
 1200 FORMAT(8F10.4)
      WRITE(1,1200)(ZERO(I),I=1,4),(BIAS(J),J=1,4)
C.... WRITE OUT AIRPLANE DATA TO FILE
      WRITE(1,1160)S,CBAR,SPAN,WEIGHT,AIX,AIXZ,AIY,AIZ
      WRITE(1,1160)H,U1
      STOP
      END
```

A.7) MINC MMLE (NEWTON)

Description: The main program of the MMLE (NEWTON) routine acts as the executive, calling subroutines as needed. Initially it reads the input data (as output from SETUP) for the starting conditions of the case. If the case converges to a solution, the nondimensional derivatives and confidence levels are computed.

Flow chart:



ORIGINAL PAGE IS OF POOR QUALITY

Listing:

```
RT-11 LINK  V06.01C      Load Map        Fri 26-Feb-82 00:10:05
NEWTON.SAV       Title:  MAIN     Ident:  FORV02

Section   Addr    Size    Global   Value   Global   Value   Global   Value

 . ABS.   000000 001000   (RW,I,GBL,ABS,OVR)
                          $USRSW   000000  $RF2A1   00[illegible]00  .VIR     000000
                          $NLCHN   000006  $HRDWR   000006  $WASIZ   000152
                          $LRECL   000210
 $OHAND   001000 000106   (RW,I,GBL,REL,CON)
                          $OVRH    001002  O$READ   001024  O$DONE   001036
                          $ODF1    001102  $ODF2    001104
 $OTABL   001106 000130   (RW,D,GBL,REL,OVR)
 OTS$I    001236 024762   (RW,I,LCL,REL,CON)
                          $$OTSI   001236  AINT     001236  $INTR    001254
                          ALOG10   001340  ALOG     001344  $CVTFB   001720
                          $CVTFI   001720  $CVTCB   001734  $CVTCI   001734
                          $CVTDB   001734  $CVTDI   001734  CIC$     001746
                          CID$     001746  CLC$     001746  CLD$     001746
                          $DI      001746  CIF$     001756  CLF$     001756
                          $RI      001756  CIL$     002064  CLI$     002070
                          $CVTIF   002072  $CVTIC   002106  $CVTID   002106
                          CCI$     002120  CDI$     002120  $IC      002120
                          $ID      002120  CFI$     002134  $IR      002134
                          RCI$     002220  GCO$     003234  FCO$     003242
                          ECO$     003246  DCO$     003254  EXP      004176
                          ADF$IS   004536  $ADDF    004544  ADF$SS   004556
                          $ADR     004556  ADF$PS   004562  ADF$MS   004566
                          SUF$IS   004602  $SUBF    004610  SUF$SS   004622
                          $SBR     004622  SUF$PS   004626  SUF$MS   004632
                          DIF$PS   004646  DIF$MS   004652  DIF$IS   004664
                          $DIVF    004672  DIF$SS   004704  $DVR     004704
                          MUF$PS   004710  MUF$MS   004714  MUF$IS   004726
                          $MULF    004734  MUF$SS   004746  $MLR     004746
                          $OTI     005000  $$OTI    005002  $SETOP   005220
                          $$SET    006764  COS      007260  SIN      007314
                          $XFI     007652  XFI$     007664  $PWRI    007664
                          $INITI   010142  $CLOSE   010260  $CSTMI   011036
                          $CSTM    011042  OCI$     011154  ICI$     011162
                          $ECI     011176  OCO$     011356  ICO$     011364
                          $DUMPL   011562  $OPNER   011710  $CHKER   011746
                          $IOEXI   011772  $EOL     012040  EOL$     012042
                          $ERRTB   012156  $ERRS    012263  $FCALL   016024
                          $FCHNL   016042  $FIO     016704  $$FIO    016710
                          $GETFI   020054  $GETRE   020112  $TTYIN   020166
                          IFR$     020322  $IFR     020326  $$IFR    020332
                          IFR$$    020364  IFW$     020406  $IFW     020412
                          $$IFW    020416  IFW$$    020454  $$OPCL   020524
                          $$$ERR   020636  $$$DIS   020660  $OSTMI   020762
                          $OSTM    020766  $PUTRE   022116  RWD$     022424
                          $RWD     022430  IRR$     022536  $IRR     022542
                          IRW$     022566  $IRW     022572  $GETIN   023146
                          $SETIN   023204  DEF$     023312  $DEF     023316
                          $PUTBL   023412  $GETBL   023622  $EOFIL   024006
                          $EOF2    024022  SAVRG$   024042  THRD$    024220
                          SAVR4$   024222  $STPS    024306  STP$     024314
                          $STP     024314  FOO$     024320  $EXIT    024340
                          $OTIS    024464  $$OTIS   024466  IUR$     024606
                          $IUR     024612  IUW$     024756  $IUW     024762
                          TVL$     025576  $TVL     025576  TVF$     025604
                          $TVF     025604  TVD$     025612  $TVD     025612
                          TVQ$     025620  $TVQ     025620  TVP$     025626
                          $TVP     025626  TVI$     025634  $TVI     025634
                          $WAIT    025770  $XFF     026032  XFF$     026044
                          $VRINT   026206
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
OTS$P    026220 000034   (RW,D,GBL,REL,OVR)
SYS$I    026274 000000   (RW,I,LCL,REL,CON)
USER$I   026274 000000   (RW,I,LCL,REL,CON)
$CODE    026274 034312   (RW,I,LCL,REL,CON)
                         $$OTSC   026274  AGIRL    047340   AEAT     057320
                         AMAKE    060700  AZOT     061346   REDUCE   061676
OTS$O    062606 001036   (RW,I,LCL,REL,CON)
                         $$OTSO   062606  $OPEN    062606
SYS$O    063644 000000   (RW,I,LCL,REL,CON)
$DATAP   063644 006310   (RW,D,LCL,REL,CON)
OTS$D    072154 000062   (RW,D,LCL,REL,CON)
                         NHCLN$   072160
OTS$S    072236 000004   (RW,D,LCL,REL,CON)
                         $AOTS    072236
SYS$S    072242 000000   (RW,D,LCL,REL,CON)
$DATA    072242 003274   (RW,D,LCL,REL,CON)
USER$D   075536 000000   (RW,D,LCL,REL,CON)
.$$$$.   075536 000000   (RW,D,GBL,REL,OVR)
ALLDIM   075536 000004   (RW,D,GBL,REL,OVR)
SUBWRT   075542 000002   (RW,D,GBL,REL,OVR)
MATRIX   075544 001210   (RW,D,GBL,REL,OVR)
DIMFAC   076754 000240   (RW,D,GBL,REL,OVR)
VECTOR   077214 000516   (RW,D,GBL,REL,OVR)
INITIC   077732 000010   (RW,D,GBL,REL,OVR)
ARRAYS   077742 010444   (RW,D,GBL,REL,OVR)
INSTRM   110406 000054   (RW,D,GBL,REL,OVR)
GEOMTR   110462 000040   (RW,D,GBL,REL,OVR)
NUMBER   110522 000070   (RW,D,GBL,REL,OVR)
COMMND   110612 000006   (RW,D,GBL,REL,OVR)
Segment size = 110620 = 18632. words

Overlay region  000001  Segment 000001
OTS$I    110622 000000   (RW,I,LCL,REL,CON)
SYS$I    110622 000000   (RW,I,LCL,REL,CON)
USER$I   110622 000000   (RW,I,LCL,REL,CON)
$CODE    110622 001620   (RW,I,LCL,REL,CON)
                         AADD  @ 110622  AMULT @ 111362
OTS$O    112442 000000   (RW,I,LCL,REL,CON)
SYS$O    112442 000000   (RW,I,LCL,REL,CON)
$DATAP   112442 000076   (RW,D,LCL,REL,CON)
OTS$D    112540 000000   (RW,D,LCL,REL,CON)
OTS$S    112540 000000   (RW,D,LCL,REL,CON)
SYS$S    112540 000000   (RW,D,LCL,REL,CON)
$DATA    112540 000142   (RW,D,LCL,REL,CON)
USER$D   112702 000000   (RW,D,LCL,REL,CON)
Segment size = 002060 = 536.   words

Overlay region  000001  Segment 000002
OTS$I    110622 000000   (RW,I,LCL,REL,CON)
SYS$I    110622 000000   (RW,I,LCL,REL,CON)
USER$I   110622 000000   (RW,I,LCL,REL,CON)
$CODE    110622 001772   (RW,I,LCL,REL,CON)
                         ALOAD @ 110622  ALOAD1   111020   ASPIT @ 111416
                         ASPIT1@ 112024
OTS$O    112614 000000   (RW,I,LCL,REL,CON)
SYS$O    112614 000000   (RW,I,LCL,REL,CON)
$DATAP   112614 000336   (RW,D,LCL,REL,CON)
OTS$D    113152 000000   (RW,D,LCL,REL,CON)
OTS$S    113152 000000   (RW,D,LCL,REL,CON)
SYS$S    113152 000000   (RW,D,LCL,REL,CON)
$DATA    113152 000130   (RW,D,LCL,REL,CON)
USER$D   113302 000000   (RW,D,LCL,REL,CON)
Segment size = 002460 = 664.   words

Overlay region  000002  Segment 000003
OTS$I    113304 000000   (RW,I,LCL,REL,CON)
SYS$I    113304 000000   (RW,I,LCL,REL,CON)
USER$I   113304 000000   (RW,I,LCL,REL,CON)
$CODE    113304 002116   (RW,I,LCL,REL,CON)
                         SOLVE @ 113304  INV    @ 114424
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
 OTS$D    115422 000000   (RW,I,LCL,REL,CON)
 SYS$D    115422 000000   (RW,I,LCL,REL,CON)
 $DATAP   115422 000160   (RW,D,LCL,REL,CON)
 OTS$D    115602 000000   (RW,D,LCL,REL,CON)
 OTS$S    115602 000000   (RW,D,LCL,REL,CON)
 SYS$S    115602 000000   (RW,D,LCL,REL,CON)
 $DATA    115602 000074   (RW,D,LCL,REL,CON)
 USER$D   115676 000000   (RW,D,LCL,REL,CON)
 Segment size = 002372 = 637.   words

Overlay region  000002  Segment 000004
 OTS$I    113304 000146   (RW,I,LCL,REL,CON)
                        SORT     113304  ABS     113434
 SYS$I    113452 000000   (RW,I,LCL,REL,CON)
 USER$I   113452 000000   (RW,I,LCL,REL,CON)
 $CODE    113452 003220   (RW,I,LCL,REL,CON)
                        CRAMER@ 113452  DIAGIN  116104
 OTS$D    116672 000000   (RW,I,LCL,REL,CON)
 SYS$D    116672 000000   (RW,I,LCL,REL,CON)
 $DATAP   116672 001032   (RW,D,LCL,REL,CON)
 OTS$D    117724 000000   (RW,D,LCL,REL,CON)
 OTS$S    117724 000000   (RW,D,LCL,REL,CON)
 SYS$S    117724 000000   (RW,D,LCL,REL,CON)
 $DATA    117724 000646   (RW,D,LCL,REL,CON)
 USER$D   120572 000000   (RW,D,LCL,REL,CON)
 Segment size = 005266 = 1371.   words

Transfer address = 026274, High limit = 120570 = 20668. words
```

```
FORTRAN IV      V02.5-2    Mon 15-Feb-82 00:40:25               PAGE 001

0001          PROGRAM MAIN
        C
        C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
        C     *                                                           *
        C     *      mini-MMLE - KANSAS UNIVERSITY FLIGHT RESEARCH LAB    *
        C     *      ===============================================      *
        C     *                                                           *
        C     * NEWTON-RAPHSON METHOD FOR OBTAINING STABILITY DERIVATIVES *
        C     *                                                           *
        C     * MEASURED STATES:                                          *
        C     *       LONG: Q, V, ALP, THA, QDO       , AND A-Z           *
        C     *       LATR: P, R, BTA, PHI, PDOT,    JT, AND A-Y          *
        C     *  -   -   -   -   -   -   -   -   -   -   -   -   -   -    *
        C     *                                                           *
        C     *      MAIN PROGRAM OF THE MAXIMUM LIKELIHOOD ESTIMATOR     *
        C     *      TECHNIQUE, (MMLE). THIS PROGRAM IS DERIVED FROM      *
        C     *      THE 'BONES' PROGRAM THAT WAS ORIGINALLY DEVELOPED    *
        C     *      BY NASA.                                             *
        C     *                     ROBERT CLARKE    2-JUN-81             *
        C     *         MODIFIED    ROBERT CLARKE   12-AUG-81             *
        C     *         MODIFIED    ROBERT CLARKE   15-FEB-82             *
        C     *                                                           *
        C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
        C
        C     - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
        C
0002          COMMON /ALLDIM/ MAX,MAT
0003          COMMON /SUBWRT/ ISUB
0004          COMMON /MATRIX/ A,B,AA,BB,AP,BP,D1
0005          COMMON /DIMFAC/ ADIM,BDIM
```

```
      COMMON /VECTOR/ U,Z,XT1,XT2,XT4,XT5,ZERO,BIAS,IBIAS
      COMMON /INITIC/ ALT1,VEL1
      COMMON /ARRAYS/ D2,APHI,PHI1,DUM,BJI,XJI,SUM,APR
      COMMON /INSTRM/ XAX,ZAX,XAY,YAY,ZAY,XAZ,ZAZ,THAI,COSTHI,
     1                SINTHI,GRV
      COMMON /GEOMTR/ S,CBAR,SPAN,GWGHT,AIXXB,AIXZB,AIYYB,AIZZB
      COMMON /NUMBER/ JKMM,JKM,JKMM1,HH,TIME,TT,NN,NNM1,L,
     1                LL,ITR,MU,MZ,MX,MXP1,I,IA,J,JK,K,KJ,NI,
     2                ITHIN,ERRSUM
      COMMON /COMMND/ LONG,LATR,EXTRA,BYPASS,MAPR
      DIMENSION AP(8,4),BP(8,3),XT4(4),DUM(25,4),XT2(7),ZERO(5)
      DIMENSION XT5(25),APR(25),ADIM(5,4),BDIM(5,4),BIAS(5)
      DIMENSION Z(7,3),U(3,3),D2(7),APHI(5,4),XT1(7),PHI1(5,4),
     1          D1(8,7),A(5,4),B(5,4),BJI(25,4),XJI(25,8),SUM(25,25)
      LOGICAL*1 LONG,LATR,EXTRA,BYPASS,AA(5,4),BB(5,4)
      DIMENSION PB(25),XT3(7),R(5,4),RI(5,4),AAA(5,4),
     1          BBB(5,4),D54(5,4),DD4(5,4)
      BYTE INAME(15),BANNER(80,4),ANS
      DATA LONG,LATR,EXTRA,BYPASS /4*.FALSE./
      DATA AA,BB /40*.FALSE./
      DATA ADIM,BDIM /40*1.0/
      DATA INAME(15),ANS /0,'N'/
      DATA GRV /32.174/
C
C     - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
C
C****************** INPUT USER DEFINED SETUP DATA
 2000 CONTINUE
      TYPE 3000
 3000 FORMAT(/////,' ',10X,'INDICATE TYPE OF RUN!',
     1          /,' ',10X,'IF LONGITUDINAL         TYPE "L"',
     2          /,' ',10X,'IF LATERAL-DIRECTIONAL TYPE "D"',
     3          /,'$',10X,'SELECT RUN! ')
      ACCEPT 100, ANS
  100 FORMAT(A1)
      IF (ANS.EQ.'L') LONG = .TRUE.
      IF (ANS.EQ.'D') LATR = .TRUE.
      IF (.NOT.(LONG.OR.LATR)) TYPE 3001
 3001 FORMAT(10X,'WRONG ANSWER')
      IF (.NOT.(LONG.OR.LATR)) GO TO 2000
C****************** SET DEFAULTS
      NI          = 25
      MZ          = 7
      TIME        = 0.
C****************** APRIORI WEIGHTING FACTOR
      FACT        = 1.
C****************** ATTACH INPUT DATA FILE
      TYPE 3002
 3002 FORMAT(/,'$',10X,'ENTER FILE NAME FOR SETUP DATA: ')
      ACCEPT 101, (INAME(IABC),IABC=1,14)
  101 FORMAT(14A1)
      OPEN (UNIT=2,NAME=INAME,TYPE='OLD',ACCESS='SEQUENTIAL',
     1      READONLY,FORM='FORMATTED',RECORDSIZE=132)
C****************** CLEAR OUT OLD FILE NAME
      DO 1000 IABC=1,14
      INAME(IABC) = ' '
 1000 CONTINUE
      READ(2,102) ((BANNER(I,J),I=1,80),J=1,4)
  102 FORMAT(80A1)
      READ(2,103) NN,ITR,MZ,MAPR,ISUB
  103 FORMAT(7I10)
      READ(2,104) HH,XLA
  104 FORMAT(8F10.4)
C****************** CHECK ON NON-DIMENSIONAL DERIVATIVES
      TYPE 3003
 3003 FORMAT('$',10X,'DIMENSIONAL INPUT DERIVATIVES? (Y OR N): ')
      ACCEPT 100, ANS
      IF (ANS.EQ.'Y') BYPASS = .TRUE.
C****************** CHECK ON EXTRA DATA
```

```
      TYPE 3004
 3004 FORMAT('$',10X,'EXTRA DATA?  (Y OR N): ')
      ACCEPT 100, ANS
      IF (ANS.EQ.'Y') EXTRA = .TRUE.
C****************** LOAD MATRICES
      MAX        = 5
      CALL ALOAD(4,A,B,AAA,BBB)
      IEND       = AAA(MAX,1)
      DO 1001 I=1,IEND
      JEND       = AAA(MAX,2)
      DO 1002 J=1,JEND
      IF (AAA(I,J).EQ.1.) AA(I,J) = .TRUE.
 1002 CONTINUE
      JEND       = BBB(MAX,2)
      DO 1003 J=1,JEND
      IF (BBB(I,J).EQ.1.) BB(I,J) = .TRUE.
 1003 CONTINUE
 1001 CONTINUE
      MAX        = 8
      CALL ALOAD(1,D1,D1,D1,D1)
C******************* SET DIMENSIONS AND INITIALIZE [AP] AND [BP] MATRICES
      AP(8,1)    = 7
      BP(8,1)    = 7
      AP(8,2)    = 4
      BP(8,2)    = 3
      CALL AZOT(AP)
      CALL AZOT(BP)
C******************* IF TEST THEN OUTPUT MATRICES
      IF (ISUB.GE.2) CALL ASPIT(D1)
      MAX        = 5
      IF (ISUB.GE.2) CALL ASPIT(AAA)
      IF (ISUB.GE.2) CALL ASPIT(BBB)
      NNM1       = NN-1
      MU         = B(MAX,2)+.01
      MX         = A(MAX,2)+.01
C******************* READ INSTRUMENTATION DATA
      READ(2,104) XAX,ZAX,XAY,YAY,ZAY,XAZ,ZAZ,THAI
      SINTHI     = SIN(THAI)
      COSTHI     = COS(THAI)
C******************* READ IN ZEROS AND BIASES
      READ (2,104) (ZERO(I),I=1,MX),(BIAS(IA),IA=1,MX)
C******************* READ AIRPLANE GEOMETRIC DATA AND INITIAL CONDITIONS
      READ(2,105) S,CBAR,SPAN,GWGHT,AIXXB,AIXZB,AIYYB,AIZZB
  105 FORMAT(8F12.4)
      READ(2,105) ALT1,VEL1
C******************* FORM [R] MATRIX
      MAX        = 5
      R(5,1)     = 4.
      R(5,2)     = 4.
      CALL AZOT(R)
      DO 1004 I=1,4
      R(I,I)     = 1.
 1004 CONTINUE
      IF (LATR) R(1,2) =-AIXZB/AIXXB
      IF (LATR) R(2,1) =-AIXZB/AIZZB
C******************* COPY [R] INTO [RI]
      CALL AMAKE(RI,R)
C******************* INVERT [R] IF NECESSARY
      IF (LATR) CALL INV(RI,MAX)
C******************* FORM [AP] AND [BP] MATRICES
      DO 1005 I=1,6
      DO 1006 J=1,4
      AP(I,J)     = 1.
 1006 CONTINUE
      DO 1007 J=1,3
      BP(I,J)     = 1.
 1007 CONTINUE
 1005 CONTINUE
      IF (LONG) AP(6,1) = 0.
```

```
      IF (LONG) AP(6,2) = 1./GRV
      IF (LONG) AP(6,3) = 0.
      IF (LONG) AP(6,4) = 0.
      AP(7,3)      = VEL1/GRV
      DO 1008 I=1,3
      IF (LONG) BP(6,I) = 1./GRV
      BP(7,I)      = VEL1/GRV
 1008 CONTINUE
C****************** FORM DIMENSIONALIZING MATRICES FOR [A] AND [B]
      TEMPA        = 518.7-ALT1*0.00358
      IF (TEMPA.LT.390.) TEMPA = 390.
      PRESA        = 2116.22*(1.-0.0000068784*ALT1)**5.2532
      RHO          = PRESA/(1716.56*TEMPA)
      ADIM(5,1)    = A(5,1)
      ADIM(5,2)    = A(5,2)
      BDIM(5,1)    = B(5,1)
      BDIM(5,2)    = B(5,2)
      AMSS         = GWGHT/GRV
      QBAR         = .5*RHO*VEL1*VEL1
C****************** IF GWGHT = 0.0 THEN OUTPUT DIMENSIONAL DERIVATIVES
      IF (GWGHT.EQ.0.) GO TO 2002
      IF (LATR) GO TO 2001
C****************** LONGITUDINAL CASE
      QS           = QBAR*S
      QSC          = QBAR*S*CBAR
      QSCC         = QBAR*S*CBAR*CBAR
      ADIM(1,1)    = (QSCC)/(2.*VEL1*AIYYB)
      ADIM(1,2)    = (QSC)/(VEL1*AIYYB)
      ADIM(1,3)    = (QSC)/(AIYYB)
      BDIM(1,1)    = (QSC)/(AIYYB)
      BDIM(1,2)    = (QSC)/(AIYYB)
      BDIM(1,3)    = (QSC)/(AIYYB)
      ADIM(2,2)    = (QS)/(AMSS*VEL1)
      ADIM(2,3)    = (QS)/(AMSS)
      BDIM(2,1)    = (QS)/(AMSS)
      BDIM(2,2)    = (QS)/(AMSS)
      BDIM(2,3)    = (QS)/(AMSS)
      ADIM(3,2)    = (QS)/(AMSS*VEL1**2)
      ADIM(3,3)    = (QS)/(AMSS*VEL1)
      BDIM(3,1)    = (QS)/(AMSS*VEL1)
      BDIM(3,2)    = (QS)/(AMSS*VEL1)
      BDIM(3,3)    = (QS)/(AMSS*VEL1)
      GO TO 2002
 2001 CONTINUE
C****************** LATERAL-DIRECTIONAL CASE
      QS           = QBAR*S
      QSB          = QBAR*S*SPAN
      QSBB         = QBAR*S*SPAN*SPAN
      ADIM(1,1)    = (QSBB)/(2.*VEL1*AIXXB)
      ADIM(1,2)    = (QSBB)/(2.*VEL1*AIXXB)
      ADIM(1,3)    = (QSB)/(AIXXB)
      BDIM(1,1)    = (QSB)/(AIXXB)
      BDIM(1,2)    = (QSB)/(AIXXB)
      BDIM(1,3)    = (QSB)/(AIXXB)
      ADIM(2,1)    = (QSBB)/(2.*VEL1*AIZZB)
      ADIM(2,2)    = (QSBB)/(2.*VEL1*AIZZB)
      ADIM(2,3)    = (QSB)/(AIZZB)
      BDIM(2,1)    = (QSB)/(AIZZB)
      BDIM(2,2)    = (QSB)/(AIZZB)
      BDIM(2,3)    = (QSB)/(AIZZB)
      ADIM(3,3)    = (QS)/(AMSS*VEL1)
      BDIM(3,1)    = (QS)/(AMSS*VEL1)
      BDIM(3,2)    = (QS)/(AMSS*VEL1)
      BDIM(3,3)    = (QS)/(AMSS*VEL1)
 2002 CONTINUE
      MXP1         = MX+1
      YY           = 0.
      XX           = 1.
      DO 1009 I=1,MX
```

```
          XT4(I)         = 0.
          XT3(I)         = 0.
          XY             = XX+ZERO(I)+BIAS(I)
          DO 1010 J=1,MU
          YY             = YY+AAA(I,J)+BBB(I,J)
    1010  XX             = XX+AAA(I,J)+BBB(I,J)
          YY             = YY+AAA(I,MX)
    1009  XX             = XX+AAA(I,MX)
          JKMM           = YY+.01
          JKM            = XX+.01
          JKMM1          = JKM-1
C****************** INITIALIZE MATRICES
          SUM(NI,1)      = JKM
          SUM(NI,2)      = JKM
          MAX            = NI
          CALL AZOT(SUM)
C****************** SELECT APRIORI OPTION THRU MAPR
          IF (MAPR) 2003,2004,2005
    2005  DO 1011 IB=1,JKM
          READ(2,106) (SUM(IB,IA),IA=1,JKM)
     106  FORMAT(6E12.4)
          IF (ISUB.GE.2) PRINT 107, (SUM(IB,IA),IA=1,JKM)
     107  FORMAT(' ',6E12.4)
          DO 1011 IA=1,JKM
    1011  SUM(IB,IA)=SUM(IB,IA)*FACT
          APR(IB)        = SUM(IB,IB)
          GO TO 2004
    2003  READ(2,106) (APR(IA),IA=1,JKMM1)
          DO 1012 IA=1,JKM
    1012  APR(IA)        = APR(IA)*FACT
          IF (ISUB.GE.2) PRINT 107, (APR(AI),IA=1,JKMM1)
    2004  CONTINUE
C **************** CLOSE INPUT DATA FILE
          CLOSE(UNIT=2)
C***************** ENTER NAME OF DATA FILE WITH FLIGHT TEST DATA
          TYPE 3006
    3006  FORMAT(/,'$',10X,'ENTER FILE NAME CONTAINING MEASURED DATA: ')
          ACCEPT 101,(INAME(IABC),IABC=1,14)
          OPEN(UNIT=4,NAME=INAME,TYPE='OLD',ACCESS='SEQUENTIAL',
         1     FORM='UNFORMATTED',READONLY,BUFFERCOUNT=2)
          TYPE 3007
    3007  FORMAT(/,'$',10X,'ENTER FILE NAME TO HOLD PREDICTED DATA: ')
          ACCEPT 101,(INAME(IABC),IABC=1,14)
          OPEN(UNIT=3,NAME=INAME,TYPE='NEW',ACCESS='SEQUENTIAL',
         1     FORM='UNFORMATTED',DISPOSE='SAVE',BUFFERCOUNT=2)
C****************** PRINT OUT INPUT DATA
          PRINT 3008
    3008  FORMAT('- - - - - - - - - - - - - - - - - - - - - - - - - - -',
         1     ' - - - - - - - - - - - - - - - - - - - - - - - - - - - -')
          DO 1013 J=1,4
          PRINT 108, (BANNER(I,J),I=1,80)
     108  FORMAT(10X,80A1)
    1013  CONTINUE
          PRINT 3008
          PRINT 3009
    3009  FORMAT(24X,'. . . . . . .  INITIAL CONDITIONS  . . . . . . . '/)
          PRINT 3010, NN,ITR,ISUB,NZ,HH,XLA
    3010  FORMAT(8X,'NUMBER OF DATA POINTS          :      ',I3,
         1       8X,'MAXIMUM NUMBER OF ITERATIONS :      ',I3,/,
         2       8X,'SUBROUTINE TEST FLAG          :      ',I3,
         3       8X,'NUMBER OF STATES              :      ',I3,/
         4       8X,'DATA SAMPLING TIME  INTERVAL :',F8.4,
         4       8X,'DIAGONAL MULTIPLYING FACTOR  :',F8.4,/)
          PRINT 3008
          PRINT 3700, QBAR,VEL1,S,SPAN,CBAR,AIXXB,AIYYB,AIZZB,AIXZB,GWGHT
    3700  FORMAT(/,10X,'FLIGHT CONDITION AND VEHICLE CHARACTERISTICS',
         1       /,15X,'DYNAMIC PRESSURE = ',F10.2,
         2         10X,'VELOCITY         = ',F10.2,
         3       /,15X,'WING AREA        = ',F10.2,
         4         10X,'WING SPAN        = ',F10.2,
```

