NASA CR-165,932

NASA Contractor Report 165932

Development of Flying Qualities Criteria for Single-Pilot Instrument Flight Operations: Interim Report

Aharon Bar-Gill, W. Barry Nixon
and George E. Miller

FLIGHT RESEARCH LABORATORY
PRINCETON UNIVERSITY
Princeton, N.J. 08544

CONTRACT NAS1-15764
June 1982

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665
AC 804 827-3966

NASA Contractor Report 165932

Development of Flying Qualities Criteria for Single-Pilot Instrument Flight Operations: Interim Report

Aharon Bar-Gill, W. Barry Nixon
and George E. Miller

FLIGHT RESEARCH LABORATORY
PRINCETON UNIVERSITY
Princeton, N.J. 08544

CONTRACT NAS1-15764
June 1982

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665
AC 804 827-3966

N82-29288#

ABSTRACT

Research is being conducted to develop flying qualities criteria for Single Pilot Instrument Flight Rule (SPIFR) operations. Significant progress has been made with regard to most of the key issues encompassed in the SPIFR research program. The ARA aircraft has been modified and adapted for SPIFR operations. Aircraft configurations to be flight-tested have been chosen and matched on the ARA in-flight simulator, implementing modern control theory algorithms. Mission planning and experimental matrix design have been completed. Microprocessor software for the onboard data acquisition system has been debugged and flight-tested. Flight-path reconstruction procedure and the associated FORTRAN program are at a final stage of development. Work has begun on algorithms associated with the statistical analysis of flight test results and the SPIFR flying qualities criteria deduction.

PREFACE

This investigation is being conducted by the Flight Research Laboratory at Princeton University, Princeton, New Jersey under Contract No. NAS1-15764 for the NASA Langley Research Center. This is the first annual technical report, and it reflects the SPIFR research effort through May 1981.

The principal investigator for the study is Professor Robert F. Stengel. He is assisted by W. Barry Nixon, senior technical staff member, George E. Miller, technical staff member, Aharon Bar-Gill, graduate student, Thomas O. Williams, technical staff member, Barton C. Reavis, technical associate and electronic technicians Louis Pokrocos, Thomas Frobose and Karl Thomas.

TABLE OF CONTENTS

TABLE OF CONTENTS (cont.)

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

Variables	Description
$\underline{a}_B$	acceleration vector in body axes, "g"
a_x, a_y, a_z	cartesian components of $\underline{a}_B$, "g"
C	implicit model following gain matrix
F	system dynamics matrix
$\underline{f}$	nonlinear functions for vehicle equations of motion
G	control effects matrix
g	gravitational acceleration, ft/sec^2
H	observation matrix transformation matrix
h	altitude, ft
$\underline{h}$	nonlinear measurement functions
I	identity matrix
K	Kalman gain matrix
L	transformation matrix
$M_{(\ )}$	pitch moment stability-and-control derivative
$n_{(\ )}$	random noise associated with the ( ) variable
P	state covariance matrix
p	roll rate, deg/sec
Q	process noise covariance matrix
q	pitch rate, deg/sec
R	measurement noise covariance matrix
r	yaw rate, deg/sec
S	intermediate command output matrix
s	Laplace transform variable
T	duration of flight segment to be reconstructed, sec

t	time, sec
u	x-axis velocity, ft/sec
$\underline{u}$	control vector
$\underline{V}_{air}$	airspeed vector, ft/sec
v	y-axis velocity, ft/sec
$\underline{v}$	measurement noise
w	z-axis velocity, ft/sec
$\underline{w}$	process noise vector
$\underline{W}_I$	wind vector in inertial frame, ft/sec
$X_{(\ )}$	(aerodynamic + thrust)-force along the x-axis derivative
x_I	axial position in inertial frame, ft
$\underline{x}$	state vector
y_I	lateral position in inertial frame, ft
$\underline{Y}$	command vector
$Z_{(\ )}$	(aerodynamic + thrust)-force along the z-axis derivative
z_I	vertical position in inertial frame, ft
$\underline{z}$	observation vector

Variables (Greek)

α	angle of attack, deg
β	angle of sideslip, deg
δE	elevator deflection, deg
δF	flap deflection, deg
δT	throttle deflection, percent
θ	pitch attitude angle, deg
Σ	summation
Φ	state transition matrix
ϕ	roll attitude angle, deg
ψ	yaw attitude angle, deg
$\tilde{\omega}$	body angular rate skew matrix
$\underline{\omega}$	angular rate vector

Superscripts

B	transformation to body axes
I	transformation to inertial axes

Subscripts

A	kinematic model A
ARA	Avionics Research Aircraft
B	body-axis frame
	from body axes (with transformation matrix)
	kinematic model B
	feedback
	backward filter
comm	commanded (desired) value
F	feed forward
I	inertial frame
	from inertial axes (with transformation matrix)
i	navigation station sequencing index
k	sampling instant index
M	model
o	nominal value
q	sensitivity to pitch rate
u	sensitivity to x-axis velocity
w	sensitivity to z-axis velocity
δE	sensitivity to elevator deflection
δF	sensitivity to flap deflection
δT	sensitivity to throttle deflection

Punctuation

$(\dot{\ })$	derivative of quantity with respect to time
$(\underline{\ })$	vector quantity
$\delta(\ )/\delta(\ )$	partial derivative of one variable with respect to another

$\Delta( )$	perturbation variable
$( )^*$	steady-state variable
$( )^T$	transpose of a vector or matrix
$( )^{-1}$	inverse of a matrix
$(\hat{ })$	estimated value of a variable
$( )^\#$	pseudoinverse of a matrix
s( )	sin( )
c( )	cos( )

Acronyms

A/D	analog-to-digital
ADF	automatic direction finder
ARA	Avionics Research Aircraft
CDU	control-display unit
CHR	Cooper-Harper Rating
D/A	digital-to-analog
DME	distance measuring equipment
FBW	fly-by-wire
FRL	Flight Research Laboratory
GA	general aviation
GDOP	geometric dilution of precision
IAS	indicated airspeed
IFR	instrument flight rule
I/O	input/output
POR	pilot opinion rating
PROM	programmable read-only memory
RAM	random access memory
SBC	single-board computer
SPIFR	single-pilot instrument flight rule
TAS	true airspeed
VFR	visual flight rule
VOR	very-high-frequency omni-range

1. INTRODUCTION

1.1 BACKGROUND AND GOALS

This investigation of Single-Pilot Instrument Flight Rule (SPIFR) flying qualities criteria focuses on General Aviation (GA) operations. General Aviation plays an important role in this nation's transportation network (there are about 200,000 active GA aircraft, with the projected number for 1990 being about 300,000), but the difficulty of piloting and the inherent hazards associated with the SPIFR flight regime pose obstacles to continued growth of this mode of transportation (Ref. 1).

An important effect which contributes to an increased hazard for SPIFR operation is the low-frequency dynamic response of a GA aircraft, which does not have to comply with any federal aviation regulation (Ref. 2). As a result, most contemporary GA aircraft have, at best, a marginally stable phugoid mode which may become divergent under wind shear conditions (Ref. 3). This dynamic response problem generally can be coped with under VFR conditions, although it increases pilot's workload significantly. In typical commercial flight, the IFR workload is shared by two pilots; however, GA IFR flight often is controlled by a single pilot. Airframe dynamic deficiencies, finite capabilities of the human operator, and the often limited capabilities of communications and navigation equipment available in typical GA aircraft compound the flight problem under SPIFR conditions.

Prior research has addressed separately various issues, which coupled together, result in this unique flight mission/regime. For example, Ref. 4 and 5 look into the dynamic response characteristics of GA aircraft, Ref. 6 presents the effect of advanced cockpit controls and displays, and Ref. 7 addresses the pilot workload issue. The SPIFR research initiated by the Flight Research Laboratory (FRL) at Princeton University is an integrated theoretical and flight test program, whose principal objectives are:

- To pursue the trends revealed in previous research,
- To develop new methodologies for analysis of complete SPIFR missions,
- To obtain statistically significant flying qualities criteria for single-pilot instrument flight operations.

1.2 ORGANIZATION OF THE REPORT

Chapter 2 describes the preparation of the ARA for SPIFR mission flights and the onboard experimental setup -- in particular, the hardware and software aspects associated with the data acquisition process. Chapter 3 presents theoretical aspects of the SPIFR research, including modern estimation and control theory algorithms for in-flight simulation and flight path reconstruction. Chapter 4 refers to preliminary flights and to the post-flight data preprocessing procedure verification. Conclusions are contained in Chapter 5. The four appendices contain additional theoretical derivations, program listings for onboard and post-flight processing, description of computer systems employed in this research, and the hardware scheme of the unique DME integration into the SPIFR experimental setup.

2. AIRCRAFT AND DATA ACQUISITION SYSTEM PREPARATION

This chapter describes the preparation of the in-flight simulator and of the onboard digital data acquisition system for SPIFR flight testing. Extensive engineering and technical effort was required for aircraft modifications and rewiring, for new avionics system installation, and for onboard experimental setup integration. The results of this effort are summarized in the following sections.

2.1 AIRCRAFT SYSTEM MODIFICATIONS

The Avionics Research Aircraft (ARA) is a Ryan Navion (N5113K) that has been modified into a fly-by-wire (FBW), variable-stability aircraft (Fig. 2-1). It is capable of simulating a variety of other aircraft using feedback control and command augmentation. The ARA is equipped to measure attitude, angular rates, and linear accelerations in three axes, aerodynamic angles (α, β), airspeed, altitude, and a number of other flight variables. Details of the ARA FBW system can be found in Ref. 8.

The evaluation pilot is to fly a SPIFR mission with the ARA responding as a desired configuration. In an emergency, the safety pilot can override the FBW system and take direct control of the aircraft, (Fig. 2-2).

To be used with the SPIFR program, the ARA had to undergo extensive modifications:

- Design and installation of a modular instrument panel.
- Acquisition and installation of a modern navigation/ communication instrument package.
- Addition of secondary workload devices in the cockpit.

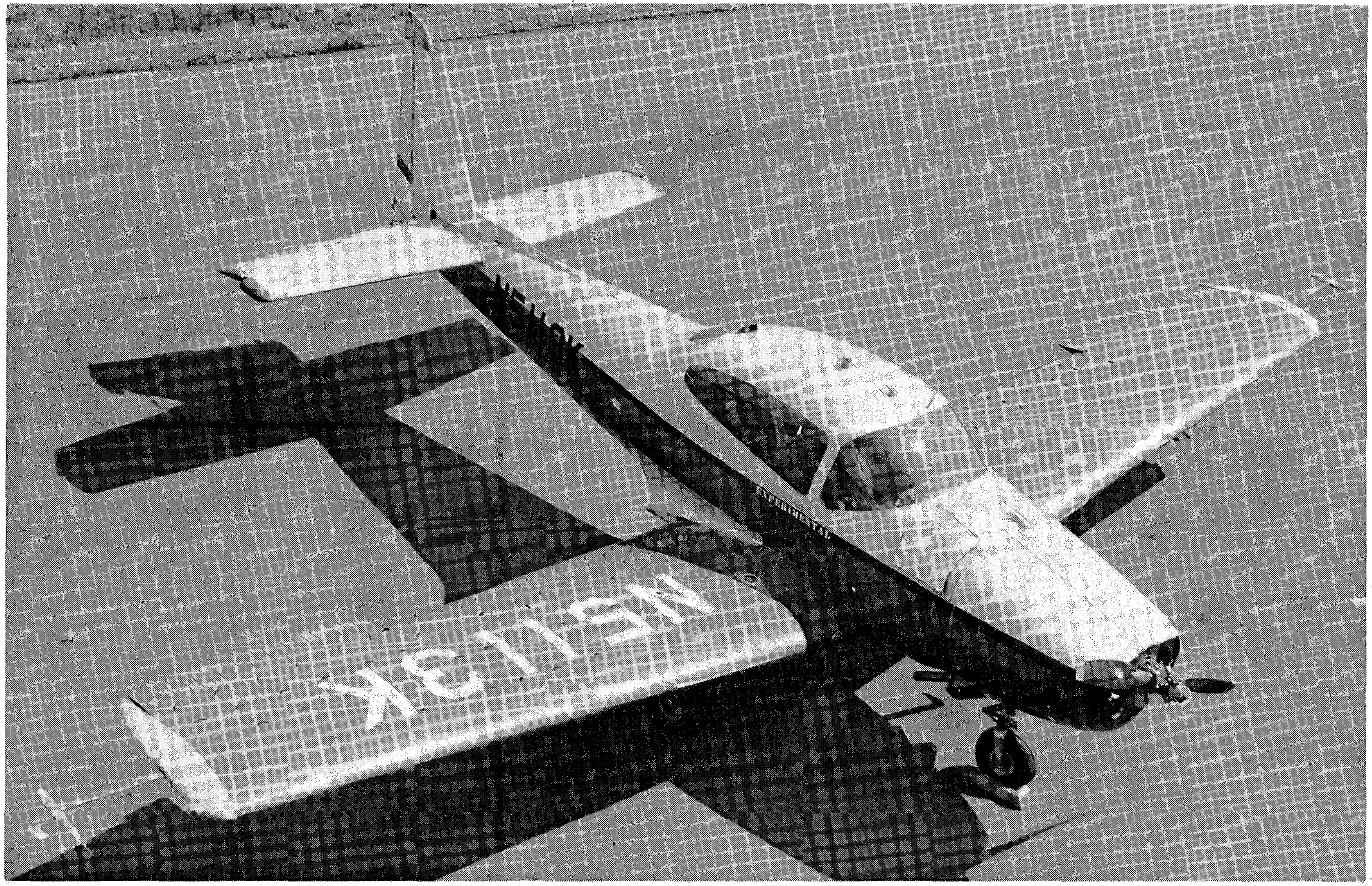


Figure 2-1. Avionics Research Aircraft, Navion N5113K.

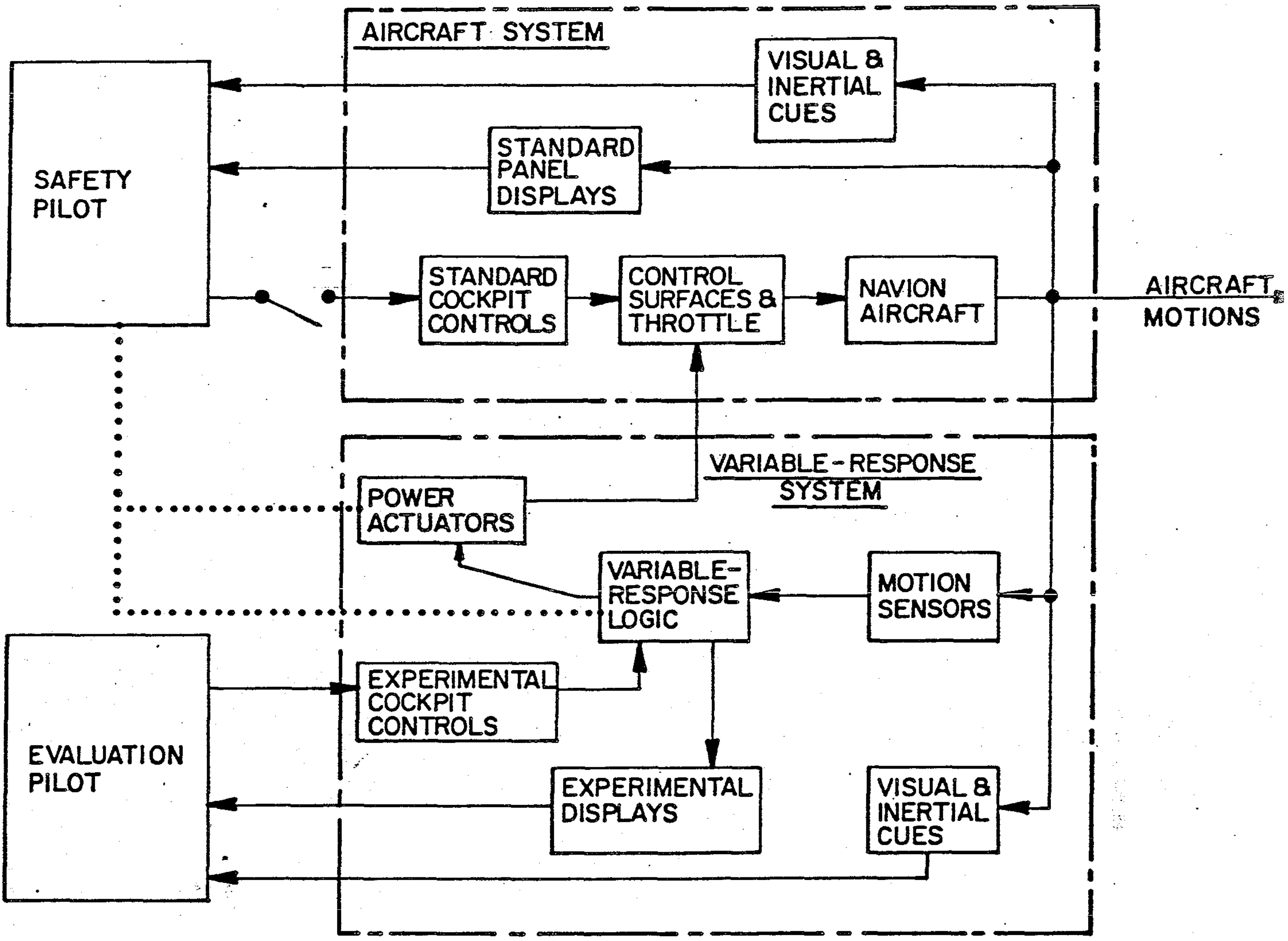


Figure 2-2. Overview of the ARA in-flight Simulator System.

Figure 2-3 illustrates the ARA's modular display panel configuration, with the evaluation pilot's station on the left, the safety pilot's station on the right, and the Bendix BX-2000 navigation/communication stack separating the two. The Distance Measuring Equipment (DME) readout is mounted on a switching panel at the top of the radio stack. The Very-high-frequency Omni-Range (VOR) navigation/communication unit is located under the switching panel. The blank space below this unit is reserved for the Automatic Direction Finder (ADF) and for the transponder.

The DME unit has been integrated into the onboard experimental setup, maintaining the capability to sequence the available navigational stations automatically (through microprocessor control). The importance of this option is discussed in Section 3.5. The technical implementation details are presented in Appendix D.

The safety pilot's panel is a permanent fixture, with conventional instruments and elements for control of the variable-stability system. The latter occupy the right side of the panel and the lower and middle consoles. The evaluation pilot's panel can be removed as a unit to facilitate installation of alternate panels for other investigations. Secondary workload meters, lights, and switches also have been added to the panel.

The secondary workload meters are additional instruments slaved to the onboard microprocessor, which occasionally forces the needles into their "red zones". The evaluation pilot is instructed to keep them "green". Alternately, the pilot can be asked to extinguish lights turned on (pseudorandomly) by the microprocessor program. It is also possible to

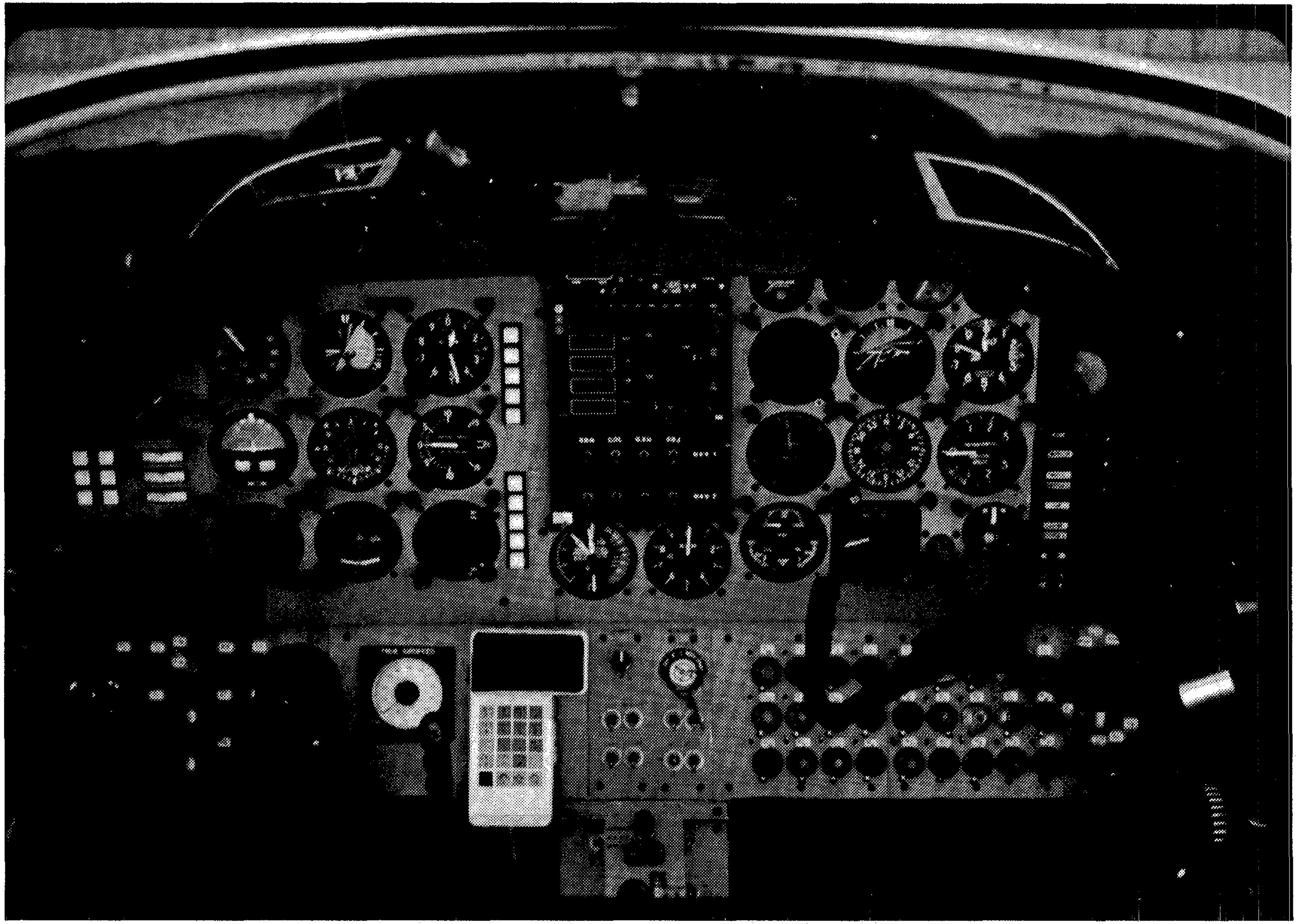


Figure 2-3. Cockpit Displays of the Avionics Research Aircraft. Modular SPIFR Evaluation Pilot Panel at Left.

simulate typical communications workload by blending audio inputs from a pre-recorded tape with specific instructions radioed from the ground on the flight test frequency.

2.2 INSTRUMENTATION AND DATA RECORDING SYSTEM

The SPIFR digital data acquisition system is illustrated in Fig. 2-4. It is built around the SPIFR microcomputer, which uses the Z-80A central processing unit and the Am9511 mathematics processor in a Multibus^{TM} architecture. As currently configured, the SPIFR microcomputer contains 48K bytes of RAM (random access memory) and 16K bytes of PROM (programmable read-only memory). It accepts 32 analog inputs and produces 6 analog outputs.

The ARA's safety pilot communicates with the SPIFR Microcomputer through a hand-held control/display unit (CDU), the Termiflex HT/4. The pilot is able to start and stop processing or recording through the CDU, change stored numerical values, and so on. Conversely, the CDU can display internally triggered error messages to the safety pilot. The evaluation pilot normally is unaware of the SPIFR Microcomputer's operation, other than through secondary workload stimuli and responses.

Analog and digital inputs and outputs shown in Fig. 2-4 are, for the most part, self-explanatory. Tables 2-1 and 2-2 contain lists of inputs and outputs. The SPIFR Microcomputer obtains its analog inputs from the Digital Avionics Research System (DARE) junction box (J-Box) previously installed in the ARA for another Langley Research Center program. Thus, there is a high degree of "plug compatibility" between the SPIFR and DARE programs.

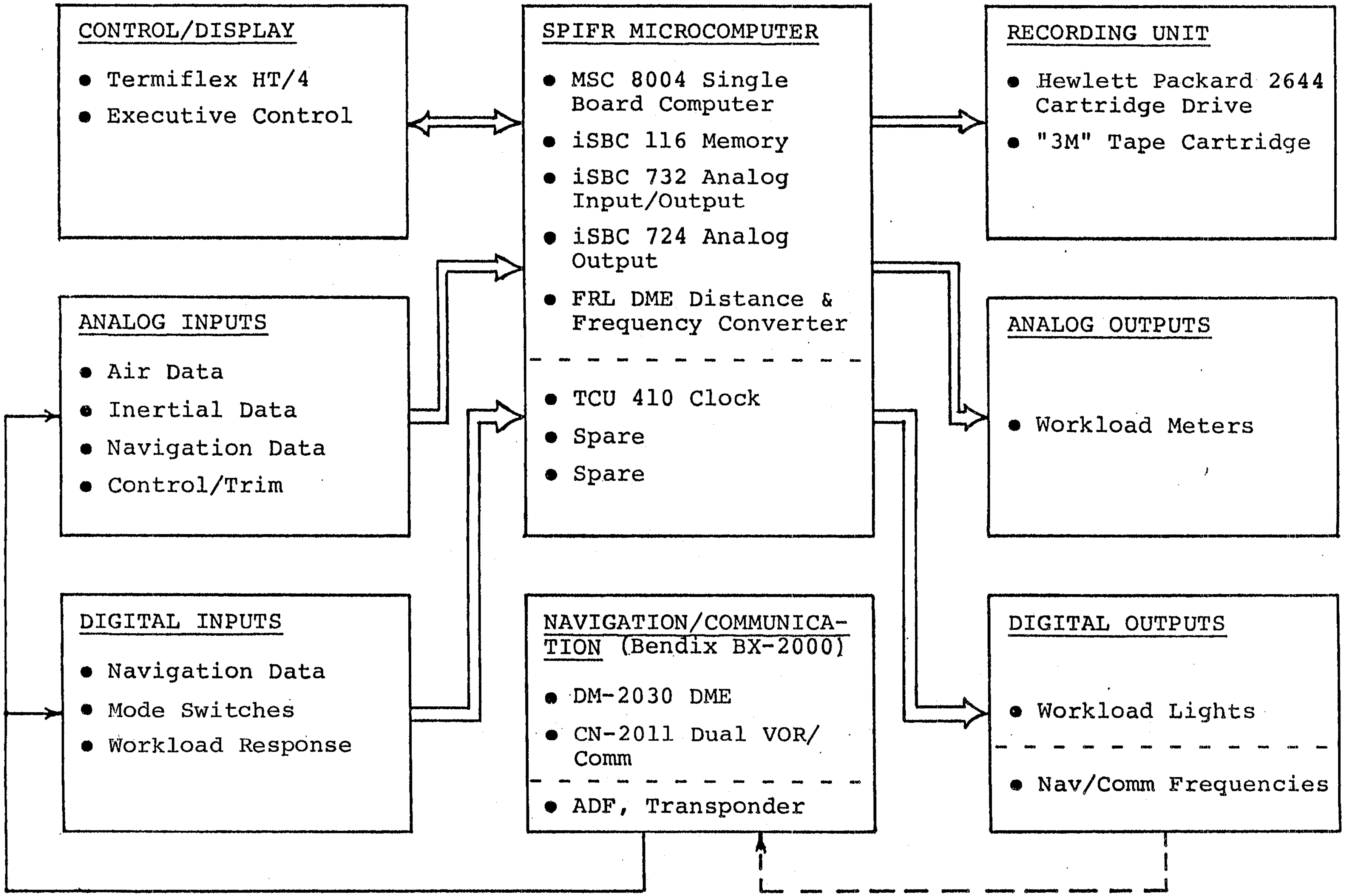


Figure 2-4. SPIFR Digital Data Recording System.

TABLE 2-1

Input Assignments For SPIFR Digital Data Recording System

Analog Inputs

1. Control Column Angle
2. Throttle Setting
3. Flap Command
4. Angle of Attack
5. Pitch Angle
6. Pitch Rate
7. Airspeed
8. Control Wheel Angle
9. Foot Pedals
10. Sideslip Angle
11. Roll Angle
12. Yaw Angle
13. Roll Rate
14. Yaw Rate
15. Pitch Trim
16. Roll Trim
17. Yaw Trim
18. Normal Acceleration
19. Axial Acceleration
20. Lateral Acceleration
21. VOR#1 Azimuth
22. VOR#2 Azimuth
23. Glide Slope Elevation
24. MLS Azimuth
25. MLS Elevation
26. Radar Altitude
27. Barometric Altitude
28. Stick Force (3rd Year)
29. Simulated Turbulence Level
30. Landing Gear
31. Wind Shear
32. System Engage

Digital Inputs

1. DME Distance
2. VOR#1 Frequency
3. VOR#2 Frequency
4. DME Frequency
5. Time
6. Barometric Altitude
7. ADF Bearing
8. Variable-Stability System Status
9. Pilot Mode Switches (8)
10. Avionics System Status (8)

TABLE 2-2

Output Assignments For SPIFR Digital Data Recording System

Analog Outputs

1. Secondary workload meter #1
2. Secondary workload meter #2
3. Secondary workload meter #3
4. Spare
5. Spare
6. Spare

Digital Outputs

1. DME tuning
2. DME station indicator
3. Pilot workload lights
4. Avionics System status lights
5. Tape recorder

A presampling filter (16 Hz break-point frequency) has been introduced for each analog channel to filter out the engine-vibration-induced noise.

Figure 2-4 also illustrates the digital radio tuning feature that will be put to use during the second phase of the project. Error budget analyses conducted during the first phase confirmed the superiority of DME over VOR for position fixing, even at the short ranges to be used in our flight tests. Consequently, it is advantageous to substitute multiple DME measurements for VOR measurements in flight data reduction. The BX-2000 DME unit can acquire and lock on a new station in considerably less than one second; this feature will be used in DME-only "round-robin" position fixing for flight path determination.

The digital tape recording unit is the Hewlett Packard (HP) 2644 terminal, which houses two DC100A magnetic tape cartridge drive units. Its built-in memory enables transition from one cartridge to the other without losing any information. Such a pair of cartridges has a storage capability of about 220K bytes, which is more than enough for a complete SPIFR mission run.

To accomodate the complete experimental setup, a pallet to fit into the ARA-aircraft behind the pilots' seats has been designed and built by the FRL technical staff. It weighs 215 lb. and uses the same mounting brackets as the DARE pallet.

2.3 SOFTWARE DEVELOPMENT

The SPIFR program focuses on the low-frequency dynamic response of the airframe and on navigation-related information, whose rate of change is low as well. On the other hand, as discussed in Section 3.5, simulated SPIFR flight duration has to be about 30 min, during which all the data channels have to be recorded at least every second. Thus, the main objectives of the onboard software design are to:

- Sample the analog data at a high enough rate to avoid aliasing.
- Compress the high frequency data so that the most significant flight test information can be recorded efficiently with minimal error.
- Trigger preprogrammed sequences of the secondary workload devices (lights, dummy meters).
- Enable the safety pilot to operate the data acquisition system via the hand-held CDU.

The information recorded in flight can be separated into "slow" and "fast" variables. The "slow" variables are principally the positional measurements, which can be sampled once per second with minimal aliasing effect. The "fast" variables -- for example, angular rates and linear accelerations -- are sampled 10 times per second. For the sake of data compaction, they are averaged and recorded once each second. The simple averaging process is analogous to "low-pass" filtering. Thus, low-frequency information is passed with little modification, while high-frequency signals are attenuated.

The HP 2644's recording format uses 16-bit binary words. The SBC 732 A/D board is designed to fill in the 12 left-most bit positions of a 16-bit field, and an appropriate shift is performed to comply with the standard output format. Appendix B contains additional detail with regard to the software aspects of the SPIFR onboard data acquisition system, plus the complete listing of the microprocessor assembly program.

3. THEORETICAL ASPECTS OF EXPERIMENT DESIGN AND FLIGHT PATH RECONSTRUCTION

This chapter starts with the presentation of the 6-DOF dynamic model of aircraft motion, as it is applied in the subsequent sections. Section 3.2 discusses the output command algorithm and its implementation to set up a priority list of aircraft configurations to be simulated in the first SPIFR flight test series. The experimental matrix design, based on statistical reasoning,follows in Section 3.3; the application of the chosen configurations on the ARA-in-flight simulator, via the implicit model following algorithm appears in Section 3.4. SPIFR mission planning (Section 3.5) is based mainly on mathematical-statistical modeling of the en-route navigational errors. Finally, the algorithm for post-flight optimal smoothing and flight path reconstruction is presented in Section 3.6.

3.1 AIRCRAFT DYNAMIC MODEL

The general formulation of a nonlinear dynamic model of a system is:

$$\dot{\underline{x}} = \underline{f}\ (\underline{x},\underline{u}) \qquad (3\text{-}1)$$

where $\underline{x}$ is the state vector and $\underline{u}$ is the control vector. The state vector $\underline{x}$ used here contains three components each of translational rate (u,v,w), translational position (x_I, y_I, z_I), angular rate (p, q, r) and angular attitude (ϕ, θ, ψ). Both body and inertial axis frames are taken right-handed and with z pointing downward. The translational rate equation of the aircraft mathematical model is:

$$\dot{\underline{V}}_{air} = \underline{a}_B + \tilde{\omega}\underline{V}_{air} + H_I^B\underline{g}_I \qquad (3\text{-}2)$$

The airspeed, expressed in body axes, is:

$$\underline{V}_{air} = [u \ v \ w]^T \tag{3-3}$$

Acceleration, expressed in body axes, is:

$$\underline{a}_B = [a_x \ a_y \ a_z]^T \tag{3-4}$$

The angular rate cross-product-equivalent matrix $\tilde{\omega}$ is defined as:

$$\tilde{\omega} \triangleq \begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix} \tag{3-5}$$

The gravity vector in an assumed local level/local north inertial axis system is:

$$\underline{g}_I \triangleq [0 \ 0 \ g]^T \tag{3-6}$$

The transformation matrix H_I^B <u>from</u> inertial (I) <u>to</u> body (B) axes, with $[\psi \ \theta \ \phi]$ Euler rotations in the specified order, is:

$$H_I^B \triangleq \begin{bmatrix} c\psi c\theta & s\psi c\theta & -s\theta \\ c\psi s\theta s\phi - s\psi c\phi & s\psi s\phi s\theta + c\psi c\phi & c\theta s\phi \\ c\psi s\theta c\phi + s\psi s\phi & s\psi s\theta c\phi - c\psi s\phi & c\theta c\phi \end{bmatrix} \tag{3-7}$$

where

$$\begin{aligned} s(\ ) &\triangleq \sin(\ ) \\ c(\ ) &\triangleq \cos(\ ) \end{aligned} \qquad (3\text{-}8)$$

The second equation of the aircraft motion 6-DOF mathematical model describes the transformation of body-axis rates to Euler angle rates, and it is

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = L_B^I \begin{bmatrix} p \\ q \\ r \end{bmatrix} \qquad (3\text{-}9)$$

where

$$L_B^I \triangleq \frac{1}{c\theta} \begin{bmatrix} c\theta & s\theta s\phi & s\theta c\phi \\ 0 & c\theta c\phi & -c\theta s\phi \\ 0 & s\phi & c\phi \end{bmatrix} \qquad (3\text{-}10)$$

The third aircraft equation combines the effects of airspeed $\underline{V}_{air}$ and of the wind vector $\underline{W}_I$ (expressed in inertial axes) to compute translational rate:

$$\dot{\underline{x}}_I = H_B^I \underline{V}_{air} + \underline{W}_I \qquad (3\text{-}11)$$

where $\underline{x}_I$ is the position vector expressed in inertial axes:

$$\underline{x}_I \triangleq [x_I \ y_I \ z_I]^T \qquad (3\text{-}12)$$

Based on the orthonormality of H_I^B in eq. (3-7):

$$H_B^I = (H_I^B)^{-1} = (H_I^B)^T \qquad (3\text{-}13)$$

The following relationships constitute the algebraic part of the model, yielding the output or the measurement models. The airspeed absolute value is:

$$|\underline{V}_{air}| = (u^2 + v^2 + w^2)^{1/2} \qquad (3\text{-}14)$$

The angle of attack is given by:

$$\alpha = \tan^{-1}(\frac{w}{u}) \qquad (3\text{-}15)$$

The sideslip angle is:

$$\beta = \tan^{-1}(\frac{v}{u}) \qquad (3\text{-}16)$$

The definition of the aerodynamic angles with respect to body axes is compatible with the actual measurement mechanization in the ARA.* Assuming that the origin of the inertial frame is at sea level, the altitude h is:

$$h = -z_I \qquad (3\text{-}17)$$

The acceleration vector $\underline{a}_B$ of eq. (3-2) and (3-4) reflects the effect of aerodynamic and thrust forces, which act on the airframe. For example, the linearized version of eq. (3-1) for the longitudinal case is:

$$\begin{bmatrix} \Delta\dot{u} \\ \Delta\dot{w} \end{bmatrix}_{\underline{a}_B} = \begin{bmatrix} X_u & X_w & X_q \\ Z_u & Z_w & Z_w \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta w \\ \Delta q \end{bmatrix} + \begin{bmatrix} X_{\delta E} & X_{\delta T} & X_{\delta F} \\ Z_{\delta E} & Z_{\delta T} & Z_{\delta F} \end{bmatrix} \begin{bmatrix} \Delta\delta E \\ \Delta\delta T \\ \Delta\delta F \end{bmatrix} \qquad (3\text{-}18)$$

The control vector here is:

---

* The sideslip angle definition differs from the conventional definition, which is:

$$\beta = \sin^{-1}\left(\frac{v}{|\underline{V}_{air}|}\right)$$

$$\underline{u} = [\Delta\delta E \;\; \Delta\delta T \;\; \Delta\delta F]^T \tag{3-19}$$

where δE is the elevator deflection, δT is the throttle travel and δF is the flap deflection.

To complete this illustration of aerodynamic and thrust effects within the context of longitudinal dynamics, the pitch moment (about the center of gravity of the airplane) equation must be introduced:

$$\Delta\dot{q} = M_u \Delta u + M_w \Delta w + M_q \Delta q + M_{\delta E} \Delta\delta E + M_{\delta T} \Delta\delta T + M_{\delta F} \Delta\delta F \tag{3-20}$$

Combining eq. (3-18) with eq. (3-20) and fully accounting for the physical effects reflected in eq. (3-2) to (3-10), we obtain:

$$\begin{bmatrix} \Delta\dot{u} \\ \Delta\dot{w} \\ \Delta\dot{q} \\ \Delta\dot{\theta} \end{bmatrix} = \begin{bmatrix} X_u & X_w & -w+X_q & -gc\theta \\ Z_u & Z_w & u+Z_q & -gs\theta \\ M_u & M_w & M_q & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta w \\ \Delta q \\ \Delta\theta \end{bmatrix} + \begin{bmatrix} X_{\delta E} & X_{\delta T} & X_{\delta F} \\ Z_{\delta E} & Z_{\delta T} & Z_{\delta F} \\ M_{\delta E} & M_{\delta T} & M_{\delta F} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\delta E \\ \Delta\delta T \\ \Delta\delta F \end{bmatrix} \tag{3-21}$$

Equation (3-21) is of the form of a state equation:

$$\underline{\Delta\dot{x}} = F \, \underline{\Delta x} + G \, \underline{\Delta u} \tag{3-22}$$

where F is the state matrix and G - the control matrix.

3.2 CANDIDATE SPIFR CONFIGURATIONS VIA THE OUTPUT COMMAND ALGORITHM

The basic assumption underlying the following derivation is that if a configuration requires large state and control variations to retrim from one nominal SPIFR flight equilibrium to another, it may also be problematic for the pilot.* Thus, to pick the candidate configurations for SPIFR in-flight simulation, we first choose initial configuration parameters and flight equilibrium. Then we examine the required variations in state and control variables, which correspond to various retrimming requirements. The retrimming requirement may be formulated in terms of flight path variables, e.g., variation in airspeed ΔV^* or in flight path angle $\Delta\gamma^*$.

The mathematical formulation uses the output command algorithm (Ref. 9). The following <u>output</u> equation is added to the state equation eq. (3-22):

$$\underline{\Delta y} = H_x \underline{\Delta x} + H_u \underline{\Delta u} \tag{3-23}$$

where $\underline{\Delta y}$ represents the required retrimming flight-path variations. An ideal transition to the new flight equilibrium is assumed:

$$\left.\begin{aligned} 0 &= F\underline{\Delta x}^* + G\underline{\Delta u}^* \\ \underline{\Delta y}_{comm} &= H_x\underline{\Delta x}^* + H_u\underline{\Delta u}^* \end{aligned}\right\} \tag{3-24}$$

where ()* symbolizes the steady-state variations in state and control that correspond to $\underline{\Delta y}$ comm.

As shown in Ref. 9, the solution to eq. (3-24) is:

---

* As used here, a configuration is a set of aerodynamic coefficients which characterizes the dynamic response of the aircraft.

$$\underline{\Delta x}^{*} = - F^{-1}GS\ \underline{\Delta y}_{comm} \tag{3-25}$$

$$\underline{\Delta u}^{*} = S\ \underline{\Delta y}_{comm} \tag{3-26}$$

where:

$$S \triangleq (-H_{x}F^{-1}G + H_{u})^{-1} \tag{3-27}$$

As a result of application of the output command algorithm, variations in the following aerodynamic parameters have received priority in the context of SPIFR flight testing:

a) X_u , Z_u , Z_α , M_u

b) $Z_{\delta E}$, $Z_{\delta T}$, $M_{\delta T}$

Stability and control derivatives to be varied in the flight tests fall into two categories: those that affect only trim and those that affect both trim and stability. Control derivatives (list in (b) above) fall into the first category because, as demonstrated by eq. (3-21), they appear in the control matrix G, thus affecting $\underline{\Delta x}$* and $\underline{\Delta u}$*. Stability derivatives (list in (a) above) fall into the second category because they appear in the F matrix, thus affecting both trim and stability.

The ranges of variation of the aerodynamic parameters must reflect the trends in GA aircraft design. These are discussed in the context of the experimental matrix design in the next section.

3.3 EXPERIMENTAL MATRIX DESIGN

The high-priority list of configurations of the previous section has been limited to seven configurations,as we must tradeoff between:

- Number of configurations to be flight-tested.
- Number of replications of SPIFR mission with a given configuration (important for statistical soundness).
- Number of evaluation pilots.

All of this must be done under the constraint of about 25 flight hours.

These tradeoff considerations resulted in the following:

- 15 configurations (nominal ARA and plus/minus variations of each of the 7 coefficients).
- Two test pilots plus one GA pilot.
- Numbers of replications are shown, along with all the other information relevant to the experimental matrix, in Table 3.1. The pluses and the minuses to the right of the numbers of replications describe how many positive and how many negative parameter variations (with respect to nominal) are simulated for each of these numbers.

Parameter to be varied	Test pilot 1	Test pilot 2	GA pilot	Number of mission runs
Nominal	2	2	2	6
X_u	2±	2±	3±+	7
Z_u	2±	2±	3±+	7
M_u	2±	2±	3±+	7
Z_α	2±	2±	3±+	7
$M_{\delta T}$	2±	2±	3±+	7
$Z_{\delta E}$	2±	2±	2±	6
$Z_{\delta T}$	2±	2±	2±	6
			Sum =	53

Table 3.1: Experimental Matrix for the First SPIFR Flight Series.

The ranges of the variations in the aerodynamic parameters are intended to reflect possible trends in GA aircraft design. For example, if the design goal is a configuration with a shorter body, an increase in elevator area may be required in order to preserve its moment effectiveness $M_{\delta E}$. Such an area change may affect the vertical force sensitivity of the elevator $\Delta Z_{\delta E}<0$. On the other hand, introduction of a canard control surface may result in $Z_{\delta E}>0$. Another example may be of a configuration design that features a high wing for improved cabin visibility and wing-mounted shrouded propellers for increased thrust. As a result M_u and $M_{\delta T}$ may be affected by the variations in the aerodynamic forces and the moment arms.

3.4 IMPLEMENTATION OF SPIFR CONFIGURATIONS VIA IMPLICIT-MODEL-FOLLOWING ALGORITHM

The chosen SPIFR configurations are implemented on the in-flight simulator via the implicit-model-following algorithm (Ref. 10). State equations of the type of eq. (3-21) may be written for the nominal ARA configuration (subscript ARA) and for the configurations to be simulated (subscript M),

$$\underline{\Delta \dot{x}}_{ARA} = F_{ARA}\underline{\Delta x}_{ARA}+G_{ARA}\underline{\Delta u}_{ARA} \qquad (3\text{-}28)$$

$$\underline{\Delta \dot{x}}_{M} = F_{M}\underline{\Delta x}_{M} + G_{M}\,\underline{\Delta u}_{M} \qquad (3\text{-}29)$$

Our objective is to obtain the control vector $\underline{\Delta u}_{ARA}$, which will make the ARA respond as the required configuration. The perfect model following condition is:

$$\underline{\Delta \dot{x}}_{ARA} = \underline{\Delta \dot{x}}_{M} \tag{3-30}$$

Substituting eq. (3-28) and (3-29) into eq. (3-30) and rearranging:

$$\begin{aligned}\underline{\Delta u}_{ARA} &= G^{\#}_{ARA}[(F_M - F_{ARA})\underline{\Delta x}_{ARA} + G_M\underline{\Delta u}_M] \triangleq \\ &= C_B\underline{\Delta x}_{ARA} + C_F\underline{\Delta u}_M \end{aligned} \tag{3-31}$$

where:

$$G^{\#}_{ARA} \triangleq [G^{T}_{ARA}G_{ARA}]^{-1}G^{T}_{ARA} \tag{3-32}$$

Eq. (3-30) renders:

$$\underline{\Delta x}_{ARA} = \underline{\Delta x}_{M} \tag{3-33}$$

Thus, $\underline{\Delta x}_{ARA}$ is the solution of eq. (3-29). A block diagram of the derived algorithm is presented in Figure 3.1.