```
     5        /,15X,'WING MAC          = ',F10.2,
     6          10X,'IXXB              = ',F10.1,
     7        /,15X,'IYYB              = ',F10.1,
     8          10X,'IZZB              = ',F10.1,
     9        /,15X,'IXZB              = ',F10.1,
     1          10X,'WEIGHT            = ',F10.2)
      PRINT 3701,XAX,XAY,XAZ,YAY,ZAX,ZAY,ZAZ,THAI
 3701 FORMAT(  15X,'INSTRUMENT OFFSETS FROM CG',
     1        /,20X,'X-DIRECTION OFFSETS (+ = INSTR FORWARD OF CG)',
     2        /,25X,'A-X   ',F7.3,3X,'A-Y   ',F7.3,3X,'A-Z   ',F7.3,
     3        /,20X,'Y-DIRECTION OFFSETS (+ = INSTR RIGHT OF CG)',
     4        /,25X,'      ',10X,'A-Y   ',F7.3,
     5        /,20X,'Z-DIRECTION OFFSETS (+ = INSTR BELOW CG)',
     6        /,25X,'A-X   ',F7.3,3X,'A-Y   ',F7.3,3X,'A-Z   ',F7.3,
     7        /,20X,'PITCH ANGULAR OFFSET FROM BODY AXES ',
     8              '(+ = PITCH UP)'
     9       ,/,25X,'THETA (MEASURED IN RADIAN UNITS)   ',F7.3)
      IF (LONG) PRINT 3011
 3011 FORMAT(/,10X,'ZERO AND BIAS CONTROL',/
     1          10X,'PITCH RATE VELOCITY   ALPHA     THETA    ',
     2              'PITCH ACCL    A-X      A-Z')
      IF (LATR) PRINT 3012
 3012 FORMAT(/,10X,'ZERO AND BIAS CONTROL',/
     1          10X,'ROLL RATE  YAW RATE    BETA      PHI     ',
     2              'ROLL ACCL  YAW ACCEL   A-Y')
      PRINT 109, (ZERO(I),I=1,MX),(BIAS(IA),IA=1,MX-1)
  109 FORMAT(10X,8F10.5)
      PRINT 110,(D1(IBCD,IBCD),IBCD=1,7)
  110 FORMAT(/,10X,'DIAGONAL ELEMENTS OF THE WEIGHTING MATRIX D1:',/
     1            5X,7F13.3,/)
      PRINT 3008
      MAX          = 5
C****************** PRINT OUT THE INPUT [R], [A], AND [B] MATRICES
      PRINT 3013
 3013 FORMAT(10X,'INITIAL INPUT MATRICES [R], [A], AND [B].',/,
     1        10X,'A STAR (*) FOLLOWING THE VALUE OF A MATRIX',
     2            ' ELEMENT INDICATES THAT',/,
     3        10X,'THE RESPECTIVE DERIVATIVE IS NOT ESTIMATED BY'
     4           ,' THE MMLE METHOD.')
      IF (.NOT.BYPASS) PRINT 3014
 3014 FORMAT(10X,'(DERIVATIVES ARE NON-DIMENSIONAL)')
      IF (BYPASS) PRINT 3015
 3015 FORMAT(10X,'(DERVATIVES ARE DIMENSIONAL)')
      PRINT 9000
 9000 FORMAT(/,10X,'MATRIX [R]')
      CALL ASPIT(R)
      PRINT 3016
 3016 FORMAT(/,10X,'STABILITY MATRIX [A]')
      CALL ASPIT1(A,AA)
      PRINT 3017
 3017 FORMAT(/,10X,'CONTROL MATRIX [B]')
      CALL ASPIT1(B,BB)
      PRINT 3008
C****************** MULTIPLY [A] AND [B] BY [RI]
      MAX          = 5
      MAT          = 5
      CALL AMULT(RI,A,A,D54)
      CALL AMULT(RI,B,B,D54)
C****************** IF NOT BYPASS THEN FORM DIMENSIONAL DERIVATIVES
      IF (BYPASS) GO TO 2006
      DO 1014 IABC=1,MX
      DO 1015 IDEF=1,MX
      A(IABC,IDEF)= A(IABC,IDEF)*ADIM(IABC,IDEF)
 1015 CONTINUE
      DO 1016 IDEF=1,MU
      B(IABC,IDEF)= B(IABC,IDEF)*BDIM(IABC,IDEF)
 1016 CONTINUE
 1014 CONTINUE
 2006 CONTINUE
C****************** STARTING ITERATION LOOP
      REWIND 4
```

```
      TT          = TIME-HH
      DO 1017 LM=1,NN
      TT          = TT+HH
C****************** READ IN MEASURED RESPONSES FROM DATA FILE
C****************** READ IN LONGITUDINAL DATA
      IF (LONG) READ(4) Z(4,1),Z(1,1),Z(7,1),Z(6,1),U(1,1),AM6,AM7,AM8,
     1                 AM9,AM10,AM11
      IF (LONG.AND.EXTRA) READ(4) Z(2,1),Z(3,1),Z(5,1),AM15,AM16,AM17
      U(2,1)      = 0.
C****************** READ IN LATERAL DIRECTIONAL DATA
      IF (LATR) READ(4) AM1,AM2,AM3,AM4,AM5,Z(4,1),Z(1,1),Z(7,1),Z(2,1),
     1                 U(1,1),U(2,1)
      IF (LATR.AND.EXTRA) READ(4) AM12,AM13,AM14,Z(3,1),Z(5,1),Z(6,1)
      IF (ISUB.GE.6) PRINT 3050,(Z(IA,1),IA=1,7),U(1,1),U(2,1),TT
 3050 FORMAT(5X,10(1PE12.4))
 1017 CONTINUE
      REWIND 4
      DO 1018 IA=1,NI
      XT5(IA)     = 0.
      PB(IA)      = 0.
 1018 CONTINUE
      IZE         = 1
      DO 1019 IA=1,MX
      IF (ZERO(IA)) 2007,2008,2007
 2007 IZE         = IZE+1
 2008 CONTINUE
 1019 CONTINUE
C****************** MAIN MMLE LOOP FOR INT ITERATIONS
      DO 1020 LL=1,ITR
C****************** REWIND TAPES FOR EACH ITERATION
      REWIND 4
      REWIND 3
      PRINT 3018, LL
 3018 FORMAT(//,10X,'ITERATION: ',I3)
      MAX         = 5
C****************** CALL SPECIAL MATRIX OUTPUT ROUTINE
      PRINT 3019
 3019 FORMAT(/,10X,'ESTIMATES OF THE STATE MATRICES')
      PRINT 3020
 3020 FORMAT(/,10X,'STABILITY MATRIX [RI]*[A]')
      CALL ASPIT1(A,AA)
      PRINT 3021
 3021 FORMAT(/,10X,'CONTROL MATRIX [RI]*[B]')
      CALL ASPIT1(B,BB)
      MAX         = 5
      MAT         = 5
      CALL AEAT (A,HH,PHI1,APHI,D54,DD4)
      U(3,1)      = 1.
      U(3,2)      = 1.
      U(3,3)      = 1.
      XJI(NI,1)   = JKM
      XJI(NI,2)   = MX
      BJI(NI,1)   = JKM
      BJI(NI,2)   = MX
      SUM(NI,1)   = JKM
      SUM(NI,2)   = JKM
C****************** INITIALIZE AND READ DATA FROM TAPE
      DO 1021 IJK=1,JKM
      DO 1021 JKL=1,IJK
 1021 SUM(IJK,JKL)= 0.
      MAX         = NI
      CALL AZOT(XJI)
C****************** READ IN THE FIRST TWO SERIES OF
C****************** MEASURED DATA FROM THE DATA DISK FILE
C****************** READ IN FIRST TIME POINT OF LONGITUDINAL DATA
      IF (LONG) READ(4) XT1(4),XT1(1),XT1(7),XT1(6),U(1,1),
     1                 AM6,AM7,AM8,AM9,AM10,AM11
      IF (LONG.AND.EXTRA) READ(4) XT1(2),XT1(3),XT1(5),
     1                               AM15,AM16,AM17
      U(2,1)      = 0.
```

```
C****************** READ IN FIRST TIME POINT LATERAL DIRECTIONAL DATA
      IF (LATR) READ(4) AM1,AM2,AM3,AM4,AM5,XT1(4),XT1(1),XT1(7),
     1                  XT1(2),U(1,1),U(2,1)
      IF (LATR.AND.EXTRA) READ(4) AM12,AM13,AM14,XT1(3),XT1(5),
     1                            XT1(6)
C
C******************* TRANSLATE FROM INSTRUMENT AXES AND LOCATION
C******************* FOR CORRECT INITIAL CONDITIONS FOR MMLE PREDICTIONS
C****************** LONGITUDINAL CASE
      IF (LONG) AXI         = COSTHI*XT1(6)+SINTHI*XT1(7)
      IF (LONG) AZI         =-SINTHI*XT1(6)+COSTHI*XT1(7)
      IF (LONG) XT1(4)      = XT1(4)-THAI
      IF (LONG) XT1(6)      = AXI
      IF (LONG) XT1(7)      = AZI
C***************** LATERAL-DIRECTIONAL CASE
      IF (LATR) PI1         = COSTHI*XT1(1)+SINTHI*XT1(2)
      IF (LATR) RI1         =-SINTHI*XT1(1)+COSTHI*XT1(2)
      IF (LATR) XT1(1)      = PI1
      IF (LATR) XT1(2)      = RI1
C
C****************** READ IN SECOND TIME POINT OF LONGITUDINAL DATA
      IF (LONG) READ(4) XT2(4),XT2(1),XT2(7),XT2(6),U(1,2),
     1                  AM6,AM7,AM8,AM9,AM10,AM11
      IF (LONG.AND.EXTRA) READ(4) XT2(2),XT2(3),XT2(5),
     1                            AM15,AM16,AM17
      U(2,2)      = 0.
C****************** READ IN SECOND TIME POINT OF LATERAL-DIRECTIONAL DATA
      IF (LATR) READ(4) AM1,AM2,AM3,AM4,AM5,XT2(4),XT2(1),XT2(7),
     1                  XT2(2),U(1,2),U(2,2)
      IF (LATR.AND.EXTRA) READ(4) AM12,AM13,AM14,XT2(3),XT2(5),XT2(6)
C
C******************* TRANSLATE FROM INSTRUMENT AXES AND LOCATION
C******************* FOR CORRECT INITIAL CONDITIONS FOR MMLE PREDICTIONS
C****************** LONGITUDINAL CASE
      IF (LONG) AXI         = COSTHI*XT2(6)+SINTHI*XT2(7)
      IF (LONG) AZI         =-SINTHI*XT2(6)+COSTHI*XT2(7)
      IF (LONG) XT2(4)      = XT2(4)-THAI
      IF (LONG) XT2(6)      = AXI
      IF (LONG) XT2(7)      = AZI
C***************** LATERAL-DIRECTIONAL CASE
      IF (LATR) PI2         = COSTHI*XT2(1)+SINTHI*XT2(2)
      IF (LATR) RI2         =-SINTHI*XT2(1)+COSTHI*XT2(2)
      IF (LATR) XT2(1)      = PI2
      IF (LATR) XT2(2)      = RI2
C
C******************* DELETE UNWEIGHTED DATA  FROM TIME HISTORY
      DO 1022 IABC =1,7
      IF (D1(IABC,IABC).NE.0) GO TO 2009
      XT1(IABC)    = 0.
      XT2(IABC)    = 0.
 2009 CONTINUE
 1022 CONTINUE
      IC           = 0
      DO 1023 I=1,MX
 1023 XJI(JKM,I)   = XT2(I)
      IF (LL-1) 2010,2011,2010
 2010 DO 1024 IA=1,MX
      IF (ZERO(IA)) 2012,2013,2012
 2012 IC           = IC + 1
      XT3(IA)      = XT3(IA)+PB(JKM-IZE+IC)
      XT1(IA)      = XT1(IA)+XT3(IA)
      XJI(JKM,IA)  = XJI(JKM,IA)+XT3(IA)
      XT2(IA)      = XJI(JKM,IA)
 2013 CONTINUE
 1024 CONTINUE
      IC           = 0
C******************* PRINT OUT VARIABLE BIAS AND ZERO LABELS
      PRINT 3005
 3005 FORMAT(/,12X,'VARIABLE BIAS:        VARIABLE ZERO:')
      DO 1025 IA=1,MX
```

```
      IF (BIAS(IA)) 2014,2015,2014
 2014 IC            = IC+1
      XT4(IA)       = XT4(IA)+PB(JKMM+IC)
C****************** LINES REMOVED TO CORRECT AS IN NASA TN
C      XT1(IA)      = XT1(IA)-XT4(IA)
C      XT2(IA)      = XT2(IA)-XT4(IA)
C      XJI(JKM,IA) = XT2(IA)
 2015 CONTINUE
C****************** PRINT OUT VARIABLE BIAS AND ZERO INFORMATION
      PRINT 111,XT4(IA),XT3(IA)
  111 FORMAT(12X,1F12.4,10X,1F12.4)
 1025 CONTINUE
 2011 CONTINUE
      DO 1026 IA=1,JKMM
      XT5(IA)       = XT5(IA)+PB(IA)
 1026 CONTINUE
      DO 1027 IA=1,MZ
      D2(IA)        = 0.
      Z(IA,1)       = XT1(IA)
      Z(IA,2)       = XT2(IA)
 1027 CONTINUE
      IC            =0
      DO 1028 I=1,MX
      IF (ZERO(I)) 2016,2017,2016
 2016 IC            = IC+1
      XJI(JKM-IZE+IC,I) = 1.
 2017 CONTINUE
 1028 CONTINUE
C****************** CALL AGIRL (MAIN SUBROUTINE
C****************** FOR THE PARAMETER IDENTIFICATION)
      CALL AGIRL
      MAX           = NI
      DO 1029 IA=1,JKM
 1029 SUM(IA,IA)  = SUM(IA,IA)*XLA
      IF (ISUB.GE.1) CALL ASPIT(SUM)
      SUM(NI,1)    = JKM-1
      SUM(NI,2)    = JKM-1
      PRINT 3022,LL
 3022 FORMAT(/,10X,'ITERATION ',I3,'  COMPLETED')
      PRINT 3008
      IF(LL-ITR) 2018,2019,2019
 2019 CONTINUE
C****************** COMPUTE FINAL OUTPUT
C****************** MULTIPLY [RI]*[A] AND [RI]*[B] BY [R]
      MAX           = 5
      CALL AMULT(R,A,A,D54)
      CALL AMULT(R,B,B,D54)
C****************** NONDIMENSIONALIZE THE DERIVATIVE OUTPUT
      DO 1500 IABC=1,MX
      DO 1501 IDEF=1,MX
      A(IABC,IDEF)= A(IABC,IDEF)/ADIM(IABC,IDEF)
 1501 CONTINUE
      DO 1502 IDEF=1,MU
      B(IABC,IDEF)= B(IABC,IDEF)/BDIM(IABC,IDEF)
 1502 CONTINUE
 1500 CONTINUE
C****************** OUTPUT DERIVATIVES
      IF (GWGHT.NE.0.) PRINT 3600
 3600 FORMAT(/,10X,'NONDIMENSIONAL STABILITY AND CONTROL MATRICES!',
     1       )
      IF (GWGHT.EQ.0.) PRINT 3650
 3650 FORMAT(/,10X,'DIMENSIONAL STABILITY AND CONTROL MATRICES!',
     1       )
      PRINT 3016
      CALL ASPIT1(A,AA)
      PRINT 3017
      CALL ASPIT1(B,BB)
      CALL CRAMER(SUM,MU,MX,MZ,ERRSUM)
      IF (ISUB.GE.2) CALL ASPIT(SUM)
      PRINT 3008
```

```
            STOP
      2018  CONTINUE
C****************** UPDATE AND COMPUTE NEW DERIVATIVES
            CALL SOLVE(SUM,PB)
            NB          = SUM(NI,1)+.01
            IF (ISUB.GE.1) PRINT 106,(PB(I),I=1,NB)
            IJ          = 0
            DO 1030 I=1,MX
            DO 1031 J=1,MU
            IF (.NOT.BB(I,J)) GO TO 2021
      2020  IJ          = IJ+1
            B(I,J)      = B(I,J)+PB(IJ)
      2021  CONTINUE
      1031  CONTINUE
            DO 1030 J=1,MX
            IF (.NOT.AA(I,J)) GO TO 2023
      2022  IJ          = IJ+1
            A(I,J)      = A(I,J)+PB(IJ)
      2023  CONTINUE
      1030  CONTINUE
      1020  CONTINUE
            END
```

Subroutine AGIRL

Description: Subroutine AGIRL performs the parameter identification and computation of first and second gradients of the cost function.

Listing:

```
FORTRAN IV      V02.5-2    Fri 26-Feb-82 00:06:42              PAGE 001

            SUBROUTINE AGIRL
      C
      C     - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      C     THIS SUBROUTINE PERFORMS THE ACTUAL PARAMETER IDENTIFICATION.
      C     THE KEY MATRICES, VECTORS AND VARIABLES ARE :
      C
      C     [SUM]    = THIS MATRIX CONTAINS THE SECOND GRADIENT IN THE LOWER
      C                TRIANGULAR AND DIAGONAL LOCATIONS, AND THE OFF-DIAGONAL
      C                A PRIORI WEIGHTINGS IN THE UPPER TRIANGULAR. THE DIAGONAL
      C                A PRIORI WEIGHTINGS ARE STORED IN THE [APR] MATRIX. THE
      C                FIRST GRADIENT APPEARS AS AN EXTRA COLUMN IN [SUM]
      C                ( THE JKM COLUMN ).
      C     [Z],[U]  = MEASURED VALUES OF OBSERVATIONS AND CONTROLS
      C     [XT1]    = COMPUTED VALUES FOR OBRESVATIONS
      C     [XT2]    = COMPUTED VALUES FOR OBSERVATIONS
      C     [XT3]    = VARIABLE ZEROS ON THE OBSERVATIONS
      C     [XT4]    = VARIABLE BIAS ON THE OBSERVATIONS
      C     [XT5]    = DIFFERENCE BETWEEN ESTIMATED COEFFICIENTS AND THE
      C                A PRIORI VALUES
      C     MX       = NUMBER OF STATES
      C     MU       = NUMBER OF CONTROLS
      C     MZ       = NUMBER OF OBSERVATIONS
      C     - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      C
            COMMON /ALLDIM/ MAX,MAT
```

```
      COMMON /SUBWRT/ ISUB
      COMMON /MATRIX/ A,B,AA,BB,AP,BP,D1
      COMMON /VECTOR/ U,Z,XT1,XT2,XT4,XT5,ZERO,BIAS,IBIAS
      COMMON /ARRAYS/ D2,APHI,PHI1,DUM,BJI,XJI,SUM,APR
      COMMON /INSTRM/ XAX,ZAX,XAY,YAY,ZAY,XAZ,ZAZ,THAI,COSTHI,
     1                SINTHI,GRV
      COMMON /NUMBER/ JKMM,JKM,JKMM1,HH,TIME,TT,NN,NNM1,L,
     1                LL,ITR,MU,MZ,MX,MXP1,I,IA,J,JK,K,KJ,NI,
     2                ITHIN,ERRSUM
      COMMON /COMMND/ LONG,LATR,EXTRA,BYPASS,MAPR
      DIMENSION AP(8,4),BP(8,3),XT4(4),DUM(25,4),XT2(7)
      DIMENSION XT5(25),APR(25),ZERO(5),BIAS(5)
      DIMENSION Z(7,3),U(3,3),D2(7),APHI(5,4),XT1(7),PHI1(5,4),
     1          D1(8,7),A(5,4),B(5,4),BJI(25,4),XJI(25,8),SUM(25,25)
      LOGICAL*1 LONG,LATR,EXTRA,BYPASS,AA(5,4),BB(5,4)
      IF (ISUB.GE.1) PRINT 2001
 2001 FORMAT('  SUBROUTINE AGIRL',
C
C     - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
C
      ANPT         = FLOAT(NNM1) + 1.
C****************** TIME LOOP
      TT           = TIME - HH
      DO 41 I=2,NNM1
      TT           = TT + HH
      DO 28 JK=1,JKM
      DO 28 J=MXP1,MZ
   28 XJI(JK,J)    = 0.
      DO 170 IA=1,MX
  170 XJI(JKM,IA) = XT2(IA)
C****************** READ IN MEASURED RESPONSES FROM DATA FILE
C****************** READ IN LONGITUDINAL DATA
      IF (LONG) READ(4) Z(4,3),Z(1,3),Z(7,3),Z(6,3),U(1,3),AM6,AM7,AM8,
     1                  AM9,AM10,AM11
      IF (LONG.AND.EXTRA) READ(4) Z(2,3),Z(3,3),Z(5,3),AM15,AM16,AM17
      IF (LONG) U(2,3) = 0.
C****************** READ IN LATERAL DIRECTIONAL DATA
      IF (LATR) READ(4) AM1,AM2,AM3,AM4,AM5,Z(4,3),Z(1,3),Z(7,3),Z(2,3),
     1                  U(1,3),U(2,3)
      IF (LATR.AND.EXTRA) READ(4) AM12,AM13,AM14,Z(3,3),Z(5,3),Z(6,3)
C****************** DELETE UNWEIGHTED DATA FROM INPUT
      DO 2525 IABC=1,7
      IF (D1(IABC,IABC).NE.0.) GO TO 2524
      Z(IABC,3)    = 0.
 2524 CONTINUE
 2525 CONTINUE
      MAX          = NI
      CALL AZOT(BJI)
      JK           = 0
      DO 44 J=1,MX
      DO 43 K=1,MU
      BJI(JKM,J)   = BJI(JKM,J)+B(J,K)*(U(K,3)+U(K,2))*.5
      IF (.NOT.BB(J,K)) GO TO 43
      JK           = JK +1
      XJI(JK,J+MX)= U(K,2)*BP(J+MX,K)
      BJI(JK,J)    = .5*(U(K,2)+U(K,1))
   43 CONTINUE
      DO 44 K=1,MX
      IF (.NOT.AA(J,K)) GO TO 44
      JK           = JK + 1
C****************** SPECIAL OPTION IF FIRST TIME THROUGH
      IF (LL-1) 2,2,4
    2 CONTINUE
      BJI(JK,J)    = .5*(Z(K,2)+Z(K,1))
      XJI(JK,J+MX)= Z(K,2)*AP(J+MX,K)
C****************** IF ALL STATES [Z(1,X) TO Z(4,X)] ARE NOT MEASURED
C****************** THEN REPLACE THESE 0. VALUES TO KEEP FROM HAVING
C****************** A SINGULAR SUM MATRIX
      IF (D1(K,K).NE.0.) GO TO 44
    4 CONTINUE
```

```
      81 FORMAT(10X,8F12.4)
C****************** SAVE TIME HISTORIES ON DISK FILE
         WRITE (3) (XT2(IA),IA=1,7)
      80 CONTINUE
         DO 91 J=1,JKM
         DO 91 I4=J,JKM
         DO 92 K=1,MZ
      92 SUM(I4,J)    = SUM(I4,J)+XJI(I4,K)*D1(K,K)*XJI(J,K)
      91 CONTINUE
         DO 69 IA=1,MZ
         Z(IA,1)      = Z(IA,2)
      69 Z(IA,2)      = Z(IA,3)
         U(1,1)       = U(1,2)
         U(2,1)       = U(2,2)
         U(1,2)       = U(1,3)
         U(2,2)       = U(2,3)
      41 CONTINUE
C****************** PRINT OUT CASE INFORMATION
         ERRSUM       = SUM(JKM,JKM)/ANPT
         PRINT 607, ERRSUM
     607 FORMAT(/,10X,'WEIGHTED ERROR SUM = ',1PE12.4)
         MAX          = 8
         PRINT 608
     608 FORMAT(10X,'WEIGHTED ERRORS:')
         PRINT 606,(D2(IA)*D1(IA,IA)/ANPT,IA=1,MZ)
     606 FORMAT(10X,10(1PE12.4),/)
         DO 888 IJK=1,JKM
     888 SUM(IJK,JKM)= SUM(JKM,IJK)
         IF (MAPR) 180,181,180
     180 DO 182 IB=1,JKM
         SUM(IB,JKM) =-XT5(IB)*APR(IB)+SUM(IB,JKM)
         SUM(IB,IB)   = SUM(IB,IB)+APR(IB)
         IBM1         = IB-1
         DO 182 IA=1,IBM1
     182 SUM(IB,IA)   = SUM(IB,IA)+SUM(IA,IB)
     181 CONTINUE
     531 FORMAT(10X,'  WOULD YOU BELIEVE.......... ',//)
         IF (ISUB.GE.2) PRINT 531
         RETURN
         END
```

Subroutine AEAT

Description: Subroutine AEAT computes the transition matrix and its integral, $e^{A\Delta t}$ and $\int_0^{\Delta t} e^{A\tau} d\tau$, respectively. It uses the Taylor series expansion to ten terms for these calculations.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-82 00:29:58              PAGE 001

         SUBROUTINE AEAT (A,T,PHI,APHI,A2,A3)
C
C        - - - - - - - - - - - - - - - - - - - - - - - - - - - -
C        THIS SUBROUTINE COMPUTES THE TRANSITION MATRIX
C        AND ITS INTEGRAL USING A TAYLOR SERIES EXPANSION
C        TO 10 TERMS.
```

```
      BJI(JK,J)    = (XT2(K)+XT1(K))*.5
      XJI(JK,J+MX)= XT2(K)*AP(J+MX,K)
   44 CONTINUE
      MAX          = NI
      MAT          = 5
      XJI(NI,2)    = MX
      CALL AMULT(XJI,PHI1,XJI,DUM)
      CALL AMULT(BJI,APHI,DUM,DUM)
      CALL AADD(1.0,DUM,1.0,XJI,XJI)
      XJI(NI,2)    = MZ
      IBIAS        = 0
      DO 162 IA=1,MX
      IF (BIAS(IA)) 163,162,163
  163 IBIAS        = IBIAS+1
      DO 175 IB=1,MZ
  175 XJI(JKMM+IBIAS,IB)= 0.
C****************** CORRECTION FROM NASA TN
      XJI(JKMM+IBIAS,IA+MX)= 1.
C
  162 CONTINUE
      JKMM1        = JKM-1
      DO 7 JK=1,JKMM
      DO 7 L=MXP1,MZ
      DO 7 K=1,MX
      XJI(JK,L)    = XJI(JK,L)+A(L-MX,K)*XJI(JK,K)*AP(L,K)
    7 CONTINUE
      DO 9 L=MXP1,MZ
C****************** CORRECTION FROM NASA TN
C****************** ADD VARIABLE BIAS TO STATES
      XJI(JKM,L)   = XT4(L-MX)
C
      DO 8 K=1,MU
      XJI(JKM,L)   = XJI(JKM,L)+B(L-MX,K)*U(K,3)*BP(L,K)
    8 CONTINUE
      DO 9 K=1,MX
      XJI(JKM,L)   = XJI(JKM,L)+A(L-MX,K)*XJI(JKM,K)*AP(L,K)
    9 CONTINUE
C****************** TRANSLATE TO INSTRUMENT AXES AND LOCATION
C****************** LONGITUDINAL CASE
      IF (LONG) AXI          = COSTHI*XJI(JKM,6)-SINTHI*XJI(JKM,7)
      IF (LONG) AZI          = SINTHI*XJI(JKM,6)+COSTHI*XJI(JKM,7)
      IF (LONG) PI           = COSTHI*AM7       -SINTHI*AM9
      IF (LONG  QDOTI        = XJI(JKM,5)
      IF (LONG) QI           = XJI(JKM,1)
      IF (LONG) RI           = SINTHI*AM7       +COSTHI*AM9
      IF (LONG) XJI(JKM,4) = XJI(JKM,4)+THAI
      IF (LONG) XJI(JKM,6) = AXI+(ZAX*QDOTI-XAX*(QI**2+RI**2))/GRV
      IF (LONG) XJI(JKM,7) = (AZI+(-XAZ*QDOTI-ZAZ*(QI**2+PI**2))/GRV)
C****************** LATERAL-DIRECTIONAL CASE
      IF (LATR) PI           = COSTHI*XJI(JKM,1)-SINTHI*XJI(JKM,2)
      IF (LATR) RI           = SINTHI*XJI(JKM,1)+COSTHI*XJI(JKM,2)
      IF (LATR) PIDOT        = COSTHI*XJI(JKM,5)-SINTHI*XJI(JKM,6)
      IF (LATR) RIDOT        = SINTHI*XJI(JKM,5)+COSTHI*XJI(JKM,6)
      IF (LATR) XJI(JKM,1) = PI
      IF (LATR) XJI(JKM,2) = RI
      IF (LATR) XJI(JKM,5) = PIDOT
      IF (LATR) XJI(JKM,6) = RIDOT
      IF (LATR) XJI(JKM,7) = XJI(JKM,7)+(-ZAY*PIDOT+XAY*RIDOT
     1                       -YAY*(PI**2+RI**2))/GRV
      DO 3 J=1,MZ
      XT1(J)       = XT2(J)
      XT2(J)       = XJI(JKM,J)
      XJI(JKM,J)   = Z(J,3)-XT2(J)
    3 CONTINUE
      DO 27 K=1,MZ
      D2(K)        = D2(K)+XJI(JKM,K)**2
   27 CONTINUE
      MAX          = NI
C****************** IF TEST THEN PRINT OUT TIME HISTORIES
      IF (ISUB.GE.6) PRINT 81,(XT2(IA),IA=1,7),TT
```

```
C
C      [A]     = STABILITY MATRIX
C      T       = DELTA TIME INCREMENT
C      [PHI]   = TRANSITION MATRIX
C      [APHI]  = INTEGRAL OF THE TRANSITION MATRIX
C      [A2]    = DUMMY MATRIX
C      [A3]    = DUMMY MATRIX
C      - - - - - - - - - - - - - - - - - - - - - - - - - -
C
       COMMON /ALLDIM/ MAX,MIX
       COMMON /SUBWRT/ ISUB
       DIMENSION A(1),PHI(1),A2(1),APHI(1),A3(1)
       IF (ISUB.GE.3) PRINT 2001
  2001 FORMAT('  SUBROUTINE AEAT')
C
C      - - - - - - - - - - - - - - - - - - - - - - - - - -
C
C****************** FIND AND SET MATRIX DIMENSIONS
       MAX2      = MAX*2
       II        = A(MAX)
       PHI(MAX)  = A(MAX)
       PHI(MAX2)= A(MAX)
       CALL AZOT(PHI)
       CALL AMAKE(APHI,PHI)
       CALL AMAKE(A3,PHI)
       MI        =-MAX
C****************** SET PHI EQUAL TO IDENTITY MATRIX
       DO 1 I=1,II
       MI        = MI+MAX
       PHI(MI+I)= 1.
     1 CONTINUE
C****************** PERFORM TAYLOR SERIES SUMMATION
       CALL AMAKE(A2,PHI)
       G         = 1.
       DO 2 I=1,10
       BB        = I
       G         = G*T/BB
       CALL AADD(1.,APHI,G,A2,APHI)
       CALL AMULT(A,A2,A2,A3)
       CALL AADD(1.,PHI,G,A2,PHI)
     2 CONTINUE
C****************** TRANSPOSE MATRICES
       DO 10 I=1,II
       DO 10 J=1,I
       JI        = (I-1)*MAX+J
       IJ        = (J-1)*MAX+I
       TEMP      = PHI(IJ)
       PHI(IJ)   = PHI(JI)
       PHI(JI)   = TEMP
       TEMP      = APHI(IJ)
       APHI(IJ)  = APHI(JI)
    10 APHI(JI)  = TEMP
C****************** PRINT OUT MATRICES FOR TEST
       IF (ISUB.LT.3) RETURN
       PRINT 1000
  1000 FORMAT(10X,'PHI MATRIX')
       CALL ASPIT(PHI)
       PRINT 1001
  1001 FORMAT(10X,'APHI MATRIX')
       CALL ASPIT(APHI)
       RETURN
       END
```

Subroutine AMAKE

Description: Subroutine AMAKE moves a copy of matrix Y into X.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-82 00:30:24              PAGE 001

              SUBROUTINE AMAKE(X,Y)
      C
      C       - - - - - - - - - - - - - - - - - - - - - - - - - -
      C       THIS SUBROUTINE GENERATES A MATRIX [X] THAT IS
      C       A COPY OF MATRIX [Y].
      C
      C       [X] = NEW MATRIX, COPY OF [Y]
      C       [Y] = MATRIX TO BE COPIED
      C       - - - - - - - - - - - - - - - - - - - - - - - - - -
      C
              COMMON /ALLDIM/ MAX,MIX
              COMMON /SUBWRT/ ISUB
              DIMENSION X(1),Y(1)
              IF (ISUB.GE.4) PRINT 2001
        2001 FORMAT('  SUBROUTINE AMAKE')
      C
      C       - - - - - - - - - - - - - - - - - - - - - - - - - -
      C
      C****************** FIND MATRIX DIMENSIONS
              MAX2    = MAX*2
              IIM1    = Y(MAX)-1.
              JJM1    = Y(MAX2)-1.
              LEND    = JJM1*MAX+1
      C****************** PERFORM MATRIX COPYING
              DO 1 L=1,LEND,MAX
              KEND    = L+IIM1
              DO 1 K=L,KEND
            1 X(K)    = Y(K)
      C****************** SET MATRIX DIMENSIONS
              X(MAX) = Y(MAX)
              X(MAX2)= Y(MAX2)
              RETURN
              END
```

Subroutine AZOT

Description: Subroutine AZOT initializes the elements of a matrix to zero.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-82 00:30:46              PAGE 001

              SUBROUTINE AZOT(X)
      C
      C       - - - - - - - - - - - - - - - - - - - - - - - - - -
      C       THIS SUBROUTINE SETS ALL ELEMENTS OF A MATRIX
      C       TO ZERO.
      C
```

```
      C      [X] = MATRIX TO BE ZEROED
      C      - - - - - - - - - - - - - - - - - - - - - - -
      C
             COMMON /ALLDIM/ MAX,MIX
             COMMON /SUBWRT/ ISUB
             DIMENSION X(1)
             IF (ISUB.GE.4) PRINT 2001
        2001 FORMAT('  SUBROUTINE AZOT')
      C
      C      - - - - - - - - - - - - - - - - - - - - - - - - -
      C
      C****************** FIND MATRIX DIMENSIONS
             MAX2    = MAX*2
             IIM1    = X(MAX)-1.
             JJM1    = X(MAX2)-1.
             LEND    = JJM1*MAX+1
      C****************** PERFORM MATRIX INITIALIZATION
             DO 1 L=1,LEND,MAX
             KEND    = L+IIM1
             DO 1 K=L,KEND
           1 X(K)    = 0.
             RETURN
             END
```

Subroutine REDUCE

Description: Subroutine REDUCE factors a symmetric matrix by Cholesky's matrix decomposition method. This factoring is used for updating the coefficients and for calculation of the confidence levels.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-82 00:31:08              PAGE 001

             SUBROUTINE REDUCE(A)
      C
      C      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      C      THIS SUBROUTINE FACTORS A SYMMETRIC MATRIX [A] BY THE
      C      'CHOLESKY'S' MATRIX DECOMPOSITION. THE MATRIX IS FACTORED
      C      INTO:
      C
      C           [LI] * [D] * [LI*].
      C      [L] = LOWER TRIANGULAR MATRIX WITH UNITY DIAGONAL ELEMENTS
      C      [D] = DIAGONAL MATRIX
      C         I  : DENOTES INVERSE OPERATION
      C         *  : DENOTES TRANSPOSE OPERATION
      C      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      C
             COMMON /SUBWRT/ ISUB
             DIMENSION A(25,25)
             IF (ISUB.GE.3) PRINT 2001
        2001 FORMAT('  SUBROUTINE REDUCE')
      C
      C      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      C
      C****************** FACTOR MATRIX
             N       = A(25,1)
```

```
      NM1    = N-1
      DO 20 K=1,NM1
      KP1    = K+1
      KM1    = K-1
      AKKI   = 1./A(K,K)
      DO 20 I=KP1,N
      AKKIK  = A(I,K)*AKKI
      DO 10 J=I,N
   10 A(J,I) = A(J,I)-AKKIK*A(J,K)
      A(I,K) =-AKKIK
      IF (KM1.EQ.0) GO TO 20
      DO 15 J=1,KM1
   15 A(I,J) = A(I,J)-AKKIK*A(K,J)
   20 CONTINUE
C****************** [L] IS NOW STORED IN LOWER TRIANGULAR PART OF [A]
C****************** EXCEPT FOR DIAGONAL, WHICH CONTAINS [D]
      RETURN
      END
```

Subroutine AADD

Description: Subroutine AADD adds scalar multiples of two matrices.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-82 00:31:30            PAGE 001

      SUBROUTINE AADD (G,X,H,Y,Z)
C
C     - - - - - - - - - - - - - - - - - - - - - - - - - - -
C     THIS SUBROUTINE ADDS SCALAR MULTIPLES OF TWO
C     MATRICES AS FOLLOWS:
C
C         [Z] = G*[X] + H*[Y]    WITH : G = 1.0.
C     ( NO CHECKING IS MADE FOR MATRIX COMPATIBILITY )
C     - - - - - - - - - - - - - - - - - - - - - - - - - - -
C
      COMMON /ALLDIM/ MAX,MIX
      COMMON /SUBWRT/ ISUB
      DIMENSION X(1),Y(1),Z(1)
      IF (ISUB.GE.4) PRINT 2001
 2001 FORMAT('  SUBROUTINE AADD')
C
C     - - - - - - - - - - - - - - - - - - - - - - - - - - -
C
C****************** FIND MATRIX DIMENSIONS
      G      = 1.
      MAX2   = MAX*2
      II     = X(MAX)
      JJ     = X(MAX2)
      JEND   = (JJ-1)*MAX+1
      IIM1   = II-1
C****************** PERFORM MATRIX SCALAR ADDITION
      DO 53 J=1,JEND,MAX
      KEND   = J+IIM1
      DO 53 K=J,KEND
   53 Z(K)   = G*X(K)+H*Y(K)
      Z(MAX) = X(MAX)
      Z(MAX2)= X(MAX2)
      RETURN
      END
```

Subroutine AMULT

Description: Subroutine AMULT computes the matrix product of two matrices. It places the result into a third matrix which cannot be either of the first two matrices.

Listing:

```
FORTRAN IV      V02.5-2     Sun 17-Jan-82 00:31:52                PAGE 001

0001         SUBROUTINE AMULT (A,B,C,D)
       C
       C     - - - - - - - - - - - - - - - - - - - - - - - - - - -
       C     THIS SUBROUTINE COMPUTES THE PRODUCT OF TWO
       C     MATRICES AND PLACES THE RESULT IN A THIRD MATRIX.
       C
       C          [D] = [A]*[B] ;  [C] = [D]
       C     ( NO CHECKING IS MADE FOR MATRIX COMPATIBILITY )
       C     - - - - - - - - - - - - - - - - - - - - - - - - - - -
       C
0002         COMMON /ALLDIM/ MAX,MIX
0003         COMMON /SUBWRT/ ISUB
0004         DIMENSION A(1),B(1),C(1),D(1)
0005         IF (ISUB.GE.4) PRINT 2001
0007    2001 FORMAT('  SUBROUTINE AMULT')
       C
       C     - - - - - - - - - - - - - - - - - - - - - - - - - - -
       C
       C******************* FIND AND SET MATRIX DIMENSIONS
0008         MAX2    = MAX*2
0009         MIX2    = MIX*2
0010         II      = A(MAX)
0011         D(MAX)  = A(MAX)
0012         C(MAX)  = A(MAX)
0013         JJ      = A(MAX2)
0014         KK      = B(MIX2)
0015         D(MAX2)= B(MIX2)
0016         C(MAX2)= B(MIX2)
       C******************* PERFORM MATRIX MULTIPLICATION
0017         JE      = (JJ-1)*MAX
0018         KE      = (KK-1)*MAX
0019         DO 20 I=1,II
0020         KEND    = KE+I
0021         JEND    = JE+I
0022         L       = 1
0023         DO 20 K=I,KEND,MAX
0024         D(K)    = 0.
0025         JB      = L
0026         DO 10 J=I,JEND,MAX
0027         D(K)    = A(J)*B(JB)+D(K)
0028      10 JB      = JB + 1
0029      20 L       = L + MIX
       C******************* COPY MATRIX [D] INTO [C]
0030         CALL AMAKE(C,D)
0031         RETURN
0032         END
```

Subroutine ALOAD

Description: Subroutine ALOAD loads matrices from the input file created by SETUP.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-82 00:33:02              PAGE 001

0001          SUBROUTINE ALOAD (N,A,B,C,D)
         C
         C    - - - - - - - - - - - - - - - - - - - - - - - - - -
         C    THIS SUBROUTINE LOADS MATRICES [A],[B],[C] AND [D]
         C    FROM AN INPUT FILE.  THE VARIABLE N SPECIFIES THE
         C    NUMBER OF MATRICES TO BE LOADED.
         C    - - - - - - - - - - - - - - - - - - - - - - - - - -
         C
0002          COMMON /SUBWRT/ ISUB
0003          DIMENSION A(1),B(1),C(1),D(1)
0004          IF (ISUB.GE.4) PRINT 2001
0006     2001 FORMAT('  SUBROUTINE ALOAD')
         C
         C    - - -  - - - - - - - - - - - - - - - - - - - - - -
         C
         C******************* PERFORM MATRIX LOADING
0007          CALL ALOAD1(A)
0008          IF (N.LT.2) RETURN
0010          CALL ALOAD1(B)
0011          IF (N.LT.3) RETURN
0013          CALL ALOAD1(C)
0014          IF (N.LT.4) RETURN
0016          CALL ALOAD1(D)
0017          RETURN
0018          END
```

Subroutine ALOAD1

Description: Subroutine ALOAD1 actually reads the input from the SETUP file for storing into the matrix A.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-82 00:33:23              PAGE 001

0001          SUBROUTINE ALOAD1(A)
         C
         C    - - - - - - - - - - - - - - - - - - - - - - - - -
         C    THIS SUBROUTINE PERFORMS THE ACTUAL LOADING OF
         C    A MATRIX FROM THE INPUT FILE.
         C
         C    [A] = MATRIX TO BE LOADED
         C    ( FIRST RECORD CONTAINS THE MATRIX DIMENSIONS )
         C    - - - - - - - - - - - - - - - - - - - - - - - - -
         C
0002          COMMON /ALLDIM/ MAX,MIX
0003          COMMON /SUBWRT/ ISUB
0004          DIMENSION A(1)
0005          IF (ISUB.GE.5) PRINT 2001
0007     2001 FORMAT('  SUBROUTINE ALOAD1')
```

```
      C
      C     - - - - - - - - - - - - - - - - - - - - - - - - -
      C
      C******************* READ MATRIX DIMENSIONS
            READ(2,100) II,JJ
        100 FORMAT(8X,I2,I10)
            KE       =(JJ-1)*MAX
            DO 10 I=1,II
            KEND     = I+KE
      C******************* READ MATRIX
         10 READ(2,1001) (A(K),K=I,KEND,MAX)
      C******************* SET MATRIX DIMENSIONS
            A(MAX)   = II
            A(MAX*2)= JJ
       1001 FORMAT(8F12.6)
            RETURN
            END
```

Subroutine ASPIT

Description: Subroutine ASPIT prints out a matrix.

Listing:

```
FORTRAN IV      V02.5-2     Sun 17-Jan-82 00:33:45              PAGE 001

            SUBROUTINE ASPIT(X)
      C
      C     - - - - - - - - - - - - - - - - - - - - - - - - -
      C     THIS SUBROUTINE PRINTS OUT A MATRIX.
      C
      C     [X] = MATRIX TO BE PRINTED OUT
      C     - - - - - - - - - - - - - - - - - - - - - - - - -
      C
            COMMON /ALLDIM/ MAX,MIX
            COMMON /SUBWRT/ ISUB
            DIMENSION X(1)
            IF (ISUB.GE.4) PRINT 2001
       2001 FORMAT('  SUBROUTINE ASPIT')
      C
      C     - - - - - - - - - - - - - - - - - - - - - - - - -
      C
        100 FORMAT(10X,' DIMENSION ',I3,' BY ',I3)
        101 FORMAT(10X,10(1PE12.4))
      C******************* FIND MATRIX DIMENSIONS
            MAX2     = MAX*2
            II       = X(MAX)
            JJ       = X(MAX2)
      C******************* PERFORM MATRIX OUTPUT
            PRINT 100, II,JJ
            KE       = (JJ-1)*MAX
            DO 1 I=1,II
            KEND     = I+KE
          1 PRINT 101, (X(K),K=I,KEND,MAX)
            RETURN
            END
```

Subroutine ASPIT1

Description: Subroutine ASPIT1 is a special form of ASPIT. In this form the printing of the matrix is combined with printing of "*" or " " which show whether a matrix element is a variable or not.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-82 00:34:06               PAGE 001

0001          SUBROUTINE ASPIT1(X,XX)
         C
         C     - - - - - - - - - - - - - - - - - - - - - - - - -
         C     SUBROUTINE USED FOR THE PRINTOUT OF MATRICES.
         C     ( SPECIAL OUTPUT FOR [A] AND [B] MATRICES )
         C
         C     [X]  = MATRIX TO BE PRINTED OUT
         C     [XX] = MATRIX WHICH SHOWS VARIABLES
         C     - - - - - - - - - - - - - - - - - - - - - - - - -
         C
0002          COMMON /ALLDIM/ MAX,MIX
0003          COMMON /SUBWRT/ ISUB
0004          DIMENSION X(1)
0005          LOGICAL*1 XX(1)
0006          BYTE CHAR(4)
0007          IF (ISUB.GE.4) PRINT 2001
0009     2001 FORMAT('  SUBROUTINE ASPIT1')
         C
         C     - - - - - - - - - - - - - - - - - - - - - - - - -
         C
0010      100 FORMAT(10X,'DIMENSION ',I3,' BY',I3)
0011      101 FORMAT(10X,5(1PE12.4,A1))
         C****************** FIND MATRIX DIMENSIONS
0012          MAX2    = MAX*2
0013          II      = X(MAX)
0014          JJ      = X(MAX2)
         C****************** PERFORM MATRIX OUTPUT
0015          PRINT 100, II,JJ
0016          KE      = (JJ-1)*MAX
0017          DO 1 I=1,II
0018          KEND    = I+KE
0019          DO 2 K=I,KEND,MAX
         C****************** '*' INDICATES THAT A VALUE IS NOT A VARIABLE
         C****************** IN EITHER THE [A] OR [B] MATRICES
0020          CHAR((K-I)/MAX+1)=' '
0021          IF (.NOT.XX(K)) CHAR((K-I)/MAX+1)='*'
0023        2 CONTINUE
0024        1 PRINT 101, ((X(K),CHAR((K-I)/MAX+1)),K=I,KEND,MAX)
0025          RETURN
0026          END
```

Subroutine SOLVE

Description: The subroutine SOLVE is used to solve the set of linear equations, Ax = b, where A is a symmetrical matrix.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-92 00:34:29              PAGE 001

0001          SUBROUTINE SOLVE(A,X)
        C
        C     - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
        C     THE FOLLOWING SYSTEM OF LINEAR EQUATIONS IS SOLVED BY
        C     THIS SUBROUTINE :
        C
        C         [A]*[X] = [B].
        C     [A] = SYMMETRICAL MATRIX
        C     [B] = N+1 COLUMN OF [A] MATRIX
        C     N   = DIMENSION OF SYSTEM
        C     - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
        C
0002          COMMON /SUBWRT/ ISUB
0003          DIMENSION A(25,25),X(25)
0004          IF (ISUB.GE.3) PRINT 2001
0006     2001 FORMAT('  SUBROUTINE SOLVE')
        C
        C     - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
        C
        C******************* FACTOR SYMMETRIC MATRIX [A] INTO [L]
0007          CALL REDUCE(A)
0008          N       = A(25,1)
0009          NM1     = N-1
0010          NP1     = N+1
        C******************* MULTIPLY [L] * [B]
0011          DO 70 I=2,N
0012          X(I)    = A(I,NP1)
0013          IM1     = I-1
0014          DO 70 J=1,IM1
0015       70 X(I)    = X(I)+A(I,J)*A(J,NP1)
        C****  *********** MULTIPLY BY [DI]
0016          A(1,NP1)= A(1,NP1)/A(1,1)
0017          DO 80 I=2,N
0018       80 A(I,NP1)= X(I)/A(I,I)
        C******************* MULTIPLY BY [L*] TO FORM [L*]*[DI]*[L]*[B]
0019          DO 90 I=1,NM1
0020          X(I)    = A(I,NP1)
0021          IP1     = I+1
0022          DO 90 J=IP1,N
0023       90 X(I)    = X(I)+A(J,I)*A(J,NP1)
0024          X(N)    = A(N,NP1)
0025          RETURN
0026          END
```

Subroutine INV

Description: Subroutine INV inverts a general matrix. Gaussion elimination is used without pivoting. This is allowed, since the matrix is near-diagonal (R) and well conditioned.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-82 00:34:53               PAGE 001

            SUBROUTINE INV(A,MAX)
      C
      C     - - - - - - - - - - - - - - - - - - - - - - - - - - -
      C     THIS SUBROUTINE INVERTS A GENERAL MATRIX IN PLACE
      C     USING GAUSSIAN ELIMINATION (NO PIVOTING)
      C
      C     [A] = MATRIX TO BE INVERTED
      C     MAX = NUMBER OF ROWS IN MATRIX
      C     - - - - - - - - - - - - - - - - - - - - - - - - - - -
      C
            COMMON /SUBWRT/ ISUB
            DIMENSION A(MAX,4)
            IF (ISUB.GE.3) PRINT 2001
       2001 FORMAT('  SUBROUTINE INV')
      C
      C     - - - - - - - - - - - - - - - - - - - - - - - - - - -
      C
      C****************** COMPUTE INVERSE OF [A] AND STORE RESULT IN [A]
            N       = A(MAX,1)
            DO 80 K=1,N
            BIGA    = A(K,K)
            DO 50 I=1,N
            IF (I.EQ.K) GO TO 50
            A(I,K) =-A(I,K)/BIGA
         50 CONTINUE
            DO 60 I=1,N
            IF (I.EQ.K) GO TO 60
            DO 55 J=1,N
            IF (J.EQ.K) GO TO 55
            A(I,J) = A(I,J)+A(I,K)*A(K,J)
         55 CONTINUE
         60 CONTINUE
            DO 70 J=1,N
            IF (J.EQ.K) GO TO 70
            A(K,J) = A(K,J)/BIGA
         70 CONTINUE
         80 A(K,K) = 1./BIGA
            RETURN
            END
```

Subroutine CRAMER

Description: Subroutine CRAMER computes the confidence levels based upon Cramèr-Rao bounds. This routine assumes that the a priori option has not been used, since their contribution to the second gradient has not been removed.

Listing:

```
FORTRAN IV      V02.5-2    Sun 17-Jan-82 00:35:30                PAGE 001

           SUBROUTINE CRAMER(SUM,MU,MX,MZ,ERRSUM)
      C
      C    - - - - - - - - - - - - - - - - - - - - - - - - -
      C
      C    THIS SUBROUTINE COMPUTES THE CRAMER-RAO BOUNDS
      C    ALSO KNOWN AS THE CONFIDENCE LEVELS OF THE
      C    ESTIMATED DERIVATIVES.
      C
      C    [SUM]  = SECOND GRADIENT MATRIX
      C    MU     = NUMBER OF CONTROL INPUTS
      C    MX     = NUMBER OF STATES
      C    MZ     = NUMBER OF OBSERVATIONS
      C    ERRSUM = WEIGHTED ERROR
      C    - - - - - - - - - - - - - - - - - - - - - - - - -
      C
           COMMON /ALLDIM/ MAX,MIX
           COMMON /SUBWRT/ ISUB
           COMMON /MATRIX/ A,B,AA,BB,AP,BP,D1
           COMMON /DIMFAC/ ADIM,BDIM
           COMMON /GEOMTR/ S,CBAR,SPAN,GWGHT,A1,A2,A3,A4
           DIMENSION AC(5,4),BC(5,4),ACPER(5,4),BCPER(5,4)
           DIMENSION ADIM(5,4),BDIM(5,4)
           DIMENSION AP(8,4),BP(8,3)
           DIMENSION D1(8,7),A(5,4),B(5,4),SUM(25,25)
           LOGICAL*1 AA(5,4),BB(5,4)
           IF (ISUB.GE.3) PRINT 2001
      2001 FORMAT('  SUBROUTINE CRAMER')
      C
      C    - - - - - - - - - - - - - - - - - - - - - - - - -
      C
      C****************** NORMALLY THE APRIORI CONTRIBUTION TO HESSIAN
      C****************** IS SUBTRACTED FOR THIS COMPUTATION BUT THIS
      C****************** ROUTINE ASSUMES NO APRIORI OPTIONS ARE BEING
      C****************** USED AND HENSE THERE ARE NO CONTRIBUTIONS
      C****************** SET DIMENSIONS OF BOUND MATRICES
           AC(5,1)    = MX
           ACPER(5,1)= MX
           AC(5,2)    = MX
           ACPER(5,2)= MX
           BC(5,1)    = MX
           BCPER(5,1)= MX
           BC(5,2)    = MU
           BCPER(5,2)= MU
      C****************** OBTAIN DIAGONAL ELEMENTS OF INVERSE
           CALL DIAGIN(SUM)
      C****************** COMPUTE BOUNDS
           WTS     = 0.
           DO 1 I=1,MZ
           IF (D1(I,I).NE.0.) WTS = WTS + 1.
         1 CONTINUE
           COEFF  = ERRSUM/WTS
           IF (ISUB.GE.3) PRINT 10,ERRSUM,COEFF,WTS
        10 FORMAT(' ERRSUM = ',F12.4,' COEFF = ',F12.4,' WTS = ',F12.4)
           L      = 0
```