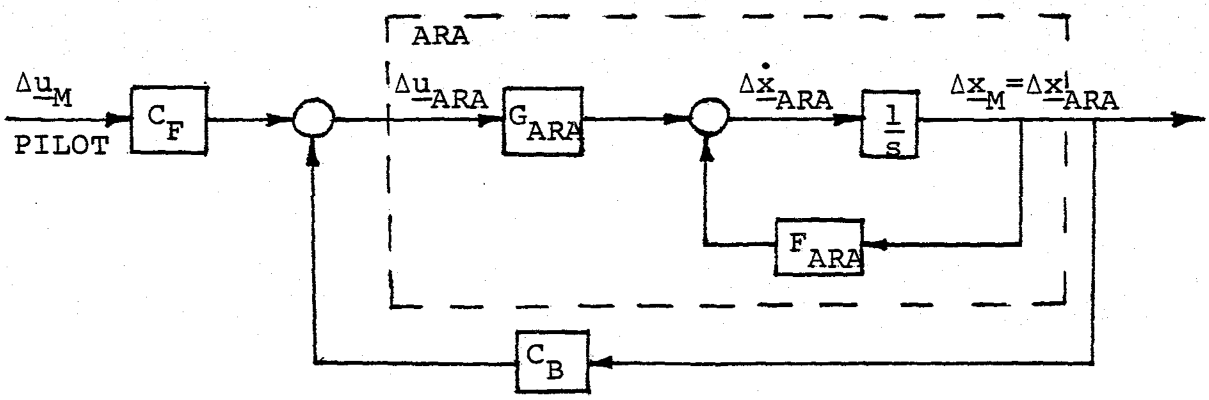


Figure 3.1. Block Diagram for Implementation of a SPIFR Configuration on the ARA In-Flight Simulator.

3.5 EFFECTS OF NAVIGATIONAL ACCURACY AND THE "LEARNING CURVE" EFFECT ON MISSION PLANNING

To simulate realistic SPIFR conditions, the mission has to contain several typical flight-path segments, including

- Climb, acceleration and cruise with airspeed retrimming.
- Holding pattern.
- Deceleration and descent.
- Localizer and glide slope interception.
- Approach and missed-approach go-around.

Also, a realistic VOR-radial navigation should consist of at least once switching navigational stations in the "TO"-mode and of a leg in the "FROM"-mode. The above considerations roughly size the SPIFR mission simulation to a minimum flight duration of about 30 minutes and the geometry shown in Fig. 3-2.

One problem associated with deciding the flight path geometry is the "learning curve" effect. The "learning curve" is the ability of a human being to improve his performance by taking advantage of past experience. Flying all missions along the same trajectory invokes memorization by the pilot,reducing the navigation workload to a level unrealistic for a SPIFR-type mission. To cope with this issue, additional flight path variants have been devised (Fig. 3-3, 3-4, 3-5). All variants are of comparable structure and flight duration.

The other problem associated with the decision of flight path geometry is navigational accuracy. Conclusions of the following discussion with regard to this issue have been implemented with the SPIFR missions of figures 3-2 to 3-5 and, as will be shown, they also contribute to post-flight flight-path reconstruction accuracy improvement.

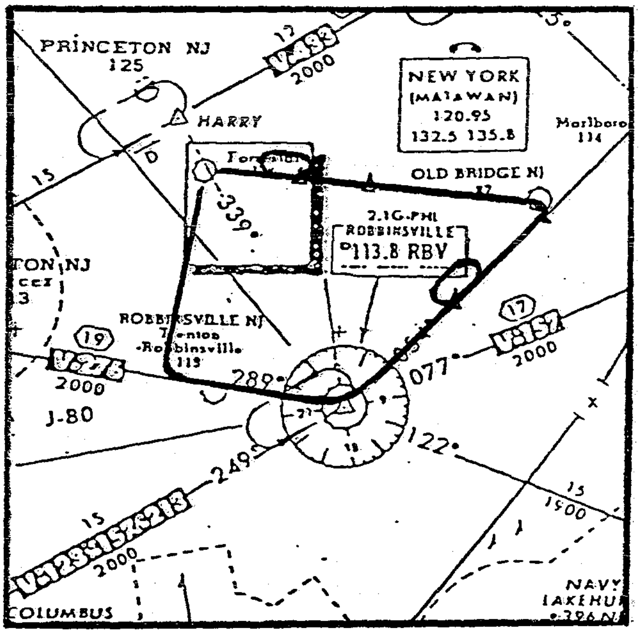


Figure 3-2. Variant I.

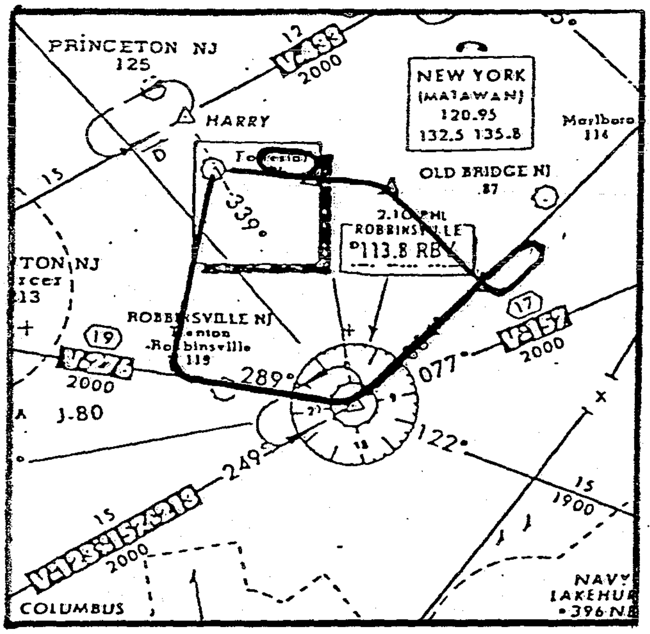


Figure 3-3. Variant II.

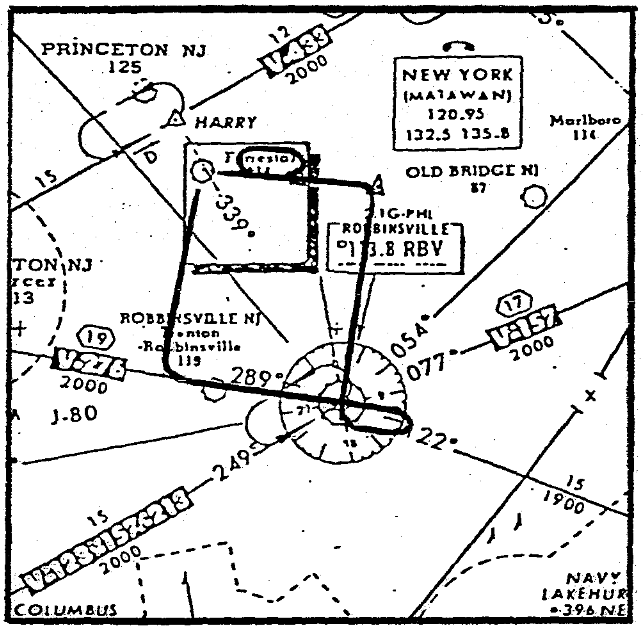


Figure 3-4. Variant III.

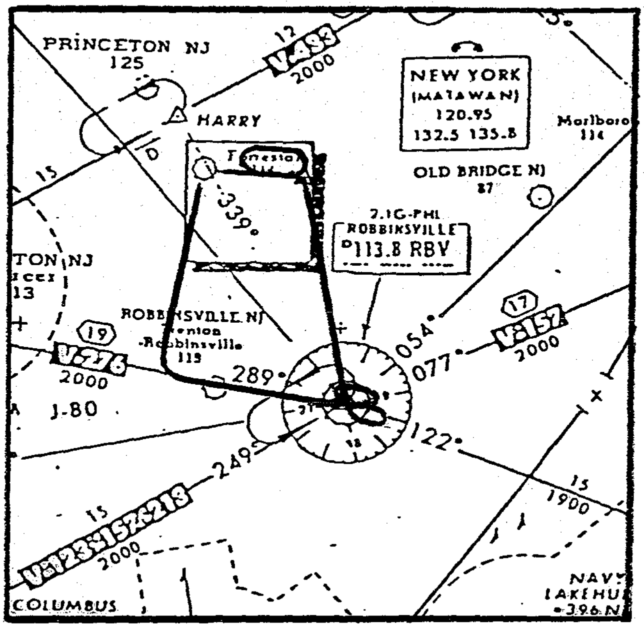


Figure 3-5. Variant IV.

Figures 3-2 to 3-5. SPIFR Flight Path Variants.

The standard navigational modes for GA are VOR/VOR, VOR/DME and DME/DME. At least two navigation stations are required to achieve a horizontal "fix" of the aircraft's position. With proper geometry any of these combinations can provide a fix; however the accuracy of the fix is subject to several factors. The accuracy requirements have been imposed by the Federal Aviation Administration (FAA), and their numerical values appear in Ref. 11.

$$\left.\begin{aligned} \sigma_\psi &\equiv \sigma_{VOR} = 1.9^\circ \\ \sigma_R &= \sigma_{DME} = .15\%\text{ range or }.1\text{ mile: whichever is larger} \end{aligned}\right\} \qquad (3\text{-}34)$$

These navigational errors are with respect to a single ground <u>station</u>. The position errors associated with a navigational <u>mode</u> have to be computed accounting for the Geometric Dilution of Precision (GDOP) effect. GDOP is an inaccuracy due to the nonperpendicularity of the lines connecting the aircraft with the engaged stations.

Applying analytical geometry to the typical situation depicted in Fig. 3-6 <u>and</u> assuming that the two navigational-

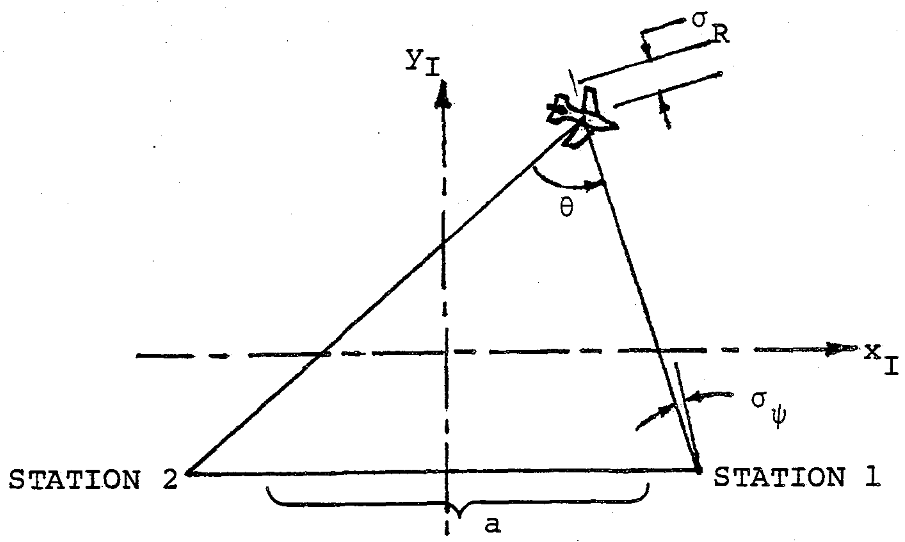


Figure 3-6. Ground Stations Engagement in the VOR/VOR or the DME/DME Modes.

stations' errors are statistically uncorrelated, we obtain:

$$\sigma = (1/s\theta)\,[\sigma_1^2 + \sigma_2^2]^{1/2} \tag{3-35}$$

The $1/s\theta$ term reflects the GDOP effect. For angles between radials in the vicinity of $\theta = 0°$ or $\theta = 180°$, the position error becomes very large, becoming infinite in the limit. To improve position accuracy using two similar ground stations while flying a given leg, it is desirable that the stations be as nearly perpendicular as possible. For VOR/VOR, eq. (3-35) can be rewritten as,

$$\sigma = \frac{a\sigma_\psi}{\sqrt{2}\, s^2\theta}\,(1+s^2\theta)^{1/2} \tag{3-36}$$

For DME/DME, eq. (3-35) becomes

$$\sigma = \sqrt{2}\,\sigma_R/s\theta \tag{3-37}$$

Numerical values of eq. (3-34) and dependence of $\sigma_{VOR/VOR}$ on the $\sin^2\theta$ suggest that this navigational mode is much less accurate than the DME/DME mode. For example, at a range of 50 miles from both stations and for $\theta = 30°$ the VOR/VOR error is 2.71 miles, while the DME/DME error is 0.28 miles and the results favour the DME/DME pairing at greater ranges. Based on this observation and on the feasibility of microprocessor-controlled sequential engagements of several DME stations, this technique will be employed to improve the flight path reconstruction accuracy. In particular, this accuracy improvement may be achieved by making use of redundant measurements, while applying the optimal Kalman filtering/ smoothing algorithm.

3.6 OPTIMAL SMOOTHING OF FLIGHT-TEST RECORDS AND FLIGHT-PATH RECONSTRUCTION

One way to smooth flight test records is to pass the data through a filter, which chops off the high frequency content of the recorded information. A better way is to account for the particular system's dynamic characteristics. This can be done applying the extended Kalman filter algorithm.

For post-flight analysis, even higher accuracy may be achieved by accounting for the "future" information. This improvement is obtained by using an optimal smoothing algorithm. An additional advantage of this algorithm is that it estimates flight path variables that have not been measured directly. Thus, smoothing and flight path reconstruction are obtained via a single algorithm implementation (as in Ref. 12, 13). For this application we need the system state eq. (3-1), which constitutes a concise representation of eq. (3-2) to (3-13),

$$\dot{\underline{x}} = \underline{f}\ (\underline{x},\underline{u}) + \underline{w} \tag{3-38}$$

and the measurement equation,

$$\underline{z} = \underline{h}\ (\underline{x},\underline{u}) + \underline{v} \tag{3-39}$$

which stands for relationships of the type of eq. (3-14) to (3-17). Equation (3-38) differs from eq. (3-1) by the additional term $\underline{w}$, which is referred to in the literature as "process noise". The vector $\underline{v}$ in eq. (3-39) is the "measurement noise".

Equations (3-18) to (3-31) need not be used in the post-flight optimal smoothing and flight-path reconstruction because the accelerations $\underline{a}_B$ are measured directly. Equations (3-2) to (3-17) constitute a kinematic model, as they do not reflect the dynamic mechanism by which $\underline{a}_B$ is actually produced.

The differential equations of the kinematic model (3-2), (3-9) and (3-11) constitute the "state model" and the algebraic relationships (3-14) to (3-17) the "measurement model". Even without accounting for the $\underline{a}_B$ -producing-mechanism, the kinematic model is nonlinear and high-dimensional. Thus, it is more efficient to tackle it in two steps. This is made possible by the fact that the SPIFR experimental setup records both $[p\ q\ r]^T$ and $[\phi\ \theta\ \psi]^T$. The first step is to smooth these six measurements. Treating all six as state variables and using eq. (3-9) we may write:

$$\begin{array}{l}\text{STATE}\\ \text{MODEL A}\end{array}\left\{\begin{aligned} \dot p &= n_{\dot p}\\ \dot q &= n_{\dot q}\\ \dot r &= n_{\dot r}\\ \dot\phi &= p+\tan\theta\,(s\phi*q+c\phi*r)\\ \dot\theta &= c\phi*q - s\phi*r\\ \dot\psi &= \frac{1}{c\theta}(s\phi*q + c\phi*r)\end{aligned}\right. \tag{3-40}$$

$$\begin{array}{l}\text{MEASUREMENT}\\ \text{MODEL A}\end{array}\left\{\ \underline{z}_A = H_A\underline{x}_A + \underline{v}_A\right. \tag{3-41}$$

The state vector for model A is:

$$\underline{x}_A = [p\ q\ r\ \phi\ \theta\ \psi]^T \tag{3-42}$$

The process noise vector (with $n_{\dot p}$, $n_{\dot q}$ and $n_{\dot r}$ random and unknown angular acceleration inputs) is:

$$\underline{w}_A = [n_{\dot p}\ n_{\dot q}\ n_{\dot r}\ 0\ 0\ 0]^T \tag{3-43}$$

The measurement noise vector for model A is:

$$\underline{v}_A = [n_p \; n_q \; n_r \; n_\phi \; n_\theta \; n_\psi]^T \qquad (3\text{-}44)$$

The measurement vector $\underline{z}_A$ in (3-41) contains the measured values of $\underline{x}_A$. Thus, the observation matrix H_A is an identity matrix.

Before elaborating on the optimal smoother algorithm implementation based on eq. (3-40) and (3-41), the kinematic model for the second step is now derived. We assume that the time histories,

$$\hat{p}(t),\ \hat{q}(t),\ \hat{r}(t);\ \hat{\phi}(t),\ \hat{\theta}(t),\ \hat{\psi}(t) \qquad (3\text{-}45)$$

and the associated matrices,

$$\hat{H}^I_B(t) \quad , \quad \hat{H}^B_I(t) \quad , \quad \hat{\underline{\omega}}(t) \qquad (3\text{-}46)$$

are given, having completed the first step. The 6-component state vector for the next step is,

$$\underline{x}_B \doteq [x_I y_I z_I \; u \; v \; w]^T \qquad (3\text{-}47)$$

and the 6-component input vector is,

$$\underline{u}_B = [w_{x_I} w_{y_I} w_{z_I} (a_x - s\hat{\theta}g)(a_y + c\hat{\theta}s\hat{\phi}g)(a_x + c\hat{\theta}c\hat{\phi}g)]^T \qquad (3\text{-}48)$$

The input vector contains components of the <u>true</u> wind and accelerations, $\underline{w}_I$ and $\underline{a}_B$. In this context, the actual values of these measured variables are interpreted as inputs and their measurement inaccuracies as process noise*:

---

* Appendix A presents an improved wind model.

$$\underline{w}_B = [0\ 0\ 0\ n_{a_x}\ n_{a_y}\ n_{a_z}]^T \tag{3-49}$$

Unlike the first step, in which the $\underline{x}_A$ components have been <u>smoothed</u> optimally, this step <u>reconstructs</u> the $\underline{x}_B$ components with eq. (3-2) and (3-11) constituting the state model B:

$$\begin{array}{l}\text{STATE}\\ \text{MODEL B}\end{array} \left\{ \begin{array}{l} \begin{bmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{z}_I \end{bmatrix} = \hat{H}_B^I(t) \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \begin{bmatrix} w_{x_I} \\ w_{y_I} \\ w_{z_I} \end{bmatrix} \\[2ex] \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \hat{\underset{\sim}{\omega}}(t) \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \begin{bmatrix} a_x - s\hat{\theta} g \\ a_y + c\hat{\theta} s\hat{\phi} g \\ a_z + c\hat{\theta} c\hat{\phi} g \end{bmatrix} + \begin{bmatrix} n_{a_x} \\ n_{a_y} \\ n_{a_z} \end{bmatrix} \end{array} \right. \tag{3-50}$$

Equations (3-14) to (3-17) plus VOR and DME measurement equations constitute the measurement model B:

$$\begin{array}{l}\text{MEASUREMENT}\\ \text{MODEL B}\end{array} \left\{ \begin{array}{lcl} |\underline{V}_{air}| & = & (u^2+v^2+w^2)^{1/2} + n_V \\[1ex] \alpha & = & \tan^{-1}(\frac{w}{u}) + n_\alpha \\[1ex] \beta & = & \tan^{-1}(\frac{v}{u}) + n_\beta \\[1ex] h & = & -z_I \quad + n_h \\[1ex] r_{DME_{si}} & = & [(x_I-x_{si})^2+(y_I-y_{si})^2+(z_I-z_{si})^2]^{1/2}+n_{DME} \\[1ex] \theta_{VOR_{si}} & = & tg^{-1}[(y_I-y_{si})/(x_I-x_{si})] + n_{VOR} \end{array} \right. \tag{3-51}$$

where si symbolizes the distance $r_{DME_{si}}$ or angle $\theta_{VOR_{si}}$ being measured with respect to navigational station i. The measurement noise vector is:

$$\underline{v}_B = [n_v \; n_\alpha \; n_\beta \; n_h \; n_{DME_1} \; n_{DME_2} \; \cdots \; n_{VOR_1} \; n_{VOR_2} \cdots]^T \qquad (3\text{-}52)$$

Estimates of measurement biases and scale factor errors may be obtained at the expense of significant increase in state vector dimension. Such an increase in dimension may affect not only the computing cost but also the computational accuracy.

Examination of eq. (3-40) to (3-51) shows that both models A and B are nonlinear. Thus the extended Kalman smoother algorithm has to be applied (Ref. 14). This algorithm is implemented as a combination of forward- and backward-running Kalman filters. The extended Kalman filter algorithm constitutes an adaptation of the linear Kalman filter theory to nonlinear situations. It propagates the nonlinear dynamic model between measurements and utilizes a locally linearized model for the measurement updates.

The following is the discrete formulation of the extended Kalman smoother algorithm, applied to the dynamic model of the system, which constitutes of the state model (eq. (3-38)) and measurement model (eq. (3-39)). The propagation of the estimated states $\hat{\underline{x}}$ and of the state covariance matrix P between measurements, for forward filtering uses

$$\dot{\hat{\underline{x}}}(t) = \underline{f}[\hat{\underline{x}}(t), \underline{u}(t)] \qquad (3\text{-}53)$$

$$P_k(-) = \Phi_{k-1} P_{k-1}(+) \Phi_{k-1}^T + Q_{k-1} \tag{3-54}$$

where Q is the process noise covariance matrix and Φ is the transition matrix obtained after <u>local</u> linearization of eq. (3-38) into:

$$\dot{\underline{x}} = F\underline{x} + G\underline{u} + L\underline{w} \tag{3-55}$$

In order not to create inaccuracies due to numerical differentiation, analytical derivation of the Jacobian matrices (F, G and L) has been carried out for both models A and B; this is documented in Appendix A. The Kalman gain matrix for <u>forward</u> filtering is,

$$K_k = P_k(-) H_k^T [H_k P_k(-) H_k^T + R_k]^{-1} \tag{3-56}$$

where R is the measurement noise covariance matrix and H is obtained by <u>local</u> linearization of eq. (3-39) as

$$\underline{z}_k = H_k \underline{x} + \underline{v}_k \tag{3-57}$$

State and covariance updates account for measurements as

$$\hat{\underline{x}}_k(+) = \hat{\underline{x}}(t) + K_k[\underline{z}_k - \underline{h}_k(\underline{x}(t))] \tag{3-58}$$

$$P_k(+) = [I - K_k H_k] P_k(-) \tag{3-59}$$

where $\hat{\underline{x}}(t)$ is obtained by integration of eq. (3-53) from t_{k-1} to t_k.

The filter processing of the raw data renders the state estimates before the measurement update $\hat{\underline{x}}_k(-) \equiv \hat{\underline{x}}(t)$ and after the measurement update $\hat{\underline{x}}_k(+)$ and the associated covariance matrices $P_k(-)$ and $P_k(+)$. The smoother algorithm uses this information as input and running backwards in time produces the improved estimates of the states $\hat{\underline{x}}_{k/n}$ and of the covariance matrix $P_{k/n}$. The first step is the computation of the state matrix F_k:

$$F_k = f[\hat{\underline{x}}_k(+)] \tag{3-60}$$

The state matrix is used to calculate the state transition matrix Φ_k. The state transition matrix and the input covariance matrices render matrix A_k:

$$A_k = P_k(+)\Phi_k^T P_{k+1}^{-1}(-) \tag{3-61}$$

Using the input state estimates $\hat{\underline{x}}_k(-)$ and $\hat{\underline{x}}_k(+)$ and the associated covariance matrices $P_k(+)$ and $P_k(-)$ along with A_k, the smoothed and reconstructed states $\hat{\underline{x}}_{k/n}$ are obtained:

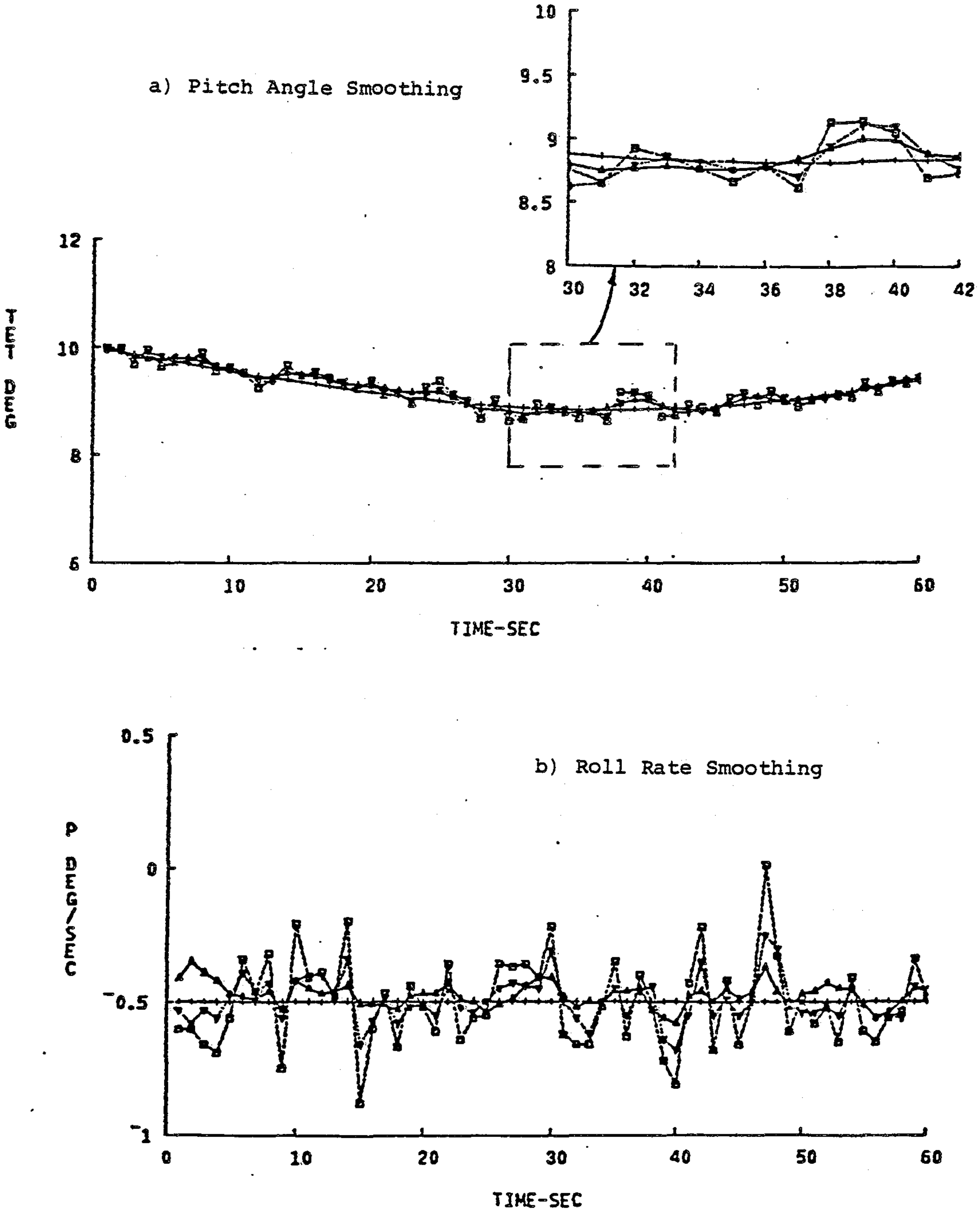
$$\hat{\underline{x}}_{k/n} = \hat{\underline{x}}_k(+) + A_k[\hat{\underline{x}}_{k+1/n} - \hat{\underline{x}}_{k+1}(-)] \tag{3-62}$$

$$P_{k/n} = P_k(+) + A_k[P_{k+1/n} - P_{k+1}(-)]A_k^T \tag{3-63}$$

This complete optimal estimation algorithm, which performs post-flight data smoothing and flight-path reconstruction has been coded in FORTRAN (Appendix B) and verified by application to a computer-generated SPIFR trajectory.

Examples of the optimal flight path reconstruction algorithm's application to the generic flight-test data records are given in Fig. 3-7 for a coordinated climbing turn flight segment of 60 seconds. Figure 3-7a) to f) present reconstructed measurements demonstrating both state variable reconstruction and improvement with respect to data corrupted by noise. The symbol convention used in these figures is: (+) for nominal, (□) for corrupted, (▽) for filtered and (△) for smoothed time histories. Line segments are used to link results but they do not suggest a functional relationship.

Figures 3-7a) and b) represent the optimal smoothing of the angular states. As may have been expected, the "derivative" states (e.g., Fig. 3-7b) are noisier than the "integral" states (e.g., Fig. 3-7a). In a sense, this distinction is applicable to the airspeed versus aerodynamic angle measurements, which reflect the atmospheric turbulence effect. As follows from the translational submodel formulation, to reconstruct these measurements (e.g., Fig. 3-7c) and d)), the states u, v and w are first estimated. The typical lag introduced by filtering is more apparent in some of the figures; it is then reduced by the smoother. The trajectory reconstruction is represented in Fig. 3-7e), f) and g). Note that optimal smoothing improves the filtered state estimates and also shrinks the position uncertainty ellipsoid.



a) Pitch Angle Smoothing

b) Roll Rate Smoothing

Figure 3-7: Examples of application of the optimal flight path reconstruction algorithm to the climbing turn pseudo-flight-test data.

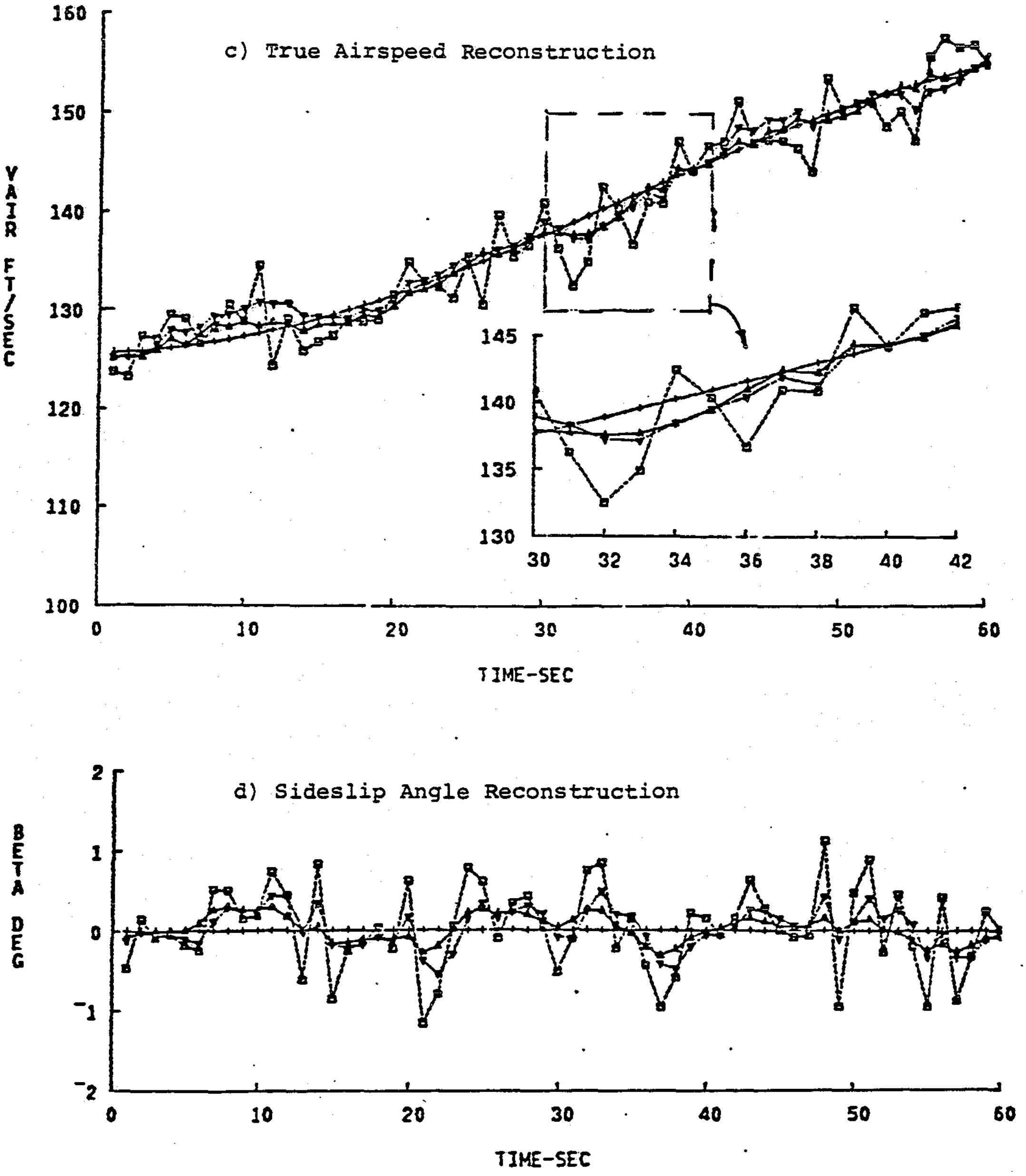


Figure 3-7: Examples of application of the optimal flight path (cont.) reconstruction algorithm to the climbing turn pseudo-flight-test data.

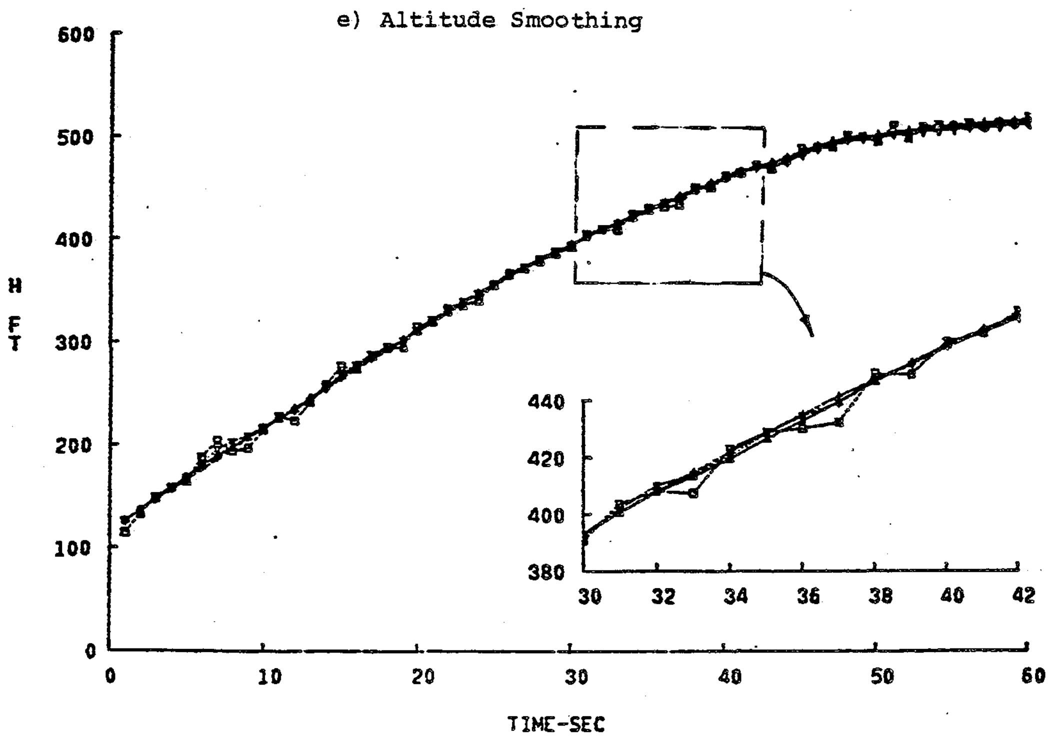
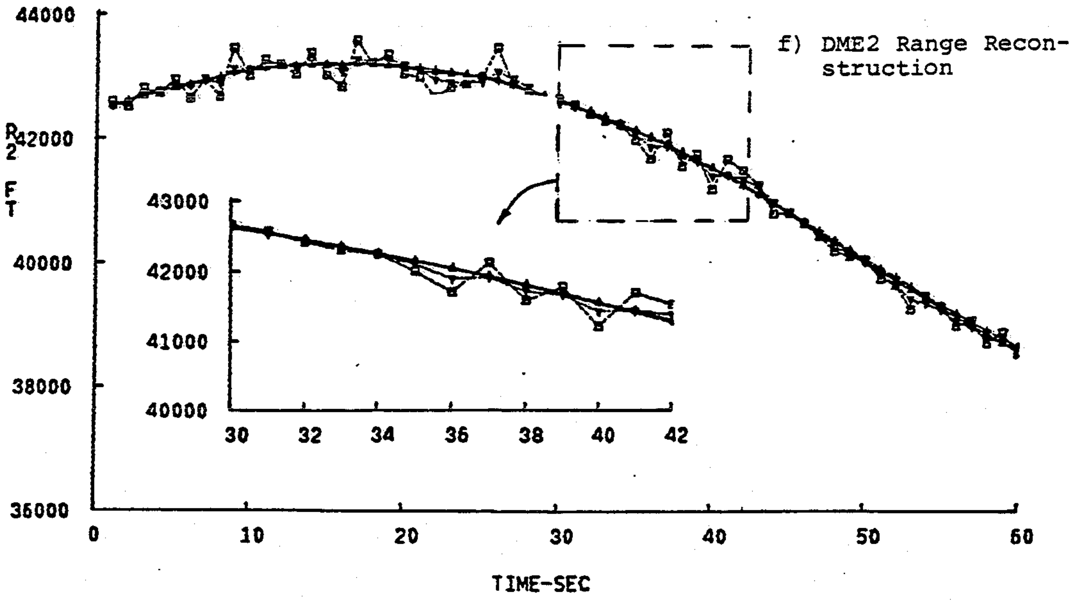
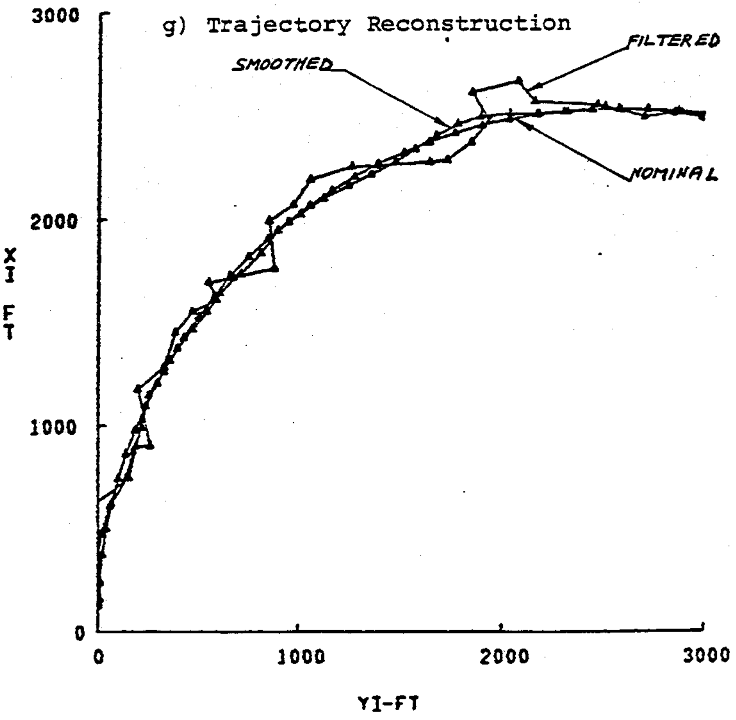


Figure 3-7: (cont.) Examples of application of the optimal flight path reconstruction algorithm to the climbing turn pseudo-flight-test data.



f) DME2 Range Reconstruction



g) Trajectory Reconstruction

Figure 3-7: (cont.) Examples of application of the optimal flight path reconstruction algorithm to the climbing turn pseudo-flight-test data.

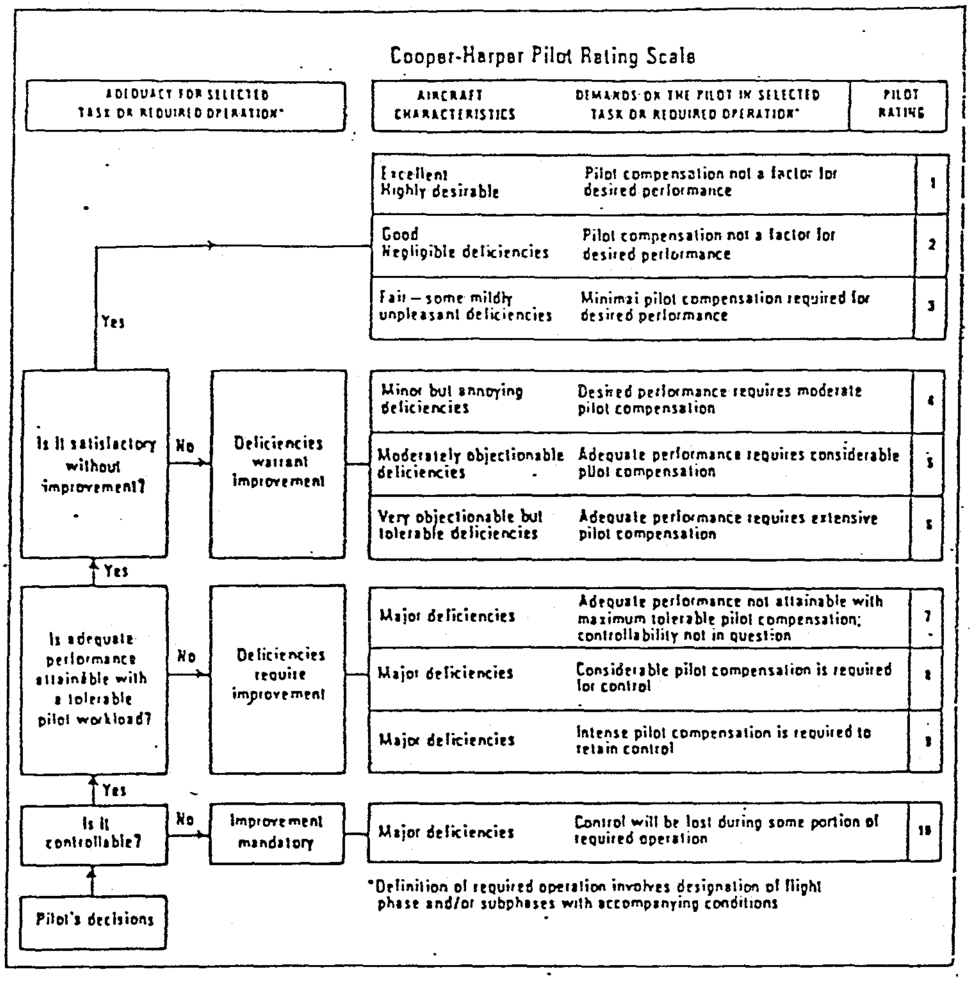
4. PRELIMINARY FLIGHTS

Flight test results are the important objectives of the SPIFR program. As the human operator is an integral part of the control and guidance loop, Pilot Opinion Ratings (POR) constitute important experimental results. Both the Cooper-Harper Rating (CHR), which is a performance rating (Ref. 15) and the workload rating (M.I.T. scale, Ref. 15) are significant. As we debrief the evaluation pilot with regard to both the complete mission and to its specific segments, knee-pad-size versions of both scales and of the grading sheet have been prepared (Fig. 4-1).

To test the complete SPIFR-mission-simulation concept, a series of preliminary flights has been carried out. Its main objectives were to verify the realism of simulation of SPIFR regime environment, the in-flight configuration matching capability and the data acquision and reduction process.

After extensive hangar checks of the aircraft system modifications, of the new navigation/communication package and of the onboard experimental setup, the proposed instrument tracks (Fig. 3-2 to 3-5) were flown - totalling to about 10 flight hours.

These preliminary flights have shown that the tasks appear to simulate SPIFR missions, which are realistic in both geometry and workload. Using the knee-pad-size POR scales and grading sheet the in-flight debriefing can be carried out without interfering with the mission. The in-flight configuration matching capability with regard to each

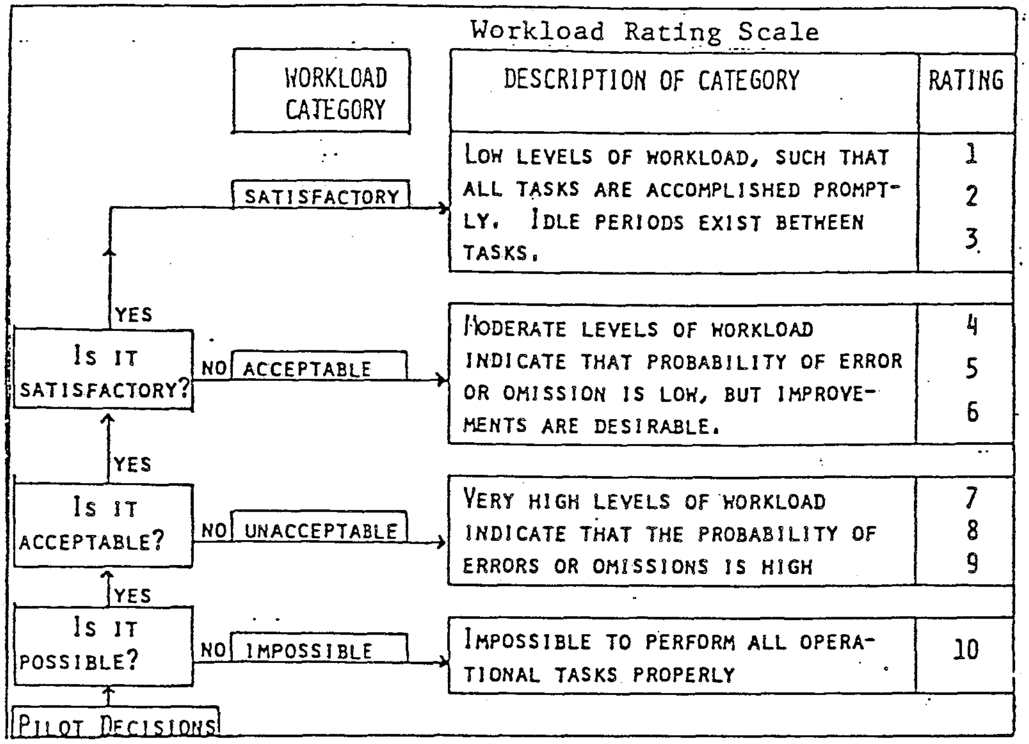


AIRCRAFT CHARACTERISTICS	DEMANDS ON THE PILOT IN SELECTED TASK OR REQUIRED OPERATION*	PILOT RATING
Excellent Highly desirable	Pilot compensation not a factor for desired performance	1
Good Negligible deficiencies	Pilot compensation not a factor for desired performance	2
Fair – some mildly unpleasant deficiencies	Minimal pilot compensation required for desired performance	3
Minor but annoying deficiencies	Desired performance requires moderate pilot compensation	4
Moderately objectionable deficiencies	Adequate performance requires considerable pilot compensation	5
Very objectionable but tolerable deficiencies	Adequate performance requires extensive pilot compensation	6
Major deficiencies	Adequate performance not attainable with maximum tolerable pilot compensation; controllability not in question	7
Major deficiencies	Considerable pilot compensation is required for control	8
Major deficiencies	Intense pilot compensation is required to retain control	9
Major deficiencies	Control will be lost during some portion of required operation	10

*Definition of required operation involves designation of flight phase and/or subphases with accompanying conditions

a) Performance POR Scale

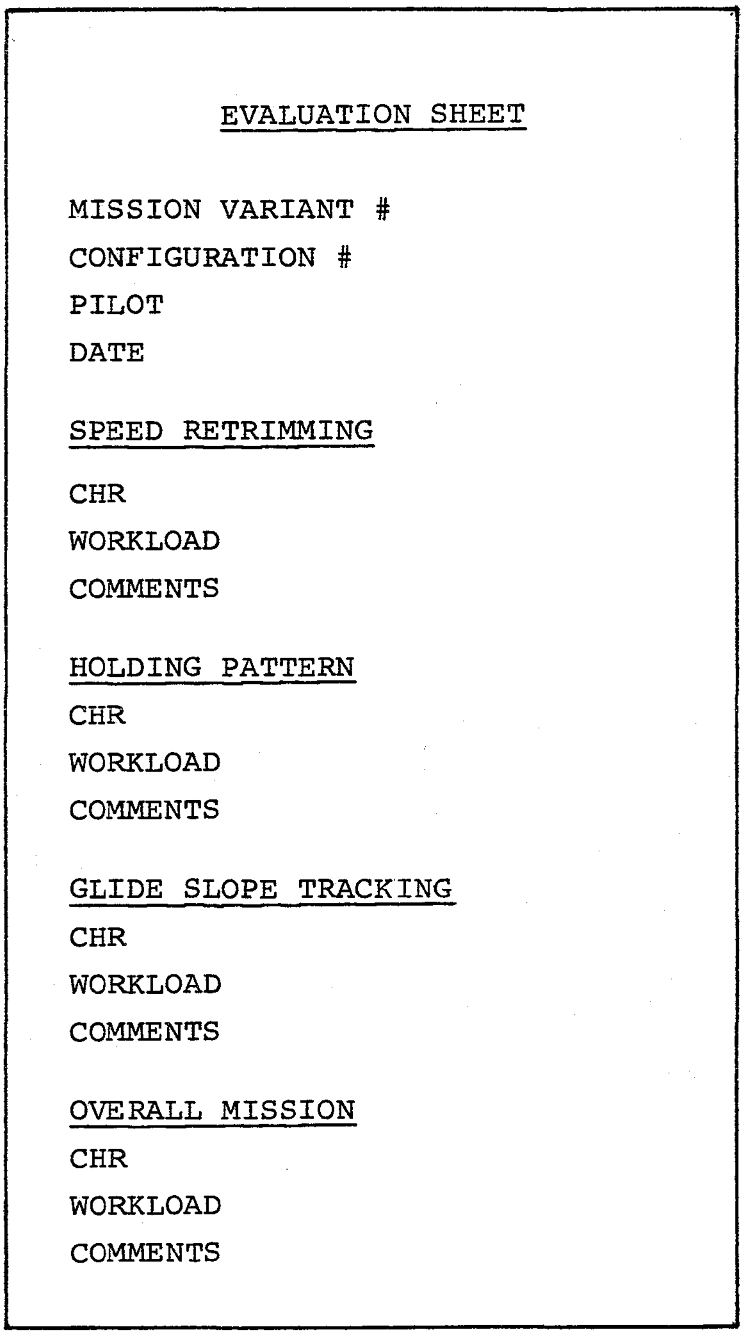
Figure 4-1: Knee-pad Versions of the Performance and Workload PORs and of the Evalution Sheet.



WORKLOAD CATEGORY	DESCRIPTION OF CATEGORY	RATING
SATISFACTORY	LOW LEVELS OF WORKLOAD, SUCH THAT ALL TASKS ARE ACCOMPLISHED PROMPTLY. IDLE PERIODS EXIST BETWEEN TASKS.	1 2 3
ACCEPTABLE	MODERATE LEVELS OF WORKLOAD INDICATE THAT PROBABILITY OF ERROR OR OMISSION IS LOW, BUT IMPROVEMENTS ARE DESIRABLE.	4 5 6
UNACCEPTABLE	VERY HIGH LEVELS OF WORKLOAD INDICATE THAT THE PROBABILITY OF ERRORS OR OMISSIONS IS HIGH	7 8 9
IMPOSSIBLE	IMPOSSIBLE TO PERFORM ALL OPERATIONAL TASKS PROPERLY	10

b) Workload POR Scale

Figure 4-1: (cont.) Knee-pad Versions of the Performance and Workload PORs and of the Evalution Sheet.



EVALUATION SHEET

MISSION VARIANT #
CONFIGURATION #
PILOT
DATE

SPEED RETRIMMING

CHR
WORKLOAD
COMMENTS

HOLDING PATTERN

CHR
WORKLOAD
COMMENTS

GLIDE SLOPE TRACKING

CHR
WORKLOAD
COMMENTS

OVERALL MISSION

CHR
WORKLOAD
COMMENTS

c) Evaluation Sheet.

Figure 4-1: (cont.) Knee-pad Versions of the Performance and Workload PORs and of the Evalution Sheet.

of the priority configurations has been confirmed. The data collection and reduction process has been verified by comparison of timings and directions of deflection of various controls reported by the safety pilot to those obtained by recorded data processing.

5. CONCLUSION

This report summarizes the first phase of the SPIFR project, which constitutes an integrated theoretical and flight-test research effort, which addresses stability-and-control, avionics and human factor effects in single pilot IFR situations. The first phase activities were aimed at basic research system preparation and, consequently, at conducting the first flight test series of the SPIFR four-year program.

Most of the goals of the first phase of the SPIFR program have been achieved. The basic research system has been put together and successfully flight-tested, including the ARA aircraft modifications and the onboard digital data acquisition system. A modern navigation/communication system has been installed, and a new instrument panel has been designed to accomodate flexibility in introduction of additional instrumentation and workload devices. The data collection system has been built around the Z-80 microprocessor. The microprocessor performs the analog channels sampling, preliminary processing and transfer of the data records to the digital recorder. The software required for these onboard manipulations has been developed, debugged and flight-tested. In parallel, the post-flight data preprocessing software and procedure development has been completed and tried out with actual in-flight recorded data. Mission planning and experimental matrix design have been carried out accounting for navigational accuracy and the "learning curve effect" and for the amount-of-flight-hours-constraint. Applying theoretical algorithms the aerodynamic configurations for the

SPIFR program have been obtained and implemented on the ARA aircraft. A preliminary flight-test series has been conducted to check whether realistic SPIFR conditions are obtained and to verify the in-flight configurations matching. These flight tests have confirmed that the SPIFR mission simulation is realistic both in geometry and in workload.

During the second phase of the SPIFR project, the first flight series will be completed and the collected data will be analysed, including the subjective PORs. Finally, statistical regression analysis will be applied to render flying qualities criteria for Single Pilot Instrument Flight Rule operations.

The structure of the research program as summarized in this report will render quantitative criteria with regard to the effects of airframe dynamic response, workload level, and pilot experience on the SPIFR flight regime. Furthermore, the ARA is now ready to conduct a broad range of additional flight experiments associated with single-pilot instrument flight.

APPENDIX A

DERIVATION OF THE LINEARIZED VERSIONS OF THE SIMPLIFIED AND THE IMPROVED KINEMATIC MODELS

A.1 IMPROVED WIND MODEL

The kinematic state model B (eq. (3-50)) assumes that ideal, constant wind measurements $\underline{w}_I$ are available along the flight path. Although we may disregard the high-frequency turbulence disturbances, it is difficult to obtain wind measurements along a flight path with an acceptable degree of reliability. A solution is to adjoin a wind model to state model B, estimating its parameters along with $\underline{x}_B$. A reasonable low-frequency wind model may constitute of a constant component and a linear variation with altitude,

$$\underline{w}_I(h) = \underline{w}_{I_o} + \frac{\partial \underline{w}_I}{\partial z_I} (z_I - z_{I_o}) \qquad \text{(A-1)}$$

where $\underline{w}_{I_o}$ is a constant wind vector at reference altitude $-z_{I_o}$ and $\frac{\partial \underline{w}_I}{\partial z_I}$ is a vector of slope of wind variation with altitude. Adjoining eq. (A-1) to eq. (3-50), we obtain:

$$\begin{bmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{z}_I \end{bmatrix} = \hat{H}_B^I(t) \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \begin{bmatrix} w_{x_{I_o}} + \frac{\partial w_{x_I}}{\partial z_I} (z_I - z_{I_o}) \\ w_{y_{I_o}} + \frac{\partial w_{y_I}}{\partial z_I} (z_I - z_{I_o}) \\ w_{z_{I_o}} + \frac{\partial w_{z_I}}{\partial z_I} (z_I - z_{I_o}) \end{bmatrix} \tag{A-2}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \hat{\tilde{\omega}}(t) \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \begin{bmatrix} a_x - s\hat{\theta}\, g \\ a_y + c\hat{\theta} s\hat{\Phi} g \\ a_z + c\hat{\theta} c\hat{\phi} g \end{bmatrix} + \begin{bmatrix} n_{a_x} \\ n_{a_y} \\ n_{a_z} \end{bmatrix} + \hat{H}_I^B(t) * \frac{\partial \underline{w}_I}{\partial z_I} * [\dot{z}_I] \tag{A-3}$$

Carrying through the mathematical steps necessary to transfer the state variable z_I derivative to the left-hand side,we obtain the state model B in the following form:

$$\begin{bmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{z}_I \end{bmatrix} = \hat{H}_B^I(t) \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \begin{bmatrix} w_{x_{I_o}} + \frac{\partial w_{x_I}}{\partial z_I} (z_I - z_{I_o}) \\ w_{y_{I_o}} + \frac{\partial w_{y_I}}{\partial z_I} (z_I - z_{I_o}) \\ w_{z_{I_o}} + \frac{\partial w_{z_I}}{\partial z_I} (z_I - z_{I_o}) \end{bmatrix} \tag{A-4}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \hat{\tilde{\omega}}(t) \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \begin{bmatrix} a_x - s\hat{\theta}\, g \\ a_y + c\hat{\theta} s\hat{\phi} g \\ a_z + c\hat{\theta} c\hat{\phi} g \end{bmatrix} + \begin{bmatrix} n_{a_x} \\ n_{a_y} \\ n_{a_z} \end{bmatrix} + H_{uvw} * \hat{H}_I^B(t) \frac{\partial \underline{w}_I}{\partial z_I} \tag{A-5}$$

$$\begin{bmatrix} \dot{w}_{x_{I_o}} \\ \dot{w}_{y_{I_o}} \\ \dot{w}_{z_{I_o}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{A-6}$$

$$\begin{bmatrix} \dfrac{\partial \dot{w}_{x_I}}{\partial z_I} \\ \dfrac{\partial \dot{w}_{y_I}}{\partial z_I} \\ \dfrac{\partial \dot{w}_{z_I}}{\partial z_I} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{A-7}$$

$$H_{uvw} \triangleq [\hat{H}^I_{B_{31}} \quad \hat{H}^I_{B_{32}} \quad \hat{H}^I_{B_{33}}] \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{A-8}$$

The kinematic state model B of eq. (A-2) to (A-8) renders improved flight path reconstruction and an estimate of constant wind and of wind gradient along the flight path.

A.2 LINEARIZATION OF THE SIMPLIFIED KINEMATIC MODEL

To apply the extended Kalman filter algorithm the non-linear kinematic model has to be linearized. The linearized version of the simplified kinematic model has been derived analytically and is presented in this section. The result for state model A [eq. (3-40)] is:

$$\underset{\dot{\underline{x}}_A}{\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}} = \underset{F_A}{\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & tg\theta s\phi & tg\theta c\phi & qtg\theta c\phi - rtg\theta s\phi & (qs\phi + rc\phi)/c^2\theta & 0 \\ 0 & c\phi & -s\phi & -qs\phi - rc\phi & 0 & 0 \\ 0 & s\phi/c\theta & c\phi/c\theta & qc\phi/c\theta - rs\phi/c\theta & (qs\phi + rc\phi)\dfrac{s\theta}{c^2\theta} & 0 \end{bmatrix}} \underset{\underline{x}_A}{\begin{bmatrix} p \\ q \\ r \\ q \\ \theta \\ \psi \end{bmatrix}} + \underset{\underline{w}_A}{\begin{bmatrix} n_{\dot{p}} \\ n_{\dot{q}} \\ n_{\dot{r}} \\ 0 \\ 0 \\ 0 \end{bmatrix}} \qquad \text{(A-9)}$$

Rearranging eq. (3-50) of state model B:

$$\underset{\dot{\underline{x}}_B}{\begin{bmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{z}_I \\ \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix}} = \underset{F_B}{\left[\begin{array}{c|c} & \hat{H}_B^I(t) \\ \underset{6x3}{0} & \\ \cline{2-2} & \hat{\tilde{\omega}}(t) \end{array}\right]} \underset{\underline{x}_B}{\begin{bmatrix} x_I \\ y_I \\ z_I \\ u \\ v \\ w \end{bmatrix}} + \underset{G_B}{[I]} \underset{\underline{u}_B}{\begin{bmatrix} w_{x_I} \\ w_{y_I} \\ w_{z_I} \\ a_x - s\hat{\theta}g \\ a_y + c\hat{\theta}s\hat{\Phi}g \\ a_z + c\hat{\Theta}c\hat{\Phi}g \end{bmatrix}} + \underset{\underline{w}_B}{\begin{bmatrix} 0 \\ 0 \\ 0 \\ n_{a_x} \\ n_{a_y} \\ n_{a_z} \end{bmatrix}} \qquad \text{(A-10)}$$

The linearized version of measurement model B [eq. (3-51)]:

$$\underbrace{\begin{bmatrix} |\underline{V}_{air}| \\ \alpha \\ \beta \\ h \\ r_{DME_{s1}} \\ r_{DME_{s2}} \\ \theta_{VOR_{s1}} \\ \theta_{VOR_{s2}} \end{bmatrix}}_{\underline{z}_B} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & \frac{u}{a_o^{1/2}} & \frac{v}{a_o^{1/2}} & \frac{w}{a_o^{1/2}} \\ 0 & 0 & 0 & \frac{-w}{a_o-v^2} & 0 & \frac{u}{a_o-v^2} \\ 0 & 0 & 0 & \frac{-v}{a_o-w^2} & \frac{u}{a_o-w^2} & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ \frac{x_I-x_{s1}}{a_{31}^{1/2}} & \frac{y_I-y_{s1}}{a_{31}^{1/2}} & \frac{z_I-z_{s1}}{a_{31}^{1/2}} & 0 & 0 & 0 \\ \frac{x_I-x_{s2}}{a_{32}^{1/2}} & \frac{y_I-y_{s2}}{a_{32}^{1/2}} & \frac{z_I-z_{s2}}{a_{32}^{1/2}} & 0 & 0 & 0 \\ \frac{-(y_I-y_{s1})}{a_{31}-(z_I-z_{s1})^2} & \frac{x_I-x_{s1}}{a_{31}-(z_I-z_{s1})^2} & 0 & 0 & 0 & 0 \\ \frac{-(y_I-y_{s2})}{a_{32}-(z_I-z_{s2})^2} & \frac{x_I-x_{s2}}{a_{32}-(z_I-z_{s2})^2} & 0 & 0 & 0 & 0 \end{bmatrix}}_{H_B} \underbrace{\begin{bmatrix} x_I \\ y_I \\ z_I \\ u \\ v \\ w \end{bmatrix}}_{\underline{x}_B} + \underbrace{\begin{bmatrix} n_v \\ n_\alpha \\ n_\beta \\ n_h \\ n_{DME_1} \\ n_{DME_2} \\ n_{VOR_1} \\ n_{VOR_2} \end{bmatrix}}_{\underline{v}_B} \qquad \text{(A-11)}$$

$$\text{with:} \quad \left.\begin{aligned} a_o &= u^2+v^2+w^2 \\ a_{3i} &= (x_I-x_{si})^2+(y_I-y_{si})^2+(z_I-z_{si})^2 \end{aligned}\right\} \qquad \text{(A-12)}$$

A.3 LINEARIZATION OF THE IMPROVED KINEMATIC MODEL.

The improvement to the kinematic model elaborated in the first section of this appendix refers to state model B. The linearized version of (A-3) is:

$$\underset{\dot{\underline{x}}_B}{\begin{bmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{z}_I \\ \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{w}_{x_{I_o}} \\ \dot{w}_{y_{I_o}} \\ \dot{w}_{y_{I_o}} \\ \frac{\partial \dot{w}_{x_I}}{\partial z_I} \\ \frac{\partial \dot{w}_{y_I}}{\partial z_I} \\ \frac{\partial \dot{w}_{z_I}}{\partial z_I} \end{bmatrix}} = \underset{F_B}{\begin{bmatrix} F_{11} & F_{12} & F_{13} & F_{14} \\ F_{21} & F_{22} & F_{23} & F_{24} \\ F_{31} & F_{32} & F_{33} & F_{34} \\ F_{41} & F_{42} & F_{43} & F_{44} \end{bmatrix}} \underset{\underline{x}_B}{\begin{bmatrix} x_I \\ y_I \\ z_I \\ u \\ v \\ w \\ w_{x_{I_o}} \\ w_{y_{I_o}} \\ w_{z_{I_o}} \\ \frac{\partial w_{x_I}}{\partial z_I} \\ \frac{\partial w_{y_I}}{\partial z_I} \\ \frac{\partial w_{z_I}}{\partial z_I} \end{bmatrix}} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ a_x - s\theta g \\ a_y + c\hat{\theta} s\hat{\phi} g \\ a_z + c\hat{\theta} c\hat{\phi} g \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ n_{a_x} \\ n_{a_y} \\ n_{a_z} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{A-13}$$

where each F_{ij} stands for a 3x3 matrix:

$$F_{21} = F_{31} = F_{41} = F_{32} = F_{42} = F_{23} = F_{33} = F_{43} = F_{34} = F_{44} = [0] \qquad \text{(A-14)}$$

$$F_{13} = [I] \qquad \text{(A-15)}$$

$$F_{14} = (z_I - z_{I_o})[I] \qquad \text{(A-16)}$$

$$F_{11} = \begin{bmatrix} 0 & 0 & \dfrac{\partial w_{x_I}}{\partial z_I} \\ 0 & 0 & \dfrac{\partial w_{y_I}}{\partial z_I} \\ 0 & 0 & \dfrac{\partial w_{z_I}}{\partial z_I} \end{bmatrix} \qquad \text{(A-17)}$$

$$F_{12} = \hat{H}^I_B(t) \qquad \text{(A-18)}$$

$$F_{22} = \begin{bmatrix} \hat{H}^I_{B_{31}}\left(\hat{H}^B_I \dfrac{\partial \underline{w}_I}{\partial z_I}\right)_1 & \hat{r}+\hat{H}^I_{B_{32}}\left(\hat{H}^B_I \dfrac{\partial \underline{w}_I}{\partial z_I}\right)_1 & -\hat{q}+\hat{H}^I_{B_{33}}\left(\hat{H}^B_I \dfrac{\partial \underline{w}_I}{\partial z_I}\right)_1 \\ -\hat{r}+\hat{H}^I_{B_{31}}\left(\hat{H}^B_I \dfrac{\partial \underline{w}_I}{\partial z_I}\right)_2 & \hat{H}^I_{B_{32}}\left(\hat{H}^B_I \dfrac{\partial \underline{w}_I}{\partial z_I}\right)_2 & \hat{p}+\hat{H}^I_{B_{33}}\left(\hat{H}^B_I \dfrac{\partial \underline{w}_I}{\partial z_I}\right)_2 \\ \hat{q}+\hat{H}^I_{B_{31}}\left(\hat{H}^B_I \dfrac{\partial \underline{w}_I}{\partial z_I}\right)_3 & -\hat{p}+\hat{H}^I_{B_{32}}\left(\hat{H}^B_I \dfrac{\partial \underline{w}_I}{\partial z_I}\right)_3 & \hat{H}^I_{B_{33}}\left(\hat{H}^B_I \dfrac{\partial \underline{w}_I}{\partial z_I}\right)_3 \end{bmatrix} \qquad \text{(A-19)}$$

$$F_{24} = H_{uvw}{}^{*}\, \hat{H}^B_I(t) \qquad \text{(A-20)}$$

APPENDIX B

PROGRAM LISTINGS

B.1 ONBOARD ASSEMBLY PROGRAMMING

The microprocessor prepares the data in block units. Each block unit is a memory buffer of 1024 data words (each data word contains two bytes or 16 bits). For 38 information channels, for example, taken every second such a block can be filled with 26-sec worth of data (plus 36 dummy words: 38x26+36 = 1024). A data block is written into the digital cartridge recordwise. Each record contains 128 data words, i.e., under normal conditions eight records complete a block transfer. Occasionally, to make sure that no data is missed, a record may be written repeatedly onto the cartridge. Thus, the first task for the preprocessing software (Appendix C) is to reconstruct the original data blocks, each constituting of eight records of $\frac{8\text{x}128-36}{38} = 26$ information-seconds.

B.2 GENERIC SPIFR FLIGHT PATH AND IN-FLIGHT MEASUREMENTS SIMULATION (FORTRAN)

This program creates a generic SPIFR mission trajectory, simulates the associated in-flight measurements and corrupts them with pseudo-random noise. Knowing the uncorrupted values of the measured variables, the algorithm for optimal smoothing and flight path reconstruction may be verified.

B.3 OPTIMAL SMOOTHING AND FLIGHT PATH RECONSTRUCTION ALGORITHM (FORTRAN)

As demonstrated in the following listings, one of the key issues in optimal smoothing and flight path reconstruction is computer storage management -- in particular, between the forward and the backward passes over the measured data records. The smoothing algorithm, which consists of eq. (3-60) to (3-63), requires knowledge of the state-vector before and after each measurement update and of the corresponding covariance matrices. For state model A, e.g., the state vector has six components. This means that following the filtering pass 84 values have to be stored for each measurement instant. A SPIFR mission simulation is about 30 minutes long and after preprocessing the measurement update interval is standardized to be 1 sec for all variables. Thus, the temporary storage facility has to "remember" 84x30x60 values.