```
      DO 2 I=1,MX
      DO 3 J=1,MU
      BC(I,J)= 0.
      IF (.NOT.BB(I,J)) GO TO 3
      L       = L+1
      BC(I,J)= SQRT(ABS(SUM(L,L))*COEFF)
    3 CONTINUE
      DO 4 J=1,MX
      AC(I,J)= 0.
      IF (.NOT.AA(I,J)) GO TO 4
      L       = L+1
      AC(I,J)= SQRT(ABS(SUM(L,L))*COEFF)
    4 CONTINUE
      IF (AA(I,4)) L=L+1
    2 CONTINUE
C******************* NON-DIMENSIONALIZE STABILITY AND CONTROL DERIVATIVES
C******************* AND COMPUTE BOUNDS AS A PERCENTAGE OF DERIVATIVES
      DO 50 I=1,MX
      DO 60 J=1,MX
      AC(I,J)=AC(I,J)/ADIM(I,J)
      ACPER(I,J)=0.
      IF (A(I,J).NE.0.) ACPER(I,J)=AC(I,J)/A(I,J)*100.
   60 CONTINUE
      DO 70 J=1,MU
      BC(I,J)=BC(I,J)/BDIM(I,J)
      BCPER(I,J)=0.
      IF (B(I,J).NE.0.) BCPER(I,J)=BC(I,J)/B(I,J)*100.
   70 CONTINUE
   50 CONTINUE
C******************* PRINT OUT CRAMER RAO BOUNDS
  100 FORMAT(10X,' DIMENSION ',I3,' BY ',I3)
  101 FORMAT(10X,5(1PE12.4,' (',OPF10.3,') '))
      IF (GWGHT.NE.0.) PRINT 6
    6 FORMAT(/,10X,'NON-DIMENSIONAL [AC] MATRIX,',
     1             ' (PERCENTAGE OF DERIVATIVE)')
      IF (GWGHT.EQ.0.) PRINT 66
   66 FORMAT(/,10X,'    DIMENSIONAL [AC] MATRIX,',
     1             ' (PERCENTAGE OF DERIVATIVE)')
      PRINT 100, MX,MX
      DO 80 I=1,MX
      PRINT 101, ((AC(I,J),ACPER(I,J)),J=1,MX)
   80 CONTINUE
      IF (GWGHT.NE.0.) PRINT 7
    7 FORMAT(/,10X,'NON-DIMENSIONAL [BC] MATRIX,',
     1             ' (PERCENTAGE OF DERIVATIVE)')
      IF (GWGHT.EQ.0.) PRINT 77
   77 FORMAT(/,10X,'    DIMENSIONAL [BC] MATRIX,',
     1             ' (PERCENTAGE OF DERIVATIVE)')
      PRINT 100, MX,MU
      DO 90 I=1,MX
      PRINT 101, ((BC(I,J),BCPER(I,J)),J=1,MU)
   90 CONTINUE
      RETURN
      END
```

C-3

Subroutine DIAGIN

Description: This subroutine calculates the diagonal elements of the inverse of a symmetric matrix. These values are used in the calculation of the Cramer-Rao bounds.

Listing:

```
FORTRAN IV      V02.5-2     Sun 17-Jan-82 00:41:36              PAGE 001

0001            SUBROUTINE DIAGIN(A)
          C
          C     - - - - - - - - - - - - - - - - - - - - - - - - - -
          C     THIS SUBROUTINE FINDS THE DIAGONAL ELEMENTS OF THE
          C     INVERSE OF A SYMMETRIC MATRIX, [A].
          C     - - - - - - - - - - - - - - - - - - - - - - - - - -
          C
0002            COMMON /SUBWRT/ ISUB
0003            DIMENSION A(25,25)
0004            IF (ISUB.GE.3) PRINT 2001
0006       2001 FORMAT('  SUBROUTINE DIAGIN')
          C
          C     - - - - - - - - - - - - - - - - - - - - - - - - - -
          C
          C****************** PERFORM MATRIX MANIPULATIONS
0007            CALL REDUCE(A)
0008            N       = A(25,1)
0009            NM1     = N-1
0010            DO 90 I=1,NM1
0011            A(I,I) = 1./A(I,I)
0012            IP1     = I+1
0013            DO 90 J=IP1,N
0014         90 A(I,I) = A(I,I)+A(J,I)**2/A(J,J)
0015            A(N,N) = 1./A(N,N)
0016            RETURN
0017            END
```

ORIGINAL PAGE IS
OF POOR QUALITY

A.8) MINC TIME HISTORY PLOTTINC (HARD COPY)

Description: The HPLOT program is the executive for the hard copy plotting routine. This program reads the measured and predicted time histories and calls either the LPLOT or DPLOT routines to plot longitudinal or lateral-directional cases, respectively.

Listing:

```
RT-11 LINK  V06.01C      Load Map          Sat 27-Feb-82 05:08:17
HPLOT .SAV       Title:  HPLOT   Ident:   FORV02

Section  Addr   Size     Global  Value    Global  Value    Global  Value

 . ABS.  000000 001000    (RW,I,GBL,ABS,OVR)
                         $USRSW  0^0000   $RF2A1  000000   .VIR    000000
                         $NLCHN  000006   $HKDWR  000006   $SYSV$  000012
                         $WASIZ  000152   $LRECL  000210
 $OHAND  001000 000106    (RW,I,GBL,REL,CON)
                         $OVRH   001002   O$REAT  001024   O$DONE  001036
                         $ODF1   001102   $ODF2   001104
 $OTABL  001106 000034    (RW,D,GBL,REL,OVR)
 OTS$I   001142 024374    (RW,I,LCL,REL,CON)
                         $$OTSI  001142   AINT    001142   $INTR   001160
                         $CVTFB  001244   $CVTFI  001244   $CVTCB  001260
                         $CVTCI  001260   $CVTDB  001260   $CVTDI  001260
                         CIC$    001272   CID$    001272   CLC$    001272
                         CLD$    001272   $DI     001272   CIF$    001302
                         CLF$    001302   $RI     001302   CIL$    001410
                         CLI$    001414   $CVTIF  001416   $CVTIC  001432
                         $CVTID  001432   CCI$    001444   CDI$    001444
                         $IC     001444   $ID     001444   CFI$    001460
                         $IR     001460   RCI$    001544   GCO$    002560
                         FCO$    002566   ECO$    002572   DCO$    002600
                         ADF$IS  003522   $ADDF   003530   ADF$SS  003542
                         $ADR    003542   ADF$PS  003546   ADF$MS  003552
                         SUF$IS  003566   $SUBF   003574   SUF$SS  003606
                         $SBR    003606   SUF$PS  003612   SUF$MS  003616
                         DIF$PS  003632   DIF$MS  003636   DIF$IS  003650
                         $DIVF   003656   DIF$SS  003670   $DVR    003670
                         MUF$PS  003674   MUF$MS  003700   NUF$IS  003712
                         $MULF   003720   MUF$SS  003732   $MLR    003732
                         $OTI    003764   $$OTI   003766   $SETOP  004204
                         $$SET   005750   COS     006244   SIN     006300
                         $INITI  006636   ABS     006754   $CLOSE  006772
                         OCI$    007550   ICI$    007556   $ECI    007572
                         OCO$    007752   ICO$    007760   $DUMPL  010156
                         ENC$    010304   $ENC    010310   DEC$    010316
                         $DEC    010322   END$    010506   ERR$    010520
                         $END    010532   $ERR    010550   $OPNER  010572
                         $CHKER  010630   $IOEXI  010654   $EOL    010722
                         EOL$    010724   $ERRTB  011040   $ERRS   011145
                         $FCHNL  014706   $FIO    015550   $$FIO   015554
                         $GETFI  016720   $GETRE  016756   $TTYIN  017032
                         IFR$    017166   $IFR    017172   $$IFR   017176
                         IFR$$   017230   IFW$    017252   $IFW    017256
                         $$IFW   017262   IFW$$   017320   ILW$    017370
                         $ILW    017374   TVS$    017514   $TVS    017516
                         $$OPCL  020330   $$$ERR  020442   $$$DIS  020464
                         $OSTMI  020566   $OSTM   020572   $PUTRE  021722
                         IRR$    022230   $IRR    022234   IRW$    022260
                         $IRW    022264   $GETIN  022640   $SETIN  022676
                         DEF$    023004   $DEF    023010   $PUTBL  023104
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
                          $GETBL   023314  $EOFIL   023500  $EOF2    023514
                          SAVRG$   023534  THRD$    023712  SAVR4$   023714
                          $STPS    024000  STP$     024006  $STP     024006
                          FOO$     024012  $EXIT    024032  $OTIS    024156
                          $$OTIS   024160  IUR$     024300  $IUR     024304
                          IUW$     024450  $IUW     024454  TVL$     025270
                          $TVL     025270  TVF$     025276  $TVF     025276
                          TVD$     025304  $TVD     025304  TVQ$     025312
                          $TVQ     025312  TVP$     025320  $TVP     025320
                          TVI$     025326  $TVI     025326  $WAIT    025462
                          $VRINT   025524
OTS$P     025536 000054   (RW,D,GBL,REL,OVR)
SYS$I     025612 004502   (RW,I,LCL,REL,CON)
                          IB$ERM   025612  IB$ROU   026572  IB$INI   027034
                          IB$WAT   027042  IB$UNI   027074  IB$PRI   027454
                          IB$SEC   027476  IB$LSN   027542  IBRECV   030262
                          IBARCV   030272  IBSEND   030640  IBASND   030650
                          IBSEOI   030662  IB$TLK   031132  IBUNL    031500
                          IB$UNL   031554  IBUNT    031632  IB$UNT   031706
                          IB$ERR   031764  IGETC    032162  STRPAD   032234
USER$I    032314 000000   (RW,I,LCL,REL,CON)
$CODE     032314 020176   (RW,I,LCL,REL,CON)
                          $$OTSC   032314  PSC      036312  PCLR     036356
                          SCL      036502  PEN      037340  PLT      037356
                          IPLT     040206  LBL      041012  CSIZ     041362
                          CPLT     042732  XAX      043074  YAX      046320
                          INTGET   051636
OTS$O     052512 001036   (RW,I,LCL,REL,CON)
                          $$OTSO   052512  $OPEN    052512
SYS$O     053550 000000   (RW,I,LCL,REL,CON)
$DATAP    053550 004732   (RW,D,LCL,REL,CON)
OTS$D     060502 000052   (RW,D,LCL,REL,CON)
                          NHCLN$   060506
OTS$S     060554 000004   (RW,D,LCL,REL,CON)
                          $AOTS    060554
SYS$S     060560 000206   (RW,D,LCL,REL,CON)
                          IB$1ST   060560  IB$DEV   060606  $SYSLB   060762
                          $LOCK    060764  $CRASH   060765
$DATA     060766 032406   (RW,D,LCL,REL,CON)
USER$D    113374 000000   (RW,D,LCL,REL,CON)
.$$$$.    113374 000000   (RW,D,GBL,REL,OVR)
ADDRES    113374 000002   (RW,D,GBL,REL,OVR)
XYSCLP    113376 000020   (RW,D,GBL,REL,OVR)
SCALES    113416 000020   (RW,D,GBL,REL,OVR)
PLOT      113436 000020   (RW,D,GBL,REL,OVR)
IB.ERR    113456 000024   (RW,I,GBL,REL,OVR)
IB.DR     113502 000026   (RW,I,GBL,REL,OVR)
IBCSR     113530 000004   (RW,D,GBL,REL,OVR)
IB.LSN    113534 000010   (RW,I,GBL,REL,OVR)
IBLSN     113544 000042   (RW,D,GBL,REL,OVR)
IB.TLK    113606 000010   (RW,I,GBL,REL,OVR)
IBTRM     113616 000014   (RW,D,GBL,REL,OVR)
IBTLK     113632 000006   (RW,D,GBL,REL,OVR)
Segment size = 113640 = 19408. words

Overlay region  000001   Segment 000001
OTS$I     113642 000000   (RW,I,LCL,REL,CON)
SYS$I     113642 000000   (RW,I,LCL,REL,CON)
USER$I    113642 000000   (RW,I,LCL,REL,CON)
$CODE     113642 003652   (RW,I,LCL,REL,CON)
                          LPLOT @ 113642
OTS$O     117514 000000   (RW,I,LCL,REL,CON)
SYS$O     117514 000000   (RW,I,LCL,REL,CON)
$DATAP    117514 003376   (RW,D,LCL,REL,CON)
OTS$D     123112 000000   (RW,D,LCL,REL,CON)
OTS$S     123112 000000   (RW,D,LCL,REL,CON)
SYS$S     123112 000000   (RW,D,LCL,REL,CON)
$DATA     123112 000302   (RW,D,LCL,REL,CON)
USER$D    123414 000000   (RW,D,LCL,REL,CON)
Segment size = 007552 = 1973.  words
```

```
Overlay region  000001  Segment 000002
 OTS$I     113642 000000   (RW,I,LCL,REL,CON)
 SYS$I     113642 000000   (RW,I,LCL,REL,CON)
 USER$I    113642 000000   (RW,I,LCL,REL,CON)
 $CODE     113642 004234   (RW,I,LCL,REL,CON)
                          DPLOT @ 113642
 OTS$O     120076 000000   (RW,I,LCL,REL,CON)
 SYS$O     120076 000000   (RW,I,LCL,REL,CON)
 $DATAP    120076 003544   (RW,D,LCL,REL,CON)
 OTS$D     123642 000000   (RW,D,LCL,REL,CON)
 OTS$S     123642 000000   (RW,D,LCL,REL,CON)
 SYS$S     123642 000000   (RW,D,LCL,REL,CON)
 $DATA     123642 000322   (RW,D,LCL,REL,CON)
 USER$D    124164 000000   (RW,D,LCL,REL,CON)
 Segment size = 010322 = 2153.  words

Transfer address = 032314, High limit = 124162 = 21561. words
```

```
FORTRAN IV      V02.5-2    Sat 27-Feb-82 05:06:00             PAGE 001

0001            PROGRAM HPLOT
          C.... PROGRAM TO PRODUCE HARD COPY PLOTS OF LONGITUDINAL
          C.... AND LATERAL-DIRECTIONAL STATE TIME HISTORIES
0002            DIMENSION Z(7,200),U(2,200),ZP(7,200)
0003            LOGICAL*1 LONG,LATR,EXTRA,MORE
0004            BYTE ANS,NAME(15)
0005            DATA LONG,LATR,EXTRA,MORE /4*.FALSE./
0006            DATA TIME /0./
0007            DATA NAME(15) /0/
0008        100 CONTINUE
          C.... ASK FOR SAMPLE PER SECOND DATA
0009            TYPE 101
0010        101 FORMAT('$',10X,'Input the integral # of sps for the test data: ')
0011            ACCEPT 102,ISPS
0012        102 FORMAT(1I10)
0013            TIMINC = FLOAT(200/ISPS)
0014            DELT   = 1./(FLOAT(ISPS))
          C.... ASK FOR PLOTTER SPEED
0015            TYPE 103
0016        103 FORMAT('$',10X,'Input the plotter speed in cm/sec (1-25): ')
0017            ACCEPT 102,ISPD
          C.... ASK IF LONG AUTO-SEQUENCE
0018            TYPE 1
0019          1 FORMAT('$',10X,'Longitudianl auto-sequence? (Y or N) ')
0020            ACCEPT 2, ANS
0021          2 FORMAT(1A1)
0022            IF (ANS.EQ.'Y') LONG = .TRUE.
0024            IF (LONG) GO TO 3
          C.... ASK IF LATR AUTO-SEQUENCE
0026            TYPE 4
0027          4 FORMAT('$',10X,'Lateral-dirctional auto-sequence? (Y or N) ')
0028            ACCEPT 2, ANS
0029            IF (ANS.EQ.'Y') LATR = .TRUE.
0031            IF (.NOT.LATR) GO TO 100
          C.... OPEN INPUT DATA FILES
0033          3 CONTINUE
0034            TYPE 5
0035          5 FORMAT('$',10X,'Enter the measured data file name: ')
0036            ACCEPT 6,(NAME(I),I=1,14)
0037          6 FORMAT(14A1)
0038            OPEN (UNIT=1,NAME=NAME,TYPE='OLD',ACCESS='SEQUENTIAL',
              .        READONLY,BUFFERCOUNT=1,FORM='UNFORMATTED')
          C.... CHECK FOR EXTRA DATA
0039            TYPE 8
0040          8 FORMAT('$',10X,'Does this file contain extra data? (Y or N) ')
0041            ACCEPT 2,ANS
0042            IF (ANS.EQ.'Y') EXTRA = .TRUE.
```

```
C.... CLEAR OUT OLD DATA FILE NAME
      DO 10 I=1,14
      NAME(I) = ' '
   10 CONTINUE
      TYPE 7
    7 FORMAT('$',10X,'Enter the predicted data file name: ')
      ACCEPT 6,(NAME(I),I=1,14)
      OPEN (UNIT=2,NAME=NAME,TYPE='OLD',ACCESS='SEQUENTIAL',
     .     READONLY,BUFFERCOUNT=1,FORM='UNFORMATTED')
   25 CONTINUE
C.... INITIALIZE ARRAYS
      DO 26 II=1,200
      DO 27 IJ=1,7
      Z(IJ,II) =-1.E10
      ZP(IJ,II) = 0.
   27 CONTINUE
      U(1,II) = 0.
      U(2,II) = 0.
   26 CONTINUE
C.... READ DATA INTO ARRAYS
      DO 1000 J=1,200
      JEND = J
      IF (LONG) READ (1,END=2000) Z(4,J),Z(1,J),Z(7,J),Z(6,J),U(1,J),
     .                          A,A,A,A,A,A
      IF (LONG.AND.EXTRA) READ (1,END=2000) Z(2,J),Z(3,J),Z(5,J),
     .                                      A,A,A
      IF (LONG) U(2,J) = 0.
      IF (LATR) READ (1,END=2000) A,A,A,A,A,Z(4,J),Z(1,J),
     .                          Z(7,J),Z(2,J),U(1,J),U(2,J)
      IF (LATR.AND.EXTRA) READ (1,END=2000) A,A,A,Z(3,J),
     .                                      Z(5,J),Z(6,J)
 1000 CONTINUE
C.... SET MORE EQUAL TO TRUE
      MORE = .TRUE.
 2000 CONTINUE
C.... IF T = 0. THEN SET THE INITIAL CONDITIONS
      IF (TIME.NE.0.) GO TO 1988
      DO 1999 J=1,7
      ZP(J,1) = (Z(J,1)+Z(J,2))/2.
      IF (ZP(J,1).LT.-1.E9) ZP(J,1) = 0.0
      ZP(J,2) = ZP(J,1)
 1999 CONTINUE
      IF (JEND.GT.200) JEND = 200
      DO 3001 J=3,JEND
      READ (2,END=2002) (ZP(IA,J),IA=1,7)
 3001 CONTINUE
      GO TO 2002
 1988 CONTINUE
      IF (JEND.GT.200) JEND = 200
      DO 2001 J=1,JEND
      READ (2,END=2002) (ZP(IA,J),IA=1,7)
 2001 CONTINUE
 2002 CONTINUE
C.... PLOT DATA
      IF (LONG) CALL LPLOT(Z,ZP,U,TIME,TIMINC,DELT,ISPD)
      IF (LATR) CALL DPLOT(Z,ZP,U,TIME,TIMINC,DELT,ISPD)
C.... IF MORE IS .TRUE. DISPLAY CHANGE PAPER MESSAGE
C.... AND GET READY FOR MORE DATA
      IF (.NOT.MORE) GO TO 3535
      TYPE 2525
 2525 FORMAT('$',10X,'MORE DATA ON FILE: (Continue when ready)')
      ACCEPT 2,ANS
C.... RESET MORE DATA FLAG
      MORE = .FALSE.
C.... INCREMENT THE TIME COUNTER BY TIMINC
      TIME = TIME+TIMINC
      GO TO 25
 3535 CONTINUE
      STOP
      END
```

Subroutine LPLOT

Description: The LPLOT subroutine plots the longitudinal time history traces.

Listing:

```
FORTRAN IV      V02.5-2    Sat 27-Feb-82 04:40:18              PAGE 001

           SUBROUTINE LPLOT (Z,ZP,U,TIMEO,TIMINC,DELT,IOD)
     C.... SUBROUTINE TO DO HARD COPY PLOTS LONGITUDINAL INPUT
           COMMON /ADDRES/ INSTR
           DIMENSION Z(7,200),ZP(7,200),U(2,200)
           REAL LOGAIN(9)
           BYTE ANS,MESSAG(6)
           DATA LOGAIN /57.3,1.00,57.3,57.3,57.3,1.00,1.00,57.3,0.00/
           DATA MESSAG /'V','S','0','0',';',0/
     C.... DEFINE TIC SIZE
           TIC    = TIMINC/10.
     C.... SET UP PLOTTER
           CALL PSC (5)
           CALL PCLR
     C.... ENCODE SPEED FOR OUTPUT TO PLOTTER
           ENCODE (2,1001,MESSAG(3)) ISPD
      1001 FORMAT(I2)
     C.... CHANGE ENCODED BLANKS BACK TO ZEROS
           IF (MESSAG(3).EQ.' ') MESSAG(3)='0'
     C.... SEND SPEED TO PLOTTER (ASSUMED AT ADDRESS 5)
           CALL IBSEND (MESSAG,5,5)
           CALL PEN
           IPEND = 200
     C.... FIND END OF CONTROL INPUT DATA
           DO 101 IC=1,200
           IF (U(1,201-IC).NE.0.) GO TO 102
           IPEND = 200-IC
       101 CONTINUE
     C.... LEAVE LOOP EARLY IPEND POINTS TO THE LAST DATA POINT
       102 CONTINUE
     C.... LONGITUDINAL
     C     1. PITCH RATE
     C     2. AIRSPEED
     C     3. ANGLE OF ATTACK
     C     4. PITCH ATTITUDE
     C     5. PITCH RATE ACCEL.
     C     6. LONGITUDINAL ACCEL.
     C     7. NORMAL ACCEL.
     C     8. ELEVATOR PSN.
     C     9.    * * *    (BLANK)
     C.... LONGITUDINAL AXES AND LABELS
           DO 500 KT=1,8
     C.... SET LINE TYPE TO CONTINUOUS LINES
           CALL IBSEND ('LT;',-1,5)
           IF(KT.EQ.1) GO TO 601
           IF(KT.EQ.2) GO TO 602
           IF(KT.EQ.3) GO TO 603
           IF(KT.EQ.4) GO TO 604
           IF(KT.EQ.5) GO TO 605
           IF(KT.EQ.6) GO TO 606
           IF(KT.EQ.7) GO TO 607
           IF(KT.EQ.8) GO TO 608
       601 CONTINUE
           CALL IBSEND ('IP 2000,5640,9500,6640;',-1,INSTR)
           CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
          .          'TIME IN SECONDS     ')
           CALL YAX(10.,4.,0.,0.,4.0,-20.0,10.,2.,-1.,
          .          'Q IN DEG/SEC        ')
           GO TO 640
```

```
  602 CONTINUE
      CALL IBSEND ('IP 2000,4440,9500,5440;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-10.0,5.,2.,-1.,
     .         'U IN FT/SEC         ')
      GO TO 640
  603 CONTINUE
      CALL IBSEND ('IP 2000,3240,9500,4240;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-10.0,5.,2.,-1.,
     .         'ALPHA IN DEG        ')
      GO TO 640
  604 CONTINUE
      CALL IBSEND ('IP 2000,2040,9500,3040;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-10.0,5.,2.,-1.,
     .         'THA IN DEG          ')
      GO TO 640
  605 CONTINUE
      TYPE 752
  752 FORMAT(10X,'INSERT NEW PAPER FOR PLOTS; (cr when finished)')
      ACCEPT 180,ANS
  180 FORMAT(A1)
      CALL IBSEND ('IP 2000,5640,9500,6640;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-50.0,25.,2.,-1.,
     .         'QDOT IN DEG/S**2    ')
      GO TO 640
  606 CONTINUE
      CALL IBSEND ('IP 2000,4440,9500,5440;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-0.20,.10,2.,-1.,
     .         'AX IN G             ')
      GO TO 640
  607 CONTINUE
      CALL IBSEND ('IP 2000,3240,9500,4240;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-2.00,0.5,2.,-1.,
     .         'AZ IN G             ')
      GO TO 640
  608 CONTINUE
      CALL IBSEND ('IP 2000,2040,9500,3040;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-10.0,5.,2.,-1.,
     .         'DE IN DEG           ')
  640 CONTINUE
C.... PLOT DATA
      TIME = TIMEO
C.... SET LINE TYPE TO CONTINUOUS LINES
      CALL IBSEND ('LT;',-1,5)
      DO 503 IA=1,IPEND
C.... CHECK TO SEE IF CHANNEL IS MEASURED; IF NOT DON'T PLOT
      IF (Z(KT,IA).LT.-1.E9) GO TO 503
      TIME = TIME+DELT
      IF (KT.NE.8) CALL PLT (TIME,LOGAIN(KT)*Z(KT,IA),-2)
      IF (KT.EQ.8) CALL PLT (TIME,LOGAIN(KT)*U(1,IA),-2)
  503 CONTINUE
      CALL PEN
C.... IF CONTROL TYPE PLOT GO TO BOTTOM OF DO LOOP
      IF (KT.EQ.8) GO TO 500
      TIME = TIMEO
C.... SET LINE TYPE TO DASHED LINES
      CALL IBSEND('LT2,2;',-1,5)
      DO 504 IA=1,IPEND
```

```
      TIME = TIME+DELT
      CALL PLT (TIME,LDGAIN(KT)*ZP(KT,IA),-2)
  504 CONTINUE
      CALL PEN
  500 CONTINUE
      RETURN
      END
```

Subroutine DPLOT

Description: The DPLOT subroutine plots the lateral-directional time history traces.

Listing:

```
FORTRAN IV      V02.5-2    Sat 27-Feb-82 04:41:00                 PAGE 001

      SUBROUTINE DPLOT (Z,ZP,U,TIMEO,TIMINC,DELT,ISPD)
C.... SUBROUTINE TO DO HARD COPY PLOTS LATERAL-DIRECTIONAL
      COMMON /ADDRES/ INSTR
      DIMENSION Z(7,200),ZP(7,200),U(2,200)
      REAL LDGAIN(9)
      BYTE ANS,MESSAG(6)
      DATA LDGAIN /57.3,57.3,57.3,57.3,57.3,57.3,1.00,57.3,57.3/
      DATA MESSAG /'V','S','0','0','!',0/
C.... DEFINE TIC SIZE
      TIC    = TIMINC/10.
C.... SET UP PLOTTER
      CALL PSC (5)
      CALL PCLR
C.... ENCODE SPEED FOR OUTPUT TO PLOTTER
      ENCODE (2,1001,MESSAG(3)) ISPD
 1001 FORMAT(I2)
C.... CHANGE ENCODED BLANKS BACK TO ZEROS
      IF (MESSAG(3).EQ.' ') MESSAG(3)='0'
C.... SEND SPEED TO PLOTTER (ASSUMED AT ADDRESS = 5)
      CALL IBSEND (MESSAG,5,5)
      CALL PEN
      IPEND = 200
C.... FIND END OF CONTROL INPUT DATA
      DO 101 IC=1,200
      IF (U(1,201-IC).NE.0.) GO TO 102
      IPEND = 200-IC
  101 CONTINUE
C.... LEAVE LOOP EARLY IPEND POINTS TO THE LAST DATA POINT
  102 CONTINUE
  752 FORMAT(10X,'INSERT NEW PAPER FOR PLOTS! (cr when finished)')
  180 FORMAT(A1)
C.... LATERAL DIRECTIONAL
C      1. ROLL RATE
C      2. YAW RATE
C      3. SIDESLIP ANGLE
C      4. BANK ANGLE
C      5. ROLL RATE ACCEL.
C      6. YAW RATE ACCEL.
C      7. LONGITUDINAL ACCEL.
C      8. AILERON DEFLECTION
C      9. RUDDER DEFLECTION
C.... LATERAL DIRECTIONAL AXES AND LABELS
      DO 300 KT=1,9
C.... SET LINE TYPE TO CONTINUOUS LINES
      CALL IBSEND ('LT!',-1,5)
```

```
      IF(KT.EQ.1) GO TO 701
      IF(KT.EQ.2) GO TO 702
      IF(KT.EQ.3) GO TO 703
      IF(KT.EQ.4) GO TO 704
      IF(KT.EQ.5) GO TO 705
      IF(KT.EQ.6) GO TO 706
      IF(KT.EQ.7) GO TO 707
      IF(KT.EQ.8) GO TO 708
      IF(KT.EQ.9) GO TO 709
701   CONTINUE
      CALL IBSEND ('IP 2000,5640,9500,6640;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-20.0,10.,2.,-1.,
     .         'P IN DEG/SEC        ')
      GO TO 740
702   CONTINUE
      CALL IBSEND ('IP 2000,4440,9500,5440;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-20.0,10.,2.,-1.,
     .         'R IN DEG/SEC        ')
      GO TO 740
703   CONTINUE
      CALL IBSEND ('IP 2000,3240,9500,4240;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-10.0,5.,2.,-1.,
     .         'BTA IN DEG          ')
      GO TO 740
704   CONTINUE
      CALL IBSEND ('IP 2000,2040,9500,3040;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-10.0,5.,2.,-1.,
     .         'PHI IN DEG          ')
      GO TO 740
705   CONTINUE
      TYPE 752
      ACCEPT 180,ANS
      CALL IBSEND ('IP 2000,5640,9500,6640;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-50.0,25.,2.,-1.,
     .         'PDOT IN DEG/S**2    ')
      GO TO 740
706   CONTINUE
      CALL IBSEND ('IP 2000,4440,9500,5440;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-50.0,25.,2.,-1.,
     .         'RDOT IN DEG/S**2    ')
      GO TO 740
707   CONTINUE
      CALL IBSEND ('IP 2000,3240,9500,4240;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-.20,0.10,2.,-1.,
     .         'AY IN G             ')
      GO TO 740
708   CONTINUE
      CALL IBSEND ('IP 2000,2040,9500,3040;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .         'TIME IN SECONDS     ')
      CALL YAX(10.,4.,0.,0.,4.0,-10.0,5.,2.,-1.,
     .         'DA IN DEG           ')
      GO TO 740
709   CONTINUE
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      CALL IBSEND ('IP 2000,840,9500,1840;',-1,INSTR)
      CALL XAX(10.,4.,0.,2.,10.,TIMEO,TIC,5.,-1.,
     .            'TIME IN SECONDS       ')
      CALL YAX(10.,4.,0.,0.,4.0,-10.0,5.,2.,-1.,
     .            'DR IN DEG             ')
  740 CONTINUE
C.... PLOT DATA
      TIME = TIMEO
C.... SET LINE TYPE TO CONTINUOUS LINES
      CALL IBSEND ('LT;',-1,5)
      DO 803 IA=1,IPEND
C.... CHECK TO SEE IF CHANNEL IS MEASURED; IF NO DON'T PLOT
      IF (Z(KT,IA).LT.-1.E9) GO TO 803
      TIME = TIME+DELT
      IF ((KT.NE.8).AND.(KT.NE.9))
     .   CALL PLT (TIME,LDGAIN(KT)*Z(KT,IA),-2)
      IF (KT.EQ.8) CALL PLT (TIME,LDGAIN(KT)*U(1,IA),-2)
      IF (KT.EQ.9) CALL PLT (TIME,LDGAIN(KT)*U(2,IA),-2)
  803 CONTINUE
      CALL PEN
C.... IF CONTROL TYPE PLOT GO TO BOTTOM OF DO LOOP
      IF ((KT.EQ.8).OR.(KT.EQ.9)) GO TO 300
      TIME = TIMEO
C.... SET LINE TYPE TO DASHED LINES
      CALL IBSEND('LT2,2;',-1,5)
      DO 804 IA=1,IPEND
      TIME = TIME+DELT
      CALL PLT (TIME,LDGAIN(KT)*ZP(KT,IA),-2)
  804 CONTINUE
      CALL PEN
  300 CONTINUE
      RETURN
      END
```

A.9) MINC PLOTTER LIBRARY ROUTINES

Description: The PLTLIB library is a library of subroutines for performing various hard copy graphics commands on the Hewlett Packard 7225B plotter. These routines are all FORTRAN programs, and they communicate to the plotter using the IEEE 488 (GPIB) interface bus. The routines IBSEND and IBRECV are the MINC routines for data output and input, respectively.

Listing:

```
FORTRAN IV      V02.5-2     Tue 10-Nov-81 01:00:56              PAGE 001

           SUBROUTINE PSC (IDD)
      C         10-NOV-81
      C******************* SUBROUTINE TO SET THE GP-IB BUSS ADDRESS
      C******************* CODE TO BE USED FOR I/O FOR PLOTTER COMMANDS
      C******************* AND TO CLEAR THE ARRAYS USED FOR SPECIAL LABELS
           COMMON /ADDRES/ INSTR
           COMMON /XYSCLF/ SCLP1,SCLP2,SCLP3,SCLP4
           DATA INSTR /5/
           DATA SCLP1,SCLP2,SCLP3,SCLP4 /4*0./
      C******************* SET ADDRESS FOR PLOTTER COMMANDS
           IF ((IDD.GT.0).AND.(IDD.LT.31)) INSTR = IDD
           RETURN
           END
```

```
FORTRAN IV      V02.5-2     Tue 10-Nov-81 01:03:33              PAGE 001

           SUBROUTINE PCLR
      C         10-NOV-81
      C******************* SUBROUTINE TO INITIALIZE THE PLOTTER
      C******************* TO ITS DEFAULT STATUS (LEAVING P1 & P2 UNAFFECTED)
           COMMON /ADDRES/ INSTR
           BYTE MESSAG(10)
           LOGICAL*1 OK,EOS
      C******************* INITIALIZE PLOTTER
           CALL IBSEND ('DF;',-1,INSTR)
      C******************* CLEAR ERROR IF ANY EXIST
           CALL IBSEND ('OE;',-1,INSTR)
           CALL IBRECV (MESSAG,25,INSTR)
           IPTR       = 1
           IEND       = 10
           IERR       = INTGET(MESSAG,IPTR,IEND,OK,EOS)
           IF (IERR.EQ.0) RETURN
           TYPE 10, IERR
        10 FORMAT(10X,'HP PLOTTER ERROR NUMBER ',I3,' OCCURED.',/,
         .         10X,'SEE PAGE 47 OF MANUAL FOR ERROR DEFINATIONS.')
           RETURN
           END
```

```
FORTRAN IV      V02.5-2    Tue 10-Nov-81 01:07:01              PAGE 001

0001          SUBROUTINE SCL (X1,X2,Y1,Y2)
      C          10-NOV-81
      C****************** SUBROUTINE TO SET THE SCALING POINTS P1 AND P2
      C****************** TO BE USED IN DEFINING THE USER SCALE
0002          COMMON /ADDRES/ INSTR
0003          COMMON /SCALES/ XP1,XP2,YP1,YP2
0004          COMMON /PLOT  / XRATIO,YRATIO,XKNST,YKNST
0005          BYTE MESSAG(25)
0006          LOGICAL*1 OK,EOS
      C****************** READ POSITION OF P1 AND P2
0007     2525 CONTINUE
0008          ENCODE(25,2424,MESSAG(1)) '
0009     2424 FORMAT(1A25)
0010          CALL IBSEND ('OP;',-1,INSTR)
0011          CALL IBRECV (MESSAG,25,INSTR)
0012          IPTR=1
0013          IEND=25
0014          IP1X         = INTGET(MESSAG,IPTR,IEND,OK,EOS)
0015          IF (.NOT.OK) GO TO 2525
0017          IF (OK) IPTR = IPTR+1
0019          IP1Y         = INTGET(MESSAG,IPTR,IEND,OK,EOS)
0020          IF (.NOT.OK) GO TO 2525
0022          IF (OK) IPTR = IPTR+1
0024          IP2X         = INTGET(MESSAG,IPTR,IEND,OK,EOS)
0025          IF (.NOT.OK) GO TO 2525
0027          IF (OK) IPTR = IPTR+1
0029          IP2Y         = INTGET(MESSAG,IPTR,IEND,OK,EOS)
0030          IF (.NOT.OK) GO TO 2525
      C****************** SET SCALES AND PLOT INFORMATION
0032          XNUM         = FLOAT (IP2X-IP1X)
0033          YNUM         = FLOAT (IP2Y-IP1Y)
0034          XP1          = X1
0035          XP2          = X2
0036          YP1          = Y1
0037          YP2          = Y2
0038          XRATIO       = XNUM/(XP2-XP1)
0039          YRATIO       = YNUM/(YP2-YP1)
0040          XKNST        = FLOAT(IP1X)-XP1*XRATIO
0041          YKNST        = FLOAT(IP1Y)-YP1*YRATIO
      C****************** SET INPUT WINDOW TO DEFAULT VALUE
0042          CALL IBSEND ('IW;',-1,INSTR)
0043          RETURN
0044          END
```

```
FORTRAN IV      V02.5-2    Tue 10-Nov-81 01:09:42              PAGE 001

0001          SUBROUTINE PEN
      C          10-NOV-81
      C****************** SUBROUTINE TO RAISE PEN
0002          COMMON /ADDRES/ INSTR
      C****************** RAISE PEN
0003          CALL IBSEND('PU;',-1,INSTR)
0004          RETURN
0005          END
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
FORTRAN IV      V02.5-2     Tue 10-Nov-81 01:16:04                PAGE 001

0001          SUBROUTINE PLT (XCORD,YCORD,I)
        C          10-NOV-81
        C****************** SUBROUTINE TO PLOT THE USER X,Y COORDINATE
        C****************** SPECIFIED (AFTER CONVERTING TO ABSOLUTE UNITS)
        C****************** UNDER USER PEN CONTROL
0002          COMMON /ADDRES/ INSTR
0003          COMMON /PLOT  / XRATIO,YRATIO,XKNST,YKNST
0004          BYTE MESSAG(25)
        C****************** CHECK WHETHER I IS AN EVEN OR ODD INTEGER
0005          IE           = (I/2)*2
        C****************** IF I > 0; THEN CHANGE PEN BEFORE MOVING
0006          IF ((I.GT.0).AND.(I.EQ.IE)) CALL IBSEND ('PD;',-1,INSTR)
0008          IF ((I.GT.0).AND.(I.NE.IE)) CALL IBSEND ('PU;',-1,INSTR)
        C****************** PLOT X,Y DATA
0010          XSCL         = XCORD*XRATIO+XKNST
0011          YSCL         = YCORD*YRATIO+YKNST
0012          IF (XSCL.LT.32767.) GO TO 500
0014          TYPE 998
0015      998 FORMAT('  X-POINT IS TO LARGE TO PLOT')
0016          RETURN
0017      500 CONTINUE
0018          IF (YSCL.LT.32767.) GO TO 501
0020          TYPE 999
0021      999 FORMAT('  Y-POINT IS TO LARGE TO PLOT')
0022          RETURN
0023      501 CONTINUE
0024          IXSCL        = XSCL
0025          IYSCL        = YSCL
0026          ENCODE(25,1000,MESSAG(1)) 'PA     ,     ;            '
0027     1000 FORMAT(1A25)
0028          ENCODE(5,1001,MESSAG(3)) IXSCL
0029     1001 FORMAT(I5)
0030          ENCODE(5,1001,MESSAG(9)) IYSCL
0031          CALL IBSEND (MESSAG,14,INSTR)
        C****************** IF I < 0; THEN CHANGE PEN AFTER MOVING
        C****************** IF I >= 0; NO CHANGES ARE MADE HERE
0032          IF (I.GE.0) RETURN
0034          IF (I.EQ.IE) CALL IBSEND ('PD;',-1,INSTR)
0036          IF (I.NE.IE) CALL IBSEND ('PU;',-1,INSTR)
0038          RETURN
0039          END
```

```
FORTRAN IV      V02.5-2     Tue 10-Nov-81 01:21:56                PAGE 001

0001          SUBROUTINE IPLT (XINC,YINC,I)
        C          10-NOV-81
        C****************** SUBROUTINE TO PLOT THE USER DELTA X,Y COORDINATE
        C****************** SPECIFIED (AFTER CONVERTING TO ABSOLUTE UNITS)
        C****************** UNDER USER CONTROL OF PEN
0002          COMMON /ADDRES/ INSTR
0003          COMMON /PLOT  / XRATIO,YRATIO,XKNST,YKNST
0004          BYTE MESSAG(25)
        C****************** CHECK WHETEHR I IS AN EVEN OR ODD INTEGER
0005          IE           = (I/2)*2
        C****************** IF I > 0; THEN CHANGE PEN BEFORE MOVING
0006          IF ((I.GT.0).AND.(I.EQ.IE)) CALL IBSEND ('PD;',-1,INSTR)
0008          IF ((I.GT.0).AND.(I.NE.IE)) CALL IBSEND ('PU;',-1,INSTR)
        C****************** PLOT DELTA X,Y DATA
0010          XSCL         = XINC*XRATIO
0011          YSCL         = YINC*YRATIO
0012          IF (XSCL.LT.32767.) GO TO 500
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
            TYPE 998
        998 FORMAT('  X-POINT IS TO LARGE TO PLOT')
            RETURN
        500 CONTINUE
            IF (YSCL.LT.32767.) GO TO 501
            TYPE 999
        999 FORMAT('  Y-POINT IS TO LARGE TO PLOT')
            RETURN
        501 CONTINUE
            IXSCL        = XSCL
            IYSCL        = YSCL
            ENCODE(25,1000,MESSAG(1)) 'PR     ,     ;              '
       1000 FORMAT(1A25)
            ENCODE (5,1001,MESSAG(3)) IXSCL
       1001 FORMAT(I5)
            ENCODE (5,1001,MESSAG(9)) IYSCL
            CALL IBSEND (MESSAG,14,INSTR)
      C################### IF I < 0; THEN CHANGE PEN AFTER MOVING
      C################### IF I >= 0; NO CHANGES ARE MADE HERE
            IF (I.GE.0) RETURN
            IF (I.EQ.IE) CALL IBSEND ('PD;',-1,INSTR)
            IF (I.NE.IE) CALL IBSEND ('PU;',-1,INSTR)
            RETURN
            END
```

```
FORTRAN IV      V02.5-2     Tue 10-Nov-81 01:23:29              PAGE 001

            SUBROUTINE OFS (XINC,YINC)
      C          10-NOV-81
      C################### SUBROUTINE TO OFFSET THE ORGIN FROM ITS PRESENT
      C################### LOCATION TO A NEW LOCATION BY THE DELTA INCREMENTS
            COMMON /SCALES/ XP1,XP2,YP1,YP2
      C################### ADD INCREMENTS TO SCALES AND CALL SCL ROUTINE
            X1           = XP1-XINC
            X2           = XP2-XINC
            Y1           = YP1-YINC
            Y2           = YP2-YINC
            CALL SCL (X1,X2,Y1,Y2)
            RETURN
            END
```

```
FORTRAN IV      V02.5-2     Tue 10-Nov-81 01:27:58              PAGE 001

            SUBROUTINE LBL (ARRAY)
      C          10-NOV-81
      C################### SUBROUTINE TO LABEL CHARACTER ARRAYS ON THE
      C################### PLOTTER AT ARBITRARY X,Y LOCATIONS
            COMMON /ADDRES/ INSTR
            BYTE MESSAG(84),ARRAY(80)
            DATA MESSAG /84*' '/
      C################### COPY LABEL INTO MESSAG
            DO 20 I=1,80
            MESSAG(I+2)  = AR AY(I)
         20 CONTINUE
      C################### CHECK LABEL FOR ALL BLANK CHARACTERS
            DO 21 I=1,80
```

```
0009          IF (ARRAY(I).NE.' ') GO TO 22
0011       21 CONTINUE
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ TO GET HERE ALL CHARACTERS MUST BE BLANK
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ THEREFORE EXIT EARLY
0012          RETURN
0013       22 CONTINUE
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ PAD STRING TC GET 84 CHARACTERS
0014          CALL STRPAD(MESSAG,84)
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ MOVE LABEL TO FRONT OF ARRAY IF IT IS NOT THERE
0015       43 CONTINUE
0016          IF (MESSAG(3).NE.' ') GO TO 44
0018          DO 45 IC=3,83
0019          MESSAG(IC)  = MESSAG(IC+1)
0020       45 CONTINUE
0021          MESSAG(84)  = ' '
0022          GO TO 43
0023       44 CONTINUE
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ FIND FIRST NON-BLANK CHARACTER LOOKING FROM THE
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ BACK TO THE FRONT  (THIS BECOMES THE LAST CHAR)
0024          ISAVE       = 80
0025          DO 10 I=1,80
0026          IF (MESSAG(83-I).EQ.' ') GO TO 10
0028          ISAVE       = 81-I
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ LEAVE LOOP EARLY SINCE NON-BLANK IS FOUND
0029          GO TO 13
0030       10 CONTINUE
0031       13 CONTINUE
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ ADD LABEL INSTRUCTION TO MESSAG
0032          ENCODE(2,1000,MESSAG(1)) 'LB'
0033     1000 FORMAT(1A2)
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ ADD TERMINATION CHARACTER TO LABEL
0034          MESSAG(ISAVE+3) = 3
0035          MESSAG(ISAVE+4) = '$'
0036          ISAVE       = ISAVE+4
0037          CALL IBSEND (MESSAG,ISAVE,INSTR)
0038          RETURN
0039          END
```

```
FORTRAN IV      V02.5-2     Tue 10-Nov-81 01:29:32              PAGE 001

0001          SUBROUTINE CSIZ (H,AR,AOR,SL)
        C        10-NOV-81
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ SUBROUTINE TO SPECIFY THE SIZE AND SHAPE OF CHAR
0002          COMMON /ADDRES/ INSTR
0003          LOGICAL*1 OK,EOS
0004          BYTE MESSAG(25)
0005          DATA DGR /57.29578/
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ CONVERT ANGLES TO RADIAN UNITS
0006          AORR        = AOR/DGR
0007          SLR         = SL/DGR
        C♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦ READ POSITION OF P1 AND P2
0008     2525 CONTINUE
0009          ENCODE(25,1000,MESSAG(1)) '                              '
0010     1000 FORMAT(1A25)
0011          CALL IBSEND ('OP$',-1,INSTR)
0012          CALL IBRECV (MESSAG,25,INSTR)
0013          IPTR=1
0014          IEND=25
0015          IP1X        = INTGET(MESSAG,IPTR,IEND,OK,EOS)
0016          IF (.NOT.OK) GO TO 2525
0018          IF (OK) IPTR = IPTR+1
0020          IP1Y        = INTGET(MESSAG,IPTR,IEND,OK,EOS)
0021          IF (.NOT.OK) GO TO 2525
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      IF (OK) IPTR = IPTR+1
      IP2X         = INTGET(MESSAG,IPTR,IEND,OK,EOS)
      IF (.NOT.OK) GO TO 2525
      IF (OK) IPTR = IPTR+1
      IP2Y         = INTGET(MESSAG,IPTR,IEND,OK,EOS)
      IF (.NOT.OK) GO TO 2525
C******************* SET SCALES AND PLOT INFORMATION
      XNUM         = FLOAT (IP2X-IP1X)
      YNUM         = FLOAT (IP2Y-IP1Y)
C******************* CALCULATE PAPER RATIO
      PR           = XNUM/YNUM
C******************* AS A FUNCTION OF P1 & P2
      W            = (H/AR)/PR
      IF (W.GT.127.999) W = 127.999
      IF (H.GT.127.999) H = 127.999
      ENCODE(25,1000,MESSAG(1)) 'SR          ,          ;      '
      ENCODE(8,1001,MESSAG(3)) W
 1001 FORMAT(F8.3)
      ENCODE(8,1001,MESSAG(12)) H
      CALL IBSEND (MESSAG,20,INSTR)
C******************* SET ANGLE OF ROTATION
      RISE         = 100*SIN(AORR)
      RUN          = 100*COS(AORR)
      ENCODE(25,1000,MESSAG(1)) 'DI          ,          ;      '
      ENCODE(8,1001,MESSAG(3)) RUN
      ENCODE(8,1001,MESSAG(12)) RISE
      CALL IBSEND(MESSAG,20,INSTR)
C******************* SET ANGLE OF SLANT
      SLR          = SIN(SLR)/COS(SLR)
      ENCODE(25,1000,MESSAG(1)) 'SL          ;                 '
      ENCODE(8,1001,MESSAG(3)) SLR
      CALL IBSEND (MESSAG,11,INSTR)
      RETURN
      END
```

FORTRAN IV   V02.5-2   Tue 10-Nov-81 01:31:21   PAGE 001

```
      SUBROUTINE CPLT (CW,CH)
C          10-NOV-81
C******************* SUBROUTINE TO MOVE THE PEN THE NUMBER
C******************* OF USER SPECIFIED CHARACTER UNITS
      COMMON /ADDRES/ INSTR
      BYTE MESSAG(25)
C******************* CALL MOVE ROUTINE
      ENCODE(25,1000,MESSAG(1)) 'CP          ,          ;      '
 1000 FORMAT(1A25)
      ENCODE(8,1001,MESSAG(3)) CW
 1001 FORMAT(F8.3)
      ENCODE(8,1001,MESSAG(12)) CH
      CALL IBSEND (MESSAG,20,INSTR)
      RETURN
      END
```

```
FORTRAN IV      V02.5-2    Tue 10-Nov-81 01:33:59                PAGE 001

0001          SUBROUTINE LIM (X1,X2,Y1,Y2)
       C         10-NOV-81
       C****************** SUBROUTINE TO SET THE WINDOW LIMIT POINTS W1 AND W2
       C****************** IN USER UNITS (PLOTTING OUT OF LIMITS WILL RAISE
       C****************** THE PEN)
0002          COMMON /ADDRES/ INSTR
0003          COMMON /SCALES/ XP1,XP2,YP1,YP2
0004          COMMON /PLOT  / XRATIO,YRATIO,XKNST,YKNST
0005          BYTE MESSAG(26)
       C****************** DETERMINE LOCATION OF LIMIT POINTS
0006          IXW1         = X1*XRATIO+XSCL
0007          IYW1         = Y1*YRATIO+YSCL
0008          IXW2         = X2*XRATIO+XSCL
0009          IYW2         = Y2*YRATIO+XSCL
       C****************** INPUT WINDOW LIMITS
0010          ENCODE(26,1000,MESSAG(1)) 'IW     ,     ,     ,     ;'
0011     1000 FORMAT(1A25)
0012          ENCODE(5,1001,MESSAG(3)) IXW1
0013     1001 FORMAT(I5)
0014          ENCODE(5,1001,MESSAG(9)) IYW1
0015          ENCODE(5,1001,MESSAG(15)) IXW2
0016          ENCODE(5,1001,MESSAG(21)) IYW2
0017          CALL IBSEND (MESSAG,26,INSTR)
0018          RETURN
0019          END
```

```
FORTRAN IV      V02.5-2    Tue 10-Nov-81 01:37:17                PAGE 001

0001          SUBROUTINE DIG (XCORD,YCORD)
       C         10-NOV-81
       C****************** SUBROUTINE TO DIGITIZE POINTS ON THE PLOTTER
0002          COMMON /ADDRES/ INSTR
0003          COMMON /PLOT  / XRATIO,YRATIO,XKNST,YKNST
0004          BYTE MESSAG(25)
0005          LOGICAL*1 OK,EOS
       C****************** SET PLOTTER FOR DIGITIZING
0006          CALL IBSEND ('DC;',-1,INSTR)
0007          CALL IBSEND ('DP;',-1,INSTR)
       C****************** WAIT FOR DIGITIZED POINT
0008       10 CONTINUE
0009          CALL IBSEND ('OS;',-1,INSTR)
0010          ENCODE(25,1000,MESSAG(1)) '                          '
0011     1000 FORMAT(1A25)
0012          CALL IBRECV (MESSAG,10,INSTR)
0013          IPTR         = 1
0014          IEND         = 10
0015          ISTS         = INTGET (MESSAG,IPTR,IEND,OK,EOS)
0016          IF (.NOT.OK) GO TO 10
0018          ISTS         = ISTS/2
0019          ISTS         = ISTS/2
0020          IRMDR        = ISTS-(ISTS/2)*2
0021          IF (IRMDR.EQ.0) GO TO 10
       C****************** READY FOR DIG POINT
0023       20 CONTINUE
0024          CALL IBSEND ('OD;',-1,INSTR)
0025          CALL IBRECV (MESSAG,14,INSTR)
0026          CALL IBSEND ('DC;',-1,INSTR)
0027          IPTR         = 1
0028          IEND         = 25
0029          IXSCL        = INTGET (MESSAG,IPTR,IEND,OK,EOS)
0030          IF (.NOT.OK) GO TO 20
```