ONBOARD ASSEMBLY (B.1)

```
T SPIFR8 LISTING

¿¿                 ;
¿¿                 ;                    VERSION 8 SPIFFER   18.V.81
¿¿                 ;                    ADC READ, PUSHBUTTONS, BLINKING LIGHTS WORKING
¿¿                 ;                    DISPLAY MEMORY INCLUDED
¿¿                 ;                    TAPES INSTALLED
¿¿                 ;                    DATA AVERAGING IMPLEMENTED
¿¿                 ;                    SLOW CHANNELS AVERAGED AT CNT = 4
¿¿                 ;                    DME TO BE IMPLEMENTED AS SLOW CHANNEL
¿¿                 ;                    ADC CHANNEL ZERO READ TWICE TO FILL OPEN RAM SLOT
¿¿                 ;                    LEFT BY MOVING DME TO SLOW CHANNEL.
¿¿                 ;                    STILL AVERAGING OVER 12 FAST CHANNELS.
¿¿                 ;                    HAVE A BUFFER OF 38 WORDS (16 BITS WIDE)
¿¿                 ;                    INSTEAD OF 37 WORDS.
¿¿                 ;                    HAVE FOUR LIGHTS INSTEAD OF THREE
¿¿                 ;                    LIGHT4 IS IN SYNC WITH LIGHT3.
¿¿                 ;                    THE LIGHTS CODING IS ALL NEW.
¿¿                 ;
¿¿8000                       ORG        8000H
¿¿3000             ADDAM     EQU        3000H      ;ADDRESS OF A TO D CONVERTER
¿¿00C3             JMP       EQU        0C3H
¿¿                 ;CO        EQU        03BH
¿¿FFE3             INT75     EQU        0FFE3H
¿¿00C5             LITEPORT  EQU         0C5H       ;LITE DISPLAY PORT IS PORT C J1 OF 116 BOARD
¿¿00C7             LITECTRL  EQU         0C7H       ;CTRL PORT FOR J1 OF 116 BOARD
¿¿                 ;LITEPORT  EQU         0E9H        ;LITE DISPLAY PORT IS PORT B J2 OF 8004
¿¿                 ;LITECTRL  EQU         0EBH        ;CTRL PORT FOR J2 OF 8004
¿¿00E7             J18004    EQU        0E7H       ;CONTROL OF 8255 J-1
¿¿00E4             ALTPORT   EQU        0E4H       ;ALTIMETER PORT A J-1
¿¿00E5             PUSHPORT  EQU        0E5H       ;PUSHBUTTON PORT B J-1
¿¿8000  F3                   DI                    ;NO INTERRUPTS ALLOWED YET
¿¿8001  CD738A               CALL       INITPORTA  ;8255 PORT A OF J2 ON 8004
¿¿8004  CD2188               CALL       INITUSART  ;INITIALISE THE USART FOR PILOT TERMINAL
¿¿8007  CD3787               CALL       INITSEMA   ;SET UP SEMAPHORES ON ALL TEN BUFFERS
¿¿800A  CDBC87               CALL       INITRE
¿¿800D  CD4080               CALL       FILLMEM    ;FILL MEMORY WITH INTEGERS 0-10239
¿¿8010  CDD480               CALL       INITWR     ;INITIALISE EMPTYING POINTERS
¿¿                                                 ;DUMP TEN BUFFERS EACH 2048
¿¿                                                 ;TO TAPE, 256 BYTES AT A TIME
¿¿                                                 ;OVER AND OVER AGAIN.
¿¿                                                 ;THE BUFFERS ARE DESIGNATED BY
¿¿                                                 ; THE DIGITS 0 TO 9.
¿¿                                                 ;
¿¿                                                 ;DE POINTS TO THE LOCATION
¿¿                                                 ; TO BE EMPTIED.
¿¿                                                 ;
¿¿8013  CD118A               CALL       ALIGN      ;INITALISE T9511
¿¿8016  CD2A8A               CALL       OK
¿¿8019  110090               LXI        D,BUFF0    ;DE PAIR POINT TO LOCATION TO BE EMPTIED
¿¿801C  CD3883               CALL       STINT      ;ENABLE THE INTERRUPTS
¿¿                 BB:
¿¿801F  CD7687               CALL       TSTEMPTY   ;WAIT UNTIL BUFFER FULL
¿¿8022  CDA180               CALL       WRITE      ;DUMP ANOTHER 256 BYTES TO TAPE
¿¿8025  CD5880               CALL       UEBUFS     ;UPDATE THE INDICES USED
¿¿                                                 ; TO KEEP TRACK OF EMPTYING THE TEN BUFFERS.
¿¿                                                 ;  THERE ARE THREE OF THEM,
¿¿                                                 ;   TWO POINTERS AND A COUNTER.
¿¿                                                 ;
¿¿8028  C31F80             JMP        BB
¿¿                                                 ;
¿¿802B  3A6188               LDA        FLAG
¿¿802E  FE00                 CPI        0
¿¿8030  C21F80               JNZ        BB
¿¿8033  3E0A                 MVI        A.10
```

```
¿¿8035  326188             STA    FLAG
¿¿8038  3E24               MVI    A,'$'
¿¿803A  CD4182             CALL   BO
¿¿803D  C31F80             JMP    BB
¿¿                  ;LAG:  DB     0
¿¿                  FILLMEM:
¿¿8040  F5                 PUSH   PSW
¿¿8041  E5                 PUSH   H
¿¿8042  110090             LXI    D,BUFFO    ;BASE ADDRESS FOR FILLING IS BUFFO
¿¿8045  210000             LXI    H,0        ;COUNT STARTS WITH 0
¿¿8048  7D          FILLAG: MOV   A,L        ;LOWER BYTE INTO ACC
¿¿8049  12                 STAX   D          ;SOTRE IT TO MEMORY
¿¿804A  13                 INX    D          ;MOVE TO NEXT BYTE LOCATION
¿¿804B  7C                 MOV    A,H        ;UPPER BYTE INTO ACC
¿¿804C  12                 STAX   D          ;STORE IT TO MEMORY
¿¿804D  13                 INX    D          ;POINT TO NEXT BYTE LOCATION
¿¿804E  23                 INX    H          ;SIZE OF INTEGER UP BY 1
¿¿804F  7C                 MOV    A,H        ;CHECK FOR AN UPPER BOUND
¿¿8050  FE28               CPI    28H        ;
¿¿8052  C24880             JNZ    FILLAG     ;REPEAT IF BELOW UPPER BOUND
¿¿8055  E1                 POP    H
¿¿8056  F1                 POP    PSW
¿¿8057  C9                 RET
¿¿                  UEBUFS:
¿¿8058  F5                 PUSH   PSW
¿¿8059  E5                 PUSH   H
¿¿805A  C5                 PUSH   B
¿¿805B  2A6688             LHLD   EREMAIN
¿¿805E  010001             LXI    B,256      ;256 BYTES EMPYIED AT A TIME
¿¿8061  AF                 XRA    A          ;CARRY=0
¿¿8062  ED42               DSBC   B
¿¿8064  226688             SHLD   EREMAIN    ;EREMAIN=EREMAIN-256
¿¿                                           ;
¿¿                                           ;IS EREMAIN >= 256 ?
¿¿                                           ;
¿¿8067  AF                 XRA    A          ;CARRY=0
¿¿8068  ED42               DSBC   B
¿¿806A  F29D80             JP     EROOH      ;IF S=0, HL >= BC
¿¿                                           ;
¿¿                                           ;CAN NOT EMPTY
¿¿                                           ; ANOTHER BLOCK OF 256
¿¿                                           ;  FROM CURRENT BUFFER OF 2048
¿¿                                           ;
¿¿806D  CDA887             CALL   SETEMPTY   ;MARK CURRENT BUFFER EMPTY
¿¿                                           ;SET UP INDICES FOR DUMPING NEXT 2048 BLOCK
¿¿8070  210008             LXI    H,2048     ;
¿¿8073  226688             SHLD   EREMAIN    ;EREMAIN=2048
¿¿                                           ;
¿¿8076  3A6888             LDA    EWHICHB    ;SWITCH TO NEXT BUFFER
¿¿8079  3C                 INR    A          ;
¿¿807A  326888             STA    EWHICHB    ;
¿¿807D  FE0A               CPI    10
¿¿807F  C29280             JNZ    ESW        ;
¿¿8082  76                 HLT
¿¿                                           ;
¿¿                                           ;WE ARE NOW BEYOND TENTH BUFFER
¿¿                                           ; NOTE EWHICHB HAS RANGE 0 TO 9
¿¿                                           ;  SWITCH TO BUFFER 0
¿¿                                           ;
¿¿8083  3E00               MVI    A,0
¿¿8085  326888             STA    EWHICHB    ;EWHICH=0
¿¿                                           ;
¿¿                                           ;ECURRBUF CONTAINS THE STARTING
¿¿                                           ; LOCATION OF THE CURRENT 2048 BLOCK
¿¿                                           ;  BEING EMPTYIED
¿¿                                           ;
```

```
¿¿8088  110090               LXI    D,BUFFO    ;BUFFO IS STARTING LOCATION OF
¿¿                                             ; BLOCK 0 OR BUFFER 0
¿¿808B  ED537488             SDED   ECURRBUF   ;NOTE THAT DE POINTS WHERE ECURRBUF POINTS
¿¿808F  C39D80               JMP    EROOM
¿¿                   ESW:
¿¿                                             ;
¿¿                                             ;WE ARE GOING TO EMPTY A BUFFER
¿¿                                             ; 2048 BYTES HIGHER IN CORE THAN THE
¿¿                                             ;  PREVIOUS BUFFER.
¿¿                                             ;
¿¿8092  2A7488               LHLD   ECURRBUF   ;
¿¿8095  010008               LXI    B,2048
¿¿8098  09                   DAD    B
¿¿8099  227488               SHLD   ECURRBUF
¿¿809C  EB                   XCHG              ;DE POINTING WHERE ECURRBUF POINTS.
¿¿                   EROOM:
¿¿809D  C1                   POP    B
¿¿809E  E1                   POP    H
¿¿809F  F1                   POP    PSW
¿¿80A0  C9                   RET
¿¿                   ;
¿¿                   WRITE:
¿¿                                                        ;THE ONLY ERROR ALLOWED FOR NOW
¿¿                                                        ;IS TRYING TO WRITE BEYOND EOT
¿¿                                             ;WRITE ON LEFT TAPE OR RIGHT TAPE
¿¿                                             ;DEPENDING ON THE SETTING OF TSW
¿¿                                             ;TSW='L' OR 'R'.
¿¿                                             ;IF END OF TAPE OR HARD ERROR ON ONE TAPE,
¿¿                                             ;SWITCH TO ANOTHER TAPE
¿¿                                             ;
¿¿80A1  F5                   PUSH   PSW
¿¿80A2  E5                   PUSH   H
¿¿80A3  C5                   PUSH   B
¿¿80A4  ED537888             SDED   TEMPDE     ;SAVE DE PAIR IN CASE OF RETRY
¿¿                   ;       LXI    H,TEMPDE
¿¿                   ;       CALL   HDMP2
¿¿80A8  3A6988               LDA    TSW        ;WHICH TAPE DO WE USE?
¿¿80AB  FE4C                 CPI    'L'
¿¿80AD  C2C280               JNZ    WRITR
¿¿                   WRITL:
¿¿80B0  CD3481               CALL   LCTUW      ;IF YOU COME TO EOT SWITCH TO NEXT UNIT
¿¿                   ;<<<<<<<<<(MORE CODE HERE)>>>>>>>>>>>>>>>>>
¿¿                   ;<<<<<<<<<(ABOUT SWITCHING LOGIC)>>>>>>>>>>>>
¿¿80B3  CABE80               JZ     RETWL
¿¿80B6  3E52                 MVI    A,'R'
¿¿80B8  326988               STA    TSW
¿¿                   ;<<<<<<<<(RETRANSMIT LAST RECORD ONTO RIGHT TAPE)>>>>>>>>>>>
¿¿80BB  C3C280               JMP    WRITR      ; AND REWRITE RECORD ON RIGHT TAPE
¿¿80BE  C1           RETWL:  POP    B
¿¿80BF  E1                   POP    H
¿¿80C0  F1                   POP    PSW
¿¿80C1  C9                   RET
¿¿                   WRITR:
¿¿80C2  CDD581               CALL   RCTUW      ;IF YOU COME TO EOT SWITCH TO NEXT UNIT
¿¿                   ;<<<<<<(MORE CODE HERE)>>>>>>>>>>>>>>>>>>>>>>>
¿¿                   ;<<<<<<(ABOUT SWITCHING LOGIC)>>>>>>>>>>>>>>>>
¿¿80C5  CAD080               JZ     RETWR
¿¿80C8  3E4C                 MVI    A,'L'
¿¿80CA  326988               STA    TSW
¿¿                   ;<<<<<<(RETRANSMIT LAST RECORD USING)>>>>>>>>>>
¿¿80CD  C3B080               JMP    WRITL      ;  AND REWRITE RECORD ON LEFT TAPE
¿¿80D0  C1           RETWR:  POP    B
¿¿80D1  E1                   POP    H
¿¿80D2  F1                   POP    PSW
¿¿80D3  C9                   RET
¿¿                   INITWR:
```

```
¿¿80D4  F5                  PUSH    PSW
¿¿80D5  E5                  PUSH    H
¿¿80D6  3E00                MVI     A,0
¿¿80D8  326888              STA     EWHICHB
¿¿80DB  210008              LXI     H,2048
¿¿80DE  226688              SHLD    EREMAIN
¿¿80E1  3E4C                MVI     A,'L'
¿¿80E3  326988              STA     TSW
¿¿80E6  210090              LXI     H,BUFFO
¿¿80E9  227488              SHLD    ECURRBUF
¿¿80EC  110090              LXI     D,BUFFO
¿¿80EF  E1                  POP     H
¿¿80F0  F1                  POP     PSW
¿¿80F1  C9                  RET
¿¿              DEM256:
¿¿80F2  E5                  PUSH    H
¿¿                                          ;
¿¿                                          ;DE <= DE - 256
¿¿                                          ;
¿¿80F3  62                  MOV     H,D
¿¿80F4  6B                  MOV     L,E     ;HL <= DE
¿¿80F5  010001              LXI     B,256
¿¿80F8  AF                  XRA     A       ;CARRY=0
¿¿80F9  ED42                DSBC    B       ;HL - 256
¿¿80FB  54                  MOV     D,H
¿¿80FC  5D                  MOV     E,L     ;DE <= HL
¿¿80FD  E1                  POP     H
¿¿80FE  C9                  RET
¿¿              LCTUF:
¿¿                                          ;WRITE AN EOF MARK ON LCTU
¿¿80FF  3E1B                MVI     A,ESC
¿¿8101  CD4182              CALL    BO
¿¿8104  3E26                MVI     A,'&'
¿¿8106  CD4182              CALL    BO
¿¿8109  3E70                MVI     A,'p'
¿¿810B  CD4182              CALL    BO
¿¿810E  3E31                MVI     A,'1'   ;ADDRESS OF LCTU
¿¿8110  CD4182              CALL    BO
¿¿8113  3E75                MVI     A,'u'
¿¿8115  CD4182              CALL    BO
¿¿8118  3E35                MVI     A,'5'   ;WRITE FILEMARK COMMAND IS  ASCII CHAR 5
¿¿811A  CD4182              CALL    BO
¿¿811D  3E43                MVI     A,'C'
¿¿811F  CD4182              CALL            BO
¿¿8122  3E11                MVI     A,DC1
¿¿8124  CD4182              CALL    BO
¿¿8127  CD5782              CALL    BI
¿¿812A  326288              STA     SAVE
¿¿812D  CD5782              CALL    BI
¿¿8130  326388              STA     SAVE+1
¿¿8133  C9                  RET
¿¿              LCTUW:
¿¿8134  ED5B7888            LDED    TEMPDE
¿¿8138  3E1B                MVI     A,ESC
¿¿813A  CD4182              CALL    BO
¿¿813D  3E26                MVI     A,'&'
¿¿813F  CD4182              CALL    BO
¿¿8142  3E70                MVI     A,'p'
¿¿8144  CD4182              CALL    BO
¿¿8147  3E31                MVI     A,'1'
¿¿8149  CD4182              CALL    BO
¿¿814C  3E64                MVI     A,'d'
¿¿814E  CD4182              CALL    BO
¿¿8151  3E32                MVI     A,'2'
¿¿8153  CD4182              CALL    BO
¿¿8156  3E35                MVI     A.'5'
```

```
¿¿8158  CD4182               CALL    BO
¿¿815B  3E36                 MVI     A,'6'
¿¿815D  CD4182               CALL    BO
¿¿8160  3E57                 MVI     A,'W'
¿¿8162  CD4182               CALL    BO
¿¿8165  3E05                 MVI     A,ENQ
¿¿8167  CD4182               CALL    BO
¿¿816A  CD5782               CALL    BI
¿¿816D  326488               STA     SAV
¿¿8170  FE06                 CPI     ACK
¿¿8172  CA7881               JZ      NEXT
¿¿8175  C33481               JMP     LCTUW
¿¿                   NEXT:                   ;DE POINTS TO LOCATION TO BE EMPTIED
¿¿8178  010001               LXI     B,256
¿¿817B  CD7482               CALL    SEND
¿¿817E  3E11                 MVI     A,DC1
¿¿8180  CD4182               CALL    BO
¿¿8183  CD5782               CALL    BI      ;READ S OF F
¿¿8186  DA3481               JC      LCTUW
¿¿8189  326288               STA     SAVE
¿¿818C  CD5782               CALL    BI      ;READ CR
¿¿818F  DA3481               JC      LCTUW
¿¿8192  326388               STA     SAVE+1
¿¿8195  3A6288               LDA     SAVE    ;EITHER S OR F
¿¿8198  FE53                 CPI     'S'
¿¿819A  C8                   RZ              ;RETURN Z SET ON 'S', Z RESET ON 'F'
¿¿819B  ED5B7888             LDED    TEMPDE
¿¿819F  C9                   RET
¿¿                   RCTUF:
¿¿                                           ;WRITE AN EOF MARK ON LCTU
¿¿81A0  3E1B                 MVI     A,ESC
¿¿81A2  CD4182               CALL    BO
¿¿81A5  3E26                 MVI     A,'&'
¿¿81A7  CD4182               CALL    BO
¿¿81AA  3E70                 MVI     A,'p'
¿¿81AC  CD4182               CALL    BO
¿¿81AF  3E32                 MVI     A,'2'   ;ADDRESS OF RCTU
¿¿81B1  CD4182               CALL    BO
¿¿81B4  3E75                 MVI     A,'u'
¿¿81B6  CD4182               CALL    BO
¿¿81B9  3E35                 MVI     A,'5'   ;WRITE FILEMARK COMMAND IS  ASCII CHAR 5
¿¿81BB  CD4182               CALL    BO
¿¿81BE  3E43                 MVI     A,'C'
¿¿81C0  CD4182               CALL            BO
¿¿81C3  3E11                 MVI     A,DC1
¿¿81C5  CD4182               CALL    BO
¿¿81C8  CD5782               CALL    BI
¿¿81CB  326288               STA     SAVE
¿¿81CE  CD5782               CALL    BI
¿¿81D1  326388               STA     SAVE+1
¿¿81D4  C9                   RET
¿¿                   RCTUW:
¿¿81D5  ED5B7888             LDED    TEMPDE
¿¿81D9  3E1B                 MVI     A,ESC
¿¿81DB  CD4182               CALL    BO
¿¿81DE  3E26                 MVI     A,'&'
¿¿81E0  CD4182               CALL    BO
¿¿81E3  3E70                 MVI     A,'p'
¿¿81E5  CD4182               CALL    BO
¿¿81E8  3E32                 MVI     A,'2'   ;ADDRESS OF RCTU
¿¿81EA  CD4182               CALL    BO
¿¿81ED  3E64                 MVI     A,'d'
¿¿81EF  CD4182               CALL    BO
¿¿81F2  3E32                 MVI     A,'2'
¿¿81F4  CD4182               CALL    BO
¿¿81F7  3E35                 MVI     A.'5'
```

```
¿¿81F9  CD4182            CALL    BO
¿¿81FC  3E36              MVI     A,'6'
¿¿81FE  CD4182            CALL    BO
¿¿8201  3E57              MVI     A,'W'
¿¿8203  CD4182            CALL    BO
¿¿8206  3E05              MVI     A,ENQ
¿¿8208  CD4182            CALL    BO
¿¿820B  CD5782            CALL    BI
¿¿820E  326488            STA     SAV
¿¿8211  FE06              CPI     ACK
¿¿8213  CA1982            JZ      NEXT
¿¿8216  C3D581            JMP     RCTUW
¿¿                NEXT:
¿¿                                        ;DE POINTS TO LOCATION TO BE EMPTIED
¿¿8219  010001            LXI     B,256
¿¿821C  CD7482            CALL    SEND
¿¿821F  3E11              MVI     A,DC1
¿¿8221  CD4182            CALL    BO
¿¿8224  CD5782            CALL    BI      ; READ S OR F
¿¿8227  DAD581            JC      RCTUW
¿¿822A  326288            STA     SAVE
¿¿822D  CD5782            CALL    BI      ; READ CR
¿¿8230  DAD581            JC      RCTUW
¿¿8233  326388            STA     SAVE+1
¿¿8236  3A6288            LDA     SAVE    ;EITHER S OR F
¿¿8239  FE53              CPI     'S'     ;RETURN WITH Z SET ON S, Z RESET ON F
¿¿823B  C8                RZ
¿¿823C  ED5B7888          LDED    TEMPDE
¿¿8240  C9                RET
¿¿                ;*******************************************************
¿¿                ;                 CHARACTER OUTPUT ROUTINE
¿¿                ; CO OUTPUTS ONE CHARACTER FROM ACC TO TERMINAL
¿¿                ; VIA THE USART. ALL REGISTERS AND FLAGS ARE
¿¿                ; PRESERVED. THE CHARACTER  IT OUTPUTS IS IN THE ACC.
¿¿                ;*******************************************************
¿¿8241  F5        BO:     PUSH    PSW
¿¿8242  C5                PUSH    B
¿¿8243  4F                MOV     C,A     ;SAVE A REG
¿¿                BOO:
¿¿8244  00                NOP
¿¿8245  00                NOP
¿¿8246  DBED              IN      S8251A  ;GET USART STATUS
¿¿8248  E601              ANI     TXRDYA  ;CHECK TRANSMIT READY FLAG
¿¿824A  CA4482            JZ      BOO     ;NOT READY
¿¿824D  00                NOP
¿¿824E  00                NOP
¿¿824F  00                NOP
¿¿8250  00                NOP
¿¿8251  79                MOV     A,C     ;READY TO TRANSMIT , RESTORE CHAR TO A REG
¿¿8252  D3EC              OUT     D8251A  ;SEND IT
¿¿8254  C1                POP     B
¿¿8255  F1                POP     PSW
¿¿8256  C9                RET
¿¿                ;*********************************************************
¿¿                ;                 CHARACTER INPUT ROUTINE
¿¿                ; BI INPUTS ONE CHARACTER FROM TERMINAL INTO ACC
¿¿                ; VIA THE USART. THE  FLAGS AND THE ACC ARE CHANGED.
¿¿                ; THE CHARACTER IT READS IS RETURNED IN THE ACC.
¿¿                ;  IF NO TIMEOUT OCCURS, THE CARRY IS SET.
¿¿                ;  IF TIMEOUT OCCURS, THE CARRY IS RESET.
¿¿                ;  THE CHAR READ IS RETURNED IN THE ACC.
¿¿                ;********************************************************
¿¿8257  00        BI:     NOP             ;
¿¿8258  C5                PUSH    B
¿¿8259  01FFFF            LXI     B,0FFFFH
¿¿                BIO:
```

```
¿¿825C  0B        394           DCX     B
¿¿825D  78        395           MOV     A,B
¿¿825E  B1        396           ORA     C
¿¿825F  CA6F82    397           JZ      TIMEOUT
¿¿8262  DBED      398           IN      S8251A     ;GET USART STATUS
¿¿8264  E602      399           ANI     RXRDYA     ;CHECK RECIEVER READY
¿¿8266  CA5C82    400           JZ      BI0        ;
¿¿8269  DBEC      401           IN      D8251A     ;GET CHAR
¿¿826B  37        402           STC
¿¿826C  3F        403           CMC
¿¿826D  C1        404           POP     B
¿¿826E  C9        405           RET
¿¿826F  00        406  TIMEOUT: NOP
¿¿8270  37        407           STC
¿¿8271  C1        408           POP     B
¿¿8272  C9        409           RET
¿¿8273  76        410           HLT
¿¿                411  SEND:
¿¿8274  1A        412           LDAX    D
¿¿8275  CD4182    413           CALL    BO
¿¿8278  13        414           INX     D
¿¿8279  0B        415           DCX     B
¿¿827A  78        416           MOV     A,B
¿¿827B  B1        417           ORA     C
¿¿827C  C27482    418           JNZ     SEND
¿¿827F  C9        419           RET
¿¿                420  STATUS:
¿¿8280  F5        421           PUSH    PSW
¿¿8281  E5        422           PUSH    H
¿¿8282  21FC89    423           LXI     H,STAT
¿¿8285  3E1B      424           MVI     A,ESC      ;START ESCAPE SEQUENCE
¿¿8287  CD4182    425           CALL    BO
¿¿828A  3E5E      426           MVI     A,'^'
¿¿828C  CD4182    427           CALL    BO
¿¿828F  3E11      428           MVI     A,DC1
¿¿8291  CD4182    429           CALL    BO
¿¿8294  CD5782    430           CALL    BI         ;EAT ESC
¿¿8297  CD5782    431           CALL    BI         ;EAT BACLSLASH
¿¿829A  CD5782    432           CALL    BI         ;BYTE 0
¿¿829D  CD0B88    433           CALL    CO
¿¿82A0  CD5782    434           CALL    BI         ;BYTE 1
¿¿82A3  CD0B88    435           CALL    CO
¿¿82A6  CD5782    436           CALL    BI         ;BYTE 2
¿¿82A9  CD0B88    437           CALL    CO
¿¿82AC  CD5782    438           CALL    BI         ;BYTE 3
¿¿82AF  CD0B88    439           CALL    CO
¿¿82B2  CD5782    440           CALL    BI         ;BYTE 4
¿¿82B5  CD0B88    441           CALL    CO
¿¿82B8  CD5782    442           CALL    BI         ;BYTE 5
¿¿82BB  CD0B88    443           CALL    CO
¿¿82BE  CD5782    444           CALL    BI         ;BYTE 6
¿¿82C1  CD0B88    445           CALL    CO
¿¿82C4  CD5782    446           CALL    BI         ;GET CR
¿¿82C7  CD0B88    447           CALL    CO
¿¿82CA  E1        448           POP     H
¿¿82CB  F1        449           POP     PSW
¿¿82CC  C9        450           RET
¿¿                451  LCTUST:
¿¿82CD  F5        452           PUSH    PSW
¿¿82CE  E5        453           PUSH    H
¿¿82CF  3E1B      454           MVI     A,ESC
¿¿82D1  CD0B88    455           CALL    CO
¿¿82D4  3E26      456           MVI     A,'&'
¿¿82D6  CD0B88    457           CALL    CO
¿¿82D9  3E70      458           MVI     A,'p'
¿¿82DB  CD0B88    459           CALL    CO
```

```
¿¿82DE 3E31      460          MVI   A,'1'
¿¿82E0 CD0B88    461          CALL  CO
¿¿82E3 3E5E      462          MVI   A,'^'
¿¿82E5 CD0B88    463          CALL  CO
¿¿82E8 3E11      464          MVI   A,DC1
¿¿82EA CD0B88    465          CALL  CO
¿¿82ED CD5782    466          CALL  BI        ;ESC READ
¿¿82F0 CD5782    467          CALL  BI        ;BACKSLASH READ
¿¿82F3 CD5782    468          CALL  BI        ;P READ
¿¿82F6 CD5782    469          CALL  BI        ;DEVICE CODE DIGIT READ
¿¿82F9 CD5782    470          CALL  BI        ;BYTE 0 READ
¿¿82FC CD5782    471          CALL  BI        ;BYTE 1 READ
¿¿82FF CD5782    472          CALL  BI        ;BYTE 2 READ
¿¿8302 CD5782    473          CALL  BI        ;CR READ
¿¿8305 E1        474          POP   H
¿¿8306 F1        475          POP   PSW
¿¿8307 C9        476          RET
¿¿               477   ;
¿¿               478   ;            DELAY ONE MILLISECOND
¿¿               479   ;
¿¿               480   D1MS:
¿¿8308 C5        481          PUSH  B
¿¿8309 F5        482          PUSH  PSW
¿¿830A 019800    483          LXI   B,152
¿¿830D 0B        484   D1MS0: DCX   B
¿¿830E 78        485          MOV   A,B
¿¿830F B1        486          ORA   C
¿¿8310 C20D83    487          JNZ   D1MS0
¿¿8313 F1        488          POP   PSW
¿¿8314 C1        489          POP   B
¿¿8315 C9        490          RET
¿¿001B           491   ESC    EQU   1BH
¿¿000D           492   CR     EQU   0DH
¿¿00ED           493   S8251A EQU   0EDH
¿¿00EC           494   D8251A EQU   0ECH
¿¿0002           495   RXRDYA EQU   02H
¿¿0001           496   TXRDYA EQU   01H
¿¿000A           497   LF     EQU   0AH
¿¿0005           498   ENQ    EQU   05H
¿¿0006           499   ACK    EQU   06H
¿¿0011           500   DC1 EQU 11H
¿¿8316 F5        501   CNTR1: PUSH  PSW
¿¿8317 3E74      502          MVI   A,074H
¿¿8319 D3DF      503          OUT   0DFH
¿¿831B 3E10      504          MVI   A,TLOW
¿¿831D D3DD      505          OUT   0DDH
¿¿831F 3E27      506          MVI   A,THIGH
¿¿8321 D3DD      507          OUT   0DDH
¿¿8323 E3        508          XTHL
¿¿8324 E3        509          XTHL
¿¿8325 F1        510          POP   PSW
¿¿8326 C9        511          RET
¿¿8327 F5        512   CNTR0: PUSH  PSW
¿¿8328 3E36      513          MVI   A,036H
¿¿832A D3DF      514          OUT   0DFH
¿¿832C 3E0A      515          MVI   A,0AH
¿¿832E D3DC      516          OUT   0DCH
¿¿8330 3E00      517          MVI   A,00H
¿¿8332 D3DC      518          OUT   0DCH
¿¿8334 E3        519          XTHL
¿¿8335 E3        520          XTHL
¿¿8336 F1        521          POP   PSW
¿¿8337 C9        522          RET
¿¿               523   STINT:
¿¿8338 F5        524          PUSH  PSW
¿¿8339 E5        525          PUSH  H
```

```
¿¿                      ;
¿¿                      ;                PUT 'JMP BREAK' INST. AT LOCS INT75,INT75+1,INT75+2
¿¿                      ;
¿¿8333A  3EC3                   MVI      A,JMP
¿¿833C  32E3FF                  STA      INT75
¿¿833F  214F83                  LXI      H,BREAK
¿¿8342  22E4FF                  SHLD     INT75+1
¿¿                      ;
¿¿                      ;                SETUP 8214 INTERRUPT CONTROOLER
¿¿                      ;
¿¿8345  3E03                    MVI      A,03H      ;BIT 7=0 FOR MODE 0
¿¿                                                  ;BIT 0 TO 3 DEFINE INTERRUPT PRIORIRY
¿¿                                                  ;BIT 0 TO 3 SET TO 3, ENABLE 2 TO 0 ONLY
¿¿8347  D3D7                    OUT      0D7H       ; SEND TO 8214
¿¿                      ;
¿¿                      ;;               PUT Z80 IN INTERRUPT MODE 0
¿¿                      ;
¿¿8349  ED46                    IM0
¿¿834B  E1                      POP      H
¿¿834C  F1                      POP      PSW
¿¿                      ;
¿¿                      ;                TURN ON INTERRUPT SYSTEM
¿¿                      ;
¿¿834D  FB                      EI
¿¿834E  C9                      RET
¿¿                      ;
¿¿                      ;                COME HERE ON TIMER INTERRUPT
¿¿                      ;
¿¿834F  00              BREAK:  NOP
¿¿8350  F5                      PUSH     PSW
¿¿8351  E5                      PUSH     H
¿¿8352  C5                      PUSH     B
¿¿8353  D5                      PUSH     D
¿¿                      ;
¿¿8354  3A6188                  LDA      FLAG
¿¿8357  3D                      DCR      A
¿¿8358  326188                  STA      FLAG
¿¿                      ;
¿¿835B  3EFF                    MVI      A,0FFH     ;MAKE OUTPUT OF
¿¿835D  D3E8                    OUT      0E8H     ;   ZERO VOLTS
¿¿835F  ED5B5788                LDED     SAVEDE     ;RESTORE BUFFER POINTER TO DE PAIR
¿¿8363  CD4A87                  CALL     TSTFULL    ;IS BUFFER EMPTY
¿¿                      ;
¿¿                      ;                IF CNT)=0.AND. CNT(=9 READ FAST CHANS AND STORE
¿¿                      ;
¿¿8366  CDC383                  CALL     FAST       ;COLLECT FAST DATA
¿¿                      ;
¿¿                      ;                IF CNT = 4 READ SLOW CHANNELS
¿¿                      ;
¿¿8369  3A5E88                  LDA      CNT
¿¿836C  FE04                    CPI      4
¿¿836E  C27483                  JNZ      NOT4
¿¿8371  CD2484                  CALL     SLOW
¿¿                      NOT4:
¿¿                      ;
¿¿                      ;
¿¿                      ;                IF CNT=9 READ TIME AND SLOW CHANS
¿¿                      ;
¿¿8374  3A5E88                  LDA      CNT        ;
¿¿8377  FE09                    CPI      9          ;
¿¿8379  C2A783                  JNZ      NOT9       ;
¿¿                      ;
¿¿837C  213400                  LXI      H,2*26
¿¿837F  19                      DAD      D          ;ADD 2*26 TO DE PAIR TO POINT BEYOND
¿¿8380  5D                      MOV      E,L
¿¿8381  54                      MOV      D.H        ;PUT HL IN DE
```

```
¿¿8382 CDCA86            CALL    DISPLAY  ;IF THERE IS TWODIG PAIR READY, DISPLAY MEMORY
¿¿8385 CDF484            CALL    UPBUFS   ;THERE ARE TWO BUFER POINTERS AND COUNTER
¿¿                                        ;  TO UPDATE
¿¿8388 CD5785            CALL    UCLK     ;TIC THE CLOCK
¿¿838B CD4D86            CALL    FLIP     ;TOGGLE LITE 3
¿¿838E CD7A88            CALL    AVG
¿¿8391 CD3D85            CALL    UINTS
¿¿8394 CD4785            CALL    RINTS
¿¿8397 ED535788          SDED    SAVEDE
¿¿839B AF                XRA     A
¿¿839C D3E8              OUT     0E8H
¿¿839E CD9785            CALL    STATE
¿¿83A1 CD7886            CALL    INTWODIG
¿¿83A4 C3B983            JMP     NODISP
¿¿               ;
¿¿               NOT9:
¿¿                       ;
¿¿83A7 CD7A88            CALL    AVG      ;AVERAGE FAST CHANNELS
¿¿83AA CD3D85            CALL    UINTS    ;UPDATE INTERRUPT COUNT
¿¿               ;
¿¿83AD CD4785            CALL    RINTS    ;RESET INTERRUPT COUNT WHEN NECESSARY
¿¿83B0 AF                XRA     A        ;MAKE OUTPUT OF
¿¿83B1 D3E8              OUT     0E8H    ;  FIVE VOLTS
¿¿               ;
¿¿               ;
¿¿83B3 CD9785             CALL   STATE    ; PLAY WITH PBS + LITES ON CONTROL PANEL
¿¿83B6 CD7886            CALL    INTWODIG ;LOOK FOR FOR TWO DIGIT PAIR
¿¿               NODISP:
¿¿83B9 3E03               MVI    A,03H
¿¿83BB D3D7               OUT    0D7H
¿¿83BD D1                POP     D
¿¿83BE C1                POP     B
¿¿83BF E1                POP     H
¿¿83C0 F1                POP     PSW
¿¿83C1 FB                EI
¿¿83C2 C9                RET
¿¿               FAST:
¿¿83C3 3E00              MVI     A,0
¿¿83C5 CD6185            CALL    ADC0
¿¿83C8 CDEB84            CALL    STORE
¿¿               ;
¿¿83CB 3E00              MVI     A,0      ;DELE COLUMN CHANNEL .
¿¿83CD CD6185            CALL    ADC0     ;  READ ADC CHANNEL
¿¿83D0 CDEB84            CALL    STORE    ;     PUT IN BUFFER RAM
¿¿               ;
¿¿83D3 3E05              MVI     A,5      ;THETADOT CHANNEL
¿¿83D5 CD6185            CALL    ADC0     ;  READ ADC CHANNEL
¿¿83D8 CDEB84            CALL    STORE    ;     PUT IN BUFFER RAM
¿¿               ;
¿¿83DB 3E07              MVI     A,7      ;NZ-NORMAL ACCEL CHANNEL
¿¿83DD CD6185            CALL    ADC0     ;  READ ADC CHANNEL
¿¿83E0 CDEB84            CALL    STORE    ;     PUT IN RAM BUFFER
¿¿               ;
¿¿83E3 3E08              MVI     A,8      ;DELA-ROOLL WHEEL
¿¿83E5 CD6185            CALL    ADC0     ;  READ ADC CHANNEL
¿¿83E8 CDEB84            CALL    STORE    ;     PUT IN RAM BUFFER
¿¿               ;
¿¿83EB 3E09              MVI     A,9     ;DELR-PEDALS CHANNEL
¿¿83ED CD6185            CALL    ADC0     ;  READ ADC CHANNEL
¿¿83F0 CDEB84            CALL    STORE    ;     PUT IN RAM BUFFER
¿¿               ;
¿¿83F3 3E0B              MVI     A,11     ;BETA-SIDESLIP CHANNEL
¿¿83F5 CD6185            CALL    ADC0     ;  READ ADC CHANNEL
¿¿83F8 CDEB84            CALL    STORE    ;     PUT IN RAM BUFFER
¿¿               ;
¿¿83FB 3E0D              MVI     A,13     ;P-ROLL RATE CHANNEL
```

```
¿¿83FD  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿8400  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿8403  3E0E            MVI     A,14    ;R-YAW RATE CHANNEL
¿¿8405  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿8408  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿840B  3E0F            MVI     A,15    ;NY-LATERAL ACCEL CHANNEL
¿¿840D  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿8410  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿8413  3E15            MVI     A,21    ;NX-LONGITUDINAL ACCEL CHANNEL
¿¿8415  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿8418  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿841B  3E19            MVI     A,25    ;ALPHA-ANGLE OF ATTACK CHANNEL
¿¿841D  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿8420  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿8423  C9              RET
¿¿              SLOW:
¿¿              ;
¿¿8424  CD7E85          CALL    DME     ;DME DISTANCE
¿¿8427  CDEB84          CALL    STORE  ; PUT IN BUFFER RAM
¿¿              ;
¿¿842A  3E01            MVI     A,1     ;DELT-THROTTLE HANDLE
¿¿842C  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿842F  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿8432  3E02            MVI     A,2     ;DELF-FLAP LEVER CHANNEL
¿¿8434  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿8437  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿843A  3E03            MVI     A,3     ;DELET-ELEVATOR TRIM CHANNEL
¿¿843C  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿843F  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿8442  3E04            MVI     A,4     ;THETA-PITCH ANGLE CHANNEL
¿¿8444  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿8447  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿844A  3E06            MVI     A,6     ;V-VELOCITY CHANNEL
¿¿844C  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿844F  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿8452  3E0A            MVI     A,10    ;PSI-HEADING ANGLE CHANNEL
¿¿8454  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿8457  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿845A  3E0C            MVI     A,12    ;PHI-ROLL ANGLE CHANNEL
¿¿845C  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿845F  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿8462  3E10            MVI     A,16    ;DELRT-RUDDER TRIM CHANNEL
¿¿8464  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿8467  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿846A  3E11            MVI     A,17    ;DELAT-AILERON TRIM CHANNEL
¿¿846C  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿846F  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿8472  3E12            MVI     A,18    ;DELE-ELEVATOR POSITION CHANNEL
¿¿8474  CD6185          CALL    ADC0    ; READ ADC CHANNEL
¿¿8477  CDEB84          CALL    STORE   ;   PUT IN RAM BUFFER
¿¿              ;
¿¿847A  3E13            MVI     A,19    ;DELT-THROTTLE POSTION CHANNEL
¿¿847C  CD6185          CALL    ADC0    : READ ADC CHANNEL
```

```
¿¿847F CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿8482 3E14            MVI    A,20     ;DELF-FLAP POSIYION CHANNEL
¿¿8484 CD6185          CALL   ADC0     ; READ ADC CHANNEL
¿¿8487 CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿848A 3E16            MVI    A,22     ;DELA-AILERON POSITION CHANNEL
¿¿848C CD6185          CALL   ADC0     ; READ ADC CHANNEL
¿¿848F CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿8492 3E17            MVI    A,23     ;DELR-RUDDER POSITION CHANNEL
¿¿8494 CD6185          CALL   ADC0     ; READ ADC CHANNEL
¿¿8497 CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿849A 3E18            MVI    A,24     ;N1LOC-N1 LOCALISER CHANNEL
¿¿849C CD6185          CALL   ADC0     ; READ ADC CHANNEL
¿¿849F CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿84A2 3E1A            MVI    A,26     ;ALPHAP-PORT ALPHA CHANNEL
¿¿84A4 CD6185          CALL   ADC0     ; READ ADC CHANNEL
¿¿84A7 CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿84AA 3E1B            MVI    A,27     ;ALPHAS-STARBOARD ALPHA CHANNEL
¿¿84AC CD6185          CALL   ADC0     ; READ ADC CHANNEL
¿¿84AF CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿84B2 3E1C            MVI    A,28     ;LOCMLS-MLS LOCALIZER CHANNEL
¿¿84B4 CD6185          CALL   ADC0     ; READ ADC CHANNEL
¿¿84B7 CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿84BA 3E1D            MVI    A,29     ;GSMLS-MLS GLIDE SLOPE CHANNEL
¿¿84BC CD6185          CALL   ADC0     ; READ ADC CHANNEL
¿¿84BF CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿84C2 3E1E            MVI    A,30     ;LOCN2-N2 LOCALIZER CHANNEL
¿¿84C4 CD6185          CALL   ADC0     ; READ ADC CHANNEL
¿¿84C7 CDEB84          CALL   STORE    ;   PUT IN RAMBUFER
¿¿                  ;
¿¿84CA 3E1F            MVI    A,31     ;IDELF-FLAP COMMAND CHANNEL
¿¿84CC CD6185          CALL   ADC0     ; READ ADC CHANNEL
¿¿84CF CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿84D2 CD8385          CALL   DIGH     ;H
¿¿84D5 CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿84D8 CD8985          CALL   DIGLI    ;LIGHTS
¿¿84DB CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿84DE CD8D85          CALL   DIGMOD   ;MODE SWITCHES
¿¿84E1 CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿                  ;
¿¿84E4 CD9385          CALL   DIGTIM   ;TIME
¿¿84E7 CDEB84          CALL   STORE    ;   PUT IN RAM BUFFER
¿¿84EA C9              RET
¿¿                  STORE:
¿¿                  ;
¿¿                  ;
¿¿                  ;                  TAKE CONTENTS OF HL REGISTER
¿¿                  ;                  STORE AT DOUBLE  BYTE POINTED
¿¿                  ;                  TO BY DE REGISTER.
¿¿                  ;
¿¿84EB F5              PUSH   PSW
¿¿84EC 7D              MOV    A,L
¿¿84ED 12              STAX   D
¿¿84EE 13              INX    D
¿¿84EF 7C              MOV    A.H
```

```
¿¿84F0  12                  STAX    D
¿¿84F1  13                  INX     D
¿¿84F2  F1                  POP     PSW
¿¿84F3  C9                  RET
¿¿              ;
¿¿              ;
¿¿              UFBUFS:
¿¿              ;
¿¿              ;
¿¿              ;
¿¿              ;                   THERE ARE TWO POINTERS
¿¿              ;                   AND ONE COUNTER
¿¿              ;                   TO BE UPDATED WHEN CURRENT
¿¿              ;                   BUFFER IS FULL.
¿¿              ;
¿¿              ;                  FWHICHB IS FROM 0 TO 9
¿¿              ;                   AND INDICATES THE CURRENT BUFFER
¿¿              ;                   BEING FILLED
¿¿              ;
¿¿              ;                   SAVE POINTS TO THE BYTE OF
¿¿              ;                   THE CURRENT BUFFER TO BE FILLED .
¿¿              ;
¿¿              ;                   REMAIN CONTAINS THE NUMBER
¿¿              ;                   OF UNFIILED BYTES IN CURRENT BUFFER
¿¿              ;
¿¿84F4  F5                  PUSH    PSW
¿¿84F5  E5                  PUSH    H
¿¿84F6  C5                  PUSH    B
¿¿84F7  2A5988              LHLD    REMAIN    ;
¿¿84FA  014C00              LXI     B,BYTPERSEC     ;BYTPERSEC BYTES FILLED PER SECOND
¿¿84FD  AF                  XRA     A         ;CARRY=0
¿¿84FE  ED42                DSBC    B
¿¿8500  225988              SHLD    REMAIN    ;REMAIN=REMAIN-BYTPERSEC
¿¿              ;
¿¿              ;                             IS REMAIN >= BYTPERSEC
¿¿              ;
¿¿8503  AF                  XRA     A         ;CARRY=0
¿¿8504  ED42                DSBC    B
¿¿8506  F23985              JP      ROOM      ;IF S=0, HL >= BC
¿¿              ;
¿¿              ;
¿¿              ;                             CAN NOT FIT
¿¿              ;                             ANOTHER BLOCK OF BYTPERSEC BYTES
¿¿              ;                             IN CURRENT BUFFER
¿¿              ;
¿¿8509  CD9487              CALL    SETFULL   ;MARK CURRENT BUFFER FULL
¿¿              ;
¿¿850C  210008              LXI     H,2048
¿¿850F  225988              SHLD    REMAIN    ;REMAIN=2048
¿¿              ;
¿¿8512  3A5B88              LDA    FWHICHB    ;SWITCH TO NEXT BUFFER
¿¿8515  3C                  INR     A         ;
¿¿8516  325B88              STA    FWHICHB    ;
¿¿8519  FE0A                CPI     10
¿¿851B  C22D85              JNZ     SW
¿¿              ;
¿¿              ;           HLT
¿¿851E  3E00                MVI     A,0
¿¿8520  325B88              STA    FWHICHB
¿¿8523  110090              LXI     D,BUFF0
¿¿8526  ED537688            SDED    CURRBUF
¿¿852A  C33985              JMP     ROOM
¿¿              ;
¿¿852D  2A7688  SW:         LHLD    CURRBUF   ;BASE ADDRESS OF CURRENT BUFFER
¿¿8530  010008              LXI     B,2048
¿¿8533  09                  DAD     B         : INCREMENT BY 2048
```

```
¿¿8534 227688         SHLD    CURRBUF  ;UPDATED BASE ADDRESS SAVED
¿¿8537 54             MOV     D,H      ;AND PUT INTO
¿¿8538 5D             MOV     E,L      ;  THE DE PAIR
¿¿             ;
¿¿             ROOM:
¿¿8539 C1             POP     B
¿¿853A E1             POP     H
¿¿853B F1             POP     PSW
¿¿853C C9             RET
¿¿             UINTS:
¿¿853D F5             PUSH    PSW
¿¿853E 3A5E88         LDA     CNT
¿¿8541 3C             INR     A
¿¿8542 325E88         STA     CNT
¿¿8545 F1             POP     PSW
¿¿8546 C9             RET
¿¿             RINTS:
¿¿8547 F5             PUSH    PSW
¿¿8548 3A5E88         LDA     CNT
¿¿854B FE0A           CPI     10
¿¿854D C25585         JNZ     NOT10
¿¿8550 3E00           MVI     A,0
¿¿8552 325E88         STA     CNT
¿¿             NOT10:
¿¿8555 F1             POP     PSW
¿¿8556 C9             RET
¿¿             UCLK:
¿¿8557 E5             PUSH    H
¿¿8558 2A5C88         LHLD    TIM
¿¿855B 23             INX     H
¿¿855C 225C88         SHLD    TIM
¿¿855F E1             POP     H
¿¿8560 C9             RET
¿¿             ADC0:
¿¿8561 320130         STA     ADDAM+01H
¿¿8564 3E01           MVI     A,01
¿¿8566 320030         STA     ADDAM+00H
¿¿8569 3A0030  BUSY   LDA     ADDAM+00H
¿¿856C 07             RLC
¿¿856D D26985         JNC     BUSY
¿¿8570 2A0430         LHLD    ADDAM+04H
¿¿8573 C9             RET
¿¿             CDC0:
¿¿8574 6F             MOV     L,A
¿¿             ;
¿¿8575 2600           MVI     H,0
¿¿8577 C9             RET
¿¿             BDC0:
¿¿8578 6F             MOV     L,A
¿¿8579 3A5E88         LDA     CNT
¿¿857C 67             MOV     H,A
¿¿857D C9             RET
¿¿             DME:
¿¿857E 2E00           MVI     L,0
¿¿8580 2600           MVI     H,0
¿¿8582 C9             RET
¿¿             DIGH:
¿¿             ;      MVI     L,34
¿¿             ;      MVI     H,0
¿¿8583 DBE4           IN      ALTPORT         ;ALTIMETER READING
¿¿8585 6F             MOV     L,A
¿¿8586 2600           MVI     H,0
¿¿8588 C9             RET
¿¿             DIGLI:
¿¿             ;      MVI     L,35
¿¿             ;      MVI     H,0
```

```
¿¿8589   2A5288       922            LHLD    LIGHTS
¿¿858C   C9           923            RET
¿¿                    924  DIGMOD:
¿¿858D   DBE5         925            IN      PUSHPORT
¿¿858F   6F           926            MOV     L,A
¿¿8590   2600         927            MVI     H,0
¿¿                    928  ;         MVI     L,36
¿¿                    929  ;         MVI     H,0
¿¿8592   C9           930            RET
¿¿                    931  DIGTIM:
¿¿8593   2A5C88       932            LHLD    TIM
¿¿8596   C9           933            RET
¿¿                    934  STATE:
¿¿                    935  ;         THISMACHINE HAS TWO STATES
¿¿                    936  ;         STATE 0 THE MACHINE STAYS IN THIS STATE FOR 30 SECS.
¿¿                    937  ;                 AFTER 30 SECS. ELAPSE, IT TURNS ON THE PROPER LIGHT
¿¿                    938  ;                 AND SHIFTS TO STATE 1.
¿¿                    939  ;
¿¿                    940  ;         STATE 1 THE MACHINE STAYS IN STATE 1 UNTIL THE PROPER
¿¿                    941  ;                 PUSHBUTTON IS DEPRESSED.(LIGHT X GOES WITH PB X, ETC)
¿¿                    942  ;                 THE CORRESPONDING LIGHT IS TURNED OFF AND THE
¿¿                    943  ;                 MACHINE SHIFTS TO STATE 0.
¿¿8597   F5           944            PUSH    PSW
¿¿8598   E5           945            PUSH    H
¿¿8599   C5           946            PUSH    B
¿¿859A   3A4E88       947            LDA     ST
¿¿859D   B7           948            ORA     A          ;SET THE FLAGS
¿¿859E   C2C185       949            JNZ     STATE1
¿¿                    950  STATE0:
¿¿85A1   212C01       951            LXI     H,THSEDY   ;LOAD THE NUMBER OF TIMER CLICKS NEEDED FOR 30 SEC
¿¿85A4   AF           952            XRA     A          ;CLEAR CARRY
¿¿85A5   ED4B5088     953            LBCD    PTIM
¿¿85A9   ED42         954            DSBC    B
¿¿85AB   C2B785       955            JNZ     BPTIM
¿¿85AE   CDDA85       956            CALL    TONL
¿¿85B1   CD3D86       957            CALL    TONS       ;STATE=1
¿¿85B4   C3D685       958            JMP     BACK
¿¿                    959  BPTIM:
¿¿85B7   2A5088       960            LHLD    PTIM
¿¿85BA   23           961            INX     H
¿¿85BB   225088       962            SHLD    PTIM
¿¿85BE   C3D685       963            JMP     BACK
¿¿                    964  STATE1:
¿¿85C1   CD2586       965            CALL    PB         ;RETURN PUSHBUTTON STATUS IN Z FLAG
¿¿85C4   C2D685       966            JNZ     BACK
¿¿                    967  PBON:
¿¿85C7   CDFB85       968            CALL    TOFFL      ;TURN OFF LITE
¿¿85CA   CD4586       969            CALL    TOFFS      ;STATE=0
¿¿85CD   CD1C86       970            CALL    CPTIM      ;PTIM = 0
¿¿85D0   CD5C86       971            CALL    SWPBLITE     ;SWITCH TO NEXT PUSHBUTON + LITE SET
¿¿85D3   C3D685       972            JMP     BACK
¿¿                    973  BACK:
¿¿85D6   C1           974            POP     B
¿¿85D7   E1           975            POP     H
¿¿85D8   F1           976            POP     PSW
¿¿85D9   C9           977            RET
¿¿                    978  TONL:
¿¿85DA   F5           979            PUSH    PSW
¿¿85DB   3A5688       980            LDA     BINRY
¿¿85DE   FE00         981            CPI     0
¿¿85E0   C2EF85       982            JNZ     ON2
¿¿85E3   3A5288       983            LDA     LIGHTS
¿¿85E6   E6FE         984            ANI     0FEH    ;
¿¿85E8   D3C5         985            OUT     LITEPORT       ;BIT 0  PORT B   5 VOLTS
¿¿85EA   325288       986            STA     LIGHTS
¿¿85ED   F1           987            POP     PSW
```

```
¿¿85EE  C9                      RET
¿¿                      ON2:
¿¿85EF  3A5288                  LDA     LIGHTS
¿¿85F2  E6FD                    ANI     0FDH     ;
¿¿85F4  D3C5                    OUT     LITEPORT       ;BIT 1 PORT B  5 VOLTS
¿¿85F6  325288                  STA     LIGHTS
¿¿85F9  F1                      POP     PSW
¿¿85FA  C9                      RET
¿¿                      TOFFL:
¿¿85FB  F5                      PUSH    PSW
¿¿85FC  3A5688                  LDA     BINRY
¿¿85FF  FE00                    CPI     0
¿¿8601  C21086                  JNZ     OFF2
¿¿8604  3A5288                  LDA     LIGHTS
¿¿8607  F601                    ORI     01H          ;
¿¿8609  D3C5                    OUT     LITEPORT       ;BIT 0 PORT B  0 VOLTS
¿¿860B  325288                  STA     LIGHTS
¿¿860E  F1                      POP     PSW
¿¿860F  C9                      RET
¿¿                      OFF2:
¿¿8610  3A5288                  LDA     LIGHTS        ;
¿¿8613  F602                    ORI     02H
¿¿8615  D3C5                    OUT     LITEPORT      ;BIT 0 PORT B  0 VOLTS
¿¿8617  325288                  STA     LIGHTS
¿¿861A  F1                      POP     PSW
¿¿861B  C9                      RET
¿¿                      CPTIM:
¿¿861C  E5                      PUSH    H
¿¿861D  210000                  LXI     H,0
¿¿8620  225088                  SHLD    PTIM       ;CLEAR PTIM
¿¿8623  E1                      POP     H
¿¿8624  C9                      RET
¿¿                      PB:
¿¿8625  3A5688                  LDA     BINRY
¿¿8628  FE00                    CPI     0
¿¿862A  C23586                  JNZ     PB2
¿¿862D  DBE5                    IN      PUSHPORT      ;PUSHBUTOON STATUS
¿¿862F  325488                  STA     MODES
¿¿8632  E601                    ANI     01H        ;BIT 0  PORT B
¿¿8634  C9                      RET                ;STATUS RETURNED IN Z FLAG
¿¿                      PB2:
¿¿8635  DBE5                    IN      PUSHPORT
¿¿8637  325488                  STA     MODES
¿¿863A  E602                    ANI     02H        ;BIT 1  PORT B
¿¿863C  C9                      RET
¿¿                      TONS:
¿¿863D  F5                      PUSH    PSW
¿¿863E  3E01                    MVI     A,01H      ;
¿¿8640  324E88                  STA     ST         ;STATE=1
¿¿8643  F1                      POP     PSW
¿¿8644  C9                      RET
¿¿                      TOFFS:
¿¿8645  F5                      PUSH    PSW
¿¿8646  3E00                    MVI     A,0
¿¿8648  324E88                  STA     ST
¿¿864B  F1                      POP     PSW
¿¿864C  C9                      RET
¿¿                      FLIP:
¿¿                      ;                         ;TOGGLE LIGHT 3, TOGGLE LIGHT 4 TOGETHER
¿¿864D  F5                      PUSH    PSW
¿¿864E  3A5288                  LDA     LIGHTS
¿¿8651  EE04                    XRI     04H      ;FLIP BIT 2
¿¿8653  EE08                    XRI     08H      ;FLIP BIT 3
¿¿8655  325288                  STA     LIGHTS
¿¿8658  D3C5                    OUT     LITEPORT
¿¿865A  F1                      POP     PSW
```

```
¿¿865B  C9                RET
¿¿                SWPBLITE:
¿¿                ;         IF YOU ARE USING PB0 AND LITE0,
¿¿                ;         USE PB1 AND LITE1.
¿¿                ;         IF YOU ARE USING PB1 AND LITE1,
¿¿                ;         USE PB0 AND LITE0.
¿¿                ;
¿¿865C  F5                PUSH    PSW
¿¿865D  3A5688            LDA     BINRY
¿¿8660  EE01              XRI     01H
¿¿8662  325688            STA     BINRY
¿¿8665  F1                POP     PSW
¿¿8666  C9                RET
¿¿                CTIM:
¿¿8667  E5                PUSH    H
¿¿8668  210000            LXI     H,0         ;HL=0
¿¿866B  225C88            SHLD    TIM         ;SECOND COUNTER = 0
¿¿866E  E1                POP     H
¿¿866F  C9                RET
¿¿                SETCDIG:
¿¿                ;
¿¿                ;                         SET DIGIT COUNTER CNTDIG
¿¿                ;
¿¿8670  F5                PUSH    PSW
¿¿8671  3E00              MVI     A,0
¿¿8673  324D88            STA     CNTDIG    ;CNTDIG=0
¿¿8676  F1                POP     PSW
¿¿8677  C9                RET
¿¿                INTWODIG:
¿¿8678  E5                PUSH    H
¿¿8679  C5                PUSH    B
¿¿867A  D5                PUSH    D
¿¿                                          ;CHECK TO SEE IF THERE IS
¿¿                                          ;A TWO DIGIT INPUT READY
¿¿                                          ;
¿¿                                          ;IF THERE IS NOT,
¿¿                                          ;  RETURN WITH Z SET
¿¿                                          ;
¿¿                                          ;IF THERE IS,
¿¿                                          ;  RETURN WITH Z RESET
¿¿                                          ;
¿¿                                          ;
¿¿867B  CDB186            CALL    DI        ;GET DIGIT IF WAITING
¿¿867E  CAAD86            JZ      INPUTR    ; IF Z SET, NOTGHING WAITING
¿¿                ;
¿¿                ;
¿¿                                          ;Z NOT SET, SO DIGIT IN ACC
¿¿8681  E60F              ANI     0FH       ;CONVERT ASCII DIGIT TO BINARY DIGIT
¿¿                ;       CALL    HDMP
¿¿                ;       CALL    CRLF
¿¿8683  57                MOV     D,A       ;SAVE DIGIT IN D
¿¿8684  3A4D88            LDA     CNTDIG    ;CNTDIG IS DIGIT COUNTER
¿¿8687  3C                INR     A
¿¿8688  324D88            STA     CNTDIG
¿¿868B  FE02              CPI     02H       ;
¿¿868D  CA9886            JZ      INPUTE    ;IF Z SET, COUNT IS 2
¿¿                ;
¿¿                ;
¿¿                                          ;HAVE INPUT IN D, SAVE IN B
¿¿8690  7A                MOV     A,D       ;
¿¿8691  324B88            STA     INPUTS    ;SAVE FIRST INPUT DIGIT
¿¿8694  AF                XRA     A         ;Z IS SET
¿¿8695  C3AD86            JMP     INPUTR
¿¿                INPUTE:
¿¿8698  3A4B88            LDA     INPUTS    ;MOVE FIRST INPUT DIGIT IN ACC
¿¿869B  07                RLC
```

```
¿¿869C    07                  RLC
¿¿869D    07                  RLC
¿¿869E    47                  MOV     B,A       ;(B)=8*(INPUTS)
¿¿869F    3A4B88              LDA     INPUTS    ;
¿¿86A2    87                  ADD     A         ;(A)=2*(INPUTS)
¿¿86A3    80                  ADD     B         ;(A)=2*(INPUTS)+8*(INPUTS)=10*(INPUTS)
¿¿86A4    82                  ADD     D         ;ADD IN SECOND INPUT DIGIT
¿¿86A5    324C88              STA     TWODIG
¿¿86A8    3E00                MVI     A,0
¿¿86AA    324D88              STA     CNTDIG    ;CLEAR THE COUNTER
¿¿                  INPUTR:
¿¿86AD    D1                  POP     D
¿¿86AE    C1                  POP     B
¿¿86AF    E1                  POP     H
¿¿86B0    C9                  RET
¿¿                  DI:
¿¿                  ;                           ACC AND FLAGS ARE CHENGED
¿¿                  ;                           GET INPUT IF WAITING
¿¿                  ;                           OTHERWISE RETURN WITH Z SET
¿¿                  ;
¿¿                  ;                           SEE IF INPUT IS ADECIMAL DIGIT,
¿¿                  ;                           IF NOT RETURN WITH Z SET
¿¿                  ;
¿¿                  ;                           IF INPUT IS A DECIMAL DIGIT,
¿¿                  ;                           ECHO IT AND RETURN IN ACC
¿¿                  ;
¿¿                  ;
¿¿86B1    DBCD                IN      S8251
¿¿86B3    E602                ANI     RXRDY     ;IS THERE INPUT DATA YET
¿¿86B5    C8                  RZ                ;IF Z IS SET, RETURN
¿¿                  ;
¿¿                  ;                           READ THE CHAR
¿¿                  ;
¿¿86B6    DBCC                IN      D8251     ;GET CHAR
¿¿86B8    E67F                ANI     07FH      ;  CHOP OFF PARITY
¿¿                  ;
¿¿                  ;                           TEST FOR DIGIT FROM 0 TO 9
¿¿                  ;
¿¿86BA    FE30                CPI     30H       ; COMPARE WITH ASCII 0
¿¿86BC    FAC886              JM      DINODIG   ; JUMP IF LESS THAN
¿¿86BF    FE3A                CPI     3AH       ;COMPARE WITH ASCII COLON
¿¿86C1    F2C886              JP      DINODIG   ;  JUMP IF GREATER OR EQUAL
¿¿86C4    CD0B88              CALL    CO        ;HAVE LEGAL DIGIT IN ACC, Z IS RESET
¿¿86C7    C9                  RET
¿¿                  ;
¿¿                  ;
¿¿                  DINODIG:
¿¿86C8    AF                  XRA     A         ;Z IS SET
¿¿86C9    C9                  RET
¿¿                  DISPLAY:
¿¿86CA    F5                  PUSH    PSW
¿¿86CB    E5                  PUSH    H
¿¿86CC    C5                  PUSH    B
¿¿                                              ;DISPLAY MEMORY LOCATION
¿¿                                              ;CORRESPONDING TO TWODIG ADDRESS
¿¿                                              ;
¿¿86CD    3A5E88              LDA     CNT       ;INTERRUPT COUNT LOADED IN ACC
¿¿86D0    FE09                CPI     9         ; IS THIS THE NINTH INTERRUPT
¿¿86D2    C2F386              JNZ     DISPLAYR  ;   RETURN IF NOT NINE
¿¿                                              ;
¿¿                                              ;
¿¿86D5    3A4C88              LDA     TWODIG    ;LOAD TWO DIGIT NUMBER
¿¿86D8    FE00                CPI     0         ;  IF IT IS ZERO, NOTHING TO DISPLAY YET
¿¿86DA    CAF386              JZ      DISPLAYR  ;    SO RETURN
¿¿                                              ;
¿¿                                              ;
```

```
¿¿86DD  87            1186                ADD     A         ;FORM BYTE OFFSET IN ACC
¿¿86DE  D602          1187                SUI     2         ;  VALUE OF 1 IS AN OFFSET OF ZERO
¿¿                    1188                                  ;
¿¿86E0  2600          1189                MVI     H,0       ;MOVE ACC OFFSET
¿¿86E2  6F            1190                MOV     L,A       ;  HL HAS ACC OFFSET
¿¿86E3  ED4B5788      1191                LBCD    SAVEDE    ;    BASE ADDRESS OF CURRENT STORAGE IN BC
¿¿86E7  09            1192                DAD     B         ;      ADDED TO OFFSET TO FORM POINTER TO
¿¿                    1193                                  ;        MEMORY LOCATION YOU WANT TO SEE
¿¿                    1194                                  ;
¿¿86E8  CD2A87        1195                CALL    CRLF
¿¿86EB  CDF786        1196                CALL    HDMP2     ;
¿¿                    1197                                  ;
¿¿                    1198                                  ;
¿¿                    1199      ;         CALL    CRLF
¿¿                    1200      ;         LDA     TWODIG
¿¿                    1201      ;         CALL    HDMP
¿¿                    1202      ;         CALL    CRLF
¿¿                    1203      ;         LXI   H,SAVEDE
¿¿                    1204      ;         CALL    HDMP2
¿¿                    1205      ;         CALL    CRLF
¿¿86EE  3E00          1206                MVI     A,0       ;ZERO
¿¿86F0  324C88        1207                STA     TWODIG    ;  TWODIG
¿¿                    1208                                  ;
¿¿                    1209                                  ;
¿¿                    1210      DISPLAYR:
¿¿86F3  C1            1211                POP     B
¿¿86F4  E1            1212                POP     H
¿¿86F5  F1            1213                POP     PSW
¿¿86F6  C9            1214                RET
¿¿                    1215                ; DUMP THE CONTENTS OF MEMORY IN HEX
¿¿                    1216                ; H HOLDS STARTING ADDRESS OF DUMP
¿¿                    1217                ; A REG IS USED IN HDUMP
¿¿86F7  F5            1218      HDMP2:    PUSH    PSW
¿¿86F8  E5            1219                PUSH    H
¿¿86F9  23            1220                INX     H
¿¿86FA  7E            1221                MOV     A,M
¿¿86FB  CD0987        1222                CALL    HDMP
¿¿86FE  2B            1223                DCX     H
¿¿86FF  7E            1224                MOV     A,M
¿¿8700  CD0987        1225                CALL    HDMP
¿¿8703  CD2A87        1226                CALL    CRLF
¿¿8706  E1            1227                POP     H
¿¿8707  F1            1228                POP     PSW
¿¿8708  C9            1229                RET
¿¿                    1230      ;                 HDMP USES THE A REG
¿¿8709  F5            1231      HDMP:     PUSH    PSW       ;SAVE ACC
¿¿870A  F5            1232                PUSH    PSW       ;  TWICE
¿¿870B  E6F0          1233                ANI     0F0H      ;ISOLATE THE HIGH ORDER NYBBLE
¿¿870D  0F            1234                RRC
¿¿870E  0F            1235                RRC
¿¿870F  0F            1236                RRC
¿¿8710  0F            1237                RRC               ;HI DIG SHIFTED RIGHT BY 4
¿¿8711  CD2287        1238                CALL    BINHE     ;PRINT HIGH ORDER DIGIT
¿¿8714  CD0B88        1239                CALL    CO      ;PRINT ASCII FORM OF HIGH DIGIT
¿¿8717  F1            1240                POP     PSW       ;RESTORE THE ACC TO VALUE AT ENTRY
¿¿8718  E60F          1241                ANI     0FH       ;ISOLATE THE LOW ORDER NYBBLE
¿¿871A  CD2287        1242                CALL    BINHE     ;
¿¿871D  CD0B88        1243                CALL    CO      ;
¿¿8720  F1            1244                POP     PSW       ;
¿¿8721  C9            1245                RET
¿¿8722  C630          1246      BINHE:    ADI     30H       ;
¿¿8724  FE3A          1247                CPI     3AH       ;
¿¿8726  D8            1248                RC
¿¿8727  C607          1249                ADI     7H
¿¿8729  C9            1250                RET
¿¿                    1251      :                 CRLF USES ONLY THE AREG
```

```
¿¿872A F5         CRLF:   PUSH    PSW
¿¿872B 3E0D               MVI     A,0DH
¿¿872D CD0B88             CALL    CO
¿¿8730 3E0A               MVI     A,0AH
¿¿8732 CD0B88             CALL CO
¿¿8735 F1                 POP     PSW
¿¿8736 C9                 RET
¿¿                ;DATAOVRUN:
¿¿                ;       ARE WE FILLING FASTER THAN WE ARE EMPTYING
¿¿                ;        PUSH    PSW
¿¿                ;        PUSH    B
¿¿                ;        LDA     EWHICHB
¿¿                ;        MOV     B,A
¿¿                ;        LDA     FWHICHB
¿¿                ;        CMP     B
¿¿                ;        JNZ     D1
¿¿                ;                          DATA OVERRUN
¿¿                ;        MVI     A,'O'
¿¿                ;        CALL    BO
¿¿                ;        HLT
¿¿                ;D1:
¿¿                ;        POP     B
¿¿                ;        POP     PSW
¿¿                ;        RET
¿¿                ;VALIDMP:
¿¿                ;                          LET THE FILLER CATCHUP WITH EMPTYER
¿¿                ;        PUSH    PSW
¿¿                ;        PUSH    B
¿¿                ;        LDA     FILLONE
¿¿                ;        CPI     0
¿¿                ;        JNZ     V1
¿¿                                           ;ONE BUFFER HAS NOT YET BEEN FILLED
¿¿                ;V2:     LDA     FWHICHB
¿¿                ;        CPI     0
¿¿                ;        MVI     A,'W'
¿¿                ;        CALL    CO
¿¿                ;        JZ     V2
¿¿                                           ;ONE BUFFER HAS BEEN FILLED
¿¿                ;        MVI     A,1
¿¿                ;        STA     FILLONE
¿¿                ;V1:
¿¿                                           ;MAKE SURE FILLER STAYS AHEAD OF EMPTYER
¿¿                ;        LDA     EWHICHB
¿¿                ;        MOV     B,A
¿¿                ;        LDA     FWHICHB
¿¿                ;        CMP     B
¿¿                ;        LDA     FWHICHB
¿¿                ;        ADI     30H
¿¿                ;        CALL    CO
¿¿                ;        JZ      V1
¿¿                ;        POP     B
¿¿                ;        POP     PSW
¿¿                ;        RET
¿¿                INITSEMA:
¿¿                                           ;MARK ALL BUFFERS EMPTY
¿¿8737 F5                 PUSH    PSW
¿¿8738 C5                 PUSH    B
¿¿8739 E5                 PUSH    H
¿¿873A 216A88             LXI     H,SEMAPHORE       ;
¿¿873D 060A               MVI     B,10
¿¿873F 3645       S1:     MVI     M,'E'
¿¿8741 23                 INX     H          ;ADDRESS NEXT SEMIPHORE
¿¿8742 05                 DCR     B          ;ONE LESS SEMIPHORE TO INITIALISE
¿¿8743 C23F87             JNZ     S1         ;REPEAT UNTIL ALL SEMIPHORES ARE INITIALISED
¿¿8746 E1                 POP     H
¿¿8747 C1                 POP     B
```

```
¿¿8748 F1                   POP    PSW
¿¿8749 C9                   RET
¿¿             TSTFULL:
¿¿                                                ;WAIT UNTIL BUFFER EMPTY (INFINITE LOOP!)
¿¿874A F5                   PUSH   PSW
¿¿874B C5                   PUSH   B
¿¿874C E5                   PUSH   H
¿¿874D 216A88               LXI    H,SEMAPHORE
¿¿8750 3A5B88               LDA    FWHICHB
¿¿8753 0600                 MVI    B,0
¿¿8755 4F                   MOV    C,A
¿¿8756 09                   DAD    B          ;(HL):= SEMAPHORE+(FWHICHB)
¿¿8757 7E      FULL1:       MOV    A,M        ;(A):= (SEMAPHORE+(FWHICHB))
¿¿8758 FE46                 CPI    'F'        ;SEE IF BUFFER FULL
¿¿875A CC6187               CZ     OVRUN      ;WAIT TIL EMPTY
¿¿875D E1                   POP    H
¿¿875E C1                   POP    B
¿¿875F F1                   POP    PSW
¿¿8760 C9                   RET
¿¿             OVRUN:
¿¿8761 3E4F                 MVI    A,'O'
¿¿8763 CD0B88               CALL   CO
¿¿8766 3E56                 MVI    A,'V'
¿¿8768 CD0B88               CALL   CO
¿¿876B 3A5B88               LDA    FWHICHB
¿¿876E C641                 ADI    'A'
¿¿8770 CD0B88               CALL   CO
¿¿8773 C30000               JMP    0
¿¿             TSTEMPTY:
¿¿                                                ;WAIT UNTIL BUFFER FULL
¿¿8776 F5                   PUSH   PSW
¿¿8777 C5                   PUSH   B
¿¿8778 E5                   PUSH   H
¿¿8779 216A88               LXI    H,SEMAPHORE
¿¿877C 3A6888               LDA    EWHICHB
¿¿877F 0600                 MVI    B,0
¿¿8781 4F                   MOV    C,A
¿¿8782 09                   DAD    B
¿¿8783 7E      EMPTY1:      MOV    A,M        ;(A):=(SEMAPHORE+(EWHICHB))
¿¿8784 FE45                 CPI    'E'        ;SEE IF BUFFER EMPTY
¿¿8786 F5                   PUSH   PSW        ;SAVE Z FLAG
¿¿8787 3A6888               LDA    EWHICHB
¿¿878A C630                 ADI    '0'        ;CONVERT TO ASCII
¿¿             ;             CALL   CO
¿¿878C F1                   POP    PSW        ;RESTORE Z FLAG
¿¿878D CA8387               JZ     EMPTY1     ;WAIT UNTIL FULL
¿¿8790 E1                   POP    H
¿¿8791 C1                   POP    B
¿¿8792 F1                   POP    PSW
¿¿8793 C9                   RET
¿¿             SETFULL:
¿¿                                                ;MARK BuffeR FULL
¿¿8794 F5                   PUSH   PSW
¿¿8795 C5                   PUSH   B
¿¿8796 E5                   PUSH   H
¿¿8797 216A88               LXI    H,SEMAPHORE
¿¿879A 3A5B88               LDA    FWHICHB
¿¿879D 0600                 MVI    B,0
¿¿879F 4F                   MOV    C,A        ;(BC):=(FWHICHB)
¿¿87A0 09                   DAD    B          ;(HL):=SEMAPHORE+(FWHICHB)
¿¿87A1 3E46                 MVI    A,'F'      ;MARK IT FULL
¿¿87A3 77                   MOV    M,A        ;(SEMAPHORE+(FWHICHB)):='F'
¿¿87A4 E1                   POP    H
¿¿87A5 C1                   POP    B
¿¿87A6 F1                   POP    PSW
¿¿87A7 C9                   RET
```

```
ii                 SETEMPTY:
ii                                                 ;MARK BUFFER EMPTY
ii87A8  F5                  PUSH    PSW
ii87A9  C5                  PUSH    B
ii87AA  E5                  PUSH    H
ii87AB  216A88              LXI     H,SEMAPHORE
ii87AE  3A6888              LDA     EWHICHB
ii87B1  0600                MVI     B,0
ii87B3  4F                  MOV     C,A                 ;(BC):=(EWHICHB)
ii87B4  09                  DAD     B          ;(HL):=SEMAPHORE+(EWHICHB)
ii87B5  3E45                MVI     A,'E'      ;MARK IT EMPTY
ii87B7  77                  MOV     M,A        ;(SEMAPHORE+(EWHICHB)):='E'
ii87B8  E1                  POP     H
ii87B9  C1                  POP     B
ii87BA  F1                  POP     PSW
ii87BB  C9                  RET
ii                 INITRE:
ii87BC  CD7086              CALL    SETCDIG    ;ZERO DIGIT COUNTER, CNTDIG
ii87BF  CD2783              CALL    CNTR0
ii87C2  CD1683              CALL    CNTR1
ii87C5  3E0A                MVI     A,10
ii87C7  326188              STA     FLAG
ii87CA  3E80                MVI     A,80H        ;FORMAT 8255 AS MODE 0,
ii87CC  D3EB                OUT .   0EBH         ;  ALL THREE PORTS OUTPUT ON J-2 8004
ii87CE  110090              LXI     D,BUFF0      ;  SAVEDE POINTER =
ii87D1  ED535788            SDED    SAVEDE       ;  SET TO BASE ADDRESS OF ALL TEN BUFFERS
ii87D5  3E00                MVI     A,00H        ; POINTER THAT TELLS WHICH BUFFER
ii87D7  325B88              STA     FWHICHB      ;  IS BEING FILLED = 0
ii87DA  210008              LXI     H,2048     ;COUNTER TO TELL HOW MUCH
ii87DD  225988              SHLD    REMAIN     ; OF A BUFFER IS UNFILLED.
ii87E0  110090              LXI     D,BUFF0      ; POINTER TO BASE ADDRESS OF CURRENT
ii87E3  210000              LXI     H,0        ;ZERO OUT
ii87E6  225E88              SHLD     CNT        ; INTERRUPT COUNTER
ii87E9  ED537688            SDED    CURRBUF      ;  BUFFER BEING FILLED
ii                 ;         MVI     A,0          ;
ii                 ;         STA     FILLONE      ;FILLONE=0, MEANS ALL BUFFERS ARE EMPTY
ii                                               ; OF DATA.
ii87ED  3E80                MVI     A,080H
ii87EF  D3C7                OUT     LITECTRL     ;FORMAT LIGHTS
ii87F1  3E9B                MVI     A,09BH
ii87F3  D3E7                OUT     J18004       ;FORMAT PUSHBUTTONS.
ii87F5  3EFF                MVI     A,0FFH       ;TURN OFF LIGHTS
ii87F7  D3C5                OUT     LITEPORT
ii87F9  325288              STA     LIGHTS       ;  SAVE LIGHT STATUS
ii87FC  CD4586              CALL    TOFFS      ;STATE=0
ii87FF  CD1C86              CALL    CPTIM        ;PTIM=0
ii8802  CD6786              CALL    CTIM         ;CLEAR SECOND COUNTER
ii8805  3E00                MVI     A,0
ii8807  325688              STA     BINRY        ;BINRY=0, PB0 AND LITE0 FIRST
ii880A  C9                  RET
ii                 ;*****************************************************
ii                 ;                    . CHARACTER OUTPUT ROUTINE
ii                 ; CO OUTPUTS ONE CHARACTER FROM ACC TO TERMINAL
ii                 ; VIA THE USART. ALL REGISTERS AND FLAGS ARE
ii                 ; PRESERVED. THE CHARACTER  IT OUTPUTS IS IN THE ACC.
ii                 ;*****************************************************
ii880B  F5         CO:      PUSH    PSW
ii880C  C5                  PUSH    B
ii880D  4F                  MOV     C,A        ;SAVE A REG
ii880E  00         CO0:     NOP                :DELAY
ii880F  00                  NOP                ;
ii8810  DBD1                IN      S8251      ;GET USART STATUS
ii8812  E601                ANI     TXRDY      ;CHECK TRANSMIT READY FLAG
ii8814  [illegible]         JZ      CO0        [illegible]
ii8817  00                  NOP
[illegible]
```

```
¿¿8819  00          1450                NOP
¿¿881A  00          1451                NOP
¿¿881B  79          1452                MOV     A,C       ;READY TO TRANSMIT , RESTORE CHAR TO A REG
¿¿881C  D3CC        1453                OUT     D8251     ;SEND IT
¿¿881E  C1          1454                POP     B
¿¿881F  F1          1455                POP     PSW
¿¿8820  C9          1456                RET
¿¿                  1457        INITUSART:
¿¿8821  CD2B88      1458                CALL    RUSART    ;MASTER RESET SEQUENCE
¿¿8824  CD3C88      1459                CALL    MUSART    ;SET MODE IN USART
¿¿8827  CD4388      1460                CALL    CUSART    ;SET COMMAND IN USART
¿¿882A  C9          1461                RET
¿¿882B  3E80        1462        RUSART: MVI     A,80H     ;RESET 8251
¿¿882D  D3CD        1463                OUT     C8251     ;
¿¿882F  E3          1464                XTHL
¿¿8830  E3          1465                XTHL
¿¿8831  D3CD        1466                OUT     C8251
¿¿8833  E3          1467                XTHL
¿¿8834  E3          1468                XTHL
¿¿8835  3E40        1469                MVI     A,40H     ;
¿¿8837  D3CD        1470                OUT     C8251     ;
¿¿8839  E3          1471                XTHL
¿¿883A  E3          1472                XTHL
¿¿883B  C9          1473                RET
¿¿883C  3E4E        1474        MUSART: MVI     A,USMODE
¿¿883E  D3CD        1475                OUT     C8251
¿¿8840  E3          1476                XTHL
¿¿8841  E3          1477                XTHL
¿¿8842  C9          1478                RET
¿¿8843  3E37        1479        CUSART: MVI     A,USCMD   ;REST ERROR FLAGS
¿¿                  1480                                  ;   ENABLE TRANSMIT
¿¿                  1481                                  ;      ENABLE RECIEVE
¿¿                  1482                                  ;         READY DATA SET
¿¿                  1483                                  ;
¿¿                  1484                                  ;
¿¿8845  D3CD        1485                OUT C8251 ;GIVE COMMAND
¿¿8847  E3          1486                XTHL
¿¿8848  E3          1487                XTHL
¿¿8849  C9          1488                RET
¿¿884A  C9          1489                RET
¿¿                  1490                                  ;
¿¿004C              1491        BYTPERSEC  EQU   2*38   ; HOW MANY BYTES PER SEC OF DATA
¿¿012C              1492        THSEDY: EQU       300     ;NUMBER OF TIMER CLICKS NEEDED FOR 30 DELAY
¿¿                  1493                                  ;300 FOR .1 SEC TIMER CLICKS
¿¿                  1494                                  ;150 FOR .2 SEC TIMER CLICKS
¿¿0010              1495        TLOW    EQU     010H      ;
¿¿0027              1496        THIGH   EQU     027H      ;2710H=10000D
¿¿                  1497        ;TLOW   EQU     020H      ;
¿¿                  1498        ;THIGH  EQU     04EH      ;4E20H=20000D
¿¿004E              1499        USMODE: EQU     04EH      ;
¿¿0037              1500        USCMD:  EQU     037H      ;
¿¿884B  00          1501        INPUTS: DB      0
¿¿884C  00          1502        TWODIG: DB      0
¿¿884D  00          1503        CNTDIG: DB      0
¿¿00CD              1504        S8251   EQU     0CDH
¿¿00CD              1505        C8251   EQU     S8251
¿¿00CC              1506        D8251   EQU     0CCH
¿¿0002              1507        RXRDY   EQU     02H
¿¿0001              1508        TXRDY   EQU     01H
¿¿884E  0000        1509        ST:     DW      0
¿¿8850  0000        1510        PTIM:   DW      0
¿¿8852  0000        1511        LIGHTS: DW      0
¿¿8854  0000        1512        MODES:  DW      0
¿¿8856  00          1513        BINRY:  DB      0
¿¿8857  0090        1514        SAVEDE: DW      BUFFO
¿¿8859  0008        1515        REMAIN: DW      2048
```

```
¿¿885B  00           1516   FWHICHB: DB        0          ;FILL WHICH BUFFER
¿¿885C  0000         1517   TIM:     DW        0
¿¿885E  0000         1518   CNT:     DW        0
¿¿8860  00           1519   FILLONE: DB        0
¿¿8861  00           1520   FLAG:    DB        0
¿¿8862  0000         1521   SAVE:    DW        0
¿¿8864  0000         1522   SAV:     DW        0
¿¿8866  0000         1523   EREMAIN: DW        0
¿¿8868  00           1524   EWHICHB: DB        0          ;EMPTY WHICH BUFFER
¿¿8869  00           1525   TSW:     DB        0
¿¿886A  4545454545   1526   SEMAPHORE:         DB         'EEEEEEEEEE'        ;TEN BUFFER SEMIPHORES ALL 'E'
¿¿      4545454545
¿¿8874  0000         1527   ECURRBUF: DW       0
¿¿8876  0090         1528   CURRBUF: DW        BUFF0
¿¿8878  0000         1529    TEMPDE: DW      0
¿¿                   1530   AVG:
¿¿887A  F5           1531            PUSH      PSW
¿¿887B  E5           1532            PUSH      H
¿¿887C  C5           1533            PUSH      B
¿¿887D  D5           1534            PUSH      D
¿¿887E  DDE5         1535            PUSHX
¿¿8880  FDE5         1536            PUSHY
¿¿8882  3A5E88       1537            LDA       CNT        ;
¿¿8885  FE00         1538            CPI       0
¿¿8887  C29688       1539            JNZ       AVG1
¿¿                   1540   ;        MVI       A,'1'
¿¿                   1541   ;        CALL      CO
¿¿888A  CDB388       1542            CALL      PART1
¿¿888D  FDE1         1543            POPY
¿¿888F  DDE1         1544            POPX
¿¿8891  D1           1545            POP       D
¿¿8892  C1           1546            POP       B
¿¿8893  E1           1547            POP       H
¿¿8894  F1           1548            POP       PSW
¿¿8895  C9           1549            RET
¿¿8896  FE09         1550   AVG1:    CPI       9
¿¿8898  C2A788       1551            JNZ       AVG2
¿¿889B  CDFA88       1552            CALL      PART3
¿¿889E  FDE1         1553            POPY
¿¿88A0  DDE1         1554            POPX
¿¿88A2  D1           1555            POP       D
¿¿88A3  C1           1556            POP       B
¿¿88A4  E1           1557            POP       H
¿¿88A5  F1           1558            POP       PSW
¿¿88A6  C9           1559            RET
¿¿88A7  CDBC88       1560   AVG2:    CALL      PART2
¿¿88AA  FDE1         1561            POPY
¿¿88AC  DDE1         1562            POPX
¿¿88AE  D1           1563            POP       D
¿¿88AF  C1           1564            POP       B
¿¿88B0  E1           1565            POP       H
¿¿88B1  F1           1566            POP       PSW
¿¿88B2  C9           1567            RET
¿¿                   1568   PART1
¿¿88B3  21A68A       1569            LXI       H,A32
¿¿88B6  013000       1570            LXI       B,48
¿¿88B9  CDF489       1571            CALL      CLEAR
¿¿                   1572   PART2:
¿¿                   1573   ;        MVI       A,'2'
¿¿                   1574   ;        CALL      CO
¿¿                   1575   ;        CALL      CRLF
¿¿88BC  AF           1576            XRA       A
¿¿88BD  32888A       1577            STA       ITER
¿¿                   1578   TP1:
¿¿                   1579   ;        CALL      PORTAOFF
¿¿88C0  DD215788     1580            LXIX      PNTA16
```

```
¿¿88C4  FD21888A    1581            LXIY    ITER
¿¿88C8  CDBF89      1582            CALL    LD2BCDE   ;(BCDE):=A16(ITER)
¿¿                  1583    ;       CALL    WBCDE     ;DUMP BCDE REGISTERS
¿¿                  1584
¿¿88CB  CD4D89      1585            CALL    DSTACK    ;PUT(BCDE) ONTO 32 BIT STACK
¿¿88CE  DD21A68A    1586            LXIX    A32
¿¿88D2  FD21888A    1587            LXIY    ITER
¿¿88D6  CD8B89      1588            CALL    LD4BCDE   ;(BCDE):=A32(ITER)
¿¿                  1589    ;       CALL    WBCDE     ;DUMP BCDE REGISTERS
¿¿88D9  CD4D89      1590            CALL    DSTACK
¿¿88DC  CD7189      1591            CALL    FIXADD
¿¿88DF  CD5F89      1592            CALL    UDSTACK   ;(BCDE):=RESULT OF 32BIT INTEGER ADD
¿¿                  1593    ;       CALL    WBCDE     ;DUMP BCDE REGISTERS
¿¿88E2  DD21A68A    1594            LXIX    A32
¿¿88E6  FD21888A    1595            LXIY    ITER
¿¿88EA  CDA389      1596            CALL    ST4BCDE   ;(A32(ITER)):=(BCDE)
¿¿                  1597    ;       LDA     ITER
¿¿                  1598    ;       CALL    HDMP
¿¿                  1599    ;       CALL    CRLF
¿¿88ED  3A888A      1600            LDA     ITER      ;CHECK MITERATION COUNT
¿¿88F0  3C          1601            INR     A
¿¿88F1  32888A      1602            STA     ITER
¿¿88F4  FE0C        1603            CPI     12
¿¿88F6  C2C088      1604            JNZ     TP1
¿¿                  1605    ;       CALL    PORTAON
¿¿88F9  C9          1606            RET
¿¿                  1607    PART3:
¿¿88FA  AF          1608            XRA     A
¿¿88FB  32888A      1609            STA     ITER
¿¿                  1610    TP2:
¿¿88FE  DD215788    1611            LXIX    PNTA16
¿¿8902  FD21888A    1612            LXIY    ITER
¿¿8906  CDBF89      1613            CALL    LD2BCDE   ;(BCDE):=(A16(ITER))
¿¿8909  CD4D89      1614            CALL    DSTACK    ;(BCDE) ONTO 32 BIT STACK OF T9511
¿¿890C  DD21A68A    1615            LXIX    A32
¿¿8910  FD21888A    1616            LXIY    ITER
¿¿8914  CD8B89      1617            CALL    LD4BCDE   ;(BCDE):=(A32(ITER))
¿¿8917  CD4D89      1618            CALL    DSTACK    ;(BCDE) ONTO 32 BIT STACK OF T9511
¿¿891A  CD7189      1619            CALL    FIXADD    ;ADD 32 BIT INTEGERS TOGETHER
¿¿891D  110A00      1620            LXI     D,10
¿¿8920  010000      1621            LXI     B,0
¿¿8923  CD4D89      1622            CALL    DSTACK
¿¿8926  CD7E89      1623            CALL    FIXDIV    ;DIVIDE BY TEN
¿¿8929  CD5F89      1624            CALL    UDSTACK   ;(BCDE) HAS AVERAGE OF TEN VALUES
¿¿892C  DD215788    1625            LXIX    PNTA16
¿¿8930  FD21888A    1626            LXIY    ITER
¿¿8934  CDD989      1627            CALL    ST2BCDE   ;(BCDE):=(A16(ITER))
¿¿8937  3A888A      1628            LDA     ITER      ;CHECK ITERATION COUNT
¿¿893A  3C          1629            INR     A         ;
¿¿893B  32888A      1630            STA     ITER
¿¿893E  FE0C        1631            CPI     12
¿¿8940  C2FE88      1632            JNZ     TP2
¿¿8943  C9          1633            RET
¿¿                  1634    T95BUSY:
¿¿8944  F5          1635                PUSH    PSW
¿¿8945  DB05        1636    T95BUSY1:       IN      T9511CTRLPORT         ;INPUT THE STATUS WORD
¿¿8947  B7          1637                  ORA     A                 ;SET UP FLAGS
¿¿8948  FA4589      1638                  JM      T95BUSY1            ;BIT 7 SET MEANS T9511 IS BUSY
¿¿894B  F1          1639                  POP     PSW               ;
¿¿894C  C9          1640            RET
¿¿                  1641    DSTACK:
¿¿894D  F5          1642                  PUSH      PSW
¿¿894E  CD4489      1643                  CALL      T95BUSY            ;WAIT UNTIL T9511 NOT BUSY
¿¿8951  7B          1644                  MOV       A,E               ;LSB TO STACK
¿¿8952  D3D4        1645                  OUT       T9511DATAPORT
¿¿8954  7A          1646                  MOV       A,D               ;NEXT BYTE TO STACK
```

```
¿¿8955  D3D4      1647               OUT       T9511DATAPORT
¿¿8957  79        1648               MOV       A,C              ;NEXT SIG BYTE TO STACK OF T9511
¿¿8958  D3D4      1649               OUT       T9511DATAPORT
¿¿895A  78        1650               MOV       A,B              ;MSB BYTE TO STACK OF T9511
¿¿895B  D3D4      1651               OUT       T9511DATAPORT
¿¿895D  F1        1652               POP       PSW
¿¿895E  C9        1653               RET
¿¿                1654      UDSTACK:
¿¿895F  F5        1655               PUSH      PSW
¿¿8960  CD4489    1656               CALL      T95BUSY             ;WAIT UNTIL T9511 NOT BUSY
¿¿8963  DBD4      1657               IN        T9511DATAPORT    ;MSB FROM STACK OF T9511
¿¿8965  47        1658               MOV       B,A
¿¿8966  DBD4      1659               IN        T9511DATAPORT    ;NEXT BYTE FROM STACK OF T9511
¿¿8968  4F        1660               MOV       C,A
¿¿8969  DBD4      1661               IN        T9511DATAPORT    ;NEXT BYTE FROM STACK OF T9511
¿¿896B  57        1662               MOV       D,A
¿¿896C  DBD4      1663               IN        T9511DATAPORT
¿¿896E  5F        1664               MOV       E,A
¿¿896F  F1        1665               POP       PSW
¿¿8970  C9        1666               RET
¿¿                1667      FIXADD:
¿¿8971  F5        1668               PUSH      PSW
¿¿8972  CD4489    1669            CALL      T95BUSY
¿¿8975  3E2C      1670               MVI       A,02CH      ;32 BIT FIX POINT ADD
¿¿8977  D3D5      1671               OUT       0D5H        ;
¿¿8979  CDFC89    1672               CALL      STAT
¿¿897C  F1        1673               POP       PSW
¿¿897D  C9        1674               RET
¿¿                1675      FIXDIV:
¿¿897E  F5        1676               PUSH      PSW
¿¿897F  CD4489    1677            CALL      T95BUSY
¿¿8982  3E2F      1678               MVI       A,02FH      ;32 BIT FIX POINT DIVIDE
¿¿8984  D3D5      1679               OUT       0D5H          ;
¿¿8986  CDFC89    1680               CALL      STAT
¿¿8989  F1        1681               POP       PSW
¿¿898A  C9        1682               RET
¿¿                1683      LD4BCDE:
¿¿                1684      ;
¿¿                1685      ;THIS SUBROUTINE LOADS A 32 BIT ARRAY ELEMENT INTO
¿¿                1686      ;THE REGISTERS BCDE. REGISTER B HAS MOST SIGNIFICANT BYTE,
¿¿                1687      ;REGISTER E HAS THE LEAST SIGNIFICANT BYTE.
¿¿                1688      ;
¿¿                1689      ;THIS SUBROUTINE HAS A CALLING SEQUENCE
¿¿                1690      ;         LXIX      A32
¿¿                1691      ;         LXIY      ITER
¿¿                1692      ;         CALL      LD4BCDE
¿¿898B  F5        1693                PUSH         PSW
¿¿898C  E5        1694                PUSH      H
¿¿                1695
¿¿898D  FD6E00    1696            MOVRY     L,0
¿¿8990  FD6601    1697            MOVRY     H,1         ;(HL):=CONTENTS(ITER)
¿¿                1698
¿¿8993  29        1699            DAD       H
¿¿8994  29        1700            DAD       H           ;(HL):=4*CONTENTS(ITER)
¿¿                1701
¿¿8995  DDE5      1702            PUSHX
¿¿8997  C1        1703            POP       B           ;(BC):=(IX)
¿¿                1704
¿¿8998  09        1705            DAD       B           ;(HL):=A32(ITER)
¿¿                1706
¿¿                1707
¿¿8999  5E        1708               MOV    E,M
¿¿899A  23        1709               INX    H
¿¿899B  56        1710               MOV    D,M
¿¿899C  23        1711               INX    H
¿¿899D  4E        1712               MOV    C,M
```

```
¿¿899E  23                    INX     H
¿¿899F  46                    MOV     B,M       ;(BCDE):=(4 BYTES OF A32(I))
¿¿
¿¿89A0  E1                    POP     H
¿¿89A1  F1                    POP     PSW
¿¿89A2  C9                    RET
¿¿              ST4BCDE:
¿¿              ;
¿¿              ;THIS SUBROUTINE STORES THE REGISTERS BCDE INTO A  32 BIT ARRAY ELEMENT
¿¿              ; REGISTER B HAS MOST SIGNIFICANT BYTE,
¿¿              ;REGISTER E HAS THE LEAST SIGNIFICANT BYTE.
¿¿              ;
¿¿              ;THIS SUBROUTINE HAS A CALLING SEQUENCE
¿¿              ;         LXIX    A32
¿¿              ;         LXIY    ITER
¿¿              ;         CALL    ST4BCDE
¿¿89A3  F5                  PUSH        PSW
¿¿89A4  E5                  PUSH    H
¿¿89A5  C5                PUSH    B
¿¿89A6  D5                PUSH    D
¿¿
¿¿89A7  FD6E00            MOVRY   L,0
¿¿89AA  FD6601            MOVRY   H,1       ;(HL):=CONTENTS(ITER)
¿¿
¿¿89AD  29                DAD     H
¿¿89AE  29                DAD     H         ;(HL):=4*CONTENTS(ITER)
¿¿
¿¿89AF  DDE5              PUSHX
¿¿89B1  C1                POP     B         ;(BC):=(IX)
¿¿
¿¿89B2  09                DAD     B         ;(HL):=A32(ITER)
¿¿
¿¿89B3  D1                POP     D
¿¿89B4  73                MOV     M,E
¿¿89B5  23                INX     H
¿¿89B6  72                MOV     M,D
¿¿89B7  23                INX     H
¿¿
¿¿89B8  C1                POP     B
¿¿89B9  71                MOV     M,C
¿¿89BA  23                INX     H
¿¿89BB  70                MOV     M,B
¿¿
¿¿89BC  E1                POP     H
¿¿89BD  F1                POP     PSW
¿¿89BE  C9                RET
¿¿
¿¿              LD2BCDE:
¿¿              ;
¿¿              ;THIS SUBROUTINE LOADS A 16 BIT ARRAY ELEMENT INTO THE REGISTERS BCDE.
¿¿              ;REGISTER BC HAS 16 BIT ZERO
¿¿              ;REGISTER DE HAS THE 16 BIT INTEGER.
¿¿              ;
¿¿              ;THIS SUBROUTINE HAS A CALLING SEQUENCE
¿¿              ;         LXIX    PNTA16
¿¿              ;         LXIY    ITER
¿¿              ;         CALL    LD2BCDE
¿¿89BF  F5                  PUSH        PSW
¿¿89C0  E5                PUSH    H
¿¿
¿¿89C1  DD6E00            MOVRX   L,0
¿¿89C4  DD6601            MOVRX   H,1       ;(HL):=ADDR(A16)
¿¿
¿¿89C7  EB                XCHG    ;(DE):=ADDR(A16)
¿¿
¿¿89C8  FD6E00            MOVRY   L.0
```

```
¿¿89CB  FD6601          MOVRY   H,1     ;(HL):CONTENTS(ITER)
¿¿
¿¿89CE  29              DAD     H       ;(HL):=2*CONTENTS(ITER)
¿¿
¿¿89CF  19              DAD     D       ;(HL):=ADDR(A16)+2*CONTENTS(ITER)
¿¿
¿¿
¿¿89D0  5E                  MOV     E,M
¿¿89D1  23                  INX     H
¿¿89D2  56                  MOV     D,M      ;(DE) LOADED WITH 16 BIT INTEGER
¿¿                                      ;WHOSE ADDRESS IS IN (HL)
¿¿
¿¿89D3  010000          LXI     B,0     ;(BC) SET TO 0
¿¿
¿¿89D6  E1                  POP     H
¿¿89D7  F1                  POP     PSW
¿¿89D8  C9                  RET
¿¿              ST2BCDE:
¿¿              ;
¿¿              ;THIS SUBROUTINE LOADS THE REGISTERS BCDE INTO A 16 BIT ARRAY ELEMENT.
¿¿              ;REGISTER BC HAS 16 BIT ZERO
¿¿              ;REGISTER DE HAS THE 16 BIT INTEGER.
¿¿              ;
¿¿              ;THIS SUBROUTINE HAS A CALLING SEQUENCE
¿¿              ;       LXIX    PNTA16
¿¿              ;       LXIY    ITER
¿¿              ;       CALL    LD2BCDE
¿¿89D9  F5               PUSH           PSW
¿¿89DA  E5               PUSH     H
¿¿89DB  C5              PUSH    B
¿¿89DC  D5              PUSH    D
¿¿              ;
¿¿
¿¿89DD  DD6E00          MOVRX    L,0
¿¿89E0  DD6601          MOVRX   H,1     ;(HL):=ADDR(A16)
¿¿
¿¿89E3  EB              XCHG            ;(DE):=ADDR(A16)
¿¿
¿¿89E4  FD6E00          MOVRY   L,0
¿¿89E7  FD6601          MOVRY   H,1     ;(HL):=CONTENTS(ITER)
¿¿
¿¿89EA  29              DAD     H       ;(HL):=2*CONTENTS(ITER)
¿¿
¿¿89EB  19              DAD     D       ;(HL):=2*CONTENTS(ITER)+ADDR(A16)
¿¿
¿¿89EC  D1              POP     D       ;RESTORE DE PAIR AS IT WAS ON ENTERING
¿¿
¿¿
¿¿89ED  73                  MOV     M,E     ;
¿¿89EE  23                  INX     H
¿¿89EF  72                  MOV     M,D         ;(DE) STORED AT ADDRESS CONTAINED IN (HL)
¿¿
¿¿89F0  C1              POP     B       ;RESTORE BC
¿¿89F1  E1                  POP     H
¿¿89F2  F1                  POP     PSW
¿¿89F3  C9                  RET
¿¿              CLEAR:
¿¿              ;
¿¿              ;THIS SUBROUTINE HAS A CALLING SEQUENCE
¿¿              ;       LXI     H,A32
¿¿              ;       LXI     B,48
¿¿              ;         CALL    CLEAR
¿¿              ;
¿¿              ;THIS SUBROUTINE CLEARS AN ARRAY OF 32 BIT ELEMENTS.
¿¿              ;HL   IS THE BASE ADDRESS
¿¿              ;BC   IS HOW MANY BYTES TO ZERO
```

```
¿¿                ;
¿¿
¿¿89F4   54                 MOV      D,H
¿¿89F5   5D                 MOV      E,L
¿¿89F6   13                 INX      D              ;(DE):=(HL)+1
¿¿
¿¿89F7   3600               MVI      M,00H          ;ZERO INITIAL LOCATION
¿¿89F9   EDB0               LDIR                    ;NOW ZERO ALL 48 BYTES
¿¿
¿¿89FB   C9                    RET
¿¿                STAT:
¿¿89FC   F5                    PUSH  PSW
¿¿89FD   E5                    PUSH  H
¿¿89FE   DBD5     STAT2:       IN    0D5H
¿¿8A00   B7                 ORA      A
¿¿8A01   FAFE89                JM    STAT2
¿¿8A04   E61E                  ANI   01EH            ;ISOLATE BIT 4,3,2,1
¿¿8A06   CA0E8A                JZ    STAT1
¿¿8A09   3E54               MVI      A,'T'
¿¿8A0B   CD0B88             CALL     CO
¿¿                STAT1:
¿¿8A0E   E1                    POP   H
¿¿8A0F   F1                    POP   PSW
¿¿8A10   C9                    RET
¿¿                ALIGN:
¿¿8A11   F5                    PUSH   PSW
¿¿8A12   3E01                  MVI    A,01H
¿¿8A14   D3D4                  OUT    0D4H
¿¿8A16   3E00                  MVI    A,00H
¿¿8A18   D3D4                  OUT    0D4H
¿¿8A1A   3E1D                  MVI    A,01DH
¿¿8A1C   D3D5                  OUT    0D5H
¿¿8A1E   CD4489                CALL   T95BUSY
¿¿                HH:
¿¿8A21   DBD4                  IN     0D4H
¿¿8A23   FE01                  CPI    01H
¿¿8A25   CA218A                JZ     HH
¿¿8A28   F1                    POP    PSW
¿¿8A29   C9                    RET
¿¿                OK:
¿¿                ;         WRITE MESSAGE OK
¿¿8A2A   D5                 PUSH     D
¿¿8A2B   11F68A             LXI      D,OKAY
¿¿8A2E   CD4A8A             CALL     MSG
¿¿8A31   CD2A87             CALL     CRLF
¿¿8A34   D1                 POP      D
¿¿8A35   C9                 RET
¿¿                DONE:
¿¿                ;         WRITE MESSAGE DONE
¿¿8A36   D5                 PUSH     D
¿¿8A37   11FB8A             LXI      D,DONEE
¿¿8A3A   CD4A8A             CALL     MSG
¿¿8A3D   D1                 POP      D
¿¿8A3E   C9                 RET
¿¿8A3F   3E4F                  MVI    A,'O'
¿¿8A41   CD0B88                CALL   CO
¿¿8A44   3E4B                  MVI    A,'K'
¿¿8A46   CD0B88                CALL   CO
¿¿8A49   C9                    RET
¿¿                ;****************************************************
¿¿                                                  ;
¿¿                ;
¿¿                ;                                 PRINT AMESSAGE ON CONSOLE.
¿¿                ;                                 D-REGISTER POINTS TO BYTE CONTAINING LENGTH
¿¿                ;                                 OF MESSAGE. MESSAGE IS IN THE BYTESFOLLOWING
¿¿                ;                                 LENGTH BYTE.
```

```
¿¿                   ;
¿¿                   ;
¿¿                   ;                         CALLING SEQUENCE
¿¿                   ;
¿¿                   ;                         LXI       D,DONE
¿¿                   ;                         CALL      MSG
¿¿                   ;               DONE:     DB        LGTH,'DONE'
¿¿                   ;               LGTH      EQU       $-(DONE+1)
¿¿                   ;
¿¿                   ;
¿¿                   ;
¿¿                   ;****************************************************************
¿¿8A4A  F5           MSG:      PUSH      PSW       ;SAVE STATUS AND A REG
¿¿8A4B  E5                     PUSH H
¿¿8A4C  D5                     PUSH D
¿¿8A4D  1A                     LDAX      D         ;GET LENGTH OF MESSAGE
¿¿8A4E  67                     MOV       H,A       ;SAVE IT IN H
¿¿8A4F  13                     INX       D         ;POINT TO FIRST BYTE OF MESSAGE
¿¿                                                 ;   TO OUTPUT
¿¿8A50  1A           MSGO:     LDAX      D         ;BYTE OF MESSAGE INTO A
¿¿8A51  CD0B88                 CALL      CO      ;   OUTPUT CHAR
¿¿8A54  13                     INX       D         ;POINT TO NEXT BYTE
¿¿8A55  25                     DCR       H         ;   NUMBER OF CHARS TO BE OUTPUT
¿¿                                                  ;      DIMINISHED BY ONE
¿¿8A56  C2508A                 JNZ       MSGO      ;BRANCH IF THERE ARE
¿¿8A59  D1                     POP       D
¿¿8A5A  E1                     POP       H
¿¿8A5B  F1                     POP       PSW
¿¿8A5C  C9                     RET
¿¿                   WBCDE:
¿¿8A5D  F5                     PUSH      PSW
¿¿8A5E  78                     MOV       A,B
¿¿8A5F  CD0987                 CALL      HDMP
¿¿8A62  79                     MOV       A,C
¿¿8A63  CD0987                 CALL      HDMP
¿¿8A66  7A                     MOV       A,D
¿¿8A67  CD0987                 CALL      HDMP
¿¿8A6A  7B                     MOV       A,E
¿¿8A6B  CD0987                 CALL      HDMP
¿¿8A6E  CD2A87                 CALL      CRLF
¿¿8A71  F1                     POP       PSW
¿¿8A72  C9                     RET
¿¿                   INITPORTA:
¿¿8A73  F5                     PUSH      PSW
¿¿8A74  3E80                   MVI       A,080H
¿¿8A76  D3EB                   OUT       0EBH
¿¿8A78  F1                     POP       PSW
¿¿8A79  C9                     RET
¿¿                   PORTAON:
¿¿8A7A  F5                     PUSH      PSW
¿¿8A7B  3E00                   MVI       A,00H
¿¿8A7D  D3E8                   OUT       0E8H
¿¿8A7F  F1                     POP       PSW
¿¿8A80  C9                     RET
¿¿                   PORTAOFF:
¿¿8A81  F5                     PUSH      PSW
¿¿8A82  3EFF                   MVI       A,0FFH
¿¿8A84  D3E8                   OUT       0E8H
¿¿8A86  F1                     POP       PSW
¿¿8A87  C9                     RET
¿¿8A88  0000         ITER:         DW      0
¿¿8A8A  0000         I:            DW      0
¿¿8A8C  A68A         PNTA32:       DW      A32
¿¿8857               PNTA16:       EQU     SAVEDE
¿¿8A8E  0100         A16:          DW     01H,02H,03H,04H,05H,06H,07H,08H,09H,0AH,0BH,0CH
¿¿8A90  0200
```

```
¿¿8A92  0300
¿¿8A94  0400
¿¿8A96  0500
¿¿8A98  0600
¿¿8A9A  0700
¿¿8A9C  0800
¿¿8A9E  0900
¿¿8AA0  0A00
¿¿8AA2  0B00
¿¿8AA4  0C00
¿¿8AA6            1976   A32:         DS     4*20
¿¿00D5            1977   T9511CTRLPORT EQU  0D5H
¿¿00D4            1978   T9511DATAPORT EQU  0D4H
¿¿8AF6  04        1979   OKAY:    DB         4,'OK',CR,LF
¿¿8AF7  4F4B
¿¿8AF9  0D
¿¿8AFA  0A
¿¿8AFB  06        1980   DONEE:   DB         6,'DONE',CR,LF
¿¿8AFC  444F4E45
¿¿8B00  0D
¿¿8B01  0A
¿¿9000            1981            ORG        9000H
¿¿9000            1982   BUFF0:   DS         2048*10
¿¿                1983            END
¿¿     0 ERRORS
¿
¿R; T=0.53/3.15 09:23:38
¿
```