```
0032          IPTR        = IPTR+1
0033          IYSCL       = INTGET (MESSAG,IPTR,IEND,OK,EOS)
0034          IF (.NOT.OK) GO TO 20
0036          XSCL        = FLOAT (IXSCL)
0037          YSCL        = FLOAT (IYSCL)
0038          XCORD       = (XSCL-XKNST)/XRATIO
0039          YCORD       = (YSCL-YKNST)/YRATIO
0040          RETURN
0041          END
```

```
FORTRAN IV      V02.5-2     Tue 10-Nov-81 01:38:49              PAGE 001

0001          SUBROUTINE XAX (P1,P2,P3,P4,P5,P6,P7,P8,P9,XLABEL)
        C          10-NOV-81
        C****************** SUBROUTINE TO DRAW AN X AXIS AND LABEL IT
        C****************** ROUTINE TAKEN FROM AN HP-9825 ROUTINE
        C****************** WRITTEN FOR THE CESSNA AIRCRAFT CO.
0002          COMMON /ADDRES/ INSTR
0003          COMMON /XYSCLP/ SCLP1,SCLP2,SCLP3,SCLP4
0004          BYTE ARRAY(10),XLABEL(20)
0005          DATA ARRAY /10*' '/
        C****************** DEFINE AXES
0006          P15         = .1*P9
0007          P11         =-.1*P9
0008          P16         = 0.
0009          SCLP1       = P6-P7*P3
0010          SCLP2       = P6+(P1-P3)*P7
0011          HGHT        = 40./P2
0012          CALL CSIZ (HGHT,2.2,0.,0.)
0013          CALL SCL (0.,P1,0.,P2)
0014          XPLT        = P3
0015          YPLT        = P4-P11
0016          CALL PLT(XPLT,YPLT,1)
0017        4 CONTINUE
0018       10 FORMAT(1PE9.2)
0019       11 FORMAT(1F9.0)
0020       12 FORMAT(1F9.1)
0021       13 FORMAT(1F9.2)
0022       14 FORMAT(1F9.3)
0023          ENCODE (9,10,ARRAY(1)) P6+P16*P7
0024          IF (ABS(SCLP1).LT.10000.AND.ABS(SCLP2).LT.10000)
            .    ENCODE (9,11,ARRAY(1)) P6+P16*P7
0026          IF (ABS(SCLP1).LT.1000.AND.ABS(SCLP2).LT.1000)
            .    ENCODE (9,12,ARRAY(1)) P6+P16*P7
0028          IF (ABS(SCLP1).LT.100.AND.ABS(SCLP2).LT.100)
            .    ENCODE (9,13,ARRAY(1)) P6+P16*P7
0030          IF (ABS(SCLP1).LT.10.AND.ABS(SCLP2).LT.10)
            .    ENCODE (9,14,ARRAY(1)) P6+P16*P7
        C****************** MOVE NUMERIC LABEL TO FRONT OF ARRAY
0032       43 CONTINUE
0033          IF (ARRAY(1).NE.' ') GO TO 44
0035          DO 45 IC=1,9
0036          ARRAY(IC)   = ARRAY(IC+1)
0037       45 CONTINUE
0038          ARRAY(10)   = ' '
0039          GO TO 43
0040       44 CONTINUE
        C****************** FIND THE LENGTH OF THE NUMERIC LABEL
0041          ISAV        = 9
0042          DO 666 IA=1,9
0043          IF (ARRAY(10-IA).EQ.' ') GO TO 666
0045          ISAV        = 10-IA
        C****************** LEAVE LOOP EARLY
```

```
      GO TO 667
  666 CONTINUE
  667 CONTINUE
      P18         = ISAV
C******************* CHANGE THE ASCII CHARACTER 'O' INTO '0'
      DO 888 JJ=1,9
      IF (ARRAY(JJ).EQ.'O') ARRAY(JJ) = '0'
  888 CONTINUE
      P17         = 0.
      CALL PEN
      CW          =-(P18/2.+.25)
      CH          = .75*P9-.25
      CALL CPLT (CW,CH)
      CALL LBL (ARRAY)
      CW          =-(P18/2.-.25)
      CH          = .25-.75*P9
      CALL CPLT (CW,CH)
    3 CONTINUE
      P16         = P16+1.
      P17         = P17+1.
      IF (P16.LE.P5) CALL IPLT (0.,P11,2)
      IF (P16.LE.P5) CALL IPLT (1.,0.,2)
      IF (P16.LE.P5) CALL IPLT (0.,P15,2)
      IF (P17.LT.P8) GO TO 3
      IF (P16.LE.P5) GO TO 4
C******************* WRITE AXES LABEL
      DELX        =-.5*P5
      DELY        = 0.
      CALL IPLT (DELX,DELY,1)
      DO 5 I=1,20
      IF (XLABEL(21-I).EQ.' ') GO TO 5
      ISAVE       = 21-I
      GO TO 6
    5 CONTINUE
C******************* EXIT LOOP EARLY
    6 CONTINUE
      P17         = FLOAT(ISAVE)
      HGHT        = 40./P2
      CALL CSIZ (HGHT,2.2,0.,0.)
      CW          =-P17/2.
      CH          = (1.75*P9-.25)
      CALL CPLT (CW,CH)
      CALL LBL (XLABEL)
      IF (SCLP1.LT.SCLP2.AND.SCLP3.LT.SCLP4)
     .   CALL SCL(SCLP1,SCLP2,SCLP3,SCLP4)
      RETURN
      END
```

FORTRAN IV   V02.5-2   Tue 10-Nov-81 01:40:58   PAGE 001

```
      SUBROUTINE YAX (P1,P2,P3,P4,P5,P6,P7,P8,P9,YLABEL)
C        10-NOV-81
C******************* SUBROUTINE TO DRAW AN Y AXIS AND LABEL IT
C******************* ROUTINE TAKEN FROM AN HP-9825 ROUTINE
C******************* WRITTEN FOR THE CESSNA AIRCRAFT CO.
      COMMON /ADDRES/ INSTR
      COMMON /XYSCLP/ SCLP1,SCLP2,SCLP3,SCLP4
      BYTE ARRAY(10),YLABEL(20)
      DATA ARRAY /10*' '/
C******************* DEFINE AXES
      P10         =-.1*P9
      P14         = .1*P9
      P16         = 0.
```

```
      P19         = 0.
      SCLP3       = P6-P7*P4
      SCLP4       = P6+(P2-P4)*P7
      HGHT        = 40./P2
      CALL CSIZ (HGHT,2.2,0.,0.)
      CALL SCL (0.,P1,0.,P2)
      XPLT        = P3-P10
      YPLT        = P4
      CALL PLT (XPLT,YPLT,1)
    4 CONTINUE
   10 FORMAT(1PE9.2)
   11 FORMAT(1F9.0)
   12 FORMAT(1F9.1)
   13 FORMAT(1F9.2)
   14 FORMAT(1F9.3)
      ENCODE (9,10,ARRAY(1)) P6+P16*P7
      IF (ABS(SCLP3).LT.10000.AND.ABS(SCLP4).LT.10000)
     .   ENCODE (9,11,ARRAY(1)) P6+P16*P7
      IF (ABS(SCLP3).LT.1000.AND.ABS(SCLP4).LT.1000)
     .   ENCODE (9,12,ARRAY(1)) P6+P16*P7
      IF (ABS(SCLP3).LT.100.AND.ABS(SCLP4).LT.100)
     .   ENCODE (9,13,ARRAY(1)) P6+P16*P7
      IF (ABS(SCLP3).LT.10.AND.ABS(SCLP4).LT.10)
     .   ENCODE (9,14,ARRAY(1)) P6+P16*P7
C****************** MOVE NUMERIC LABEL TO FRONT OF ARRAY
   43 CONTINUE
      IF (ARRAY(1).NE.' ') GO TO 44
      DO 45 IC=1,9
      ARRAY(IC)   = ARRAY(IC+1)
   45 CONTINUE
      ARRAY(10)   = ' '
      GO TO 43
   44 CONTINUE
C****************** FIND THE LENGTH OF THE NUMERIC LABEL
      ISAV        = 9
      DO 666 IA=1,9
      IF (ARRAY(10-IA).EQ.' ') GO TO 666
      ISAV        = 10-IA
C****************** LEAVE LOOP EARLY
      GO TO 667
  666 CONTINUE
  667 CONTINUE
      P18         = ISAV
      IF (P18.GT.P19) P19=P18
C****************** CHANGE THE ASCII CHARACTER '0' INTO '0'
      DO 888 JJ=1,9
      IF (ARRAY(JJ).EQ.'0') ARRAY(JJ) = '0'
  888 CONTINUE
      P17         = 0.
      CALL PEN
      CW          = (P9*(P18/2.+1.)-P18/2.)
      CH          =-.3
      CALL CPLT (CW,CH)
      CALL LBL (ARRAY)
      CW          =-(P18/2.+P9*(P18/2.+1.))
      CH          = .3
      CALL CPLT (CW,CH)
    3 CONTINUE
      P16         = P16+1.
      P17         = P17+1.
      IF (P16.LE.P5) CALL IPLT (P10,0.,2)
      IF (P16.LE.P5) CALL IPLT (0.,1.,2)
      IF (P16.LE.P5) CALL IPLT (P14,0.,2)
      IF (P17.LT.P8) GO TO 3
      IF (P16.LE.P5) GO TO 4
C****************** WRITE AXES LABEL
      DELX        = 0.
      DELY        =-.5*P5
      CALL IPLT (DELX,DELY,1)
```

```
      DO 5 I=1,20
      IF (YLABEL(21-I).EQ.' ') GO TO 5
      ISAVE       = 21-I
      GO TO 6
    5 CONTINUE
C****************** EXIT LOOP EARLY
    6 CONTINUE
      P17         = FLOAT(ISAVE)
      CW          = P9*(P19+2.)+.5
      CH          = 0.
      CALL CPLT (CW,CH)
      HGHT        = 40./P2
      CALL CSIZ (HGHT,2.2,90.,0.)
      CW          =-P17/2.
      CH          = 0.
      CALL CPLT (CW,CH)
      CALL LBL (YLABEL)
      IF (SCLP1.LT.SCLP2.AND.SCLP3.LT.SCLP4)
     .    CALL SCL(SCLP1,SCLP2,SCLP3,SCLP4)
      RETURN
      END
```

A.10 MINC TIME HISTORY (CRT GRAPHICS)

Description: The SPLOT program is the executive for the CRT graphics plotting routine. This program reads the measured and predicted time histories and calls SCPLOT to display 200 time points of data on the CRT.

Listing:

```
FORTRAN IV      V02.5-2     Sun 28-Feb-82 03:18:38                PAGE 001

            PROGRAM SPLOT
      C.... PROGRAM TO PRODUCE SCREEN PLOTS OF LONGITUDINAL
      C.... AND LATERAL-DIRECTIONAL STATE TIME HISTORIES
            COMMON /LABELS/ ARRAY
            DIMENSION Z(7,200),U(2,200),ZP(7,200)
            LOGICAL LONG,LATR,EXTRA,MORE
            BYTE ARRAY(80),ANS,NAME(15)
            DATA LONG,LATR,EXTRA,MORE /4*.FALSE./
            DATA TIME /0./
            DATA NAME(15) /0/
        100 CONTINUE
      C.... ASK IF LONG AUTO-SEQUENCE
            TYPE 1
          1 FORMAT('$',10X,'LONGITUDINAL AUTO SEQUENCE? (Y OR N)')
            ACCEPT 2, ANS
          2 FORMAT(1A1)
            IF (ANS.EQ.'Y') LONG = .TRUE.
            IF (LONG) GO TO 3
      C.... ASK IF LATR AUTO-SEQUENCE
            TYPE 4
          4 FORMAT('$',10X,'LATERAL-DIRECTIONAL AUTO SEQUENCE? (Y OR N)')
            ACCEPT 2, ANS
            IF (ANS.EQ.'Y') LATR = .TRUE.
            IF (.NOT.LATR) GO TO 100
      C.... OPEN INPUT DATA FILES
          3 CONTINUE
            TYPE 5
          5 FORMAT('$',10X,'ENTER THE MEASURED DATA FILE NAME: ')
            ACCEPT 6,(NAME(I),I=1,14)
          6 FORMAT(14A1)
            OPEN (UNIT=1,NAME=NAME,TYPE='OLD',ACCESS='SEQUENTIAL',
           .      READONLY,BUFFERCOUNT=2,FORM='UNFORMATTED')
      C.... CHECK FOR EXTRA DATA
            TYPE 8
          8 FORMAT('$',10X,'DOES THE FILE CONTAIN EXTRA DATA? (Y OR N)')
            ACCEPT 2,ANS
            IF (ANS.EQ.'Y') EXTRA = .TRUE.
      C.... CLEAR OUT OLD DATA FILE NAME
            DO 10 I=1,14
            NAME(I) = ' '
         10 CONTINUE
            TYPE 7
          7 FORMAT('$',10X,'ENTER THE PREDICTED DATA FILE NAME: ')
            ACCEPT 6,(NAME(I),I=1,14)
            OPEN (UNIT=2,NAME=NAME,TYPE='OLD',ACCESS='SEQUENTIAL',
           .      READONLY,BUFFERCOUNT=2,FORM='UNFORMATTED')
         25 CONTINUE
      C.... EMPTY ARRAYS
            DO 26 II=1,200
            DO 27 IJ=1,7
            Z(IJ,II) = 0.
            ZP(IJ,II) = 0.
         27 CONTINUE
            U(1,II) = 0.
```

```
        U(2,II) = 0.
     26 CONTINUE
  C.... READ DATA INTO ARRAYS
        DO 1000 J=1,200
        JEND = J
        IF (LONG) READ (1,END=2000) Z(4,J),Z(1,J),Z(7,J),Z(6,J),U(1,J),
      .                             AM6,AM7,AM8,AM9,AM10,AM11
        IF (LONG.AND.EXTRA) READ (1,END=2000) Z(2,J),Z(3,J),Z(5,J),
      .                                       AM15,AM16,AM17
        IF (LONG) U(2,J) = 0.
        IF (LATR) READ (1,END=2000) AM1,AM2,AM3,AM4,AM5,Z(4,J),Z(1,J),
      .                             Z(7,J),Z(2,J),U(1,J),U(2,J)
        IF (LATR.AND.EXTRA) READ (1,END=2000) AM12,AM13,AM14,Z(3,J),
      .                                       Z(5,J),Z(6,J)
   1000 CONTINUE
  C.... SET MORE EQUAL TO TRUE
        MORE = .TRUE.
   2000 CONTINUE
  C.... SET THE INITIAL CONDITIONS ON STATES
        IF (TIME.NE.0.) GO TO 1988
        DO 1999 J=1,7
        ZP(J,1) = (Z(J,1)+Z(J,2))/2.
        ZP(J,2) = (Z(J,1)+Z(J,2))/2.
   1999 CONTINUE
   1988 CONTINUE
        DO 2001 J=3,JEND-1
        READ (2) (ZP(IA,J),IA=1,7)
   2001 CONTINUE
  C.... PLOT DATA
        CALL SCPLOT(Z,ZP,U,LONG,LATR)
  C.... IF MORE IS TRUE DISPLAY CHANGE PAPER
  C.... AND GET READY FOR MORE DATA
        IF (.NOT.MORE) GO TO 3535
        TYPE 2525
   2525 FORMAT('$',10X,'MORE DATA ON FILE: (CONTINUE WHEN READY)')
        ACCEPT 2,ANS
  C.... RESET MORE DATA FLAG
        MORE = .FALSE.
  C.... INCREMENT THE TIME COUNTER BY 20. SEC
        TIME = TIME+20.
        GO TO 25
   3535 CONTINUE
        STOP
        END
```

Subroutine SCPLOT

Description: The SCPLOT subroutine plots either a set of longitudinal or lateral-directional time history traces.

Listing:

```
FORTRAN IV      V02.5-2     Sun 28-Feb-82 03:19:11              PAGE 001

        SUBROUTINE SCPLOT(Z,ZP,U,LONG,LATR)
        COMMON /STATUS/ ISTAT(16)
        COMMON /FIVEA/ DATA,GAIN
        DIMENSION IARRAY(512)
        DIMENSION GAIN(9),GAIN1(9),GAIN2(9)
```

```
      DIMENSION Z(7,200),ZP(7,200),U(2,200)
      LOGICAL LONG,LATR
      BYTE ANS
C.... THE REQUIRED SUBROUTINES ARE:
C.... PLOT55,INIT AND GRAPH
      DATA ISTAT  /16*0/
      DATA GAIN1 /114.6,114.6,143.2,47.7,28.6,28.6,100.,143.2,143.2/
      DATA GAIN2 /114.6,2.5,143.2,95.5,28.6,100.,25.,143.2,0./
C
      IARRAY(1) = 0
      IARRAY(3) = 0
C.... LONGITUDINAL
C     1. PITCH RATE -      25 DEG/SEC
C     2. AIRSPEED -        20 FT/SEC
C     3. ANGLE OF ATTACK - 20 DEG
C     4. PITCH ATTITUDE -  30 DEG
C     5. PITCH RATE ACCEL. - 100 DEG/SEC**2
C     6. LONGITUDINAL ACCEL. - .5 G
C     7. NORMAL ACCEL. -        2 G
C     8. ELEVATOR PSN. - 20 DEG
C     9.   * * *   (BLANK)
C.... LATERAL DIRECTIONAL
C     1. ROLL RATE - 25 DEG/SEC
C     2. YAW RATE  - 25 DEG/SEC
C     3. SIDESLIP ANGLE - 20 DEG
C     4. BANK ANGLE     - 60 DEG
C     5. ROLL RATE ACCEL. - 100 DEG/SEC**2
C     6. YAW RATE ACCEL.  - 100 DEG/SEC**2
C     7. LONGITUDINAL ACCEL. - .5 G
C     8. AILERON DEFLECTION - 20 DEG
C     9. RUDDER DEFLECTION  - 20 DEG
      IF (LONG) GO TO 411
      DO 421 I=1,9
      GAIN(I) = GAIN1(I)
  421 CONTINUE
  411 CONTINUE
      IF (LATR) GO TO 402
      DO 422 I=1,9
      GAIN(I) = GAIN2(I)
  422 CONTINUE
  402 CONTINUE
C.... SET UP DO LOOP
      DO 2525 LL=1,5
      KT = LL*2-1
      KB = LL*2
      IF (KB.EQ.10) KB = 9
      IF (LONG.AND.LL.EQ.5) GO TO 2525
C.... CLEAR CRT AND FORM GRID FOR PLOTTING
      CALL INIT
      CALL PLOT55(2,1+2+32,64+128,,ISTAT)
      DO 110 K=1,235,50
      CALL PLOT55(4,1,K-1,ISTAT)
  110 CONTINUE
      CALL PLOT55(4,1,229,ISTAT)
      CALL PLOT55(5,0,1,ISTAT)
C     - - - - - - - - - - - - - - - - - - - -
C.... FORM THE MEASURED AND ESTIMATED RESPONSES
      DO 130 I=1,200
      IF(KT.LT.8) IARRAY(2*I) = Z(KT,I)*GAIN(KT)+150
      IF(KT.GE.8) IARRAY(2*I) = U(KT-7,I)*GAIN(KT)+150
      IF(KB.LT.8) IARRAY(2*I-1) = Z(KB,I)*GAIN(KB)+50
      IF(KB.GE.8) IARRAY(2*I-1) = U(KB-7,I)*GAIN(KB)+50
  130 CONTINUE
      CALL PLOT55(9,20,2,ISTAT)
      CALL PLOT55(12,,'* * *  TIME HISTORIES  * * * ',ISTAT)
      CALL PLOT55(9,50,4,ISTAT)
      CALL PLOT55(12,,' MEASURED DATA ',ISTAT)
      CALL GRAPH(400,IARRAY)
      DO 140 I=1,200
      IF (KT.GE.8) GO TO 131
      IARRAY(2*I) = ZP(KT,I)*GAIN(KT)+150
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
  131 CONTINUE
      IF (KB.GE.8) GO TO 132
      IARRAY(2*I-1) = ZP(KB,I)*GAIN(KB)+50
  132 CONTINUE
  140 CONTINUE
      CALL PLOT55(9,50,4,ISTAT)
      CALL PLOT55(12,,' ESTIMATED DATA',ISTAT)
      CALL GRAPH(400,IARRAY)
C     - - - - - - - - - - - - - - - - - - - - -
      CALL PLOT55(9,50,4,ISTAT)
      CALL PLOT55(12,,' CR WHEN READY ',ISTAT)
C     - - - - - - - - - - - - - - - - - - - - - - - - - -
      IF (LATR) GO TO 699
      KFLAG1 = 0
C.... LONGITUDINAL LABELS
      CALL PLOT55(9,50,6,ISTAT)
      IF(KT.EQ.1) GO TO 601
      IF(KT.EQ.2) GO TO 602
      IF(KT.EQ.3) GO TO 603
      IF(KT.EQ.4) GO TO 604
      IF(KT.EQ.5) GO TO 605
      IF(KT.EQ.6) GO TO 606
      IF(KT.EQ.7) GO TO 607
      IF(KT.EQ.8) GO TO 608
  610 CONTINUE
      KFLAG1 = 1
      CALL PLOT55(9,50,16,ISTAT)
      IF(KB.EQ.1) GO TO 601
      IF(KB.EQ.2) GO TO 602
      IF(KB.EQ.3) GO TO 603
      IF(KB.EQ.4) GO TO 604
      IF(KB.EQ.5) GO TO 605
      IF(KB.EQ.6) GO TO 606
      IF(KB.EQ.7) GO TO 607
      IF(KB.EQ.8) GO TO 608
  601 CALL PLOT55(12,,' Q  +/- 25 DEG/SEC        ',ISTAT)
      GO TO 640
  602 CALL PLOT55(12,,' V  +/- 20 FEET/SEC        ',ISTAT)
      GO TO 640
  603 CALL PLOT55(12,,' ALPHA  +/- 20 DEG         ',ISTAT)
      GO TO 640
  604 CALL PLOT55(12,,' THETA  +/- 30 DEG         ',ISTAT)
      GO TO 640
  605 CALL PLOT55(12,,' Q DOT  +/- 100 DEG/SEC**2',ISTAT)
      GO TO 640
  606 CALL PLOT55(12,,' AX  +/- .5 G             ',ISTAT)
      GO TO 640
  607 CALL PLOT55(12,,' AN  +/- 2 G              ',ISTAT)
      GO TO 640
  608 CALL PLOT55(12,,' DE  +/- 20 DEG           ',ISTAT)
  640 CONTINUE
      IF (KFLAG1.EQ.0) GO TO 610
C.... GO TO END OF LOOP
      IF (LONG) GO TO 751
  699 CONTINUE
C.... LATERAL DIRECTIONAL LABELS
      KFLAG2 = 0
      CALL PLOT55(9,50,6,ISTAT)
      IF(KT.EQ.1) GO TO 701
      IF(KT.EQ.2) GO TO 702
      IF(KT.EQ.3) GO TO 703
      IF(KT.EQ.4) GO TO 704
      IF(KT.EQ.5) GO TO 705
      IF(KT.EQ.6) GO TO 706
      IF(KT.EQ.7) GO TO 707
      IF(KT.EQ.8) GO TO 708
      IF(KT.EQ.9) GO TO 709
  710 CONTINUE
      KFLAG2 = 1
      CALL PLOT55(9,50,16,ISTAT)
      IF(KB.EQ.1) GO TO 701
```

```
      IF(KB.EQ.2) GO TO 702
      IF(KB.EQ.3) GO TO 703
      IF(KB.EQ.4) GO TO 704
      IF(KB.EQ.5) GO TO 705
      IF(KB.EQ.6) GO TO 706
      IF(KB.EQ.7) GO TO 707
      IF(KB.EQ.8) GO TO 708
      IF(KB.EQ.9) GO TO 709
  701 CALL PLOT55(12,,' P  +/- 25 DEG/SEC            ',ISTAT)
      GO TO 740
  702 CALL PLOT55(12,,' R  +/- 25 DEG/SEC            ',ISTAT)
      GO TO 740
  703 CALL PLOT55(12,,' BETA  +/- 20 DEG             ',ISTAT)
      GO TO 740
  704 CALL PLOT55(12,,' PHI  +/- 60 DEG              ',ISTAT)
      GO TO 740
  705 CALL PLOT55(12,,' P DOT  +/- 100 DEG/SEC**2  ',ISTAT)
      GO TO 740
  706 CALL PLOT55(12,,' R DOT  +/- 100 DEG/SEC**2  ',ISTAT)
      GO TO 740
  707 CALL PLOT55(12,,' AY  +/- .5 G                 ',ISTAT)
      GO TO 740
  708 CALL PLOT55(12,,' DA  +/- 20 DEG               ',ISTAT)
      GO TO 740
  709 CALL PLOT55(12,,' DR  +/- 20 DEG               ',ISTAT)
  740 CONTINUE
      IF(KFLAG2.EQ.0) GO TO 710
  751 CONTINUE
      ACCEPT 180,ANS
  180 FORMAT(A1)
      CALL INIT
C.... BOTTOM OF LOOP
 2525 CONTINUE
      CALL PLOT55(2,512,1+2+4+32+64,ISTAT)
      CALL PLOT55(0,-1,0,ISTAT)
      RETURN
      END
```

Subroutine INIT

Description: The INIT subroutine initializes the common block array /STATUS/ ISTAT(16) for use in the PLOT55 graphics routines.

Listing:

```
FORTRAN IV      V02.5-2     Sun 28-Feb-82 03:19:57             PAGE 001

      SUBROUTINE INIT
      COMMON/STATUS/ISTAT(16)
      DATA ISTAT/16*0/
      CALL PLOT55(13,72,,ISTAT)
      CALL PLOT55(13,74,,ISTAT)
      CALL PLOT55(2,1+512,,ISTAT)
      RETURN
      END
```

Subroutine GRAPH

Description: The GRAPH subroutine plots the actual data to the VT 105 CRT. It uses a MINC software package called PLOT55.

Listing:

```
FORTRAN IV      V02.5-2     Sun 28-Feb-82 03:20:39                PAGE 001

        SUBROUTINE GRAPH(N,IARRAY)
        COMMON/STATUS/ISTAT(16)
        DIMENSION IARRAY(512)
        NUMBER=ISTAT(8)/8
        CALL PLOT55(7,0,0,ISTAT)
        CALL PLOT55(8,512,0,ISTAT)
        CALL PLOT55(2,1+(NUMBER+1)*2,(NUMBER+1)*10,ISTAT)
        CALL PLOT55(3,-N,IARRAY,ISTAT)
        CALL PLOT55(1,1-NUMBER,,ISTAT)
        CALL PLOT55(9,10,1,ISTAT)
        END
```

A.11) MINC DATA TRANSFER (MINC TO MAINFRAME)

Description: This routine allows crude use of the MINC computer as a smart terminal. It is used to transfer files to a mainframe computer.

Listing:

```
FORTRAN IV      V02.5-2    Sun 03-Jan-82 11:50:48              PAGE 001

           PROGRAM RS232
      C
      C.... MINC PROGRAM TO TALK TO THE HONEYWELL COMPUTER
      C.... WRITTEN BY ROBERT CLARKE
      C
      C.... SET UP INTEGER ARRAY FOR DEFINING COMMUNICATION USING SLUO
           DIMENSION IADDR(4)
      C.... DIMENSION CHARACTER ARRAYS FOR INPUT AND OUTPUT
           BYTE NAME(15),SAVE(30000),LINE(132),CHARIN,CHAROT
           LOGICAL*1 RECEIV,DISKOT
      C.... SET RECEIVE FLAG TO FALSE
           DATA RECEIV /.FALSE./
      C.... SET DISK OUTPUT FLAG TO FALSE
           DATA DISKOT /.FALSE./
      C.... INITIALIZE ARRAYS
           DATA SAVE /30000*0/
           DATA NAME /15*0/
           DATA LINE /132*0/
      C.... START PROGRAM
      C.... THIS PROGRAM READS FROM SLUO
      C.... (300 BAUD, EVEN PARITY, 7 DATA & 2 STOP)
      C.... SET TT FOR SPECIAL READING
      C.... SET BITS 12, 13, AND 14 OF JSW
      C.... (NO LOCAL ECHO AND LOWER CASE INABLED)
           CALL IPOKEB ('45,112)
      C.... SET BIT 6 OF JSW (NO WAIT FOR TT INPUT)
           CALL IPOKEB ('44,64)
      C.... TERMINAL IS NOW SET UP TO READ IN ABNORMAL FORMAT
      C.... THE PROGRAM CAN GO OUT AND LOOK FOR A CHARACTER WITHOUT
      C.... GOING INTO A WAIT STATE IF ONE IS UNAVAILABLE
      C.... ATTACH THE INPUT PORT AND CHECK FOR ERRORS
         4 CONTINUE
           IERR     = MTATCH(1)
           IF (IERR.NE.0) TYPE 998, IERR
           IF (IERR.NE.0) GO TO 4
       998 FORMAT ('ERROR IN SETTING UP PORT; ERROR NUMBER = ',I2)
         5 CONTINUE
           IADDR(1) = '50100
           IADDR(2) = 0
           IADDR(3) = 0
           IADDR(4) = 0
           IERR     = MTSET(1,IADDR(1))
           IF (IERR.NE.0) TYPE 998, IERR
           IF (IERR.NE.0) GO TO 5
           TYPE *,'  TERMINAL IS SET UP WITH NO ERRORS'
      C.... RS232 PORT IS SET UP (SLUO) WITH CORRECT BAUD RATE AND
      C.... DATA COMMUNICATION PARAMETERS
      C.... THE PORT IS ALSO SET UP TO SAMPLE WITHOUT GOING INTO
      C.... WAIT STATES IF NO DATA HAS BEEN SENT
      C.... TOP OF LOOP FOR DATA COMMUNICATION
      2525 CONTINUE
      C.... LOOK FOR KEYBOARD INPUT
           IVAL     = ITTINR()
           IF (IVAL.LT.0) GO TO 100
```

```
         CHARIN   = IVAL
    C.... CHECK FOR SPECIAL ACTION KEYS (^R: RECEIVE INPUT DATA)
         IF (IVAL.EQ.18) RECEIV = .TRUE.
         IF (IVAL.EQ.18) ISAVE  = 1
         IF (IVAL.EQ.18) TYPE 2001
    2001 FORMAT(' ',/,'*****  HONEYWELL INPUT BUFFER OPENED *****',/,
      .          '$','INPUT FILE NAME FOR HONEYWELL DATA FILE: ')
         IF (IVAL.EQ.18) ACCEPT 2002, (NAME(I),I=1,14)
    2002 FORMAT(14A1)
    C.... OPEN OUTPUT FILE
         IF (IVAL.EQ.18) OPEN (UNIT=1,NAME=NAME,TYPE='NEW',
      .        ACCESS='SEQUENTIAL',FORM='FORMATTED',
      .        BUFFERCOUNT=2,DISPOSE='SAVE',RECORDSIZE=132)
         IF (IVAL.EQ.18) GO TO 100
    C.... (^D: MINC DISK DUMP OF HONEYWELL DATA)
         IF (IVAL.EQ.4) RECEIV = .FALSE.
         IF (IVAL.EQ.4) TYPE 2003
    2003 FORMAT(' ',/,'***** HONEYWELL DATA BUFFER OUTPUT TO DISK *****')
         IF (IVAL.EQ.4) DISKOT = .TRUE.
         IF (IVAL.EQ.4) GO TO 100
    C.... (^T: MINC FILE OUTPUT TO HONEYWELL)
         IF (IVAL.NE.20) GO TO 2050
         TYPE 2004
    2004 FORMAT(' ',/,'***** MINC DATA FILE TO HONEYWELL *****',/,
      .          '$','INPUT FILE NAME FOR MINC DATA FILE: ')
         ACCEPT 2002, (NAME(I),I=1,14)
         OPEN (UNIT=2,NAME=NAME,TYPE='OLD',
      .        ACCESS='SEQUENTIAL',FORM='FORMATTED',
      .        BUFFERCOUNT=1,READONLY)
    2007 CONTINUE
    C.... READ LINE FROM INPUT FILE
         READ(2,2005,END=2009) (LINE(I),I=1,132)
    2005 FORMAT(132A1)
    C.... FIND END OF LINE
         DO 344 IU =1,132
         ILEN     = 133-IU
         IF (LINE(133-IU).NE.' ') GO TO 345
     344 CONTINUE
     345 CONTINUE
    C.... SEND LINE TO HONEYWELL
    C.... SEND CR/LF TO HONEYWELL FIRST
         I         = ITTOUR(10)
         I         = MTOUT(1,10)
         I         = ITTOUR(13)
         I         = MTOUT(1,13)
    C.... SEND 15 BLANKS
         DO 66 IU=1,15
         DO 67 IU2=1,200
         W         = ABS(FLOAT(IU2))
      67 CONTINUE
         I         = MTOUT(1,32)
      66 CONTINUE
         DO 2006 J=1,ILEN
    C.... BUILD WAIT LOOP
         DO 2020 IWAIT=1,200
         W         = ABS(FLOAT(I))
    2020 CONTINUE
         CHARIN   = LINE(J)
         I         = ITTOUR(CHARIN)
         I         = MTOUT(1,CHARIN)
    2006 CONTINUE
         GO TO 2007
    2009 CONTINUE
    C.... END OF FILE IS FOUND
    C.... CLOSE FILE
         CLOSE (UNIT=2)
    C.... SEND LAST LINE OF INPUT TO HONEYWELL
    C.... SEND CR/LF TO HONEYWELL FIRST
         I         = ITTOUR(10)
```

```
        I          = MTOUT(1,10)
        I          = ITTOUR(13)
        I          = MTOUT(1,13)
        DO 2010 J=1,I
        DO 2021 IWAIT=1,200
        W          = ABS(FLOAT(I))
 2021   CONTINUE
        CHARIN     = LINE(J)
        I          = ITTOUR(CHARIN)
        I          = MTOUT(1,CHARIN)
 2010   CONTINUE
        GO TO 100
 2050   CONTINUE
C....   SEND CHARACTER
        I          = ITTOUR(CHARIN)
        I          = MTOUT(1,CHARIN)
C....   NO KEYBOARD INPUT
  100   CONTINUE
C....   CHECK FOR HONEYWELL INPUT
        I          = MTIN(1,CHAROT,1)
C....   REMOVE LF FROM HONEYWELL INPUT
        IF (CHAROT.EQ.10) I = -1
C....   IF THERE WAS A CHARACTER WRITE IT TO THE TERMINAL
        IF (I.EQ.0) I = ITTOUR(CHAROT)
        IF (CHAROT.EQ.13) I = ITTOUR(10)
C....   CHECK FOR SAVING HONEYWELL INPUT OR WRITING TO DISK
C....   CHECK FOR DISK SAVE
        IF (DISKOT) GO TO 200
C....   CKECK FOR INPUT SAVING
        IF ((I.NE.0).OR.(.NOT.RECEIV)) GO TO 2525
        IF (CHAROT.GT.126) GO TO 2525
        IF ((CHAROT.LT.7.OR.CHAROT.GT.13).AND.(CHAROT.LT.32)) GO TO 2525
        SAVE(ISAVE) = CHAROT
        ISAVE = ISAVE+1
        IF (CHAROT.EQ.13) SAVE(ISAVE) = 10
        IF (CHAROT.EQ.13) ISAVE = ISAVE+1
        IF (ISAVE.GT.30000) DISKOT = .TRUE.
        IF (.NOT.DISKOT) GO TO 2525
  200   CONTINUE
C....   RESET DISK OUTPUT FLAG
        DISKOT = .FALSE.
C....   FIND LAST OF DATA
        ILAST  = LEN(SAVE)
C....   INITIALIZE ILFOLD (POINTER TO LAST CR/LF
        ILFOLD = 1
  203   CONTINUE
        DO 201 I=ILFOLD,ILAST
C....   FIND NEXT LF (END OF LINE)
        IF (SAVE(I).EQ.10) ILF = I
        IF (SAVE(I).EQ.10) GO TO 202
  201   CONTINUE
C....   MUST BE LAST LINE COULD END WITHOUT LF
        ILF    = ILAST
  202   CONTINUE
        WRITE(1,21) (SAVE(JJ),JJ=ILFOLD,ILF-2)
   21   FORMAT(132A1)
C....   RESET ILFOLD TO NEW VALUE
        ILFOLD = ILF+1
C....   LOOP UNTIL SAVE BUFFER IS WRITTEN
        IF (ILF.NE.ILAST) GO TO 203
        CLOSE (UNIT=1)
        STOP
        END
```

A.12) MINC DATA ERROR CORRECTION

Description: This program, FIXUP, is used to correct the measured time history files for data errors.

Listing:

```
FORTRAN IV      V02.5-2    Sun 07-Mar-82 02:24:03              PAGE 001

0001        PROGRAM FIXUP
      C
      C     THIS PROGRAM IS USED TO CORRECT DATA ERRORS ON THE MMLE
      C     INPUT DATA FILES
      C
      C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      C     * MEASURED STATES:                                          *
      C     *        LONG: Q, V, ALP, THA, QDOT, A-X, AND A-Z           *
      C     *        LATR: P, R, BTA, PHI, PDOT, RDOT, AND A-Y          *
      C     *                                                           *
      C     *                     ROBERT CLARKE   18-FEB-82             *
      C     *                                                           *
      C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      C
0002        DIMENSION Z(9)
      C     NOTE: FOR THIS PROGRAM Z(8) = U(1) AND Z(9) = U(2)
0003        LOGICAL*1 LONG,LATR,EXTRA
0004        BYTE INAME(15),ANS
0005        DATA LONG,LATR,EXTRA /3*.FALSE./
0006        DATA INAME(15) /0/
0007        DATA VALUE,IPOINT /0.,0/
0008        DATA ZLLIM,ZULIM /0.,0./
0009        DATA ISTM1 /0/
      C
      C****************** INPUT USER DEFINED SETUP DATA
0010   2000 CONTINUE
0011        TYPE 3000
0012   3000 FORMAT(/////,' ',10X,'Indicate type of run:',
          1            /,' ',10X,'If Longitudinal        type "L"',
          2            /,' ',10X,'If Lateral-directional type "D"',
          3            /,'$',10X,'Select Run Type: ')
0013        ACCEPT 100, ANS
0014    100 FORMAT(A1)
0015        IF (ANS.EQ.'L') LONG = .TRUE.
0017        IF (ANS.EQ.'D') LATR = .TRUE.
0019        IF (.NOT.(LONG.OR.LATR)) TYPE 3001
0021   3001 FORMAT(10X,'WRONG ANSWER')
0022        IF (.NOT.(LONG.OR.LATR)) GO TO 2000
      C****************** ATTACH INPUT DATA FILE
      C****************** CHECK ON EXTRA DATA
0024        TYPE 3004
0025   3004 FORMAT('$',10X,'Extra data?  (Y OR N): ')
0026        ACCEPT 100, ANS
0027        IF (ANS.EQ.'Y') EXTRA = .TRUE.
      C****************** ENTER NAME OF DATA FILE WITH FLIGHT TEST DATA
0029        TYPE 3006
0030   3006 FORMAT(/,'$',10X,'Enter file name containing measured data: ')
0031        ACCEPT 101,(INAME(IABC),IABC=1,14)
0032    101 FORMAT(14A1)
0033        OPEN(UNIT=4,NAME=INAME,TYPE='OLD',ACCESS='SEQUENTIAL',
          1     FORM='UNFORMATTED',READONLY,BUFFERCOUNT=2)
      C****************** ATTACH OUTPUT DATA FILE
      C****************** ENTER NAME OF DATA FILE FOR FLIGHT TEST DATA
0034        TYPE 3008
```

```
 3008 FORMAT(/,'$',10X,'Enter file name containing output data: ')
      ACCEPT 101,(INAME(IABC),IABC=1,14)
      OPEN(UNIT=3,NAME=INAME,TYPE='NEW',ACCESS='SEQUENTIAL',
     1     FORM='UNFORMATTED',BUFFERCOUNT=2)
C******************* STARTING ITERATION LOOP
C******************* GET CHANNEL NUMBER TO CHECK FOR BAD DATA
      TYPE 3010
 3010 FORMAT(/,' ',10X,'Data which is suspect will be querried.',
     1       /,' ',10X,'Changes can be made when they are requested.',
     2       /,' ',10X,'No data can be changed to exactly zero.',
     3       /,'$',10X,'Enter the channel number to check (1-9): ')
      ACCEPT 3011,INUM
 3011 FORMAT(I10)
C******************* GET NUMBER OF DATA POINTS TO NOT WRITE TO OUTPUT
      TYPE 3020
 3020 FORMAT(/,'$',10X,
     1 'Enter the number of data points to skip writing to output: ')
      ACCEPT 3011, ISTM1
      ISTART       = ISTM1+1
  998 CONTINUE
      IPOINT       = IPOINT+1
C******************* READ IN MEASURED RESPONSES FROM DATA FILE
C******************* READ IN LONGITUDINAL DATA
      IF (LONG) READ(4,END=1000) Z(4),Z(1),Z(7),Z(6),Z(8),
     1                           AM6,AM7,AM8,AM9,AM10,AM11
      IF (LONG.AND.EXTRA) READ(4,END=1000) Z(2),Z(3),Z(5),
     1                                     AM15,AM16,AM17
      Z(9)         = 0.
C******************* READ IN LATERAL DIRECTIONAL DATA
      IF (LATR) READ(4,END=1000) AM1,AM2,AM3,AM4,AM5,Z(4),
     1                           Z(1),Z(7),Z(2),Z(8),Z(9)
      IF (LATR.AND.EXTRA) READ(4,END=1000) AM12,AM13,AM14,
     1                                     Z(3),Z(5),Z(6)
C******************* SKIP OVER DATA NOW
      IF (IPOINT.LT.ISTART) TYPE 3021, IPOINT
 3021 FORMAT(' ','SKIPPING DATA;  TIME POINT = ',I6)
      IF (IPOINT.LT.ISTART) GO TO 998
C******************* PRINT OUT DATA
      IF ((Z(INUM).GT.ZULIM).OR.(Z(INUM).LT.ZLLIM))
     1   TYPE 3012,IPOINT,Z(INUM)
 3012 FORMAT('$','TIME POINT = ',I6,'  VALUE = ',F12.6,'  CHANGE: ')
      IF ((Z(INUM).LT.ZULIM).AND.(Z(INUM).GT.ZLLIM))
     1   TYPE 3014,IPOINT,Z(INUM)
 3014 FORMAT(' ','TIME POINT = ',I6,'  VALUE = ',F12.6)
      IF ((Z(INUM).GT.ZULIM).OR.(Z(INUM).LT.ZLLIM)) ACCEPT 3013,VALUE
 3013 FORMAT(F10.1)
      IF (VALUE.NE.0.) Z(INUM) = VALUE
C******************* SET Z UPPER AND LOWER ERROR LIMITS
      ZULIM        = Z(INUM)*1.2
      IF (Z(INUM).LT.0.) ZULIM = Z(INUM)*0.8
      ZLLIM        = Z(INUM)*0.8
      IF (Z(INUM).LT.0.) ZLLIM = Z(INUM)*1.2
C******************* RESET VALUE TO ZERO
      VALUE        = 0.0
C******************* WRITE OUT LONGITUDINAL DATA
      IF (LONG) WRITE(3) Z(4),Z(1),Z(7),Z(6),Z(8),
     1                   AM6,AM7,AM8,AM9,AM10,AM11
      IF (LONG.AND.EXTRA) WRITE(3) Z(2),Z(3),Z(5),
     1                             AM15,AM16,AM17
C******************* WRITE OUT LATERAL DIRECTIONAL DATA
      IF (LATR) WRITE(3) AM1,AM2,AM3,AM4,AM5,Z(4),
     1                   Z(1),Z(7),Z(2),Z(8),Z(9)
      IF (LATR.AND.EXTRA) WRITE(3) AM12,AM13,AM14,
     1                             Z(3),Z(5),Z(6)
      GO TO 998
 1000 CONTINUE
C******************* CLOSE OUTPUT DATA FILE
      CLOSE(UNIT=3,DISPOSE='SAVE')
      STOP
      END
```

A.13) <u>MINC SUMMARY DERIVATIVE OUTPUT</u>

Description: The SUMMARY program is used for the plotting of derivative output as a function of lift coefficient. These plots are used to show trends in derivative predictions.

Listing:

```
FORTRAN IV      V02.5-2     Mon 22-Feb-82 00:34:04              PAGE 001

          PROGRAM SUMARY
    C.... SUBROUTINE TO DO HARD COPY PLOTS OF COEFFICIENT DATA
          COMMON /ADDRES/ INSTR
          BYTE ANS,MESSAG(6),LABEL(25)
          DATA LABEL /25*' '/
          DATA MESSAG /'V','S','0','0','!',0/
          DATA DGR /57.3/
    C.... SET UP PLOTTER
          CALL PSC (5)
          CALL PCLR
    C.... ASK FOR PLOTTER SPEED
          TYPE 103
      103 FORMAT('$','Input the plotter speed in cm/sec (1-25): ')
          ACCEPT 104,ISPD
      104 FORMAT(I3)
    C.... ENCODE SPEED FOR OUTPUT TO PLOTTER
          ENCODE (2,1001,MESSAG(3)) ISPD
     1001 FORMAT(I2)
    C.... CHANGE ENCODED BLANKS BACK TO ZEROS
          IF (MESSAG(3).EQ.' ') MESSAG(3)='0'
    C.... SEND SPEED TO PLOTTER
          CALL IBSEND (MESSAG,5,INSTR)
    C.... TOP OF LOOP
     1000 CONTINUE
          CALL PEN
    C.... GET QUADRANT FOR PLOT
          TYPE 105
      105 FORMAT('$','Input the quadrant for the plot (1-4): ')
          ACCEPT 104,IQD
    C.... GET MULTIPLICATION FACTOR FOR CRAMER RAO BOUNDS
          TYPE 200
      200 FORMAT('$','Input the Cramer Rao bound multiplication factor: ')
          ACCEPT 201,FACT
      201 FORMAT(F10.0)
    C.... DEFINE P1 AND P2 FOR PLOT
          IF (IQD.EQ.1) CALL IBSEND('IP 6000,4700,10000,7700;',-1,INSTR)
          IF (IQD.EQ.2) CALL IBSEND('IP 1000,4700,5000,7700;',-1,INSTR)
          IF (IQD.EQ.3) CALL IBSEND('IP 1000,1200,5000,4200;',-1,INSTR)
          IF (IQD.EQ.4) CALL IBSEND('IP 6000,1200,10000,4200;',-1,INSTR)
    C.... GET INFORMATION FOR   AXIS OF PLOT
          TYPE 106
      106 FORMAT('$','Input the label for the y-axis (up to 20 char): ')
          ACCEPT 107,(LABEL(I),I=1,20)
      107 FORMAT(20A1)
          TYPE 108
      108 FORMAT('$','Input the minimum value for the y-axis: ')
          ACCEPT 201,YMIN
          TYPE 110
      110 FORMAT('$','Input the maximum value for the y-axis: ')
          ACCEPT 201,YMAX
          YINC   = (YMAX-YMIN)/10.
          CALL XAX(5.,10.,1.,0.,4.,0.0,0.25,2.,-1.,
         .           'LIFT COEFFICIENT    ')
          CALL YAX(5.,10.,1.,0.,10.0,YMIN,YINC,5.,-1.,LABEL)
```

```
C.... SET PLOTTER LIMITS
      CALL LIM(0.,2.,YMIN,YMAX)
C.. . LOOP AND PLOT DATA
  997 CONTINUE
      TYPE 111
  111 FORMAT('$','Input CL, value, & bound (in percent, 0 to quit): ')
      ACCEPT 112, CL,VALUE,BOUND
  112 FORMAT(3F10.0)
C.... MULTIPLY BOUND BY FACTOR
      BOUND  = BOUND*FACT
      IF ((CL.EQ.0.).AND.(VALUE.EQ.0.)) GO TO 999
C.... PLOT DATA
      CALL PLT(CL,VALUE,+1)
C.... DRAW SYMBOL
      SIZE   = 0.025
      DO 800 I=1,361,18
      ANG    = FLOAT(I)/DGR
      X      = SIZE*(1.)*COS(ANG)
      Y      = SIZE*(YMAX-YMIN)*SIN(ANG)
      CALL PLT(CL+X,VALUE+Y,-2)
  800 CONTINUE
      CALL PEN
C.... DRAW BOUNDS
C.... COMPUTE UPPER BOUND
      DELT   = (BOUND*VALUE/100.)
      IF (VALUE.GE.0.) UPPER  = VALUE+SIZE*(YMAX-YMIN)
      IF (VALUE.LT.0.) UPPER  = VALUE-SIZE*(YMAX-YMIN)
      IF (VALUE.GE.0.) ALOWER = VALUE-SIZE*(YMAX-YMIN)
      IF (VALUE.LT.0.) ALOWER = VALUE+SIZE*(YMAX-YMIN)
      CALL PLT(CL,UPPER,+1)
      CALL PLT(CL,VALUE+DELT,+2)
      CALL PLT(CL-SIZE,VALUE+DELT,+2)
      CALL PLT(CL+SIZE,VALUE+DELT,+2)
      CALL PLT(CL,ALOWER,+1)
      CALL PLT(CL,VALUE-DELT,+2)
      CALL PLT(CL-SIZE,VALUE-DELT,+2)
      CALL PLT(CL+SIZE,VALUE-DELT,+2)
      CALL PEN
      GO TO 997
C.... DONE WITH PLOT GO UP AND REPEAT
  999 CONTINUE
      GO TO 1000
      END
```

PRECEDING PAGE BLANK NOT FILMED

APPENDIX B

MMLE (NEWTON) TEST CASE

This appendix contains the computer output and comparison plots for the MMLE test case A from NASA TND-7831. The comparison is good with the differences of final weighted error (Our 5.7576 x 10^{-2} vs NASA's 5.890 x 10^{-2}) attributed to floating point accuracy (Our 32 bit vs NASA's 60 bit real numbers). The overall derivatives and Cramèr-Rao bounds are compared in Table B.1 below. Figure B.1 shows the time history plots. The dashed lines are the predicted time histories, and the solid lines the measured time histories.

Table B.1 Comparison of MMLE Test Case Results*

	NASA		KU-FRL MMLE	
L_p	-1.015×10^{-1}	(9.0)	-1.2814×10^{-1}	(5.8)
L_r	2.464×10^{0}	(8.5)	2.4643×10^{0}	(8.3)
L_β	$-2.432 \times 10^{+1}$	(0.6)	$-2.4328 \times 10^{+1}$	(0.6)
L_{δ_A}	$1.447 \times 10^{+1}$	(2.5)	$1.4422 \times 10^{+1}$	(2.5)
L_{δ_R}	$1.787 \times 10^{+1}$	(2.2)	$1.7847 \times 10^{+1}$	(2.1)
L_0	4.092×10^{-1}	(1.7)	4.1006×10^{-1}	(1.7)
N_p	4.483×10^{-4}	(162.4)	6.1063×10^{-4}	(100.1)
N_r	-1.514×10^{-1}	(10.8)	-1.2918×10^{-1}	(12.5)
N_β	1.290×10^{0}	(0.8)	1.0243×10^{0}	(0.9)
N_{δ_A}	5.062×10^{-1}	(6.3)	6.7616×10^{-1}	(4.6)
N_{δ_R}	-2.125×10^{0}	(1.8)	-1.9165×10^{0}	(1.9)
N_0	-7.553×10^{-3}	(7.6)	-3.2595×10^{-3}	(17.2)
$\sin\alpha_1$	1.026×10^{-1}	(0.8)	1.1360×10^{-1}	(0.7)
Y_β	-4.670×10^{-2}	(1.2)	-4.6683×10^{-2}	(1.1)
Y_{δ_A}	2.753×10^{-3}	(33.2)	2.7210×10^{-3}	(33.2)
Y_{δ_R}	1.594×10^{-2}	(6.4)	1.5915×10^{-2}	(6.3)
Y_0	-3.035×10^{-3}	(6.9)	-3.0605×10^{-3}	(7.4)
$\dot{\phi}_0$	-8.423×10^{-2}	(-)	-8.5449×10^{-3}	(22.1)
$a_{Y_{bias}}$	0.5179		0.5217	
$\dot{p}_{bias}$	-0.02582		-0.0259	
$\dot{r}_{bias}$	-0.004472		-0.0047	

*Cramèr-Rao bound as a percentage of each derivative is shown in parentheses following the derivative.

ORIGINAL PAGE IS
OF POOR QUALITY

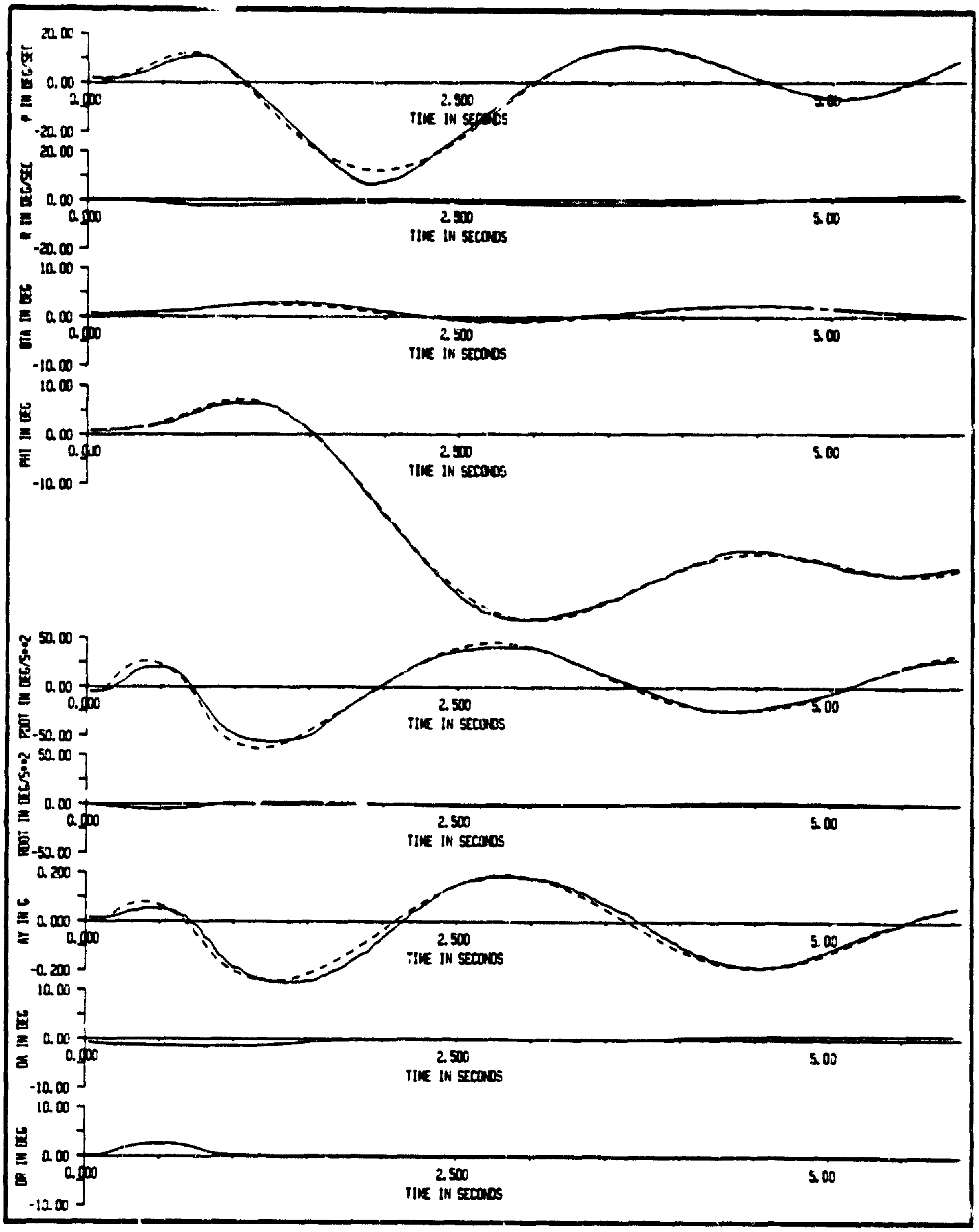


Figure B.1 Flight Time History; NASA Test Case A

Table B.2 MMLE Computer Output; Test Case A

```
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
              KU FRL MMLE RESULTS
              AIRCRAFT A CHECK CASE NASA TND-7831
              PD = 520.  VEL = 4665.  XCG = 0.250
              LATERAL DIRECTIONAL DATA
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
              . . . . . . . .  INITIAL CONDITIONS  . . . . . . . .