GENERIC TRAJECTORY (B.2)

```
      COMMON/COM/C(2000)                                                GT
      COMMON/DEGREE/DUMMY2(5)                                           GT
      COMMON/RK/DUMMY4(112)                                             GT
      COMMON/RUNOUT/DUMMY8(1)                                           GT
C                                                                       GT
  235 CONTINUE                                                          GT
      CALL MYINIT                                                       GT
      CALL MYRUN                                                        GT
      GOTO 235                                                          GT
      END                                                               GT
      SUBROUTINE MYINIT                                                 GT
      CALL INPT                                                         GT
      CALL INIT                                                         GT
      CALL DYNAMI                                                       GT
      RETURN                                                            GT
      END                                                               GT
      SUBROUTINE MYRUN                                                  GT
      COMMON/COM/C(2000)                                                GT
      COMMON/RUNOUT/DAN                                                 GT
      COMMON/STNAV/XST(7),YST(7),ZST(7),DIREQ(7),ISTDME,ISTVCR          GT
     1,R1(40),NOYES(40),SIGMA(40)                                       GT
      COMMON/DOUBLE/DSEED,DDS                                           GT
      DOUBLE PRECISION DSEED,DDS                                        GT
      EQUIVALENCE (C(203),TIME),(C(207),NSTEP)                          GT
      EQUIVALENCE (C(211),TIMPR),(C(212),TIMPLT),(C(213),TIMTTY)        GT
      EQUIVALENCE (C(241),J1),(C(244),TIMSOF)                           GT
C                                                                       GT
C*************TEMPORARILY:                                              GT
      DO 259 I=1,40                                                     GT
  259 R1(I)=0.                                                          GT
C                                                                       GT
      DAN=0.                                                            GT
C                                                                       GT
      CALL ERROR                                                        GT
C                                                                       GT
  242 CONTINUE                                                          GT
      IF(TIME.LT.TIMSOF)GOTO 82                                         GT
      DAN=2.                                                            GT
      CALL OUTPT                                                        GT
      CALL FINT                                                         GT
      GOTO 100                                                          GT
C                                                                       GT
   82 CONTINUE                                                          GT
C                                                                       GT
C***NOISE HAS TO BE DRAWN FROM RANDOM GEN FOR EACH CHANN EVERY DT AND   GT
C***BE AVAILABLE FOR SUBROUTINE OUTPT AT WHATEVER REQUIRED RECORDING    GT
C***INSTANTS.                                                           GT
C*****CALL GGNPM(DSEED,30,R1)                                           GT
      DSEED=DSEED+DDS                                                   GT
C                                                                       GT
      CALL LOGIC                                                        GT
C***R.K. LOOP                                                           GT
C                                                                       GT
      DO 246 J=1,4                                                      GT
      J1=J                                                              GT
```

```
      CALL ROTAT
      CALL DYNAM
C
      IF(TIME.LE.0.)CALL OUTPT
      CALL RKG
  246 CONTINUE
C
C
      NSTEP=NSTEP+1
      IF(TIME.LE.TIMPR.AND.TIME.LE.TIMPLT.AND.TIME.LE.TIMTTY)GOTO 242
      CALL OUTPT
      GOTO 242
  100 CONTINUE
      RETURN
      END
      BLOCK      DATA
      COMMON/DEGREE/TET0,Q0(50),PSI0J,PHI0J,WP0J(50),WR0J(50)
      COMMON/PROG/TACCX(50),TACCY(50),TACCZ(50),TPROG(50),
     1WPC(50),WRC(50),WQC(50),NTIME,TSWTCH(15),ISWTCH
      COMMON/COM/C(2000)
      COMMON/ALBET/XBETA,ZBETA,XALPHA,YALPHA
      COMMON/STNAV/XST(7),YST(7),ZST(7),DIREQ(7),ISTDME,ISTVOR
     1,R1(40),NOYES(40),SIGMA(40)
      COMMON/DOUBLE/DSEED,DDS
      DOUBLE PRECISION DSEED,DDS
C
      EQUIVALENCE (C(202),NRATE)
      EQUIVALENCE (C(214),DTPR),(C(215),DTPLT),(C(216),DTTTY)
      EQUIVALENCE (C(220),IPR),(C(244),TIMSOF)
      EQUIVALENCE (C(341),H0),(C(347),XE0),(C(348),YE0)
      EQUIVALENCE (C(277),USPED0),(C(278),VSPED0),(C(279),WSPED0)
C
      DATA H0/116./,TIMSOF/60. /,TET0/10./,PSI0J/0./,PHI0J/12.7/,
     1NTIME/15/,USPED0/125./,VSPED0/0./,WSPED0/12./
     2,TPROG/0.,60.,61.,100.,101.,160.,161.,43*600./,
     6Q0/2*.6,2*0.,2*.6,44*0./,
     7WP0J/2*-.5,2*0.,2*.5,44*0./,
     9WR0J/2*2.86,2*0.,2*-2.86,44*0./,
     BTACCX/2*.179,2*.096,2*.179,2*-.206,42*.2/,
     CTACCY/40*0.,10*0./,
     DTACCZ/2*-1.003,2*-.994,2*-1.003,2*-.985,42*-1./
      DATA NRATE/20/,DTPR/400./,DTPLT/400./,DTTTY/1./,
     1IPR/1/
     2,XBETA/0./,ZBETA/0./,XALPHA/0./,YALPHA/0./
     3,XST/7*120000./
     4,YST/7*120000./
     5,ZST/7*0./
     6,DIREQ/7*32./
     7,ISTDME/1/,ISTVOR/1/
      DATA DSEED/1.D0/,DDS/1.D0/,NOYES/40*0/,SIGMA/40*0./
      DATA ISWTCH/1/,TSWTCH/15*600./
     1,XE0/0./,YE0/0./
      END
      SUBROUTINE INPT
      INTEGER FILE(6),GO
```

```
      LOGICAL SOF
      DIMENSION NAME(8),LNAME(4)
C
      COMMON/COM/C(2000)
      COMMON/DEGREE/TETO,Q0(50),PSI0J,PHI0J,WP0J(50),WR0J(50)
      COMMON/PROG/TACCX(50),TACCY(50),TACCZ(50),TPROG(50),
     1WPC(50),WRC(50),WQC(50),NTIME,TSWTCH(15),ISWTCH
      COMMON/ALBET/XBETA,ZBETA,XALPHA,YALPHA
      COMMON/STNAV/XST(7),YST(7),ZST(7),DIREQ(7),ISTDME,ISTVOR
     1,R1(40),NOYES(40),SIGMA(40)
      COMMON/DOUBLE/DSEED,DDS
      DOUBLE PRECISION DSEED,DDS
C
      EQUIVALENCE (C(202),NRATE)
      EQUIVALENCE (C(214),DTPR),(C(215),DTPLT),(C(216),DTTTY)
      EQUIVALENCE (C(220),IPR),(C(243),IF2),(C(244),TIMSOF)
      EQUIVALENCE (C(341),H0),(C(347),XE0),(C(348),YE0)
      EQUIVALENCE (C(277),USPED0),(C(278),VSPED0),(C(279),WSPED0)
C
      NAMELIST/INP/FILE
      NAMELIST/INCN/H0,TETO,Q0,TIMSOF,PSI0J,PHI0J,WP0J,WR0J,
     1TACCX,TACCY,TACCZ,NTIME,TPROG,USPED0,VSPED0,WSPED0
     2,XBETA,ZBETA,XALPHA,YALPHA
     3,XST,YST,ZST,DIREQ,ISTDME,ISTVOR
     4,TSWTCH,XE0,YE0
      NAMELIST/PARM/NRATE,DTTTY,DTPR,DTPLT,IPR,DSEED,DDS,NOYES,SIGMA
C
      DATA NAME/4HINCN,4HPARM,4H    ,4H    ,4H    ,4H    ,4H    ,4H    /
  235 CONTINUE
      PRINT 502
  502 FORMAT(1H ,'TO   CONTINUE ENTER F,TO      STOP ENTER T')
      READ 503,SOF
  503 FORMAT(L1)
      IF(SOF)STOP
      PRINT 500
  500 FORMAT(1H ,'ENTER DESIRED FILES IN      NAMELIST INP',/)
      READ(5,INP)
      IF1=FILE(1)
      IF2=FILE(2)
      IF3=FILE(3)
      IF4=FILE(4)
      IF5=FILE(5)
      IF6=FILE(6)
      DO 220 I=1,2
      LNAME(I)=NAME(I)
  220 CONTINUE
C
      IF(IF1.EQ.5)PRINT 501,LNAME
  501 FORMAT(1H ,'ENTER      NAMELISTS ',A4,A4,'IN THIS ORDER',/)
      IF(IF3.NE.0)READ(IF1,INCN)
      IF(IF4.NE.0)READ(IF1,PARM)
C
      PRINT 504
  504 FORMAT(1H ,'TO RUN ENTER 1,TO MODIFY INPUT      DATA ENTER 0')
      READ 505,GO
```

```
  505 FORMAT(I1)
      IF(GO.NE.1)GOTO 235
      RETURN
      END
      SUBROUTINE INIT
      DIMENSION IPL(100),IPD(100)
      COMMON/COM/C(2000)
      COMMON/DEGREE/TETO,Q0(50),PSIOJ,PHIOJ,WPOJ(50),WROJ(50)
      COMMON/PROG/TACCX(50),TACCY(50),TACCZ(50),TPROG(50),WPC(50),
     1WRC(50),WQC(50),NTIME,TSWTCH(15),ISWTCH
      COMMON/RK/ARK(4),BRK(4),CRK(4),QRK(100)
C
      EQUIVALENCE (C(201),N)
      EQUIVALENCE (C(202),NRATE),(C(203),TIME),(C(204),TIMED)
      EQUIVALENCE (C(205),DT),(C(207),NSTEP)
      EQUIVALENCE (C(211),TIMPR),(C(212),TIMPLT),(C(213),TIMTTY)
      EQUIVALENCE (C(214),DTPR),(C(215),DTPLT),(C(216),DTTTY)
      EQUIVALENCE (C(217),LINE),(C(218),NPRINT),(C(219),NPLOT)
      EQUIVALENCE (C(222),NSQ2)
      EQUIVALENCE (C(242),CRAD),(C(370),GRAV1),(C(319),THETO)
      EQUIVALENCE (C(318),PHIO),(C(320),PSIO)
C
      GRAV1=32.17
C
      ARK(1)=.5
      ARK(2)=1.-1./SQRT(2.)
      ARK(3)=1.+1./SQRT(2.)
      ARK(4)=1./6.
      BRK(1)=2.
      BRK(2)=1.
      BRK(3)=1.
      BRK(4)=2.
      CRK(1)=ARK(1)
      CRK(2)=ARK(2)
      CRK(3)=ARK(3)
      CRK(4)=ARK(1)
      DO 229 I=1,100
  229 QRK(I)=0.
C
      NSQ2=1
C
      PI=4.*ATAN(1.)
      CRAD=180./PI
      THETO=TETO/CRAD
      PSIO=PSIOJ/CRAD
      PHIO=PHIOJ/CRAD
      DO 248 I=1,NTIME
      WPC(I)=WPOJ(I)/CRAD
      WQC(I)=Q0(I)/CRAD
  248 WRC(I)=WROJ(I)/CRAD
C
      IPL(1)=203
      IPD(1)=204
      N=1
      C(1)=IPL(1)
```

```
      C(101)=IPD(1)                                                     GT
      TIME=0.                                                           GT
      TIMED=1.                                                          GT
      NSTEP=0                                                           GT
      NRAT=NRATE                                                        GT
       DT=1./FLOAT(NRAT)                                                GT
C                                                                       GT
      NPRINT=0                                                          GT
       NPLOT=0                                                          GT
       LINE=60                                                          GT
       TIMPR=DTPR-.5*DT                                                 GT
       TIMPLT=DTPLT-.5*DT                                               GT
      TIMTTY=DTTTY-.5*DT                                                GT
      IF(DTTTY.GT.50.)      TIMTTY=1000.                                GT
C                                                                       GT
      RETURN                                                            GT
      END                                                               GT
      SUBROUTINE ERROR                                                  GT
      PRINT 507                                                         GT
  507 FORMAT(1H ,'RUNNING NOW',/)                                       GT
      RETURN                                                            GT
      END                                                               GT
      SUBROUTINE DYNAMI                                                 GT
      DIMENSION IPL(100),IPD(100)                                       GT
      COMMON/COM/C(2000)                                                GT
      EQUIVALENCE (C(201),N)                                            GT
      EQUIVALENCE (C(205),DT)                                           GT
      EQUIVALENCE (C(311),TET),(C(319),THETO),(C(312),PHI)              GT
      EQUIVALENCE (C(318),PHIO),(C(310),PSI),(C(320),PSIO)              GT
      EQUIVALENCE (C(334),XE),(C(335),YE),(C(336),ZE)                   GT
      EQUIVALENCE (C(341),HO),(C(343),HM),(C(347),XEO),(C(348),YEO)     GT
      EQUIVALENCE (C(271),USPEED),(C(277),USPEDO),(C(272),VSPEED)       GT
      EQUIVALENCE (C(278),VSPEDO),(C(273),WSPEED),(C(279),WSPEDO)       GT
C                                                                       GT
      N=N+1                                                             GT
       IPL(N)=310                                                       GT
       IPD(N)=314                                                       GT
      N=N+1                                                             GT
       IPL(N)=311                                                       GT
       IPD(N)=315                                                       GT
      N=N+1                                                             GT
       IPL(N)=312                                                       GT
       IPD(N)=316                                                       GT
      N=N+1                                                             GT
       IPL(N)=271                                                       GT
       IPD(N)=274                                                       GT
      N=N+1                                                             GT
       IPL(N)=272                                                       GT
       IPD(N)=275                                                       GT
      N=N+1                                                             GT
       IPL(N)=273                                                       GT
       IPD(N)=276                                                       GT
      N=N+1                                                             GT
       IPL(N)=334                                                       GT
       IPD(N)=337                                                       GT
```

```
      N=N+1
       IPL(N)=335
       IPD(N)=338
      N=N+1
       IPL(N)=336
       IPD(N)=339
C
      TET=THETO
      PSI=PSIO
      PHI=PHIO
      XE=XE0
      YE=YE0
      ZE=-H0
      HM=H0
C
      USPEED=USPED0
      VSPEED=VSPED0
      WSPEED=WSPED0
C
      DO 111 I=2,N
      C(I)=IPL(I)
  111 C(100+I)=IPD(I)
C
      RETURN
      END
      SUBROUTINE LOGIC
      COMMON/COM/C(2000)
      COMMON/PROG/TACCX(50),TACCY(50),TACCZ(50),TPROG(50),
     1WPC(50),WRC(50),WQC(50),NTIME,TSWTCH(15),ISWTCH
      EQUIVALENCE (C(302),WQ),(C(303),WR),(C(301),WP)
      EQUIVALENCE (C(531),ACCX),(C(532),ACCY),(C(533),ACCZ)
      EQUIVALENCE (C(203),TIME),(C(311),TET)
      EQUIVALENCE (C(312),PHI),(C(272),VSPEED),(C(205),DT)
      IF(TIME.LT.TSWTCH(ISWTCH))GOTO 11
      IF(ISWTCH.EQ.1)GOTO 12
      IF(ISWTCH.EQ.2)GOTO 13
      PHI=0.
      TET=-.1745
      ISWTCH=ISWTCH+1
      GOTO 11
   13 PHI=12.7*.01745
      TET=.1745
      ISWTCH=ISWTCH+1
      GOTO 11
   12 PHI=0.
      TET=.096
      ISWTCH=ISWTCH+1
   11 CONTINUE
C***FOR THIS N.L. MODEL TRANSITION BETWEEN COMPUTATIONALLY PREDICTED
C***STEADY STATES WILL ANYWAY BE ARTIFICIAL. THUS, THE TSWTCH-OPTION
C***IS DIFFERRED AND ONLY THE FIRST SQUINT-SEGMENT IS USED IN EACH RUN.
C***RESULTS OF ALL RUNS APPEND TO EACH OTHER ON FILE 10('DISP MOD'-
C***OPTION OF FILEDEF-BEFORE THE CONSECUTIVE RUNS OF THE JOB).
C***THE CREATED FILE HAS ALL DATA-VARIABLES OF A GIVEN INSTANT AS A
C***RECORD.IT'S COMPATIBLE WITH APL-LINPLOT(BLANKS BETWEEN VALUES),
```

```
C***AND A SEPARATE PROGRAM CONVERTS IT TO A TIME-VECTOR-RECORD FILE.
      WQ=SQUINT(TIME,TPROG,WQC,NTIME,1)
      WP=SQUINT(TIME,TPROG,WPC,NTIME,1)
      WR=SQUINT(TIME,TPROG,WRC,NTIME,1)
      ACCX=SQUINT(TIME,TPROG,TACCX,NTIME,1)
      ACCY=SQUINT(TIME,TPROG,TACCY,NTIME,1)
      ACCZ=SQUINT(TIME,TPROG,TACCZ,NTIME,1)
      RETURN
      END
      SUBROUTINE DYNAM
      COMMON/COM/C(2000)
C
      EQUIVALENCE (C(302),WQ),(C(311),TET),(C(315),DTET)
      EQUIVALENCE (C(321),VX),(C(323),VZ)
      EQUIVALENCE (C(334),XE),(C(336),ZE),(C(337),XED),(C(339),ZED)
      EQUIVALENCE (C(351),CEB11)
      EQUIVALENCE (C(353),CEB13),(C(357),CEB31),(C(359),CEB33)
      EQUIVALENCE (C(310),PSI),(C(312),PHI),(C(314),DPSI),(C(316),DPHI)
      EQUIVALENCE (C(301),WP),(C(303),WR)
      EQUIVALENCE (C(322),VY),(C(335),YE),(C(338),YED)
      EQUIVALENCE (C(439),CSF),(C(440),SNF),(C(441),CST),(C(442),SNT)
      EQUIVALENCE (C(352),CEB12),(C(354),CEB21),(C(355),CEB22)
      EQUIVALENCE (C(356),CEB23),(C(358),CEB32)
      EQUIVALENCE (C(531),ACCX),(C(532),ACCY),(C(533),ACCZ)
      EQUIVALENCE (C(271),USPEED),(C(272),VSPEED),(C(273),WSPEED)
      EQUIVALENCE (C(274),DUSPED),(C(275),DVSPED),(C(276),DWSPED)
      EQUIVALENCE (C(370),GRAV1)
C
      DPSI=(WR*CSF+WQ*SNF)/CST
      DTET=WQ*CSF-WR*SNF
      DPHI=WP+DPSI*SNT
      CALL DBTOI(USPEED,VSPEED,WSPEED,CEB11,CEB12,CEB13
     1,CEB21,CEB22,CEB23,CEB31,CEB32,CEB33,VX,VY,VZ)
      XED=VX
      ZED=VZ
      YED=VY
      DUSPED=-WQ*WSPEED+WR*VSPEED-GRAV1*SNT     +ACCX*GRAV1
C*****DVSPED=-WR*USPEED+WP*WSPEED+GRAV1*CST*SNF+ACCY*GRAV1-ASSUME COORD.
      DVSPED=0.
      DWSPED=+WQ*USPEED-WP*VSPEED+GRAV1*CST*CSF+ACCZ*GRAV1
C
C
      RETURN
      END
      SUBROUTINE FINISH
      COMMON/COM/C(2000)
      COMMON/REC/A3(2000,20)
C
      EQUIVALENCE (C(203),TIME),(C(217),LINE),(C(218),NPRINT)
      EQUIVALENCE (C(219),NPLOT),(C(220),IPR),(C(243),IF2),(C(242),CRAD)
      EQUIVALENCE (C(212),TIMPLT)
C
      ENTRY FINT
      WRITE(IF2,516)
  516 FORMAT(1H ,'TIME IS OVER',///)
```

```
C
      IF(NPLOT.LT.2)GOTO 518
      DO 121 I=1,NPLOT
  121 WRITE(10,531)(A3(I,J),J=1,16)
  531 FORMAT(1H ,16(F9.2,X))
      ENDFILE 10
      WRITE(IF2,517)NPLOT
  517 FORMAT(1H ,I15)
  518 CONTINUE
      RETURN
      END
      SUBROUTINE OUTPT
      COMMON/COM/C(2000)
      COMMON/RUNOUT/DAN
      COMMON/REC/A3(2000,20)
      DIMENSION RDME(7),VOR(7)
      COMMON/ALBET/XBETA,ZBETA,XALPHA,YALPHA
      COMMON/STNAV/XST(7),YST(7),ZST(7),DIREQ(7),ISTDME,ISTVOR
     1,R1(40),NOYES(40),SIGMA(40)
      EQUIVALENCE (C(336),ZE)
      EQUIVALENCE (C(203),TIME),(C(211),TIMPR),(C(212),TIMPLT)
      EQUIVALENCE (C(213),TIMTTY),(C(214),DTPR),(C(215),DTPLT)
      EQUIVALENCE (C(216),DTTTY),(C(217),LINE),(C(218),NPRINT)
      EQUIVALENCE (C(219),NPLOT),(C(242),CRAD),(C(243),IF2)
      EQUIVALENCE (C(302),WQ),(C(321),VX),(C(323),VZ),(C(334),XE)
      EQUIVALENCE (C(343),HM),(C(311),TET)
      EQUIVALENCE (C(310),PSI),(C(312),PHI)
      EQUIVALENCE (C(301),WP),(C(303),WR),(C(322),VY),(C(335),YE)
      EQUIVALENCE (C(531),ACCX),(C(532),ACCY),(C(533),ACCZ)
      EQUIVALENCE (C(271),USPEED),(C(272),VSPEED),(C(273),WSPEED)
C
C
      WPJ=CRAD*WP+NOYES(1)*SIGMA(1)*R1(1)
      WQJ=CRAD*WQ+NOYES(2)*SIGMA(2)*R1(2)
      WRJ=CRAD*WR+NOYES(3)*SIGMA(3)*R1(3)
      PSIJ=CRAD*PSI+NOYES(4)*SIGMA(4)*R1(4)
      THETJ=CRAD*TET+NOYES(5)*SIGMA(5)*R1(5)
      PHIJ=CRAD*PHI+NOYES(6)*SIGMA(6)*R1(6)
      VAIR=SQRT(USPEED**2+VSPEED**2+WSPEED**2)+NOYES(7)*SIGMA(7)*R1(7)
      VRP=VSPEED+WR*XBETA-WP*ZBETA
      USPD=USPEED
      BETA=ATAN2(VRP,USPD)
      BETAJ=CRAD*BETA+NOYES(8)*SIGMA(8)*R1(8)
      WQR=WSPEED-WQ*XALPHA+WP*YALPHA
      ALPHA=ATAN2(WQR,USPD)
      ALPHAJ=CRAD*ALPHA+NOYES(9)*SIGMA(9)*R1(9)
      HM=-ZE+NOYES(10)*SIGMA(10)*R1(10)
      DO 301 I=1,ISTDME
  301 RDME(I)=SQRT((XE-XST(I))**2+(YE-YST(I))**2+(ZE-ZST(I))**2)+
     1NOYES(10+I)*SIGMA(10+I)*R1(10+I)
      DO 302 I=1,ISTVOR
      YVOR=YE-YST(I)
      XVOR=XE-XST(I)
      REQ=DIREQ(I)/CRAD
      RUNITX=COS(REQ)
```

```
      RUNITY=SIN(REQ)                                                   GT
      VCOS=ABS((XVOR*RUNITX+YVOR*RUNITY)/SQRT(XVOR**2+YVOR**2))         GT
      RVOR1=ARCOS(VCOS)                                                 GT
      RVOR=CRAD*RVOR1+NOYES(10+ISTDME+I)*SIGMA(10+ISTDME+I)*            GT
     1R1(10+ISTDME+I)                                                   GT
  302 VOR(I)=SATF(RVOR,10.)                                             GT
      VXY=SQRT(VX*VX+VY*VY)                                             GT
      VZN=-VZ                                                           GT
      GAMV=ATAN2(VZN,VXY)                                               GT
      GAMVJ=CRAD*GAMV                                                   GT
      GAMH=ATAN2(VY,VX)                                                 GT
      GAMHJ=CRAD*GAMH                                                   GT
      ACCX=ACCX+NOYES(10+ISTDME+ISTVOR+1)*SIGMA(10+ISTDME+ISTVOR+1)*    GT
     1R1(10+ISTDME+ISTVOR+1)                                            GT
      ACCY=ACCY+NOYES(10+ISTDME+ISTVOR+2)*SIGMA(10+ISTDME+ISTVOR+2)*    GT
     1R1(10+ISTDME+ISTVOR+2)                                            GT
      ACCZ=ACCZ+NOYES(10+ISTDME+ISTVOR+3)*SIGMA(10+ISTDME+ISTVOR+3)*    GT
     1R1(10+ISTDME+ISTVOR+3)                                            GT
C***NOW,IF TRIGGERED-CONTAMINATION BY NOISE;IF NOT-BYPASSED.            GT
C                                                                       GT
C***OUTPUT OPTIONS:                                                     GT
C***  1.SHORT(TERMINAL) PRINTOUT;                                       GT
C***  2.LONG PRINTOUT OF FIRST BUNCH,SECOND OR BOTH;                    GT
C***  3.CREATION OF 1SEC-INTERVAL-FILE OF DATA(SAME AS FLIGHT FILE)-    GT
C***    -TO BE PRINTED,PLOTTED OR FURTHER PROCESSED.                    GT
C                                                                       GT
      IF(TIME.GT.0..AND.DAN.LT.1.)GOTO 879                              GT
      PRINT 508                                                         GT
      WRITE(IF2,511)TIME,VAIR,GAMVJ,GAMHJ,HM,XE,YE,PSIJ,THETJ,          GT
     1PHIJ,VOR(1),RDME(1),ALPHAJ,BETAJ                                  GT
  879 CONTINUE                                                          GT
      IF(TIME.LE.TIMPR)GOTO 241                                         GT
      IF(LINE.NE.60)GOTO 259                                            GT
      WRITE(IF2,508)                                                    GT
  508 FORMAT(1H1,' TIME     VAIR     GAMVJ     GAMEJ      HM        XE  GT
     1YE     PSIJ     THETJ     PHIJ      VOR(1)     RDME(1)   ALPHAJ   BETAJ')GT
       NPRINT=NPRINT+1                                                  GT
       LINE=1                                                           GT
  259 CONTINUE                                                          GT
      WRITE(IF2,511)TIME,VAIR, GAMVJ,GAMHJ,HM,XE,YE,PSIJ,THETJ,PHIJ,    GT
     1VOR(1),RDME(1),ALPHAJ,BETAJ                                       GT
  511 FORMAT(1H ,5F7.1,2F10.1,4F7.1,F11.1,2F7.1)                        GT
       NPRINT=NPRINT+1                                                  GT
      TIMPR=TIMPR+ITPR                                                  GT
       LINE=LINE+1                                                      GT
  241   IF(TIME.LE.TIMPLT.OR.NPLOT.GE.2000)GOTO 247                     GT
       NPLOT=NPLOT+1                                                    GT
      A3(NPLOT,1)=TIME                                                  GT
      A3(NPLOT,2)=WPJ                                                   GT
      A3(NPLOT,3)=WQJ                                                   GT
      A3(NPLOT,4)=WRJ                                                   GT
      A3(NPLOT,5)=ACCX                                                  GT
      A3(NPLOT,6)=ACCY                                                  GT
      A3(NPLOT,7)=ACCZ                                                  GT
      A3(NPLOT,8)=HM                                                    GT
```

```
      A3(NPLOT,9)=VAIR
      A3(NPLOT,10)=PSIJ
      A3(NPLOT,11)=THETJ
      A3(NPLOT,12)=PHIJ
      A3(NPLOT,13)=VOR(1)
      A3(NPLOT,14)=RDME(1)
      A3(NPLOT,15)=ALPHAJ
      A3(NPLOT,16)=BETAJ
       TIMPLT=TIMPLT+DTPLT
  247    IF(TIME.LE.TIMTTY)GOTO 240
      IF(TIME.LT.(1.5*DTTTY))PRINT 510
  510 FORMAT(1H1,' TIME        VAIR      GAMVJ      GAMHJ         XE         YE
     1    HM')
      PRINT 513,TIME,VAIR,GAMVJ,GAMHJ,XE,YE,HM
  513 FORMAT(1H ,7F10.1)
       TIMTTY=TIMTTY+DTTTY
  240 CONTINUE
C
      RETURN
      END
      FUNCTION SATF(X11,XM11)
      SATF=SIGN(AMIN1(ABS(X11),XM11),X11)
      RETURN
      END
      SUBROUTINE RCTAT
      COMMON/COM/C(2000)
      EQUIVALENCE (C(311),TET),(C(351),CEB11),(C(353),CEB13)
      EQUIVALENCE (C(357),CEB31),(C(359),CEB33)
      EQUIVALENCE (C(441),CST),(C(442),SNT)
      EQUIVALENCE (C(310),PSI),(C(312),PHI),(C(352),CEB12)
      EQUIVALENCE (C(354),CEB21),(C(355),CEB22),(C(356),CEB23)
      EQUIVALENCE (C(358),CEB32),(C(443),CSP),(C(444),SNP)
      EQUIVALENCE (C(439),CSF),(C(440),SNF)
C
      SNT=SIN(TET)
      CST=COS(TET)
      SNP=SIN(PSI)
      CSP=COS(PSI)
      SNF=SIN(PHI)
      CSF=COS(PHI)
C
      CEB11=CST*CSP
      CEB13=-SNT
      CEB31=+SNT*CSF*CSP+SNF*SNP
      CEB33=CST*CSF
      CEB12=CST*SNP
      CEB21=SNF*SNT*CSP-CSF*SNP
      CEB22=SNF*SNT*SNP+CSF*CSP
      CEB23=SNF*CST
      CEB32=CSF*SNT*SNP-SNF*CSP
      RETURN
      END
      SUBROUTINE DITOB(XI,YI,ZI,A11,A12,A13,A21,A22,A23,A31,A32,A33,
     1XO,YO,ZO)
      XO=A11*XI+A13*ZI+A12*YI
```

```
      ZO=A31*XI+A33*ZI+A32*YI                                           GT
      YO=A21*XI+A22*YI+A23*ZI                                           GT
      RETURN                                                            GT
      ENTRY DBTCI(XI,YI,ZI,A11,A12,A13,A21,A22,A23,A31,A32,A33,         GT
     1XO,YO,ZO)                                                         GT
      XO=A11*XI+A31*ZI+A21*YI                                           GT
      ZO=A13*XI+A33*ZI+A23*YI                                           GT
      YO=A12*XI+A22*YI+A32*ZI                                           GT
      RETURN                                                            GT
      END                                                               GT
      SUBROUTINE RKG                                                    GT
      DIMENSION IPL(100),IPD(100)                                       GT
      COMMON/COM/C(2000)                                                GT
      COMMON/RK/ARK(4),BRK(4),CRK(4),QRK(100)                           GT
C                                                                       GT
      EQUIVALENCE  (C(201),N)                                           GT
      EQUIVALENCE  (C(205),H),(C(241),J)                                GT
C                                                                       GT
      DO 100 I=1,N                                                      GT
      IL=C(I)                                                           GT
      ID=C(100+I)                                                       GT
      X1=C(ID)*H                                                        GT
      X2=(X1-BRK(J)*QRK(I))*ARK(J)                                      GT
      C(IL)=C(IL)+X2                                                    GT
100   QRK(I)=QRK(I)+3.*X2-CRK(J)*X1                                     GT
C                                                                       GT
      RETURN                                                            GT
      END                                                               GT
      FUNCTION SQUINT(X,TABX,TABY,NTAB,N)                               GT
      DIMENSION TABX(NTAB),TABY(NTAB)                                   GT
      IF(NTAB.NE.1)GOTO 1                                               GT
      SQUINT=TABY(NTAB)                                                 GT
      RETURN                                                            GT
    1 IF(X-TABX(N))2,3,4                                                GT
    2  N=N-1                                                            GT
      IF(N)9,9,1                                                        GT
    3  SQUINT=TABY(N)                                                   GT
      FACTOR=0.                                                         GT
      RETURN                                                            GT
    4  IF((N+1).GT.NTAB)GOTO 10                                         GT
      IF(X-TABX(N+1))5,6,7                                              GT
    6  N=N+1                                                            GT
      GOTO 3                                                            GT
    7  N=N+1                                                            GT
      GOTO 4                                                            GT
    5  FACTOR=(X-TABX(N))/(TABX(N+1)-TABX(N))                           GT
        SQUINT=(TABY(N+1)-TABY(N))*FACTOR+TABY(N)                       GT
      RETURN                                                            GT
    9  PRINT 1000,X,TABX(1)                                             GT
      Y=SQRT(-1.)                                                       GT
      STOP                                                              GT
 1000 FORMAT(1H ,10('$'),'SQUINT UNDERFLOW - INPUT =',E15.8 ,'  LESS THAGT
     1N  FIRST  ARG. TABLE     ENTRY( ',E15.8 ,' ) ' )                  GT
   10   PRINT 2000,X,TABX(NTAB)                                         GT
      Y=SQRT(-1.)                                                       GT
```

```
      STOP                                                              GTR
 2000 FORMAT(1H ,10('$'), 'SQUINT OVERFLOW -INPUT =',E15.8 ,'  GREATER  GTR
     1 THAN  LAST  ARG. TABLE      ENTRY (',E15.8 ,' ) ' )              GTR
      END                                                               GTR
```

OPTIMAL FLIGHT PATH RECONSTRUCTION (B.3)

```
FILE: OPTFILT  FORTRAN  A             PRINCETON UNIVERSITY TIME-SHARING


C*
C*OPTIMAL FILTERING PROGRAM
C*
      IMPLICIT REAL*8(A-H,O-$)
      REAL*4 PREDCT(6)
      DIMENSION SUM(6),H112(11,6),H111(6,6)
      COMMON/DATO/S2LIAC,S2XYZR,TIMSOF,X00(11),S2WIND,S2WNDZ,S2WNDY
     1,S2V,S2ALF,S2BET,S2H,S2DME,S2VOR,S2BDME,S2ZR,PERCNT,OMEGA,AGLOBE
     4,OUTL1,OUTL2,S2QA,S2QB
     2,EPS2,TLAM0,SMEW0,XST,YST,ZST,DIREQ,KDME(20),NDME1(20),NDME2(20)
     3,KDME2(20) ,NXP,NRATE,NSTEP,IDGT,ISTDME,ISTVOR,SOF
      LOGICAL GO,SOF
      NAMELIST/INP/FILE
      INTEGER FILE(2)
      NAMELIST/OK/GO
      NAMELIST/DAT/SOF,NXP,S2LIAC,S2XYZR,NRATE,TIMSOF,X00,NSTEP,IDGT
     1,S2V,S2ALF,S2BET,S2H,S2DME,S2VOR,PERCNT,OMEGA,AGLOBE
     4,OUTL1,OUTL2,S2QA,S2QB
     2,EPS2,ISTDME,ISTVOR,XST,YST,ZST,DIREQ,KDME,NDME1,NDME2,S2WIND
     3,TLAM0,SMEW0,S2WNDZ,S2BDME,S2ZR,S2WNDY,KDME2
      DIMENSION P(11,11),P0(11,11),HB(6,11),X0(11),XOUT(11)
     1,RES(6),RK(6,6),HBT(11,6),PHT(11,6),HPHT(6,6),HPHTR(6,6)
     2,HPHTRI(6,6),WKAREA(110),CKALM(11,6),DELX(11,1),RES1(6,1)
     3,XPLUS(11),AIDEN(11,11),CH(11,11),UNMKH(11,11),PPLUS(11,11)
     4,CKALMT(6,11),UNMKHT(11,11),STABK1(11,11),STABK2(11,11)
     5,STABK3(6,11),STABK4(11,11)
      DIMENSION XST(7),YST(7),ZST(7),DIREQ(7)
     1,XST7(7),YST7(7),ZST7(7),DIREQ7(7)
C***
      DEFINE FILE 12(60,2112,L,KSTEPA)
C***
    1 CONTINUE
      PRINT 1002
 1002 FORMAT(1H ,'TYPE &INP FILE= &END')
      READ(5,INP)
      INF1=FILE(1)
      INF2=FILE(2)
      PRINT 2000,FILE
 2000 FORMAT(2I10)
      PRINT 1004
 1004 FORMAT(1H ,'TYPE &DAT DATA= &END')
      READ(INF1,DAT)
      IF(SOF)GOTO 92
      PRINT 1003
 1003 FORMAT(1H ,'IF EVERYTHING O.K. TYPE &OK GO=.TRUE.&END')
      GO=.TRUE.
      READ(5,OK)
      IF(GO)GOTO 10
      GOTO 1
   10 CONTINUE
C***
      DO 331 I=1,6
  331 SUM(I)=0.0D0
C***
      PI=4.0D0*DATAN(1.0D0)
```

```
      CRAD=180.0D0/PI                                                   OP
      DO 17 I=1,NXP                                                     OP
   17 X0(I)=X00(I)                                                      OP
      KSTEP=0                                                           OP
      KSTEP1=0                                                          OP
      DO 101 I=1,NXP                                                    OP
      DO 102 J=1,NXP                                                    OP
      AIDEN(I,J)=0.0D0                                                  OP
  102 P0(I,J)=0.0D0                                                     OP
  101 CONTINUE                                                          OP
C******                                                                 OP
      DO 109 I=1,NXP                                                    OP
  109 AIDEN(I,I)=1.0D0                                                  OP
      DO 104 I=1,6                                                      OP
      DO 105 J=1,6                                                      OP
  105 RK(I,J)=0.0D0                                                     OP
  104 CONTINUE                                                          OP
      RK(1,1)=S2V                                                       OP
      RK(2,2)=S2ALF                                                     OP
      RK(3,3)=S2BET                                                     OP
      RK(4,4)=S2H                                                       OP
      RK(5,5)=S2DME                                                     OP
      RK(6,6)=S2DME                                                     OP
C***                                                                    OP
      DO 461 I=1,7                                                      OP
      XST7(I)=XST(I)                                                    OP
      YST7(I)=YST(I)                                                    OP
      ZST7(I)=ZST(I)                                                    OP
  461 DIREQ7(I)=DIREQ(I)                                                OP
      ISTVO7=ISTVOR                                                     OP
      ISTDM7=ISTDME                                                     OP
      NRAT7=NRATE                                                       OP
      TIMS7=TIMSOF                                                      OP
      NXP7=NXP                                                          OP
      S2LI7=S2LIAC                                                      OP
C***                                                                    OP
      IDME2=1                                                           OP
      IDME=1                                                            OP
  301 CONTINUE                                                          OP
      KSTEP=KSTEP+1                                                     OP
      KSTEP1=KSTEP1+1                                                   OP
C*                                                                      OP
      IFLAG=1                                                           OP
C***                                                                    OP
      IF(KDME(IDME).GT.KSTEP)GOTO 807                                   OP
C*                                                                      OP
      DO 817 I=1,NXP7                                                   OP
      DO 817 J=1,NXP7                                                   OP
  817 IF(I.NE.J)P0(I,J)=0.0D0                                           OP
      IFLAG=5                                                           OP
C*                                                                      OP
      XST7(1)=XST(NDME1(IDME))                                          OP
      YST7(1)=YST(NDME1(IDME))                                          OP
      ZST7(1)=ZST(NDME1(IDME))                                          OP
      IDME=IDME+1                                                       OP
```

```
  807 CONTINUE
C*
      IF(KDME2(IDME2).GT.KSTEP)GOTO 827
      DO 828 I=1,NXP7
      DO 828 J=1,NXP7
  828 IF(I.NE.J) P0(I,J)=0.0D0
      IFLAG=5
      XST7(2)=XST(NDME2(IDME2))
      YST7(2)=YST(NDME2(IDME2))
      ZST7(2)=ZST(NDME2(IDME2))
      IDME2=IDME2+1
  827 CONTINUE
C*
C*CALLING THE PROPAGATION-BETWEEN-MEASUREMENTS SUBROUTINE
C***
      CALL PROP27(TIMS7,PERCNT,OMEGA,AGLOBE,EPS2,S2WNDY,TLAM0
     1,SMEW0,S2WIND,S2WNDZ,X0,P0,XST7,YST7,ZST7,DIREQ7,S2ZR,S2BDME
     3,OUTL1,OUTL2,S2QA,S2QB
     2,S2LI7,S2XYZR,XOUT,P,HB,RES,IFLAG,NRAT7,NXP7,ISTDM7,ISTVO7,KSTEP)
      DO 201 I=1,6
      DO 202 J=1,NXP
  202 HBT(J,I)=HB(I,J)
  201 CONTINUE
      CALL VMULFF(P,HBT,NXP,NXP,6,NXP,NXP,PHT,NXP,IER1)
      CALL VMULFF(HB,PHT,6,NXP,6,6,NXP,HPHT,6,IER2)
      DO 211 I=1,6
      DO 212 J=1,6
  212 HPHTR(I,J)=HPHT(I,J)+RK(I,J)
  211 CONTINUE
C***
      DO 341 I=1,6
  341 SUM(I)=SUM(I)+HPHTR(I,I)
C***
      CALL LINV1F(HPHTR,6,6,HPHTRI,IDGT,WKAREA,IER3)
      CALL VMULFF(PHT,HPHTRI,NXP,6,6,NXP,6,CKALM,NXP,IER4)
      DO 221 I=1,6
  221 RES1(I,1)=RES(I)
      CALL VMULFF(CKALM,RES1,NXP,6,1,NXP,6,DELX,NXP,IER5)
      DO 231 I=1,NXP
  231 XPLUS(I)=XOUT(I)+DELX(I,1)
      CALL VMULFF(CKALM,HB,NXP,6,NXP,NXP,6,CH,NXP,IER6)
      DO 241 I=1,NXP
      DO 242 J=1,NXP
  242 UNMKH(I,J)=AIDEN(I,J)-CH(I,J)
  241 CONTINUE
C-OLD CALL VMULFF(UNMKH,P,NXP,NXP,NXP,NXP,NXP,PPLUS,NXP,IER7)
      DO 281 I=1,NXP
      DO 281 J=1,NXP
  281 UNMKHT(J,I)=UNMKH(I,J)
      DO 283 I=1,NXP
      DO 283 J=1,6
  283 CKALMT(J,I)=CKALM(I,J)
      CALL VMULFF(P,UNMKHT,NXP,NXP,NXP,NXP,NXP,STABK1,NXP,IER7)
      CALL VMULFF(UNMKH,STABK1,NXP,NXP,NXP,NXP,NXP,STABK2,NXP,IER8)
      CALL VMULFF(RK,CKALMT,6,6,NXP,6,6,STABK3,6,IER9)
```

```
      CALL VMULFF(CKALM,STABK3,NXP,6,NXP,NXP,6,STABK4,NXP,IER10)        OP
      DO 282 I=1,NXP                                                    OP
      DO 282 J=1,NXP                                                    OP
  282 PPLUS(I,J)=STABK2(I,J)+STABK4(I,J)                                OP
C***                                                                    OP
C***                                                                    OP
      IF(KSTEP1.LT.5)GOTO 253                                           OP
      KSTEP1=0                                                          OP
      DO 251 I=1,NXP                                                    OP
      DO 252 J=1,NXP                                                    OP
      IF(I.LE.J)GOTO 252                                                OP
      PPLUS(I,J)=.5D0*(PPLUS(I,J)+PPLUS(J,I))                           OP
      PPLUS(J,I)=PPLUS(I,J)                                             OP
  252 CONTINUE                                                          OP
  251 CONTINUE                                                          OP
  253 CONTINUE                                                          OP
C******                                                                 OP
      DO 741 I=1,NXP                                                    OP
  741 X0(I)=XPLUS(I)                                                    OP
      DO 742 I=1,NXP                                                    OP
      DO 742 J=1,NXP                                                    OP
  742 P0(I,J)=PPLUS(I,J)                                                OP
C***                                                                    OP
      KSTEPA=KSTEP                                                      OP
C***                                                                    OP
      WRITE(12'KSTEPA)(XPLUS(I),I=1,NXP)                                OP
     1,(XOUT(I),I=1,NXP),((PPLUS(I,J),I=1,NXP),J=1,NXP)                 OP
     2,((P(I,J),I=1,NXP),J=1,NXP)                                       OP
C***                                                                    OP
      IF(KSTEP.EQ.NSTEP)GOTO 311                                        OP
C***                                                                    OP
      GOTO 301                                                          OP
C***                                                                    OP
  311 CONTINUE                                                          OP
C***                                                                    OP
      IF(NSTEP.GT.60)GOTO 1                                             OP
      DO 342 I=1,6                                                      OP
  342 PREDCT(I)=DSQRT(SUM(I)/59)                                        OP
      PREDCT(2)=57.3*PREDCT(2)                                          OP
      PREDCT(3)=57.3*PREDCT(3)                                          OP
      PRINT 343,(PREDCT(I),I=1,6)                                       OP
  343 FORMAT(' ',F12.0,2F12.1,3F12.0)                                   OP
C***                                                                    OP
      GOTO 1                                                            OP
   92 CONTINUE                                                          OP
      STOP                                                              OP
      END                                                               OP
      BLOCK DATA                                                        OP
      IMPLICIT REAL*8(A-H,O-$)                                          OP
      COMMON/DAT0/S2LIAC,S2XYZR,TIMSOF,X00(11),S2WIND,S2WNDZ,S2WNDY     OP
     1,S2V,S2ALF,S2BET,S2H,S2DME,S2VOR,S2BDME,S2ZR,PERCNT,OMEGA,AGLOBE  OP
     4,OUTL1,OUTL2,S2QA,S2QB                                            OP
     2,EPS2,TLAM0,SMEW0,XST,YST,ZST,DIREQ,KDME(20),NDME1(20),NDME2(20)  OP
     3,KDME2(20),NXP,NRATE,NSTEP,IDGT,ISTDME,ISTVOR,SOF                 OP
      DIMENSION XST(7),YST(7),ZST(7),DIREQ(7)                           OP
```

```
      LOGICAL SOF                                                       C
      DATA SOF/ .FALSE. /,NXP/11/,S2LIAC/.4D0/,NRATE/20/,TIMSOF/1.0D0/  C
     1,X00/0.0D0,0.0D0,-116.0D0,125.0D0,0.0D0,12.0D0,5*0.0D0/           C
     2,S2V/6.25D0/,S2ALF/.000076D0/,S2WIND/.10D0/,S2WNDZ/.000001D0/     C
     3,S2BET/.000076D0/,S2H/25.0D0/,S2DME/40000.0D0/,S2VOR/.000004D0/   C
     4,NSTEP/60/,IDGT/3/,S2XYZR/25.0D0/,S2ZR/.10D0/                     C
     6,ISTDME/2/,ISTVOR/2/,S2BDME/.000025D0/,S2WNDY/.10D0/              C
     7,XST/.703D0,.6985D0,5*.706D0/                                     C
     8,YST/-1.297D0,-1.2985D0,5*-1.294D0/                               C
     9,ZST/7*0.0D0/                                                     C
     A,DIREQ/32.0D0,32.0D0,0.0D0,0.0D0,0.0D0,0.0D0,0.0D0/               C
     B,KDME/1,19*2500/,NDME1/1,2,3,1,4,15*2/,NDME2/2,3,1,4,16*3/        C
     C,OMEGA/.0000728D0/,AGLOBE/20940000.0D0/,EPS2/.0067D0/             C
     D,TLAM0/.700D0/,SMEW0/-1.300D0/                                    C
      DATA PERCNT/.000001D0/                                            C
     1,KDME2/1,19*2600/                                                 C
     2,OUTL1/20000.0D0/,OUTL2/20000.0D0/,S2QA/.81D0/,S2QB/.81D0/        C
      END                                                               C
```

```
C*
C***STATE AND COVARIANCE MATRIX PROPAGATION BETWEEN MEASUREMENTS.
C*
      SUBROUTINE PROP27(TIMSOF,PERC1,OMEG1,AGLOB1,EPS1,S2W0Y
     1,TLAM00,SMEW00,S2W0,S2WZ,X0,P0,XST,YST,ZST,DIREQ,S2ZR1,S2BDM
     3,OUTL1T,OUTL2T,S2QAT,S2QBT
     2,S2LIAC,S2XYZR,XOUT,P,HB,RES,IFLAG7,NRATE,NXP,ISTDME,ISTVOR,KSTEP)
      IMPLICIT REAL*8 (A-H,O-$)
      EQUIVALENCE (IC(202),NRATE1),(C(311),TIMSF)
      DIMENSION P0(11,11),X0(11),P01(11,11),P1(11,11),HB1(8,11),X01(11)
      EQUIVALENCE (C(467),S2LIA),(C(373),S2XYZ)
      EQUIVALENCE (IC(251),NXP1),(C(551),X01(1)),(C(593),P01(1,1))
      EQUIVALENCE (C(993),HB1(1,1)),(C(793),P1(1,1))
      DIMENSION HB(6,11),XOUT(11),P(11,11),RES(6),Z(11),ZCAL(8)
      EQUIVALENCE (C(469),XE),(C(471),YE),(C(473),ZE)
      EQUIVALENCE (C(481),USPEED),(C(483),VSPEED),(C(485),WSPEED)
      EQUIVALENCE (C(571),Z(1)),(C(1235),ZCAL(1))
      DIMENSION XST(7),YST(7),ZST(7),DIREQ(7),XST1(7),YST1(7),ZST1(7)
     1,DIREQ1(7)
      EQUIVALENCE (C(1257),XST1(1)),(C(1271),YST1(1))
      EQUIVALENCE (C(1299),DIREQ1(1)),(C(1285),ZST1(1))
      EQUIVALENCE (IC(252),ISTDM1),(IC(253),ISTVO1)
      EQUIVALENCE (IC(259),KSTEP1)
      EQUIVALENCE (C(505),OMEGA),(C(507),AGLOBE),(C(509),EPS2)
      EQUIVALENCE (C(511),TLAM0),(C(513),SMEW0)
      EQUIVALENCE (C(470),S2WNDZ)
C***
      EQUIVALENCE (C(482),WX),(C(484),WY),(C(486),WZ)
      EQUIVALENCE (C(468),S2WIND)
      COMMON/COM/C(2000)
      COMMON/INCOM/IC(500)
C***
      EQUIVALENCE (C(312),PERCNT),(C(381),S2ZR),(C(382),S2BDME)
     1,(C(494),BDME1),(C(498),BDME2),(C(474),S2WNDY)
     2,(IC(261),IFLAG),(C(502),OUTL1),(C(504),OUTL2)
     3,(C(506),S2QA),(C(508),S2QB)
      OUTL1=OUTL1T
      OUTL2=OUTL2T
      S2QA=S2QAT
      S2QB=S2QBT
      IFLAG=IFLAG7
      S2WNDY=S2W0Y
      S2ZR=S2ZR1
      S2BDME=S2BDM
      PERCNT=PERC1
C***
      OMEGA=OMEG1
      AGLOBE=AGLOB1
      EPS2=EPS1
      TLAM0=TLAM00
      SMEW0=SMEW00
      KSTEP1=KSTEP
      NRATE1=NRATE
      TIMSF=TIMSOF
      S2LIA=S2LIAC
```

```
      S2XYZ=S2XYZR
      S2WIND=S2W0
      S2WNDZ=S2WZ
      NXP1=NXP
      DO 61 I=1,NXP1
   61 X01(I)=X0(I)
      DO 62 I=1,NXP1
      DO 62 J=1,NXP1
      P01(I,J)=P0(I,J)
   62 CONTINUE
      DO 64 I=1,7
      XST1(I)=XST(I)
      YST1(I)=YST(I)
      ZST1(I)=ZST(I)
   64 DIREQ1(I)=DIREQ(I)
      ISTDM1=ISTDME
      ISTVO1=ISTVOR
C
      CALL MYINIT
      CALL MYRUN
C***
      DO 101 I=1,NXP1
      DO 102 J=1,NXP1
      IF(I.LT.J)GOTO 101
  102 P1(J,I)=P1(I,J)
  101 CONTINUE
      XOUT(1)=XE
      XOUT(2)=YE
      XOUT(3)=ZE
      XOUT(4)=USPEED
      XOUT(5)=VSPEED
      XOUT(6)=WSPEED
      XOUT(7)=WX
      XOUT(8)=WY
      XOUT(9)=WZ
C*
      XOUT(10)=BDME1
      XOUT(11)=BDME2
C***K-TH VECTOR Z TRANSFERRED FROM INIT AND VECTOR ZCAL--FROM OUTPT
      DO 111 I=1,6
  111 RES(I)=Z(I+3)-ZCAL(I)
C*
      RES(5)=RES(5)-BDME1
      RES(6)=RES(6)-BDME2
C***
      DO 52 I=1,NXP1
      DO 52 J=1,NXP1
   52 P(I,J)=P1(I,J)
      DO 63 I=1,6
      DO 63 J=1,NXP1
   63 HB(I,J)=HB1(I,J)
C
      RETURN
      END
      SUBROUTINE MYINIT
```

```
      IMPLICIT REAL*8(A-H,O-$)
      CALL INIT
      CALL DYNAMI
      RETURN
      END
      SUBROUTINE MYRUN
      IMPLICIT REAL*8(A-H,O-$)
      COMMON/COM/C(2000)
      COMMON/INCOM/IC(500)
      EQUIVALENCE (C(305),TIME),(IC(207),NSTEP)
      EQUIVALENCE (IC(241),J1),(C(311),TIMSF)
C
  242 CONTINUE
      IF(TIME.LT.TIMSF)GOTO 82
      CALL OUTPT
      GOTO 100
C
   82 CONTINUE
C
C***R.K. LOOP
C
      DO 246 J=1,4
      J1=J
      CALL DYNAM
C
      CALL RKG
  246 CONTINUE
C
C
      NSTEP=NSTEP+1
      GOTO 242
  100 CONTINUE
      RETURN
      END
      SUBROUTINE INIT
      IMPLICIT REAL*8(A-H,O-$)
      REAL*4 Z1(11),XA1(6)
      DIMENSION IPL(100),IPD(100)
      COMMON/COM/C(2000)
      COMMON/INCOM/IC(500)
      COMMON/RK/ARK(4),BRK(4),CRK(4),QRK(100)
C
      EQUIVALENCE (IC(251),NXP1),(C(467),S2LIA),(C(373),S2XYZ)
      COMMON/FQ/FB(11,11),QB(11,11)
      EQUIVALENCE (C(993),HB1(1,1)),(C(571),Z(1))
      DIMENSION HB1(8,11),XA(6),Z(11)
      EQUIVALENCE (IC(259),KSTEP1)
C
      EQUIVALENCE (IC(201),N)
      EQUIVALENCE (IC(202),NRATE1),(C(305),TIME),(C(307),TIMED)
      EQUIVALENCE (C(303),DT),(IC(207),NSTEP)
      EQUIVALENCE (C(309),CRAD),(C(301),GRAV1)
      EQUIVALENCE (C(493),ACCX),(C(495),ACCY),(C(497),ACCZ)
      EQUIVALENCE (C(313),WP),(C(315),WQ),(C(317),WR)
      EQUIVALENCE (C(329),PHI),(C(327),TET),(C(325),PSI)
```

```
      EQUIVALENCE (C(351),CEB11),(C(353),CEB12),(C(355),CEB13)
      EQUIVALENCE (C(357),CEB21),(C(359),CEB22),(C(361),CEB23)
      EQUIVALENCE (C(363),CEB31),(C(365),CEB32),(C(367),CEB33)
      EQUIVALENCE (IC(1),IPL(1)),(IC(101),IPD(1))
      EQUIVALENCE (C(468),S2WIND),(C(470),S2WNDZ)
     1,(C(381),S2ZE),(C(382),S2BDME),(C(474),S2WNDY)
     2,(IC(261),IFLAG),(C(502),OUTL1),(C(504),OUTL2)
     3,(C(506),S2QA),(C(508),S2QB)
C
      GRAV1=32.17D0
C
      ARK(1)=.5D0
      ARK(2)=1.0D0-1.0D0/DSQRT(2.0D0)
      ARK(3)=1.0D0+1.0D0/DSQRT(2.0D0)
      ARK(4)=1.0D0/6.0D0
      BRK(1)=2.0D0
      BRK(2)=1.0D0
      BRK(3)=1.0D0
      BRK(4)=2.0D0
      CRK(1)=ARK(1)
      CRK(2)=ARK(2)
      CRK(3)=ARK(3)
      CRK(4)=ARK(1)
      DO 229 I=1,100
  229 QRK(I)=0.0D0
C
      PI=4.0D0*DATAN(1.0D0)
      CRAD=180.0D0/PI
C
      IPL(1)=305
      IPD(1)=307
      N=1
      TIME=0.0D0
      TIMED=1.0D0
      NSTEP=0
      NRAT=NRATE1
       DT=1.0D0/DFLOAT(NRAT)
C***
C
      DO 306 I=1,NXP1
      DO 307 J=1,NXP1
  307 FB(I,J)=0.0D0
  306 CONTINUE
      DO 308 I=1,8
      DO 309 J=1,NXP1
  309 HB1(I,J)=0.0D0
  308 CONTINUE
      HB1(4,3)=-1.0D0
C*
      HB1(5,10)=1.0D0
      HB1(6,11)=1.0D0
      DO 311 I=1,NXP1
      DO 312 J=1,NXP1
  312 QB(I,J)=0.0D0
  311 CONTINUE
```

```
      QB(4,4)=S2LIA
      QB(5,5)=S2QA
      QB(6,6)=S2QB
      DO 314 I=1,2
  314 QB(I,I)=S2XYZ
      QB(3,3)=S2ZR
      QB(7,7)=S2WIND
      QB(8,8)=S2WNDY
      QB(9,9)=S2WNDZ
C*
      QB(10,10)=S2BDME
      QB(11,11)=S2BDME
C***
C***READ IN SMOOTHED VECTOR X OF MODEL A AND VECTOR Z AT K-TH TIME
C***POINT.Z COSTITUTES OF INPUT(ACCEL MEASUREMENTS) AND MEASUREMENT
C***MATRICES.FIRST NEEDED IN DYNAM AND SECOND--IN MAIN.
C
      IF(KSTEP1.EQ.2)GOTO 319
      READ(10) (XA1(I),I=1,6)
  319 CONTINUE
      READ(9) (Z1(I),I=1,11)
      Z(5)=Z1(5)/CRAD
      Z(6)=Z1(6)/CRAD
      Z(10)=Z1(10)/CRAD
      Z(11)=Z1(11)/CRAD
      Z(4)=Z1(4)
      Z(7)=Z1(7)
      Z(8)=Z1(8)
      Z(9)=Z1(9)
C***
      WP=XA1(1)
      WQ=XA1(2)
      WR=XA1(3)
      PHI=XA1(4)
      TET=XA1(5)
      PSI=XA1(6)
      CALL ROTAT
      ACCX=Z1(1)
      ACCY=Z1(2)
      ACCZ=Z1(3)
C*
C* IFLAG=5 FOR FIRST STEP
      IF(IFLAG.NE.5)GOTO 837
      OOR1=Z(8)
      ONR1=Z(8)
      OOR2=Z(9)
      ONR2=Z(9)
      GOTO 838
  837 CONTINUE
      IF(DABS(Z(8)-ONR1).LT.OUTL1)GOTO 839
      Z(8)=ONR1+(ONR1-OOR1)
  839 CONTINUE
      OOR1=ONR1
      ONR1=Z(8)
      IF(DABS(Z(9)-ONR2).LT.OUTL2)GOTO 840
```

```
      Z(9)=ONR2+(ONR2-OOR2)
  840 CONTINUE
      OOR2=ONR2
      ONR2=Z(9)
  838 CONTINUE
C*
      RETURN
      END
      SUBROUTINE DYNAMI
      IMPLICIT REAL*8(A-H,O-$)
      DIMENSION IPL(100),IPD(100)
      COMMON/COM/C(2000)
      COMMON/INCOM/IC(500)
C***
      DIMENSION X01(11),P01(11,11),P1(11,11)
      EQUIVALENCE (C(337),CSF),(C(339),SNF),(C(341),CST),(C(343),SNT)
     1,(C(505),OMEGA),(C(507),AGLOBE),(C(509),EPS2)
     2,(C(511),TLAMO)
     3,(C(351),CEB11),(C(352),CEB12),(C(353),CEB13)
     4,(C(354),CEB21),(C(355),CEB22),(C(356),CEB23)
     5,(C(357),CEB31),(C(358),CEB32),(C(359),CEB33)
     6,(C(494),BDME1),(C(498),BDME2),(C(312),PERCNT),(C(311),TIMSF)
     7,(C(313),WP),(C(315),WQ),(C(317),WR)
      COMMON/FQ/FB(11,11),QB(11,11)
      DIMENSION B60(11,11),FKDT(11,11),SUME(11,11),SUME1(11,11)
     1,TMAT(11,11),TMATT(11,11),PTMATT(11,11),TMPTMT(11,11)
      EQUIVALENCE (IC(251),NXP1),(C(551),X01(1)),(C(593),P01(1,1))
      EQUIVALENCE (C(793),P1(1,1))
C***
      EQUIVALENCE (IC(201),N)
      EQUIVALENCE (C(303),DT)
      EQUIVALENCE (C(327),TET),(C(325),PSI),(C(329),PHI)
      EQUIVALENCE (C(469),XE),(C(471),YE),(C(473),ZE)
      EQUIVALENCE (C(481),USPEED),(C(485),WSPEED),(C(483),VSPEED)
      EQUIVALENCE (IC(1),IPL(1)),(IC(101),IPD(1))
      EQUIVALENCE (C(482),WX),(C(484),WY),(C(486),WZ)
C
      N=N+1
       IPL(N)=469
       IPD(N)=475
      N=N+1
       IPL(N)=471
       IPD(N)=477
      N=N+1
       IPL(N)=473
       IPD(N)=479
      N=N+1
       IPL(N)=481
       IPD(N)=487
      N=N+1
       IPL(N)=483
       IPD(N)=489
      N=N+1
       IPL(N)=485
       IPD(N)=491
```

```
C***
      N=N+1
       IPL(N)=482
       IPD(N)=488
      N=N+1
       IPL(N)=484
       IPD(N)=490
      N=N+1
       IPL(N)=486
       IPD(N)=492
C*
      N=N+1
       IPL(N)=494
       IPD(N)=496
      N=N+1
       IPL(N)=498
       IPD(N)=500
C
      XE=X01(1)
      YE=X01(2)
      ZE=X01(3)
C
      USPEED=X01(4)
      VSPEED=X01(5)
      WSPEED=X01(6)
C***
      WX=X01(7)
      WY=X01(8)
      WZ=X01(9)
C*
      BDME1=X01(10)
      BDME2=X01(11)
C***
      DO 201 I=1,NXP1
  201 P1(I,1)=P01(I,1)
      DO 202 I=2,NXP1
  202 P1(I,2)=P01(I,2)
      DO 203 I=3,NXP1
  203 P1(I,3)=P01(I,3)
      DO 204 I=4,NXP1
  204 P1(I,4)=P01(I,4)
      DO 205 I=5,NXP1
  205 P1(I,5)=P01(I,5)
      DO 206 I=6,NXP1
  206 P1(I,6)=P01(I,6)
      DO 207 I=7,NXP1
  207 P1(I,7)=P01(I,7)
      DO 208 I=8,NXP1
  208 P1(I,8)=P01(I,8)
      DO 209 I=9,NXP1
  209 P1(I,9)=P01(I,9)
      DO 210 I=10,NXP1
  210 P1(I,10)=P01(I,10)
      P1(11,11)=P01(11,11)
C***
```

```
      DO 101 I=1,NXP1
      DO 102 J=1,NXP1
      IF(I.LT.J)GOTO 101
  102 P1(J,I)=P1(I,J)
  101 CONTINUE
C***
      TLAM=TLAM0+XE/AGLOBE
      COST=DCOS(TLAM)
      SINT=DSIN(TLAM)
      TANT=SINT/COST
      SIN2T=2*SINT*COST
      COS2T=COST*COST-SINT*SINT
      C1111=1.0D0-ZE/AGLOBE-.5D0*EPS2*COS2T
      C112=1/C1111
C
C
      C141=C112*(ZE/AGLOBE+.5*EPS2*COS2T)
      C111=(CEB11*USPEED+CEB21*VSPEED+CEB31*WSPEED+WX)/AGLOBE
      C211=(CEB12*USPEED+CEB22*VSPEED+CEB32*WSPEED+WY)/AGLOBE
      C241=-C112*YE*TANT/AGLOBE
      C341=C112*EPS2*SIN2T
      C56=OMEGA*(CEB11*COST-CEB13*SINT)
      C64=OMEGA*(CEB21*COST-CEB23*SINT)
      C45=OMEGA*(CEB31*COST-CEB33*SINT)
C***
      C413=OMEGA*(CEB11*SINT+CEB13*COST)
      C412=OMEGA*(CEB21*SINT+CEB23*COST)
      C411=OMEGA*(CEB31*SINT+CEB33*COST)
      CENT1=-OMEGA**2*AGLOBE*C1111*.5*SIN2T
      CENT2=-OMEGA**2*AGLOBE*C1111*COST**2
      C47=2*(+WR*CEB21-WQ*CEB31)+OMEGA*CEB12*SINT
      C48=2*(+WR*CEB22-WQ*CEB32)-C413
      C49=2*(+WR*CEB23-WQ*CEB33)-OMEGA*CEB12*COST
      C57=2*(-WR*CEB11+WP*CEB31)+OMEGA*CEB22*SINT
      C58=2*(-WR*CEB12+WP*CEB32)-C412
      C59=2*(-WR*CEB13+WP*CEB33)-OMEGA*CEB22*COST
      C67=2*(+WQ*CEB11-WP*CEB21)+OMEGA*CEB32*SINT
      C68=2*(+WQ*CEB12-WP*CEB22)-C411
      C69=2*(+WQ*CEB13-WP*CEB23)-OMEGA*CEB32*COST
C
C***
      C1122=C112*C112
      C2112=C211*C1122
      C13=C111*C1122
      C11=-EPS2*SIN2T*C13
      C31=C111*C112*EPS2*(2*COS2T-C112*EPS2*SIN2T*SIN2T)
      C33=C13*EPS2*SIN2T
      C21=-EPS2*SIN2T*C2112+C13*(-C1111+.5D0*EPS2*SIN2T*SIN2T)*YE/
     1 AGLOBE/COST/COST
      C22=-C111*C112*TANT
      C23=C2112-C13*TANT*YE/AGLOBE
      C14=C141*CEB11
      C15=C141*CEB21
      C16=C141*CEB31
      C24=C241*CEB11+C141*CEB12
```

```
      C25=C241*CEB21+C141*CEB22
      C26=C241*CEB31+C141*CEB32
      C34=C341*CEB11
      C35=C341*CEB21
      C36=C341*CEB31
      C17=C141
      C27=C241
      C28=C141
      C37=C341
      C411=OMEGA*(CEB31*SINT+CEB33*COST)
      C412=OMEGA*(CEB21*SINT+CEB23*COST)
      C413=OMEGA*(CEB11*SINT+CEB13*COST)
      C433=OMEGA*COST
      C43=C433*C413
      C53=C433*C412
      C63=C433*C411
      COST1=COST*OMEGA/AGLOBE
      SINT1=SINT*OMEGA/AGLOBE
      C4111=(C1111*COS2T+.5D0*EPS2*SIN2T*SIN2T)*OMEGA*OMEGA
      C4112=(C1111*SIN2T-      EPS2*SIN2T*COST*COST)*OMEGA*OMEGA
      C41=(WSPEED*C412-VSPEED*C411)/AGLOBE
     1-CEB11*C4111+CEB13*C4112+COST1*(WX*CEB12-WY*CEB11)+SINT1*(
     2 WY*CEB13+WZ*CEB12)
      C51=(USPEED*C411-WSPEED*C413)/AGLOBE
     1-CEB21*C4111+CEB23*C4112+COST1*(WX*CEB22-WY*CEB21)+SINT1*(
     2 WY*CEB23+WZ*CEB22)
      C61=(VSPEED*C413-USPEED*C412)/AGLOBE
     1-CEB31*C4111+CEB33*C4112+COST1*(WX*CEB32-WY*CEB31)+SINT1*(
     2 WY*CEB33+WZ*CEB32)
      C65=-C56
      C46=-C64
      C54=-C45
      FB(1,1)=C11
      FB(2,1)=C21
      FB(3,1)=C31
      FB(4,1)=C41
      FB(5,1)=C51
      FB(6,1)=C61
      FB(2,2)=C22
      FB(1,3)=C13
      FB(2,3)=C23
      FB(3,3)=C33
      FB(4,3)=C43
      FB(5,3)=C53
      FB(6,3)=C63
      FB(1,4)=CEB11+C14
      FB(2,4)=CEB12+C24
      FB(3,4)=+CEB13+C34
      FB(1,5)=CEB21+C15
      FB(2,5)=CEB22+C25
      FB(3,5)=+CEB23+C35
      FB(1,6)=CEB31+C16
      FB(2,6)=CEB32+C26
      FB(3,6)=+CEB33+C36
      FB(5,4)=-WR+C54
```

```
      FB(6,4)=WQ+C64
      FB(4,5)=WR+C45
      FB(6,5)=-WP+C65
      FB(4,6)=-WQ+C46
      FB(5,6)=WP+C56
      FB(1,7)=1.0D0+C17
      FB(2,8)=1.0D0+C141
      FB(3,9)=1.0D0
      FB(2,7)=C27
      FB(3,7)=C37
      FB(4,7)=C47
      FB(4,8)=C48
      FB(4,9)=C49
      FB(5,7)=C57
      FB(5,8)=C58
      FB(5,9)=C59
      FB(6,7)=C67
      FB(6,8)=C68
      FB(6,9)=C69
C****
C
      CALL CSTM(FB,PERCNT,TIMSF,TMAT,B60,FKDT,SUME,SUME1,NXP1,NTERMS)
      DO 600 I=1,NXP1
      DO 600 J=1,NXP1
  600 TMATT(J,I)=TMAT(I,J)
      DO 610 I=1,NXP1
      DO 610 J=1,NXP1
  610 TMPTMT(I,J)=P1(I,J)+QB(I,J)
      CALL VMULFF(TMPTMT,TMATT ,NXP1,NXP1,NXP1,NXP1,NXP1,PTMATT,NXP1,I1)
      CALL VMULFF(TMAT   ,PTMATT,NXP1,NXP1,NXP1,NXP1,NXP1,P1     ,NXP1,I2)
C
      RETURN
      END
      SUBROUTINE DYNAM
      IMPLICIT REAL*8(A-H,O-$)
      COMMON/COM/C(2000)
      COMMON/INCOM/IC(500)
C***
      EQUIVALENCE (IC(251),NXP1)
      COMMON/FQ/FB(11,11),QB(11,11)
      EQUIVALENCE (C(494),BDME1),(C(496),DBDME1)
     1,(C(498),BDME2),(C(500),DBDME2)
      EQUIVALENCE (C(793),P1(1,1))
      EQUIVALENCE (C(401),DP11),(C(402),DP21),(C(403),DP31)
      EQUIVALENCE (C(404),DP41),(C(405),DP51),(C(406),DP61)
      EQUIVALENCE (C(407),DP71),(C(408),DP81),(C(409),DP91)
      EQUIVALENCE (C(410),DP22),(C(411),DP32),(C(412),DP42)
      EQUIVALENCE (C(413),DP52),(C(414),DP62),(C(415),DP72)
      EQUIVALENCE (C(416),DP82),(C(417),DP92)
      EQUIVALENCE (C(418),DP33),(C(419),DP43),(C(420),DP53)
      EQUIVALENCE (C(421),DP63),(C(422),DP73),(C(423),DP83)
      EQUIVALENCE (C(424),DP93)
      EQUIVALENCE (C(425),DP44),(C(426),DP54),(C(427),DP64)
      EQUIVALENCE (C(428),DP74),(C(429),DP84),(C(430),DP94)
      EQUIVALENCE (C(431),DP55),(C(432),DP65),(C(433),DP75)
```

```
      EQUIVALENCE (C(434),DP85),(C(435),DP95),(C(436),DP66)             OP
      EQUIVALENCE (C(437),DP76),(C(438),DP86),(C(439),DP96)             OP
      EQUIVALENCE (C(440),DP77),(C(441),DP87),(C(442),DP97)             OP
      EQUIVALENCE (C(443),DP88),(C(444),DP98),(C(445),DP99)             OP
      EQUIVALENCE (C(319),WPD),(C(321),WQD),(C(323),WRD)                OP
      DIMENSION DPM(11,11),FP(11,11),P1(11,11)                          OP
C***                                                                    OP
      EQUIVALENCE (C(482),WX),(C(484),WY),(C(486),WZ)                   OP
      EQUIVALENCE (C(488),DWX),(C(490),DWY),(C(492),DWZ)                OP
C                                                                       OP
      EQUIVALENCE (C(315),WQ),(C(327),TET),(C(333),DTET)                OP
      EQUIVALENCE (C(499),VX),(C(503),VZ)                               OP
      EQUIVALENCE (C(469),XE),(C(473),ZE),(C(475),XED),(C(479),ZED)     OP
      EQUIVALENCE (C(351),CEB11)                                        OP
      EQUIVALENCE (C(355),CEB13),(C(363),CEB31),(C(367),CEB33)          OP
      EQUIVALENCE (C(325),PSI),(C(329),PHI),(C(331),DPSI),(C(335),DPHI) OP
      EQUIVALENCE (C(313),WP),(C(317),WR)                               OP
      EQUIVALENCE (C(501),VY),(C(471),YE),(C(477),YED)                  OP
      EQUIVALENCE (C(337),CSF),(C(339),SNF),(C(341),CST),(C(343),SNT)   OP
      EQUIVALENCE (C(353),CEB12),(C(357),CEB21),(C(359),CEB22)          OP
      EQUIVALENCE (C(361),CEB23),(C(365),CEB32)                         OP
      EQUIVALENCE (C(493),ACCX),(C(495),ACCY),(C(497),ACCZ)             OP
      EQUIVALENCE (C(481),USPEED),(C(483),VSPEED),(C(485),WSPEED)       OP
      EQUIVALENCE (C(487),DUSPED),(C(489),DVSPED),(C(491),DWSPED)       OP
      EQUIVALENCE (C(301),GRAV1)                                        OP
      EQUIVALENCE (C(511),TLAM0)                                        OP
      EQUIVALENCE (C(505),OMEGA),(C(507),AGLOBE),(C(509),EPS2)          OP
C***                                                                    OP
C                                                                       OP
      CALL DBTOI(USPEED,VSPEED,WSPEED,CEB11,CEB12,CEB13                 OP
     1,CEB21,CEB22,CEB23,CEB31,CEB32,CEB33,VX,VY,VZ)                    OP
C***                                                                    OP
      DO 101 I=1,NXP1                                                   OP
      DO 102 J=1,NXP1                                                   OP
      IF(I.LT.J)GOTO 101                                                OP
  102 P1(J,I)=P1(I,J)                                                   OP
  101 CONTINUE                                                          OP
      TLAM=TLAM0+XE/AGLOBE                                              OP
      COST=DCOS(TLAM)                                                   OP
      SINT=DSIN(TLAM)                                                   OP
      TANT=SINT/COST                                                    OP
      SIN2T=2*SINT*COST                                                 OP
      COS2T=COST*COST-SINT*SINT                                         OP
      C1111=1.0D0-ZE/AGLOBE-.5D0*EPS2*COS2T                             OP
      C112=1/C1111                                                      OP
C                                                                       OP
C                                                                       OP
      C141=C112*(ZE/AGLOBE+.5*EPS2*COS2T)                               OP
      C111=(CEB11*USPEED+CEB21*VSPEED+CEB31*WSPEED+WX)/AGLOBE           OP
      C211=(CEB12*USPEED+CEB22*VSPEED+CEB32*WSPEED+WY)/AGLOBE           OP
      C241=-C112*YE*TANT/AGLOBE                                         OP
      C341=C112*EPS2*SIN2T                                              OP
      C56=OMEGA*(CEB11*COST-CEB13*SINT)                                 OP
      C64=OMEGA*(CEB21*COST-CEB23*SINT)                                 OP
      C45=OMEGA*(CEB31*COST-CEB33*SINT)                                 OP
```

```
      XED=VX+WX+C141*C111*AGLOBE
      ZED=+VZ+WZ+C341*C111*AGLOBE
      YED=VY+WY+C141*C211*AGLOBE+C241*C111*AGLOBE
C***
      C413=OMEGA*(CEB11*SINT+CEB13*COST)
      C412=OMEGA*(CEB21*SINT+CEB23*COST)
      C411=OMEGA*(CEB31*SINT+CEB33*COST)
      CENT1=-OMEGA**2*AGLOBE*C1111*.5*SIN2T
      CENT2=-OMEGA**2*AGLOBE*C1111*COST**2
      C47=2*(+WR*CEB21-WQ*CEB31)+OMEGA*CEB12*SINT
      C48=2*(+WR*CEB22-WQ*CEB32)-C413
      C49=2*(+WR*CEB23-WQ*CEB33)-OMEGA*CEB12*COST
      C57=2*(-WR*CEB11+WP*CEB31)+OMEGA*CEB22*SINT
      C58=2*(-WR*CEB12+WP*CEB32)-C412
      C59=2*(-WR*CEB13+WP*CEB33)-OMEGA*CEB22*COST
      C67=2*(+WQ*CEB11-WP*CEB21)+OMEGA*CEB32*SINT
      C68=2*(+WQ*CEB12-WP*CEB22)-C411
      C69=2*(+WQ*CEB13-WP*CEB23)-OMEGA*CEB32*COST
C
C***
      DUSPED=-WQ*WSPEED+WR*VSPEED     +ACCX*GRAV1
     1+C45*VSPEED-C64*WSPEED
     2+C47*WX+C48*WY+C49*WZ+CENT1*CEB11+CENT2*CEB13
C*
C*
      DVSPED=-WR*USPEED+WP*WSPEED+ACCY*GRAV1
     1-C45*USPEED+C56*WSPEED
     2+C57*WX+C58*WY+C59*WZ+CENT1*CEB21+CENT2*CEB23
      DWSPED=+WQ*USPEED-WP*VSPEED+ACCZ*GRAV1
     1+C64*USPEED-C56*VSPEED
     2+C67*WX+C68*WY+C69*WZ+CENT1*CEB31+CENT2*CEB33
C
      DWX=0.0D0
      DWY=0.0D0
      DWZ=0.0D0
C*
      DBDME1=0.0D0
      DBDME2=0.0D0
C***
C
      RETURN
      END
      SUBROUTINE OUTPT
      IMPLICIT REAL*8(A-H,O-$)
      COMMON/COM/C(2000)
      COMMON/INCOM/IC(500)
      DIMENSION RDME(7),VOR(7)
C***
      DIMENSION XST1(7),YST1(7),ZST1(7),DIREQ1(7)
      EQUIVALENCE (C(1257),XST1(1)),(C(1271),YST1(1)),(C(1285),ZST1(1))
      EQUIVALENCE (C(1299),DIREQ1(1)),(IC(252),ISTDM1),(IC(253),ISTVO1)
      EQUIVALENCE (C(993),HB1(1,1))
      EQUIVALENCE (C(1235),ZCAL(1))
      DIMENSION ZCAL(8),HB1(8,11)
C
```

```
      EQUIVALENCE (C(309),CRAD)
      EQUIVALENCE (C(473),ZE)
      EQUIVALENCE (C(305),TIME)
      EQUIVALENCE (C(315),WQ),(C(499),VX),(C(503),VZ),(C(469),XE)
      EQUIVALENCE (C(327),TET)
      EQUIVALENCE (C(325),PSI),(C(329),PHI)
      EQUIVALENCE (C(313),WP),(C(317),WR),(C(501),VY),(C(471),YE)
      EQUIVALENCE (C(493),ACCX),(C(495),ACCY),(C(497),ACCZ)
      EQUIVALENCE (C(481),USPEED),(C(483),VSPEED),(C(485),WSPEED)
      EQUIVALENCE (C(507),AGLOBE),(C(509),EPS2)
      EQUIVALENCE (C(511),TLAM0),(C(513),SMEW0)
      DIMENSION XDME(7),YDME(7),ZDME(7)
C***
      TLAM=TLAM0+XE/AGLOBE
      COST=DCOS(TLAM)
      SINT=DSIN(TLAM)
      TANT=SINT/COST
      SMEW=SMEW0+YE/AGLOBE/COST
      SIN2T=2*SINT*COST
      COS2T=COST*COST-SINT*SINT
      C1111=1.0D0-.5D0*EPS2*COS2T-ZE/AGLOBE
      AC1111=AGLOBE*C1111
      COSS=DCOS(SMEW)
      SINS=DSIN(SMEW)
      RX1=EPS2*SIN2T*COST*COSS-C1111*SINT*COSS-C1111*TANT*SINS*YE/AGLOBE
      RX2=EPS2*SIN2T*COST*SINS-C1111*SINT*SINS+C1111*TANT*COSS*YE/AGLOBE
      RX3=EPS2*SIN2T*SINT+C1111*COST
C
C***VECTORS X OF MODEL A AND Z(K-TH TIME POINT) OF MODEL B(WITHOUT
C***ACCEL) TRANSFERRED FROM INIT
C
      VAIR2=USPEED**2+VSPEED**2+WSPEED**2
      VAIR=DSQRT(VAIR2)
      VRP=VSPEED
      USPD=USPEED
      BETA=DATAN2(VRP,USPD)
      WQR=WSPEED
      ALPHA=DATAN2(WQR,USPD)
      DO 301 I=1,ISTDM1
      XDME(I)=AC1111*COST*COSS-DCOS(XST1(I))*DCOS(YST1(I))*
     1 (-ZST1(I)+AGLOBE*(1.0D0-.5D0*EPS2*DCOS(2*XST1(I))))
      YDME(I)=AC1111*COST*SINS-DCOS(XST1(I))*DSIN(YST1(I))*
     1 (-ZST1(I)+AGLOBE*(1.0D0-.5D0*EPS2*DCOS(2*XST1(I))))
      ZDME(I)=AC1111*SINT      -DSIN(XST1(I))*
     1 (-ZST1(I)+AGLOBE*(1.0D0-.5D0*EPS2*DCOS(2*XST1(I))))
C***XST,YST ARE LATITUDE AND LONGITUDE
  301 RDME(I)=DSQRT(XDME(I)**2+YDME(I)**2+ZDME(I)**2)
C
C*301 RDME(I)=DSQRT((XE-XST1(I))**2+(YE-YST1(I))**2+(ZE-ZST1(I))**2)
C
      DO 302 I=1,ISTVO1
  302 VOR(I)=1.
C***
      ZCAL(1)=VAIR
      ZCAL(2)=ALPHA
```

```
      ZCAL(3)=BETA
      ZCAL(4)=-ZE
      ZCAL(5)=RDME(1)
      ZCAL(6)=RDME(2)
      ZCAL(7)=VOR(1)
      ZCAL(8)=VOR(2)
C
      HB1(5,1)=(RX1*XDME(1)+RX2*YDME(1)+RX3*ZDME(1))/RDME(1)
      HB1(6,1)=(RX1*XDME(2)+RX2*YDME(2)+RX3*ZDME(2))/RDME(2)
      HB1(5,2)=(-C1111*SINS*XDME(1)+C1111*COSS*YDME(1))/RDME(1)
      HB1(6,2)=(-C1111*SINS*XDME(2)+C1111*COSS*YDME(2))/RDME(2)
      HB1(5,3)=(-COST*COSS*XDME(1)-COST*SINS*YDME(1)-SINT*ZDME(1))
     1/RDME(1)
      HB1(6,3)=(-COST*COSS*XDME(2)-COST*SINS*YDME(2)-SINT*ZDME(2))
     1/RDME(2)
      HB1(7,1)=-(YE-YST1(1))/(RDME(1)**2-(ZE-ZST1(1))**2)
      HB1(8,1)=-(YE-YST1(2))/(RDME(2)**2-(ZE-ZST1(2))**2)
      HB1(7,2)=(XE-XST1(1))/(RDME(1)**2-(ZE-ZST1(1))**2)
      HB1(8,2)=(XE-XST1(2))/(RDME(2)**2-(ZE-ZST1(2))**2)
      HB1(1,4)=USPEED/VAIR
      HB1(2,4)=-WSPEED/(VAIR2-VSPEED**2)
      HB1(3,4)=-VSPEED/(VAIR2-WSPEED**2)
      HB1(1,5)=VSPEED/VAIR
      HB1(3,5)=USPEED/(VAIR2-WSPEED**2)
      HB1(1,6)=WSPEED/VAIR
      HB1(2,6)=USPEED/(VAIR2-VSPEED**2)
C
      RETURN
      END
      FUNCTION SATF(X11,XM11)
      IMPLICIT REAL*8(A-H,O-$)
      SATF=DSIGN(DMIN1(DABS(X11),XM11),X11)
      RETURN
      END
      SUBROUTINE ROTAT
      IMPLICIT REAL*8(A-H,O-$)
      COMMON/COM/C(2000)
      EQUIVALENCE (C(327),TET),(C(351),CEB11),(C(355),CEB13)
      EQUIVALENCE (C(363),CEB31),(C(367),CEB33)
      EQUIVALENCE (C(341),CST),(C(343),SNT)
      EQUIVALENCE (C(325),PSI),(C(329),PHI),(C(353),CEB12)
      EQUIVALENCE (C(357),CEB21),(C(359),CEB22),(C(361),CEB23)
      EQUIVALENCE (C(365),CEB32),(C(345),CSP),(C(347),SNP)
      EQUIVALENCE (C(337),CSF),(C(339),SNF)
C
      SNT=DSIN(TET)
      CST=DCOS(TET)
      SNP=DSIN(PSI)
      CSP=DCOS(PSI)
      SNF=DSIN(PHI)
      CSF=DCOS(PHI)
C
      CEB11=CST*CSP
      CEB13=-SNT
      CEB31=+SNT*CSF*CSP+SNF*SNP
```

```
      CEB33=CST*CSF
      CEB12=CST*SNP
      CEB21=SNF*SNT*CSP-CSF*SNP
      CEB22=SNF*SNT*SNP+CSF*CSP
      CEB23=SNF*CST
      CEB32=CSF*SNT*SNP-SNF*CSP
      RETURN
      END
      SUBROUTINE DITOB(XI,YI,ZI,A11,A12,A13,A21,A22,A23,A31,A32,A33,
     1XO,YO,ZO)
      IMPLICIT REAL*8(A-H,O-$)
      XO=A11*XI+A13*ZI+A12*YI
      ZO=A31*XI+A33*ZI+A32*YI
      YO=A21*XI+A22*YI+A23*ZI
      RETURN
      ENTRY DBTOI(XI,YI,ZI,A11,A12,A13,A21,A22,A23,A31,A32,A33,
     1XO,YO,ZO)
      XO=A11*XI+A31*ZI+A21*YI
      ZO=A13*XI+A33*ZI+A23*YI
      YO=A12*XI+A22*YI+A32*ZI
      RETURN
      END
      SUBROUTINE RKG
      IMPLICIT REAL*8(A-H,O-$)
      DIMENSION IPL(100),IPD(100)
      COMMON/COM/C(2000)
      COMMON/RK/ARK(4),BRK(4),CRK(4),QRK(100)
      COMMON/INCOM/IC(500)
C
      EQUIVALENCE (IC(201),N)
      EQUIVALENCE (C(303),H),(IC(241),J)
      EQUIVALENCE (IC(1),IPL(1)),(IC(101),IPD(1))
C
      DO 100 I=1,N
      IL=IPL(I)
      ID=IPD(I)
      X1=C(ID)*H
      X2=(X1-BRK(J)*QRK(I))*ARK(J)
      C(IL)=C(IL)+X2
  100 QRK(I)=QRK(I)+3.0D0*X2-CRK(J)*X1
C
      RETURN
      END
```

```
C*
C*OPTIMAL SMOOTHING PROGRAM
C*
      IMPLICIT REAL*8(A-H,O-$)
      REAL*4 XS1(11),PS1(9,9),XAS(6,1)
      COMMON/DAT0/PERCNT,DT,TLAM0,OMEGA,AGLOBE,EPS2,IDGT,NSTEP,NXP,SOF
      LOGICAL GO,SOF
      NAMELIST/INP/FILE
      INTEGER FILE(2)
      NAMELIST/OK/GO
      NAMELIST/DAT/SOF,NSTEP,NXP,PERCNT,DT,IDGT,TLAM0
     1,OMEGA,AGLOBE,EPS2
C***
      DEFINE FILE 17( 150,368,L,KSTEPG)
      DEFINE FILE 12( 150,2112,L             ,KSTEPA)
      DIMENSION XPK12(11),XMK12(11),PPK12(11,11),PMK12(11,11)
C*
      DIMENSION XPK1(6),XMK1(6),XPK(6),XMK(6)
     1,PPK1(6,6),PMK1(6,6),PPK(6,6),PMK(6,6)
     2,FK(6,6),TMAT(6,6),TMATT(6,6),PTMATT(6,6)
     3,PINV(6,6),WKAREA(60),AK(6,6),AKT(6,6)
     4,DELX(6,1),DELP(6,6),DELXES(6,1),XS(6)
     5,AKDELP(6,6),ADPAT(6,6),PS(6,6)
     6,PMK2(6,6)
C***
      DIMENSION B60(6,6),FKDT(6,6),SUME(6,6),SUME1(6,6)
C***
    1 CONTINUE
      PRINT 1002
 1002 FORMAT(1H ,'TYPE &INP FILE= &END')
      READ(5,INP)
      INF1=FILE(1)
      INF2=FILE(2)
      PRINT 2000,FILE
 2000 FORMAT(2I10)
      PRINT 1004
 1004 FORMAT(1H ,'TYPE &DAT DATA= &END')
      READ(INF1,DAT)
      IF(SOF)GOTO 92
      PRINT 1003
 1003 FORMAT(1H ,'IF EVERYTHING O.K. TYPE &OK GO=.TRUE.&END')
      GO=.TRUE.
      READ(5,OK)
      IF(GO)GOTO 10
      GOTO 1
   10 CONTINUE
C***
C***
      KSTEP=NSTEP
      KSTEP1=0
C***
      KSTEPA=KSTEP
C*
      PI=4.0D0*DATAN(1.0D0)
      CRAD=180.0D0/PI
```

```
C***
      READ(10) (XAS(I,1),I=1,6)
      READ(12'KSTEPA) (XPK12(I),I=1,11)
     1,(XMK12(I),I=1,11),((PPK12(I,J),I=1,11),J=1,11)
     2,((PMK12(I,J),I=1,11),J=1,11)
C*
      DO 431 I=1,NXP
      XPK1(I)=XPK12(I)
  431 XMK1(I)=XMK12(I)
      DO 432 I=1,NXP
      DO 432 J=1,NXP
      PPK1(I,J)=PPK12(I,J)
  432 PMK1(I,J)=PMK12(I,J)
C***
      DO 581 I=1,NXP
  581 XS(I)=XPK1(I)
      DO 582 I=1,NXP
      DO 582 J=1,NXP
  582 PS(I,J)=PPK1(I,J)
      KSTEPG=KSTEP
      DO 709 I=1,NXP
  709 XS1(I)=XS(I)
      DO 710 I=1,NXP
      DO 710 J=1,NXP
  710 PS1(I,J)=PS(I,J)
      XS1(7)=XPK12(7)
      XS1(8)=XPK12(8)
      XS1(9)=XPK12(9)
      DO 801 I=7,9
      DO 801 J=1,9
  801 PS1(I,J)=PPK12(I,J)
      DO 802 I=1,6
      DO 802 J=7,9
  802 PS1(I,J)=PPK12(I,J)
C***
      XS1(10)=XPK12(10)
      XS1(11)=XPK12(11)
      WRITE(17'KSTEPG) (XS1(I),I=1,11)
     1,((PS1(I,J),I=1,9),J=1,9)
C***
      DO 591 I=1,NXP
      DO 591 J=1,NXP
  591 FK(I,J)=0.
C***
C***
  301 CONTINUE
      KSTEP=KSTEP-1
      IF(KSTEP.LT.1)GOTO 311
      KSTEP1=KSTEP1+1
C***
      KSTEPA=KSTEP
C***
      READ(10) (XAS(I,1),I=1,6)
      READ(12'KSTEPA) (XPK12(I),I=1,11)
     1,(XMK12(I),I=1,11),((PPK12(I,J),I=1,11),J=1,11)
```

```
     2,((PMK12(I,J),I=1,11),J=1,11)
C*
      DO 441 I=1,NXP
      XPK(I)=XPK12(I)
  441 XMK(I)=XMK12(I)
      DO 442 I=1,NXP
      DO 442 J=1,NXP
      PPK(I,J)=PPK12(I,J)
  442 PMK(I,J)=PMK12(I,J)
C***
C***FK-MATRIX COMPUTATION
      WP=XAS(1,1)
      WQ=XAS(2,1)
      WR=XAS(3,1)
      PHI=XAS(4,1)
      TET=XAS(5,1)
      PSI=XAS(6,1)
      SNT=DSIN(TET)
      CST=DCOS(TET)
      SNF=DSIN(PHI)
      CSF=DCOS(PHI)
      SNP=DSIN(PSI)
      CSP=DCOS(PSI)
C***
      CEB11=CST*CSP
      CEB12=CST*SNP
      CEB13=-SNT
      CEB21=SNF*SNT*CSP-CSF*SNP
      CEB22=SNF*SNT*SNP+CSF*CSP
      CEB23=+SNF*CST
      CEB31=SNT*CSF*CSP+SNF*SNP
      CEB32=CSF*SNT*SNP-SNF*CSP
      CEB33=+CST*CSF
C***
      XE=XPK(1)
      YE=XPK(2)
      ZE=XPK(3)
      USPEED=XPK(4)
      VSPEED=XPK(5)
      WSPEED=XPK(6)
C***
      WX=XPK12(7)
      WY=XPK12(8)
      WZ=XPK12(9)
C***
      TLAM=TLAM0+XE/AGLOBE
      COST=DCOS(TLAM)
      SINT=DSIN(TLAM)
      TANT=SINT/COST
      SIN2T=2*SINT*COST
      COS2T=COST*COST-SINT*SINT
      C1111=1.0D0-ZE/AGLOBE-.5D0*EPS2*COS2T
      C112=1/C1111
C*
C*
```

```
      C141=C112*(ZE/AGLOBE+.5*EPS2*COS2T)                               OP
      C111=(CEB11*USPEED+CEB21*VSPEED+CEB31*WSPEED+WX)/AGLOBE           OP
      C211=(CEB12*USPEED+CEB22*VSPEED+CEB32*WSPEED+WY)/AGLOBE           OP
      C241=-C112*YE*TANT/AGLOBE                                         OP
      C341=C112*EPS2*SIN2T                                              OP
      C56=OMEGA*(CEB11*COST-CEB13*SINT)                                 OP
      C64=OMEGA*(CEB21*COST-CEB23*SINT)                                 OP
      C45=OMEGA*(CEB31*COST-CEB33*SINT)                                 OP
C***                                                                    OP
      C413=OMEGA*(CEB11*SINT+CEB13*COST)                                OP
      C412=OMEGA*(CEB21*SINT+CEB23*COST)                                OP
      C411=OMEGA*(CEB31*SINT+CEB33*COST)                                OP
      CENT1=-OMEGA**2*AGLOBE*C1111*.5*SIN2T                             OP
      CENT2=-OMEGA**2*AGLOBE*C1111*COST**2                              OP
      C47=2*(+WR*CEB21-WQ*CEB31)+OMEGA*CEB12*SINT                       OP
      C48=2*(+WR*CEB22-WQ*CEB32)-C413                                   OP
      C49=2*(+WR*CEB23-WQ*CEB33)-OMEGA*CEB12*COST                       OP
      C57=2*(-WR*CEB11+WP*CEB31)+OMEGA*CEB22*SINT                       OP
      C58=2*(-WR*CEB12+WP*CEB32)-C412                                   OP
      C59=2*(-WR*CEB13+WP*CEB33)-OMEGA*CEB22*COST                       OP
      C67=2*(+WQ*CEB11-WP*CEB21)+OMEGA*CEB32*SINT                       OP
      C68=2*(+WQ*CEB12-WP*CEB22)-C411                                   OP
      C69=2*(+WQ*CEB13-WP*CEB23)-OMEGA*CEB32*COST                       OP
C*                                                                      OP
      C1122=C112*C112                                                   OP
      C2112=C211*C1122                                                  OP
      C13=C111*C1122                                                    OP
      C11=-EPS2*SIN2T*C13                                               OP
      C31=C111*C112*EPS2*(2*COS2T-C112*EPS2*SIN2T*SIN2T)                OP
      C33=C13*EPS2*SIN2T                                                OP
      C21=-EPS2*SIN2T*C2112+C13*(-C1111+.5D0*EPS2*SIN2T*SIN2T)*YE/      OP
     1AGLOBE/COST/COST                                                  OP
      C22=-C111*C112*TANT                                               OP
      C23=C2112-C13*TANT*YE/AGLOBE                                      OP
      C14=C141*CEB11                                                    OP
      C15=C141*CEB21                                                    OP
      C16=C141*CEB31                                                    OP
      C24=C241*CEB11+C141*CEB12                                         OP
      C25=C241*CEB21+C141*CEB22                                         OP
      C26=C241*CEB31+C141*CEB32                                         OP
      C34=C341*CEB11                                                    OP
      C35=C341*CEB21                                                    OP
      C36=C341*CEB31                                                    OP
      C17=C141                                                          OP
      C27=C241                                                          OP
      C28=C141                                                          OP
      C37=C341                                                          OP
      C411=OMEGA*(CEB31*SINT+CEB33*COST)                                OP
      C412=OMEGA*(CEB21*SINT+CEB23*COST)                                OP
      C413=OMEGA*(CEB11*SINT+CEB13*COST)                                OP
      C433=OMEGA*COST                                                   OP
      C43=C433*C413                                                     OP
      C53=C433*C412                                                     OP
      C63=C433*C411                                                     OP
      COST1=COST*OMEGA/AGLOBE                                           OP
```

```
      SINT1=SINT*OMEGA/AGLOBE
      C4111=(C1111*COS2T+.5D0*EPS2*SIN2T*SIN2T)*OMEGA*OMEGA
      C4112=(C1111*SIN2T-      EPS2*SIN2T*COST*COST)*OMEGA*OMEGA
      C41=(WSPEED*C412-VSPEED*C411)/AGLOBE
     1-CEB11*C4111+CEB13*C4112+COST1*(WX*CEB12-WY*CEB11)+SINT1*(
     2 WY*CEB13+WZ*CEB12)
      C51=(USPEED*C411-WSPEED*C413)/AGLOBE
     1-CEB21*C4111+CEB23*C4112+COST1*(WX*CEB22-WY*CEB21)+SINT1*(
     2 WY*CEB23+WZ*CEB22)
      C61=(VSPEED*C413-USPEED*C412)/AGLOBE
     1-CEB31*C4111+CEB33*C4112+COST1*(WX*CEB32-WY*CEB31)+SINT1*(
     2 WY*CEB33+WZ*CEB32)
      C65=-C56
      C46=-C64
      C54=-C45
      FK(1,1)=C11
      FK(2,1)=C21
      FK(3,1)=C31
      FK(4,1)=C41
      FK(5,1)=C51
      FK(6,1)=C61
      FK(2,2)=C22
      FK(1,3)=C13
      FK(2,3)=C23
      FK(3,3)=C33
      FK(4,3)=C43
      FK(5,3)=C53
      FK(6,3)=C63
      FK(1,4)=CEB11+C14
      FK(2,4)=CEB12+C24
      FK(3,4)=+CEB13+C34
      FK(1,5)=CEB21+C15
      FK(2,5)=CEB22+C25
      FK(3,5)=+CEB23+C35
      FK(1,6)=CEB31+C16
      FK(2,6)=CEB32+C26
      FK(3,6)=+CEB33+C36
      FK(5,4)=-WR+C54
      FK(6,4)=WQ+C64
      FK(4,5)=WR+C45
      FK(6,5)=-WP+C65
      FK(4,6)=-WQ+C46
      FK(5,6)=WP+C56
C*
C*CALLING THE SUBROUTINE FOR COMPUTATION OF THE STATE TRANSITION MATRIX
C***
      CALL CSTM(FK,PERCNT,DT,TMAT,B60,FKDT,SUME,SUME1,NXP,NTERMS)
      DO 600 I=1,NXP
      DO 600 J=1,NXP
  600 TMATT(J,I)=TMAT(I,J)
C***
      DO 701 I=1,NXP
      DO 701 J=1,NXP
  701 PMK2(I,J)=PMK1(I,J)
C***
```

```
      CALL LINV1F(PMK2,NXP,NXP,PINV,IDGT,WKAREA,IER1)
      CALL VMULFF(PPK,TMATT,NXP,NXP,NXP,NXP,NXP,PTMATT,NXP,IER2)
      CALL VMULFF(PTMATT,PINV,NXP,NXP,NXP,NXP,NXP,AK,NXP,IER3)
      DO 601 I=1,NXP
      DO 601 J=1,NXP
  601 AKT(J,I)=AK(I,J)
      DO 602 I=1,NXP
  602 DELX(I,1)=XS(I)-XMK1(I)
      DO 603 I=1,NXP
      DO 603 J=1,NXP
  603 DELP(I,J)=PS(I,J)-PMK1(I,J)
      CALL VMULFF(AK,DELX,NXP,NXP,1,NXP,NXP,DELXES,NXP,IER4)
      DO 604 I=1,NXP
  604 XS(I)=XPK(I)+DELXES(I,1)
      CALL VMULFF(AK,DELP,NXP,NXP,NXP,NXP,NXP,AKDELP,NXP,IER5)
      CALL VMULFF(AKDELP,AKT,NXP,NXP,NXP,NXP,NXP,ADPAT,NXP,IER6)
      DO 605 I=1,NXP
      DO 605 J=1,NXP
  605 PS(I,J)=PPK(I,J)+ADPAT(I,J)
C***
      IF(KSTEP1.LT.5)GOTO 253
      KSTEP1=0
      DO 251 I=1,NXP
      DO 252 J=1,NXP
      IF(I.LE.J)GOTO 252
      PS(I,J)=.5D0*(PS(I,J)+PS(J,I))
      PS(J,I)=PS(I,J)
  252 CONTINUE
  251 CONTINUE
  253 CONTINUE
C***
      DO 711 I=1,NXP
  711 XS1(I)=XS(I)
      DO 712 I=1,NXP
      DO 712 J=1,NXP
  712 PS1(I,J)=PS(I,J)
      KSTEPG=KSTEP
      XS1(7)=XPK12(7)
      XS1(8)=XPK12(8)
      XS1(9)=XPK12(9)
      DO 803 I=7,9
      DO 803 J=1,9
  803 PS1(I,J)=PPK12(I,J)
      DO 804 I=1,6
      DO 804 J=7,9
  804 PS1(I,J)=PPK12(I,J)
      XS1(10)=XPK12(10)
      XS1(11)=XPK12(11)
      WRITE(17'KSTEPG) (XS1(I),I=1,11)
     1,((PS1(I,J),I=1,9),J=1,9)
C***COPY K-AREAS INTO (K+1)-AREAS I.E. INTO 'PREVIOUS' AREA,
C***--GOING FROM END TO BEGINNING OF FILE.
      DO 631 I=1,NXP
      DO 631 J=1,NXP
  631 PMK1(I,J)=PMK(I,J)
```

```
      DO 632 I=1,NXP
  632 XMK1(I)=XMK(I)
C***
      GOTO 301
C***
C***
  311 CONTINUE
C***
C***
      GOTO 1
   92 CONTINUE
      STOP
      END
      BLOCK DATA
      IMPLICIT REAL*8(A-H,O-$)
      COMMON/DAT0/PERCNT,DT,TLAM0,OMEGA,AGLOBE,EPS2,IDGT,NSTEP,NXP,SOF
      LOGICAL SOF
      DATA SOF/ .FALSE. /,NSTEP/60/,NXP/6/
     1,PERCNT/.000001D0/,DT/1.0D0/
     2,IDGT/3/
     3,TLAM0/.700D0/
     4,OMEGA/.0000728D0/,AGLOBE/20940000.0D0/,EPS2/.0067D0/
      END
```

```
C***THIS PROGRAM CREATES A NON-DIRECT-ACCESS-DATA-FILE FOR XAS OR
C***XBS FOR OUTPUT OR FURTHER PROCESSING
C*
      DIMENSION J(15),Z(11)
     1    ,RDME(7),XDME(7),YDME(7),ZDME(7)
      COMMON/DAT0/SOF,J,NSTEP,N,NAMES,TLAM0,AGLOBE
     1   ,EPS2,SMEW0,ISTDME,XST(7),YST(7),ZST(7)
      INTEGER FILE(2)
      DIMENSION NAMES(38),NAMES1(38),XS(11),PS(9,9),SUM(14),STDZ(10)
     1       ,BIAS(11),SUM1(14),TEMP(14)
      LOGICAL GO,SOF
      NAMELIST/INP/FILE
      NAMELIST/OK/GO
      NAMELIST/DAT/SOF,J,NSTEP,N,TLAM0,AGLOBE
     1    ,EPS2,SMEW0,ISTDME,XST,YST,ZST
C***
      DEFINE FILE 10(60,368,L,KSTEP1)
C***
    1 CONTINUE
      PRINT 1002
 1002 FORMAT(1H ,'TYPE &INP FILE= &END')
      READ(5,INP)
      INF1=FILE(1)
      INF2=FILE(2)
      PRINT 2000,FILE
 2000 FORMAT(2I10)
      PRINT 1004
 1004 FORMAT(1H ,'TYPE &DAT DATA= &END')
      READ(INF1,DAT)
      IF(SOF)GOTO 92
      PRINT 1003
 1003 FORMAT(1H ,'IF EVERYTHING O.K. TYPE &OK GO=.TRUE.&END')
      GO=.TRUE.
      READ(5,OK)
      IF(GO)GOTO 10
      GOTO 1
   10 CONTINUE
C***
      DO 331 I=1,14
      SUM1(I)=0.
  331 SUM(I)=0.
C***
      DO 991 J1=1,NSTEP
      KSTEP1=J1
      READ(10'KSTEP1) (XS(I),I=1,11)
     1, ((PS(I,J7),I=1,9),J7=1,9)
C***
      COST=COS(TLAM0+XS(1)/AGLOBE)
      XS(2)=XS(2)/COST
C***
      DO 302 I=1,9
      DO 302 J8=1,9
      IF(PS(I,J8).LT.0.)PS(I,J8)=100.
  302 PS(I,J8)=SQRT(PS(I,J8))
C***
```

```
      WRITE(11) (XS(I),I=1,9)
C***
      READ(14) (Z(I3),I3=1,11)
      IF(NSTEP.GT.150)GOTO 923
      WRITE(12,903)J1,(XS(I),I=1,6),(PS(I,I),I=1,6)
  903 FORMAT(' 'I10,12F10.3)
      WRITE(12,905)J1,(XS(I),I=7,9),(PS(I,I),I=7,9)
  905 FORMAT(' ',I10,6F20.3)
  923 CONTINUE
C***NOT TO WASTE TDISK SPACE, WHILE ACTUALLY PROCESSING
C****
      USPD=XS(4)
      VSPD=XS(5)
      WSPD=XS(6)
      XE=XS(1)
      YE=XS(2)*COST
C***XS(2) HAS BEEN REDEFINED AT THE BEGINNING OF THIS PROGRAM.
      ZE=XS(3)
      VAIR=SQRT(USPD**2+VSPD**2+WSPD**2)
      ALPHA=57.3*ATAN2(WSPD,USPD)
      BETA=57.3*ATAN2(VSPD,USPD)
      HM=-ZE
C***
      IF(J1.EQ.1)PRINT 931,XS(1),XS(2)
  931 FORMAT(' ',2F10.0)
C***
      IF(NSTEP.GT.150)GOTO 921
C***
      TLAM=TLAM0+XE/AGLOBE
      COST=COS(TLAM)
      SINT=SIN(TLAM)
      SMEW=SMEW0+YE/AGLOBE/COST
      COSS=COS(SMEW)
      SINS=SIN(SMEW)
      C1111=1.-.5*EPS2*COS(2*TLAM)
      AC1111=AGLOBE*C1111
      DO 301 I=1,ISTDME
      XDME(I)=AC1111*COST*COSS-(-ZST(I)+AGLOBE*(1.-.5*EPS2*COS(2*XST(I))
     1))*COS(XST(I))*COS(YST(I))
      YDME(I)=AC1111*COST*SINS-(-ZST(I)+AGLOBE*(1.-.5*EPS2*COS(2*XST(I))
     1))*COS(XST(I))*SIN(YST(I))
      ZDME(I)=AC1111*SINT      -(-ZST(I)+AGLOBE*(1.-.5*EPS2*COS(2*XST(I))
     1))*SIN(XST(I))
      RDME(I)=SQRT(XDME(I)**2+YDME(I)**2+ZDME(I)**2)
C*
C*    RDME(I)=SQRT((XE-XST(I))**2+(YE-YST(I))**2+(ZE-ZST(I))**2)
C*
  301 CONTINUE
C***
      WRITE(15,912)J1,Z(4),VAIR,Z(5),ALPHA,Z(6),BETA,Z(7),HM
     1,Z(8),RDME(1),Z(9),RDME(2),XS(7),XS(8),(PS(I,I),I=1,6)
C*****PRINT 911,J1,Z(4),VAIR,Z(5),ALPHA,Z(6),BETA,Z(7),HM
C****1    ,Z(8),RDME(1),Z(9),RDME(2),XS(7),XS(8)
  911 FORMAT(I4,F4.0,F4.0,4F4.1,2F6.0,4F8.0,2F4.0)
  912 FORMAT(' ',I5,2F5.0,4F4.1,2F6.0,4F8.0,2F4.0,4X,3F6.0,4X,3F5.1)
```

```
C***
      TEMP( 1)=Z( 4)-VAIR
      TEMP( 2)=Z( 5)-ALPHA
      TEMP( 3)=Z( 6)-BETA
      TEMP( 4)=Z( 7)-HM
      TEMP( 9)=Z( 8)-RDME(1)
      TEMP(10)=Z( 9)-RDME(2)
      TEMP( 5)=XS(7)
      TEMP( 6)=XS(8)
      TEMP( 7)=XS(10)
      TEMP( 8)=XS(11)
      DO 951 I51=1,10
      SUM1(I51)=SUM1(I51)+TEMP(I51)
  951 SUM (I51)=SUM (I51)+TEMP(I51)**2
      GOTO 922
  921 CONTINUE
      WRITE(15,913)J1,Z(4),VAIR,Z(5),ALPHA,Z(6),BETA,Z(7),HM
     1,Z(8),XS(1),Z(9),XS(2),XS(7),XS(8)
  913 FORMAT(' ',I5,2F5.0,4F4.1,2F6.0,4F8.0,2F4.0,4X,18X,4X,15X)
C*****PRINT 911,J1,Z(4),VAIR,Z(5),ALPHA,Z(6),BETA,Z(7),HM
C****1    ,Z(8),XS(1),Z(9),XS(2),XS(7),XS(8)
  922 CONTINUE
C****
C*****PRINT 996,J1,(XS(I),I=1,6)
  996 FORMAT(' ',I7,6F10.3)
C*****PRINT 997,J1,(XS(I),I=7,9)
  997 FORMAT(' ',I7,3F20.3)
  991 CONTINUE
C***
      IF(NSTEP.GT.60)GOTO 953
      DO 952 I=1,8
      BIAS(I)=SUM1(I)/60
      TEMP2=SUM(I)/59-BIAS(I)**2
      IF(TEMP2.LT.0.)TEMP2=100.
  952 STDZ(I)=SQRT(TEMP2)
      BIAS( 9)=SUM1( 9)/60
      BIAS(10)=SUM1(10)/60
      TEMP3=SUM( 9)/59-(BIAS( 9)+BIAS(7))**2
      TEMP4=SUM(10)/59-(BIAS(10)+BIAS(8))**2
      IF(TEMP3.LT.0.)TEMP3=100.
      IF(TEMP4.LT.0.)TEMP4=100.
      STDZ( 9)=SQRT(TEMP3)
      STDZ(10)=SQRT(TEMP4)
      WRITE(15,914)(PS(I,I),I=1,6),(STDZ(I1),I1=1,4),STDZ(9),STDZ(10)
  914 FORMAT(' ',F10.0,2F10.1,3F10.0,4X,F10.0,2F10.1,3F10.0)
      PRINT 915,(BIAS(I),I=1,4),BIAS(9),BIAS(10),(STDZ(I1),I1=1,4)
     1         ,STDZ(9),STDZ(10)
  915 FORMAT(' ',F6.0,2F6.1,3F6.0,7X,F6.0,2F6.1,3F6.0)
  953 CONTINUE
      ENDFILE 12
      ENDFILE 15
C***
      PRINT 995,(PS(I,I),I=1,6)
  995 FORMAT(' ',6F10.3)
C*** IS THERE ANOTHER RUN TO BE GENERATED?
```

```
      GOTO 1
   92 CONTINUE
      STOP
      END
      BLOCK DATA
      COMMON/DAT0/SOF,J,NSTEP,N,NAMES,TLAM0,AGLOBE
     1    ,EPS2,SMEW0,ISTDME,XST(7),YST(7),ZST(7)
      DIMENSION J(15),NAMES(38)
      LOGICAL SOF
      DATA SOF/ .FALSE. /,J/38,2,23,5,26,6,27,14,24,15,25,29,18,2*0/,
     ANSTEP/60/,N/13/
     B,TLAM0/.700/,AGLOBE/20940000./
     C,ISTDME/2/,XST/.703,.6985,5*.706/,YST/-1.297,-1.2985,5*-1.294/
     D,ZST/7*0./,EPS2/.0067/,SMEW0/-1.300/
     1,NAMES/'DME1','CLDE',' Q  ',' NZ ','WHDA','PEDR','BETA',' P  '
     2,' R  ',' NY ',' NX ','ALFA','DME2','HADT','LEDF','TRDE','THET'
     3,' V  ','PSI ','PHI ','TRDR','TRDA','PODE','PODT','PODF','PODA'
     4,'PODR','VOR1','HBAR','ALS ','LMLS','GMLS','VOR2','INDF','HDIG'
     5,'LGHT','MDSW','TIME'/
      END
```