NUMBER OF DATA POINTS          :      235     MAXIMUM NUMBER OF ITERATIONS :       5
SUBROUTINE TEST FLAG           :        0     NUMBER OF STATES             :       7
DATA SAMPLING TIME  INTERVAL   :   0.0250     DIAGONAL MULTIFLYING FACTOR  :  1.0000

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

  FLIGHT CONDITION AND VEHICLE CHARACTERISTICS
     DYNAMIC PRESSURE =       216.12        VELOCITY          =    4665.00
     WING AREA        =         0.00        WING SPAN         =       0.00
     WING MAC         =         0.00        IXXB              =   100000.0
     IYYB             =     100000.0        IZZB              =   100000.0
     IXZB             =          0.0        WEIGHT            =       0.00
     INSTRUMENT OFFSETS FROM CG
          X-DIRECTION OFFSETS (+ = INSTR FORWARD OF CG)
               A-X      0.000   A-Y      0.000   A-Z      0.000
          Y-DIRECTION OFFSETS (+ = INSTR RIGHT OF CG)
                                A-Y      0.000
          Z-DIRECTION OFFSETS (+ = INSTR BELOW CG)
               A-X      0.000   A-Y      0.000   A-Z      0.000
          PITCH ANGULAR OFFSET FROM BODY AXES (+ = PITCH UP)
               THETA (MEASURED IN RADIAN UNITS)     0.000

  ZERO AND BIAS CONTROL
  ROLL RATE  YAW RATE    BETA       PHI     ROLL ACCL  YAW ACCEL   A-Y
    0.00000   0.00000   0.00000   0.00000   1.00000   1.00000   1.00000

  DIAGONAL ELEMENTS OF THE WEIGHTING MATRIX D1:
    6.500     4860.000     2160.000     135.000       2.700      198.000      22.650

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
  INITIAL INPUT MATRICES [R], [A], AND [B].
  A STAR (*) FOLLOWING THE VALUE OF A MATRIX ELEMENT INDICATES THAT
  THE RESPECTIVE DERIVATIVE IS NOT ESTIMATED BY THE MMLE METHOD.
  (DERVATIVES ARE DIMENSIONAL)

  MATRIX [R]
   DIMENSION    4 BY   4
    1.0000E+00  0.0000E-01  0.0000E-01  0.0000E-01
    0.0000E-01  1.0000E+00  0.0000E-01  0.0000E-01
    0.0000E-01  0.0000E-01  1.0000E+00  0.0000E-01
    0.0000E-01  0.0000E-01  0.0000E-01  1.0000E+00
```

Table B.2 MMLE Computer Output; Test Case A (continued)

```
STABILITY MATRIX [A]
DIMENSION    4 BY  4
 -2.4100E-01   4.0000E-01  -1.6790E+01   0.0000E-01*
 -2.8400E-03  -4.2000E-02   1.5590E+00   0.0000E-01*
  1.1100E-01  -1.0000E+00* -3.8000E-02   6.9000E-03*
  1.0000E+00*  0.0000E-01*  0.0000E-01*  0.0000E-01*

CONTROL MATRIX [B]
DIMENSION    4 BY  3
  1.2760E+01   2.0080E+01   0.0000E-01
  3.5770E-01  -2.4450E+00   0.0000E-01
  0.0000E-01   1.4800E-02   0.0000E-01
  0.0000E-01*  0.0000E-01*  0.0000E-01
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

ITERATION:   1

ESTIMATES OF THE STATE MATRICES

STABILITY MATRIX [RI]*[A]
DIMENSION    4 BY  4
 -2.4100E-01   4.0000E-01  -1.6790E+01   0.0000E-01*
 -2.8400E-03  -4.2000E-02   1.5500E+00   0.0000E-01*
  1.1100E-01  -1.0000E+00* -3.8000E-02   6.9000E-03*
  1.0000E+00*  0.0000E-01*  0.0000E-01*  0.0000E-01*

CONTROL MATRIX [RI]*[B]
DIMENSION    4 BY  3
  1.2760E+01   2.0080E+01   0.0000E-01
  3.5770E-01   2.4450E+00   0.0000E-01
  0.0000E-01   1.4800E-02   0.0000E-01
  0.0000E-01*  0.0000E-01*  0.0000E-01

WEIGHTED ERROR SUM =    6.8304E+00
WEIGHTED ERRORS:
  1.2313E-01  9.3241E-01  2.2741E-01  5.4451E+00  4.0176E-02  4.0330E-02  2.1859E-02

ITERATION   1  COMPLETED
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

ITERATION:   2

ESTIMATES OF THE STATE MATRICES
```

ORIGINAL PAGE IS
OF POOR QUALITY

Table B.2 MMLE Computer O··put; Test Case A (continued)

```
STABILITY MATRIX [RI]*[A]
DIMENSION   4 BY  4
 -1.6620E-01   2.2610E+00  -2.4224E+01   0.0000E-01*
 -1.4010E-03  -1.1256E-01   1.0268E+00   0.0000E-01*
  1.1299E-01  -1.0000E+00* -4.6874E-02   6.9000E-03*
  1.0000E+00*  0.0000E-01*  0.0000E-01*  0.0000E-01*

CONTROL MATRIX [RI]*[B]
DIMENSION   4 BY  3
  1.4259E+01   1.7340E+01   4.1311E-01
  5.9245E-01  -1.9830E+00  -2.5236E-01
  2.2962E-03   1.5292E-02, -3.0857E-03
  0.0000E-01*  0.0000E-01* -8.5859E-03

  VARIABLE BIAS:        VARIABLE ZERO:
     -0.0248                0.0000
     -0.0047                0.0000
      0.5279                0.0000
      0.0000                0.0000

WEIGHTED ERROR SUM =    4.5788E-01
WEIGHTED ERRORS:
  9.3807E-03  4.9451E-02  1.0704E-02  3.6474E-01  9.7494E-03  6.1867E-03  7.64[illegible]
ITERATION   2  COMPLETED
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

ITERATION:   3

ESTIMATES OF THE STATE MATRICES

STABILITY MATRIX [RI]*[A]
DIMENSION   4 BY  4
 -1.2806E-01   2.4254E+00  -2.4496E+01   0.0000E-01*
  7.2221E-04  -1.2308E-01   1.0303E+00   0.0000E-01*
  1.1266E-01  -1.0000E+00* -4.6991E-02   6.9000E-03*
  1.0000E+00*  0.0000E-01*  0.0000E-01*  0.0000E-01*

CONTROL MATRIX [RI]*[B]
DIMENSION   4 BY  3
  1.4515E+01   1.7949E+01   4.1199E-01
  6.6279E-01  -1.9238E+00  -3.2635E-03
  2.7751E-03   1.5825E-02  -3.0633E-03
  0.0000E-01*  0.0000E-01*  7.9113E-03

  VARIABLE BIAS:        VARIABLE ZERO:
     -0.0259                0.0000
     -0.0047                0.0000
      0.5231                0.0000
      0.0000                0.0000
```

Table B.2 MMLE Computer Output; Test Case A (continued)

```
WEIGHTED ERROR SUM =    5.8079E-02
WEIGHTED ERRORS:
  5.2195E-03  8.5818E-03  9.7688E-03  9.4259E-03  1.1699E-02  5.4488E-03  7.9353E-03

ITERATION   3  COMPLETED
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

ITERATION:   4

ESTIMATES OF THE STATE MATRICES

STABILITY MATRIX [RI]*[A]
DIMENSION    4 BY  4
 -1.2836E-01   2.4563E+00  -2.4323E+01   0.0000E-01*
  6.2883E-04  -1.2889E-01   1.6240E+00   0.0000E-01*
  1.1362E-01  -1.0000E+00* -4.6680E-02   6.9000E-03*
  1.0000E+00*  0.0000E-01*  0.0000E-01*  0.0000E-01*

CONTROL MATRIX [RI]*[B]
DIMENSION    4 BY  3
  1.4423E+01   1.7852E+01   4.0976E-01
  6.7552E-01  -1.9174E+00  -3.2406E-03
  2.6986E-03   1.5905E-02  -3.0583E-03
  0.0000E-01*  0.0000E-01* -8.5366E-03

  VARIABLE BIAS:         VARIABLE ZERO:
      -0.0259               0.0000
      -0.0047               0.0000
       0.5214               0.0000
       0.0000               0.0000
WEIGHTED ERROR SUM =    5.7580E-02
WEIGHTED ERRORS:
  5.2909E-03  9.1528E-03  9.4850E-03  8.6684E-03  1.1589E-02  5.4764E-03  7.9182E-03

ITERATION   4  COMPLETED
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

ITERATION:   5

ESTIMATES OF THE STATE MATRICES

STABILITY MATRIX [RI]*[A]
DIMENSION    4 BY  4
 -1.2814E-01   2.4643E+00  -2.4328E+01   0.0000E-01*
  6.1063E-04  -1.2918E-01   1.0243E+00   0.0000E-01*
  1.1360E-01  -1.0000E+00* -4.6683E-02   6.9000E-03*
  1.0000E+00*  0.0000E-01*  0.0000E-01*  0.0000E-01*
```

Table B.2 MMLE Computer Output; Test Case A (continued)

```
CONTROL MATRIX [RI]=[B]
DIMENSION    4 BY  3
  1.4422E+01   1.7847E+01   4.1006E-01
  6.7616E-01  -1.9165E+00  -3.2595E-03
  2.7210E-03   1.5915E-02  -3.0605E-03
  0.0000E-01*  0.0000E-01* -8.5449E-03

  VARIABLE BIAS:        VARIABLE ZERO:
      -0.0259                0.0000
      -0.0047                0.0000
       0.5217                0.0000
       0.0000                0.0000

WEIGHTED ERROR SUM =    5.7576E-02
WEIGHTED ERRORS:
  5.2859E-03  9.1293E-03  9.4924E-03  8.6499E-03  1.1614E-02  5.4726E-03  7.9319E-03

ITERATION    5  COMPLETED
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

DIMENSIONAL STABILITY AND CONTROL MATRICES:

STABILITY MATRIX [A]
DIMENSION    4 BY  4
 -1.2814E-01   2.4643E+00  -2.4328E+01   0.0000E-01*
  6.1063E-04  -1.2918E-01   1.0243E+00   0.0000E-01*
  1.1350E-01  -1.0000E+00* -4.6683E-02   6.9000E-03*
  1.0000E+00*  0.0000E-01*  0.0000E-01*  0.0000E-01*

CONTROL MATRIX [B]
DIMENSION    4 BY  3
  1.4422E+01   1.7847E+01   4.1006E-01
  6.7616E-01  -1.9165E+00  -3.2595E-03
  2.7210E-03   1.5915E-02  -3.0605E-03
  0.0000E-01*  0.0000E-01* -8.5449E-03

    DIMENSIONAL [AC] MATRIX, (PERCENTAGE OF DERIVATIVE)
 DIMENSION    4 BY   4
  7.4722E-03 (    -5.831)  2.0537E-01 (     8.334)  1.4942E-01 (    -0.614)  0.0000E-01 (     0.000)
  6.1113E-04 (   100.081)  1.6109E-02 (   -12.470)  9.7231E-03 (     0.949)  0.0000E-01 (     0.000)
  8.0406E-04 (     0.708)  0.0000E-01 (     0.000)  5.3513E-04 (    -1.146)  0.0000E-01 (     0.000)
  0.0000E-01 (     0.000)  0.0000E-01 (     0.000)  0.0000E-01 (     0.000)  0.0000E-01 (     0.000)

    DIMENSIONAL [BC] MATRIX, (PERCENTAGE OF DERIVATIVE)
 DIMENSION    4 BY   3
  3.6127E-01 (     2.505)  3.7999E-01 (     2.129)  7.0066E-03 (     1.709)
  3.1392E-02 (     4.643)  3.7317E-02 (    -1.947)  5.6202E-04 (   -17.243)
  9.0448E-04 (    33.241)  1.0090E-03 (     6.340)  2.2526E-04 (    -7.360)
  0.0000E-01 (     0.000)  0.0000E-01 (     0.000)  1.8878E-03 (   -22.092)
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

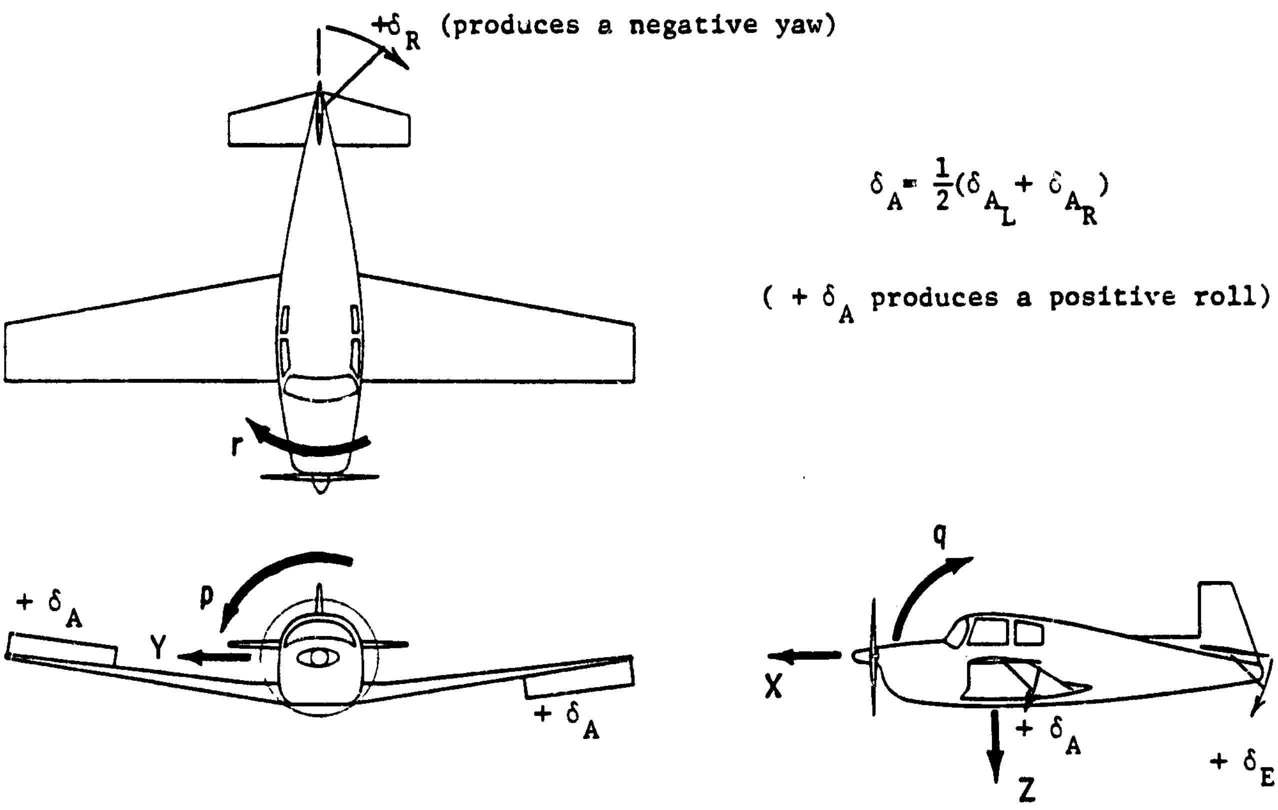
OF POOR QUALITY

APPENDIX C

TRANSFORMATION OF AXES SYSTEMS

This appendix shows the correlation between several axes systems.

Much information contained in this section is taken directly from Reference 6, which deals in depth with the problem of the different axes systems used in airplane analysis.



Note: All directions are shown positive

Figure C.1 Body Axes System Used in This Report

There are primarily five axes systems used in airplane analysis. These are described here.

1) Body Axes

"The orthogonal body-axes system is fixed within the vehicle with the X-axis along the longitudinal center line of the body, the Y-axis normal to the plane of symmetry, and the Z-axis in the plane of symmetry. This is the axes system about which aircraft instruments are usually mounted. Its main advantage in motion calculations is that vehicle moments of inertia about the axes are constant, so that the $\dot{I}$ terms can be omitted from the equations of motion. It is the logical system to which to refer velocities, accelerations, and stability and control parameters in the study of aircraft handling qualities because the pilot's orientation with respect to this frame is fixed."* *(This is the axes system used in this report.)*

2) Principal Axes

"The principal axes are an orthogonal body-fixed system for which the products of inertia are zero. The X and Z principal axes lie in the plane of symmetry; the angle between the X body axis and the X principal axes is usually small so that in many cases the body axes can be assumed to coincide with the principal axes."*

*From Reference 6.

3) Flight Stability Axes

"The flight stability axes (sometimes referred to as vehicle stability axes) are an orthogonal body-axes system fixed to the vehicle, the X-axes of which is alined with the relative wind vector when the vehicle is in a steady-state trim condition but then rotates with the vehicle after a disturbance as the vehicle changes angle of attack. This system is preferred in many stability studies because, as with other body-fixed axes, the moments of inertia about the axes remain constant and also because the motions defined are primarily those about the flight path rather than about body reference lines."*

(This is the axes system used in Reference 25.)

4) Wind-Tunnel Stability Axes

"The wind-tunnel stability axes are the system about which most wind-tunnel data are obtained. For this system the X-axis is in the same horizontal plane as the relative wind at all times . . . . The angle α between the X-axis of this system and the X-body axes is variable. (It is a constant α_0 for the flight stability axes.) This means that vehicle moments of inertia about the X-axis change. It also means that additional terms are required in the transformation equations for static-stability derivatives and for u,v,w derivatives when data are transferred to or from the wind axes or the wind-tunnel stability axes."*

---

*From Reference 6.

5) Wind Axes

"The wind axes are the system generally used in calculating motions of the vehicle as a point mass. The X-axis for this system is alined with the relative wind at all times so that vehicle moments of inertia about this axis change. As with the wind-tunnel stability axes, additional terms . . . are required in the transformation to or from the wind axes and either the body, principal, or flight stability axes, since the angle . . . between the X wind axis and the X-axis of either of these systems is variable. Also, since the lateral angle . . . between the X-axes is variable, there are additional terms . . . required in the transformations for some of the lateral derivatives between the wind axes and either of the other axes systems."*

The correlation between these axes systems is perhaps best summarized by Table C.1.

Table C.1 Designation of Force and Moment Coefficients for Different Axes Systems*

Component	Coefficients for axes system -			
	Body or principal	Flight stability	Wind-tunnel stability	Wind
X-axis force	C_X or $-C_A$	$C_{X,s}$	$-C_D$	$-C_D$
Y-axis force	C_Y	$C_{Y,s}$	C_Y	C_C
Z-axis force	C_Z or $-C_N$	$C_{Z,s}$	$-C_L$	$-C_L$
X-axis moment (roll)	C_ℓ	$C_{\ell,s}$	$C_{\ell,wt}$	$C_{\ell,w}$
Y-axis moment (pitch)	C_m	$C_{m,s}$	$C_{m,wt}$	$C_{m,w}$
Z-axis moment (yaw)	C_n	$C_{n,s}$	$C_{n,wt}$	$C_{n,w}$

*From Reference 6.

Transformation from the flight stability axes (as used in Reference 22) to the body axes used in this report involves accounting for the steady-state angle of attack (α_1). The following equation takes care of this by correcting the inertias. This is the only change required.

$$\begin{bmatrix} I_{xx,s} \\ I_{zz,s} \\ I_{xz,s} \\ I_{yy,s} \end{bmatrix} = \begin{bmatrix} \cos^2\alpha_1 & \sin^2\alpha_1 & (-)\sin^2\alpha_1 & 0 \\ \sin^2\alpha_1 & \cos^2\alpha_1 & \sin^2\alpha_1 & 0 \\ \frac{1}{2}\sin^2\alpha_1 & (-)\frac{1}{2}\sin^2\alpha_1 & \cos^2\alpha_1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I_{xx} \\ I_{zz} \\ I_{xz} \\ I_{yy} \end{bmatrix} \quad [C.1]$$

NOTE: "s" denotes stability axes; no subscript denotes body axes.

NASA Langley (Reference 4) and NASA Dryden (References 5, 16-19) both use the body axes system. They both, however, use different designations. NASA Langley uses the X, Y, Z, ℓ, m, n designation; NASA Dryden, the A, Y, N, ℓ, m, n designation. The parameters will be presented in the X, Y, Z, ℓ, m, n system in this report. Table B.2 shows the correlation between both these systems.

The symbols (i.e., Z_α', etc.) in the definition column of Table C.2 are those as predicted by the MMLE "BONES" program. For conversion from normal stability parameters (as per Reference 25) to these state vector derivatives, the reader is referred back to Tables 5.2 and 5.3.

For rigorous conversion between the various axes systems, the reader is referred to Reference 6.

Table C.2 Comparison of Non-Dimensional Derivatives

LONGITUDINAL

KU-FRL NASA Langley designation	NASA Dryden designation	DEFINITION
C_{Z_α}'	$-C_{N_\alpha}'$	$\frac{Z_\alpha' \, mU_1}{\bar{q}_1 S}$
C_{Z_u}'	$-C_{N_u}'$	$\frac{Z_u' \, mU_1}{\bar{q}_1 S}$
$C_{Z_{\delta_{E,c}}}'$	$-C_{N_{\delta_{E,c}}}$	$\frac{Z_{\delta_{E,c}}' \, mU_1}{\bar{q}_1 S}$
C_{Z_c}'	$-C_{N_o}'$	$\frac{Z_o' \, mU_1}{\bar{q}_1 S}$
C_{m_α}'	C_{m_α}'	$\frac{M_\alpha' \, I_{yy}}{\bar{q}_1 S\bar{c}}$
C_{m_q}'	C_{m_q}'	$\frac{M_q' \, 2U_1 \, I_{yy}}{\bar{q}_1 S\bar{c}^2}$
C_{m_u}'	C_{m_u}'	$\frac{M_u' \, U_1 \, I_{yy}}{\bar{q}_1 S\bar{c}}$
$C_{m_{\delta_{E,c}}}'$	$C_{m_{\delta_{E,c}}}'$	$\frac{M_{\delta_{E,c}}' \, I_{yy}}{\bar{q}_1 S\bar{c}}$
C_{m_o}'	C_{m_o}'	$\frac{M_o' \, I_{yy}}{\bar{q}_1 S\bar{c}}$
C_{X_α}'	$-C_{A_\alpha}'$	$\frac{X_\alpha' \, m}{\bar{q}_1 S}$
C_{X_u}'	$-C_{A_u}'$	$\frac{X_u' \, mU_1}{\bar{q}_1 S}$
$C_{X_{\delta_{E,c}}}'$	$-C_{A_{\delta_{E,c}}}'$	$\frac{X_{\delta_{E,c}}' \, m}{\bar{q}_1 S}$
C_{X_o}'	$-C_{A_o}'$	$\frac{X_o' \, m}{\bar{q}_1 S}$

LATERAL-DIRECTIONAL

KU-FRL NASA-Langley/-Dryden designation	Definition
C_{ℓ_p}'	$\frac{L_p' \, 2I_{xx} \, U_1}{\bar{q}_1 Sb^2}$
C_{ℓ_r}'	$\frac{L_r' \, 2I_{xx} \, U_1}{\bar{q}_1 Sb^2}$
C_{ℓ_β}'	$\frac{L_\beta' \, I_{xx}}{\bar{q}_1 Sb}$
$C_{\ell_{\delta_A}}'$	$\frac{L_{\delta_A}' \, I_{xx}}{\bar{q}_1 Sb}$
$C_{\ell_{\delta_R}}'$	$\frac{L_{\delta_R}' \, I_{xx}}{\bar{q}_1 Sb}$
C_{y_β}'	$\frac{Y_\beta' \, mU_1}{\bar{q}_1 S}$
$C_{y_{\delta_A}}'$	$\frac{Y_{\delta_A}' \, mU_1}{\bar{q}_1 S}$
$C_{y_{\delta_R}}'$	$\frac{Y_{\delta_R}' \, mU_1}{\bar{q}_1 S}$
C_{n_p}'	$\frac{N_p' \, 2U_1 \, I_{zz}}{\bar{q}_1 Sb^2}$
C_{n_r}'	$\frac{N_r' \, 2U_1 \, I_{zz}}{\bar{q}_1 Sb^2}$
C_{n_β}'	$\frac{N_\beta' \, I_{zz}}{\bar{q}_1 Sb}$
$C_{n_{\delta_A}}'$	$\frac{N_{\delta_A}' \, I_{zz}}{\bar{q}_1 Sb}$
$C_{n_{\delta_R}}'$	$\frac{N_{\delta_R}' \, I_{zz}}{\bar{q}_1 Sb}$