APPENDIX C

COMPUTER SYSTEMS FOR PREPROCESSING AND POST-FLIGHT DATA REDUCTION

Post-flight data handling begins using the HP 1000 digital computer located at Princeton University's Gas Dynamics Laboratory. The raw data is transferred to a 9-track, 1600 BPI magnetic tape that can be processed on either the IBM 4341 or the IBM 3033 computer. The following block-diagram summarizes the described procedure:

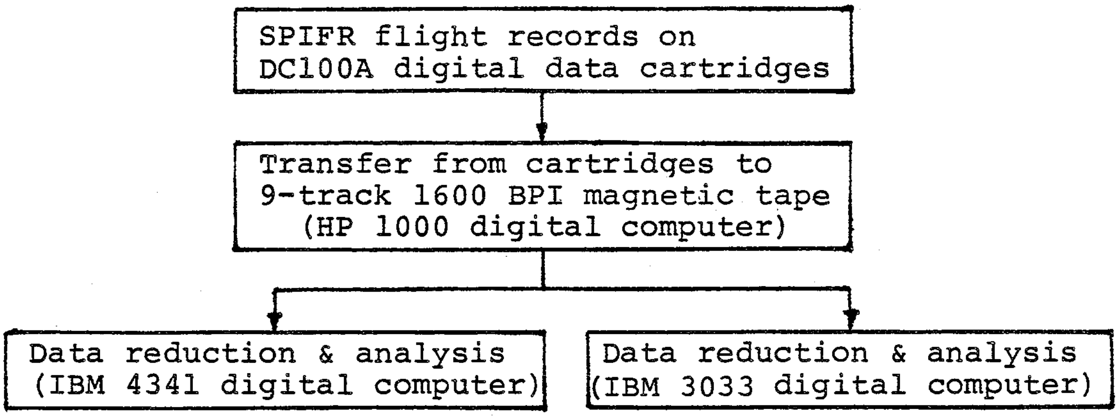


Figure C-1. Data Reduction Procedure.

The FORTRAN program CAT9 controls the transfer from the DC100A cartridges to the 9-track magnetic tape. The FORTRAN program RAWY1 converts 16-bit binary-formatted data into IBM-compatible decimal integer format and arranges the data in physical time vectors. The FORTRAN program SPIFY1 completes the preprocessing by converting the decimal integer time vectors into voltage and then into engineering units, also converting Indicated Air Speed (IAS) to True Air Speed (TAS).

The SPIFR data storage policy is to preserve both the raw flight-test data and the preprocessed data on magnetic tapes (9-track, 1600 BPI), which makes it compatible for further analysis on both the IBM 4341 and the IBM 3033 machines. Thus, two copies of the raw integer data (RAWY1 output file) and one copy with engineering unit time-vectors (SPIFY1 output file)-for further processing (analysis, tabular printouts or plotting) are preserved.

TRANSFER FROM CARTRIDGES TO TAPE

```
&CAT9  T=00004 IS ON CR00005 USING 00012 BLKS R=0000

FTN4,L
      PROGRAM CAT9(3,99), VERSION OF 4 JUNE 1981
C
C     PROGRAM TO COPY BINARY DATA FROM CASSETTE TO IBM COMPATIBLE
C     TAPE DRIVE.
C
C     LOADING THE PROGRAM
C        :RU,LOADR,*F4X,%CAT9
C
C     RUNNING THE PROGRAM
C        :RU,CAT9,P1,P2
C           WHERE P1 - IS THE LOGICAL UNIT NUMBER OF YOUR TERMINAL
C                 P2 - IS THE LOGICAL UNIT NUMBER OF THE MAG TAPE
C
C
      INTEGER IBUFF(128),IMORE,ISTAT,ITLOG,PARMS(5),NBLCK
      EQUIVALENCE (PARMS(1),LUCRT),(PARMS(2),MTLU)
      CALL RMPAR(PARMS)
      NBLCK=0
C***READ FROM LEFT CARTRIDGE LU 4

   21 CONTINUE
      CALL EXEC(1,100B+4,IBUFF,128)
C     GET STATUS
      CALL ABREG(ISTAT,ITLOG)
      WRITE(LUCRT,47)ITLOG
   47 FORMAT(I10)
C     CHECK FOR END OF FILE
      IF(IAND(ISTAT,200B) .EQ. 200B) GO TO 22
C     CHECK FOR END OF TAPE
      IF(IAND(ISTAT,40B) .EQ. 40B) GO TO 22
C     CHECK FOR END OF DATA
      IF(IAND(ISTAT,2).EQ.2)GO TO 22
C***WRITE TO TAPE
      CALL EXEC(2,100B+MTLU,IBUFF,128)
      NBLCK=NBLCK+1
      WRITE(LUCRT,31)NBLCK
   31 FORMAT(I7)
      GOTO 21
   22 CONTINUE
      IF(ITLOG.LT.128) GO TO 41
      CALL EXEC(2,100B+MTLU,IBUFF,128)
      NBLCK=NBLCK+1
      WRITE(LUCRT,31)NBLCK
   41   WRITE(LUCRT,23)
   23 FORMAT('PLUG IN NEXT CARTRIDGE AND TYPE 1 OR IF LAST-TYPE 0')
      READ(LUCRT,*) IMORE
      IF(IMORE.EQ.1)GOTO 21
C     WRITE TWO CONSECUTIVE END OF FILE MARKS
      CALL EXEC(3,0100B+MTLU)
      CALL EXEC(3,0100B+MTLU)
      STOP
```

PREPROCESSING - STEP I

```
FILE: RAWY1    FORTRAN  A1  PRINCETON UNIVERSITY TIME-SHARING SYSTEM


      COMMON/DAT0/SOF,NB
      LOGICAL GO,SOF
      NAMELIST/INP/FILE
      INTEGER*2 A3(1200),A2(1700,38)
      INTEGER*2 DATA(128)
      LOGICAL*1 DALOG(256)
      LOGICAL*1 SWLOG(256)
      INTEGER*2 DATA1(128)
      INTEGER*2 DATA2(128)
      EQUIVALENCE (DATA(1),DALOG(1)),(SWLOG(1),DATA1(1))
      INTEGER FILE(2)
      NAMELIST/OK/GO
      NAMELIST/DAT/SOF,NB
C***
    1 CONTINUE
      PRINT 1002
 1002 FORMAT(1H ,'TYPE &INP FILE= &END')
      READ(5,INP)
      INF1=FILE(1)
      INF2=FILE(2)
      PRINT 2000,FILE
 2000 FORMAT(2I10)
      PRINT 1004
 1004 FORMAT(1H ,'TYPE &DAT DATA= &END')
      READ(INF1,DAT)
      IF(SOF)GOTO 92
      PRINT 1003
 1003 FORMAT(1H ,'IF EVERYTHING O.K. TYPE &OK GO=.TRUE.&END')
      GO=.TRUE.
      READ(5,OK)
      IF(GO)GOTO 10
      GOTO 1
   10 CONTINUE
C***
      I1=0
      I2=0
C***
      READ(15,17)DATA
      DO 28 I=1,255,2
      SWLOG(I)=DALOG(I+1)
      SWLOG(I+1)=DALOG(I)
   28 CONTINUE
      GOTO 32
C***
   99 CONTINUE
      DO 30 I=1,128
   30 DATA2(I)=DATA1(I)
      READ(15,17,END=100)DATA
   17 FORMAT (128A2)
      DO 29 I=1,255,2
      SWLOG(I)=DALOG(I+1)
      SWLOG(I+1)=DALOG(I)
   29 CONTINUE
C***
      DO 31 I=1,128
```

```
      IF(DATA2(I).NE.DATA1(I))GOTO 32
   31 CONTINUE
      GOTO 99
   32 CONTINUE
C***
      I1=I1+1
      IF(I1.EQ.8)GOTO 417
      DO 421 M=1,128
      M3=M+(I1-1)*128
  421 A3(M3)=DATA1(M)
      GOTO 99
  417 DO 418 M=1,92
      M3=M+(I1-1)*128
  418 A3(M3)=DATA1(M)
C***M3=1-(128*8-36);I1=0-8;I2=NO. OF 2048-BYTE BLOCKS,
C***READ FROM TAPE INTO A3(EACH OVERWRITING THE PREVIOUS).
      I2=I2+1
      K1=26*(I2-1)+1
      K2=K1+25
C*K HAS A SPAN OF 26 AS 1024=38*26+36
C***
      J7=0
      DO 431 K=K1,K2
      IF(K.GT.NB)GOTO 437
      DO 432 J=1,38
  432 A2(K,J)=A3(J7+J)
      J7=J7+38
  431 CONTINUE
      I1=0
      GOTO 99
  100 CONTINUE
C***
      PRINT 441
  441 FORMAT(1H ,'EOT;DECREASE NB AND RERUN AS A NEW JOB')
C***
  437 CONTINUE
      KSOF=K-1
   27 FORMAT(1H ,I10)
      PRINT 27,KSOF
      DO 531 J=1,38
  531 WRITE(9)(A2(K0,J),K0=1,KSOF)
      ENDFILE 9
C***
      GOTO 1
   92 CONTINUE
      STOP
      END
      BLOCK DATA
      COMMON/DATO/SOF,NB
      LOGICAL SOF
      DATA SOF/ .FALSE. /,NB/256/
      END
```

PREPROCESSING - STEP II

```
FILE: SPIFY1   FORTRAN   A1   PRINCETON UNIVERSITY TIME-SHARING SYSTEM


      COMMON/DAT0/SOF,VSLOPE,VCONST,PHSLOP,PHCONS,N11,DPNSTD
      INTEGER FILE(2)
      DIMENSION A3(1700,38),PHSLOP(55),PHCONS(55)
      INTEGER*2 A2(1700,38)
      LOGICAL GO,SOF
      NAMELIST/INP/FILE
      NAMELIST/OK/GO
      NAMELIST/DAT/SOF,VSLOPE,VCONST,PHSLOP,PHCONS,N11,DPNSTD
    1 CONTINUE
      PRINT 1002
 1002 FORMAT(1H ,'TYPE &INP FILE= &END')
      READ(5,INP)
      INF1=FILE(1)
      INF2=FILE(2)
      PRINT 2000,FILE
 2000 FORMAT(2I10)
      PRINT 1004
 1004 FORMAT(1H ,'TYPE &DAT DATA= &END')
      READ(INF1,DAT)
      IF(SOF)GOTO 92
      PRINT 1003
 1003 FORMAT(1H ,'IF EVERYTHING O.K. TYPE &OK GO=.TRUE.&END')
      GO=.TRUE.
      READ(5,OK)
      IF(GO)GOTO 10
      GOTO 1
   10 CONTINUE
C***
      DO 501 J=1,38
  501 READ(9)(A2(I,J),I=1,N11)
C***NOW-INTO REAL PHYSICAL DATA.
      DO 512 I=1,N11
      ISIGN=0
      IF(A2(I,1).LT.0)ISIGN=1
C***65535=2**16-1,BECAUSE IF LEFTMOST OF THE 16-ZEROS-AND-ONES-FIELD
C***IS ONE,IT ITSELF IS INTERPRETTED AS MINUS AND EACH OF THE OTHER
C***15 BITS IS CHANGED(ONES TO ZEROS AND ZEROS TO ONES).
C***THUS,E.G.,A 16-ONES-FIELD IS INTERPRETTED AS -0 INSTEAD OF 2**16-1
C***AND A ONE FOLLOWED BY 15 ZEROS IS -(2**15-1) INSTEAD OF 2**15
  512 A3(I,1)=A2(I,1)+ISIGN*65535
      DO 502 I=1,N11
      DO 503 J=2,12
      ISIGN=0
      IF(A2(I,J).LT.0)ISIGN=1
  503 A3(I,J)=(((A2(I,J)+ISIGN*65535)*VSLOPE/16.+VCONST)*PHSLOP(J)+
     1PHCONS(J))
  502 CONTINUE
C***
      DO 521 I=1,N11
      ISIGN=0
      IF(A2(I,13).LT.0)ISIGN=1
  521 A3(I,13)=A2(I,13)+ISIGN*65535
      DO 522 I=1,N11
      DO 523 J=14,34
      ISIGN=0
```

```
      IF(A2(I,J).LT.0)ISIGN=1
  523 A3(I,J)=(((A2(I,J)+ISIGN*65535)*VSLOPE/16.+VCONST)*PHSLOP(J)+
     1PHCONS(J))
  522 CONTINUE
C***
      DO 514 I=1,N11
      DO 515 J=35,38
      ISIGN=0
      IF(A2(I,J).LT.0)ISIGN=1
  515 A3(I,J)=A2(I,J)+ISIGN*65535
  514 CONTINUE
C***
      DO 591 I=1,N11
  591 IF(A3(I,19).LT.0.)A3(I,19)=A3(I,19)+360.
C***
      DO 601 I=2,N11,2
  601 A3(I,1)=A3(I,13)
      DO 602 I=3,N11,2
  602 A3(I,1)=.5*(A3(I-1,1)+A3(I+1,1))
      A3(1,1)=A3(2,1)
      N111=N11-2
      DO 603 I=2,N111,2
  603 A3(I,13)=.5*(A3(I-1,13)+A3(I+1,13))
      A3(N11,13)=A3(N11-1,13)
C***
C***PRAT=PRATIO;RRAT=RRATIO
      DO 611 I=1,N11
      PRAT=(A3(I,29)+DPNSTD)/1013.3
      RRAT=PRAT**.81
      A3(I,18)=1.689*A3(I,18)/SQRT(RRAT)
      HCONST=EXP(ALOG(PRAT)/5.256)
      A3(I,29)=(1-HCONST)/.00000689
  611 CONTINUE
C***
C***NOT TO LOSE ACCURACY,THE TIME VECTORS ARE STORED UNFORMATTED,I.E.
C***USING UNFORMATTED READ(AND WRITE WHEN RETRIEVING FOR FURTHER
C***PROCESSING).
      DO 121 J=1,38
  121 WRITE(10) (A3(I,J),I=1,N11)
      ENDFILE 10
C***IS THERE ANOTHER RUN TO BE GENERATED?
      GOTO 1
   92 CONTINUE
      STOP
      END
      BLOCK DATA
      COMMON/DAT0/SOF,VSLOPE,VCONST,PHSLOP,PHCONS,N11,DPNSTD
      DIMENSION PHSLOP(55),PHCONS(55)
      LOGICAL SOF
      DATA SOF/ .FALSE. /,VSLOPE/.004884/,VCONST/-10./,
     1PHSLOP/1.,1.6583,-2.7604,-.20555,-8.2085,.2557,
     2-3.0754,4.0811,-3.4664,-.05184,.05519,2.8611,
     31.,.0508,.1020,-5.,3.1338,5.0623,
     418.2787,-8.1864,5.,-5.,2.4703,.0513,
     55.1310,-1.9589,2.4074,1.,15.275,2.8611,
```

```
     6.25,.10,1.,.1,1.,3*1.,
     717*1./,
     8PHCONS/0.,-3.2009,.458,-.02467,-.5304,.1055,
     9-.1019,-.0427,-.2048,-.0019,-.01233,13.7125,
     A0.,.492,-.0017,0.,.3805,99.9689,
     B.0984,-.4821,0.,0.,-4.0193,.513,
     C-23.8869,1.5698,3.5703,0.,950.,13.7125,
     D0.,0.,0.,0.,0.,0.,0.,0.,
     E17*0./,
     FN11/10/
     G,DPNSTL/0./
      END
```

APPENDIX D

INTEGRATION OF DISTANCE MEASURING EQUIPMENT (DME) INTO THE DATA COLLECTION SYSTEM

The DME component of the navigation/communication system has been integrated into the onboard experimental setup with the capability to sequence automatically available navigation stations and process the distance information using microprocessor control. The navigation/communication (NAV/COM) and the DME are part of the Bendix "BX-2000" product line of aircraft avionics. A digital information format is used in the Bendix NAV/COM and DME for frequency tuning. The DME receiver output to the pilot's indicator is a pulse-width signal which is compatible with digital processing techniques.

This appendix is sub-divided into sections relating to the external (microprocessor) tuning, distance signal decoding, and an overview of the DME system and specifications. The first two sections are specific to the Bendix system.

D-1. EXTERNAL DME TUNING

The Bendix DM-2031A DME receiver/transmitter has provisions for both "2 out of 5" tuning which is compatible with other manufacturers systems and a serial binary-coded-decimal (BCD) tuning. The serial tuning method is used by the Bendix NAV/COM and is implemented in the microprocessor tuning for compatibility. When the Bendix DME is installed with the Bendix NAV/COM, the DME serial tuning signal is the same one which is used for tuning the NAV receivers. As shown in Figure D-1, a switch located on the NAV/COM (Bendix CN-2011A) permits the pilot to select DME tuning paired with either NAV 1 or NAV 2. In the center-off or hold (H) position, no tuning signal is sent to the DME. Under this condition the DME continues to hold the last tuning selection and station frequency. The tuning signal contains a BCD format of the paired NAV frequency. (The NAV frequency is not the actual frequency used in the DME system, as will be explained in the overview section.)

The tuning signal is in the form of a twenty-bit asynchronous pulse-width modulated serial word. The serial data word format is shown in Figure D-2. The basic period of each word is 4.0 msec, and when supplied by the NAV/COM; the word rate is 250 Hz. However, a single word is sufficient to tune the DME. Note that the same format is used for the COM, NAV, DME and GS (glide slope) units in the Bendix product line. The first bit in the word is the synchronizing pulse. Each bit after the first is dedicated to a specific piece of information. The value of bits 2 through 7 is ignored in the current DME, but future units may use these bits as a device code.

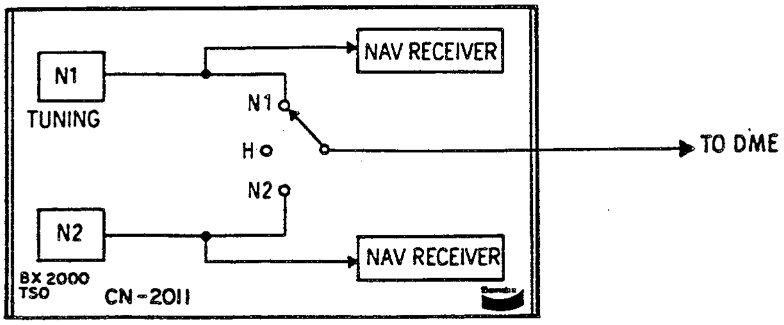
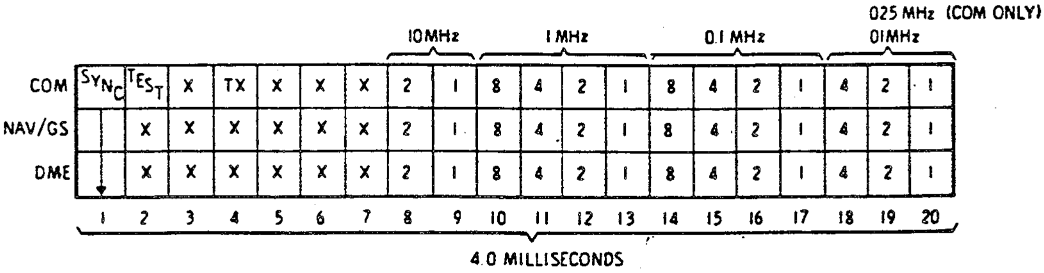


Figure D-1. DME Tuning Via NAV/COM.



								10MHz		1 MHz				0.1 MHz				025 MHz (COM ONLY) 01MHz		
COM	SYNC	TEST	X	TX	X	X	X	2	1	8	4	2	1	8	4	2	1	4	2	1
NAV/GS		X	X	X	X	X	X	2	1	8	4	2	1	8	4	2	1	4	2	1
DME		X	X	X	X	X	X	2	1	8	4	2	1	8	4	2	1	4	2	1
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

4.0 MILLISECONDS

Figure D-2. Serial Data Word Format.

The bit format is shown in Figure D-3. Syncronization, logic "1", and logic "0" bits correspond to 150-, 100- and 50-microsecond duration pulses respectively. The decoder inside the DME (as well as NAV, COM and GS) is relatively tolerant of the actual pulse width (and word length) of the incoming signal. As mentioned previously, the synchronizing pulse (bit 1) indicates the beginning of the serial data word. During the synchronizing pulse, the signal level stays at logic 1 for 150 microseconds (nominal). The Bendix circuitry samples each bit at 125 microseconds to determine if that bit is the synchronizing pulse. A similar sample is made at 75 microseconds to differentiate logic 1 and logic 0 pulses. Hence, the minor variation in the pulse widths of the tuning signal will not compromise the proper functioning of the system.

A microprocessor software program which generates the bit pulses and data word format to tune the DME was written using simple software timing loops. This program was verified using an oscilliscope to check the pulse widths and data word format. Software programming of the station sequencing was not completed in time for implementation on the test program. The alternative tuning method to be described latter is an interim solution.

Electrical (hardware) interfacing for microprocessor tuning output to the DME input is shown in Figure D-4. A signal inversion is employed at the NAV/COM's DME tuning signal output (this was not shown in Figure D-1 for clarity) and the signal is again inverted at the DME. Thus, the signals on the interconnecting wires are inverted with respect to Figure D-3. The high level (pull-up) voltage

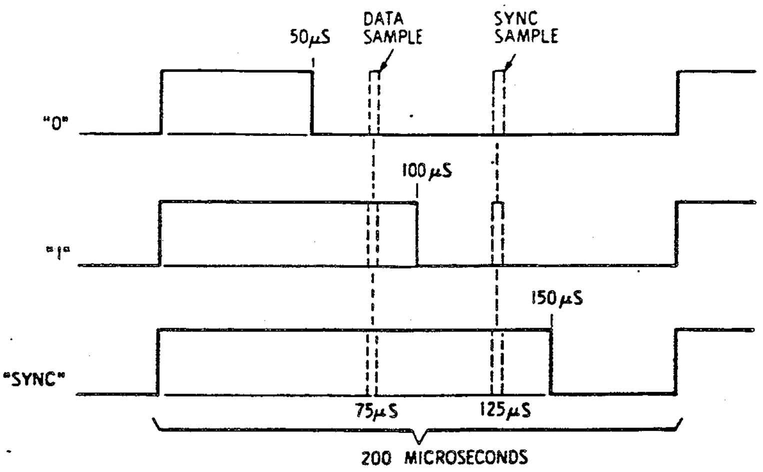


Figure D-3. Bit Format.

is 12 to 15 volts. An open collector buffer, preferrably with a 12 volt pull-up, may be used at the microprocessor side of the interface.

The alternative tuning method used in the current testing also is shown in Figure D-4. A switch located on the avionics section of the instrument panel allows the pilot to select normal NAV/COM (N) tuning or remote microprocessor (EXT) tuning. In the EXT position either the NAV 1 or NAV 2 tuning signal is routed to the DME, depending on the position of the relay shown. The relay is driven by a discrete digital output of the microprocessor. No changes in software logic were required for this implementation since the relay was driven in parallel with the "computer functioning" light on the instrument panel. The present rate of 0.5 Hz allows sufficient time for DME station lock-on and measurement of distance.

The selection switch N/EXT provides an additional function. In the EXT position, the displayed DME distance available on the one pilot's electronic course deviation instrument (ECDI) is blanked. The primary center panel DME indicator is not blanked, and the microprocessor station tuning of the DME can be verified by the safety pilot. The primary DME indicator can be switched by the safety pilot to display elapsed time or other function during flight tests.

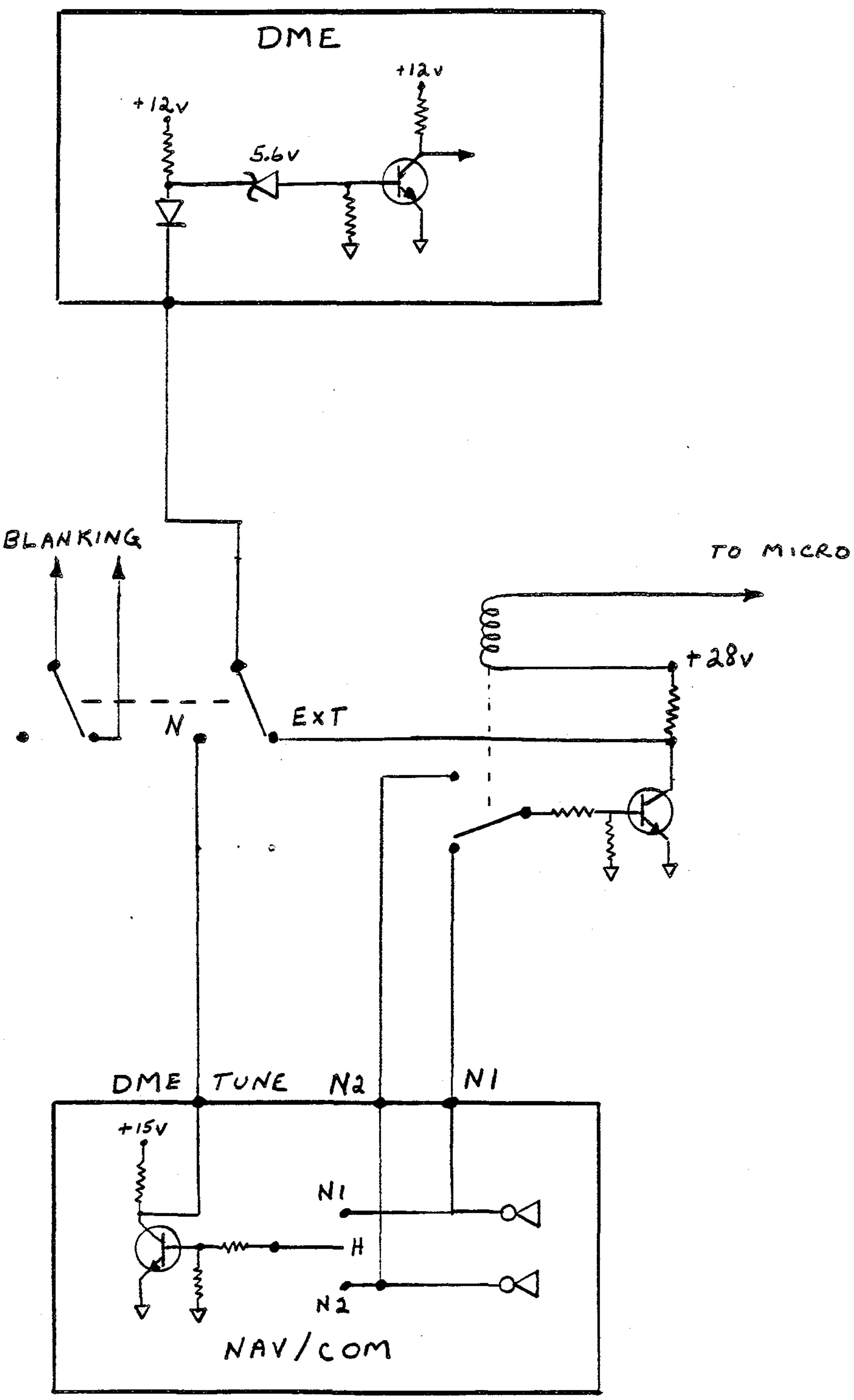


Figure D-4. DME Tuning Electrical Interface.

D-2. DME DISTANCE SIGNAL DECODING

Three signal outputs are generated by the Bendix DM-2031A DME receiver/transmitter: a pulse pair (RP1 and RP2) and a status signal (SEARCH). The time interval between RP1 and RP2 represents the slant range distance to the DME ground station. The digital logic interface, shown in Figure D-5, processes these three signals upon a DATA READ signal from the microprocessor. The distance represented the difference (RP2 - RP1) is presented to the microprocessor as a 16-bit (2-byte) word. The high-order bit of this data word is used to indicate the DME status (SEARCH).

The difference (RP2 - RP1) is measured by a 16-bit digital counter using a crystal controlled oscillator which operates at a frequency of 18 MHZ. Using the principle that RF energy travels one nautical mile and returns in 12.359 microseconds, the slant range from the aircraft to the ground station can be determined. Since the high-order bit is used for the status SEARCH signal, the maximum distance reading (15 bits) is 147 nautical miles. Although the interface clock frequency of 18 MHZ would suggest a measurement (counter) bit resolution of 27 feet, the actual resolution is determined by the processing within the Bendix DM-2031A. The LSI (large-scale-integration) chip that generates these pulses uses a 1.6 MHz clock (actually 1.61825 MHZ) which limits the (RP2 - RP1) difference increments to the equivalent of 0.05 nautical miles. Some other factors influencing measurement accuracy are discussed in the overview section.

The digital logic interface is presented in a simplified block diagram form in Figure D-6 for discussion of interface

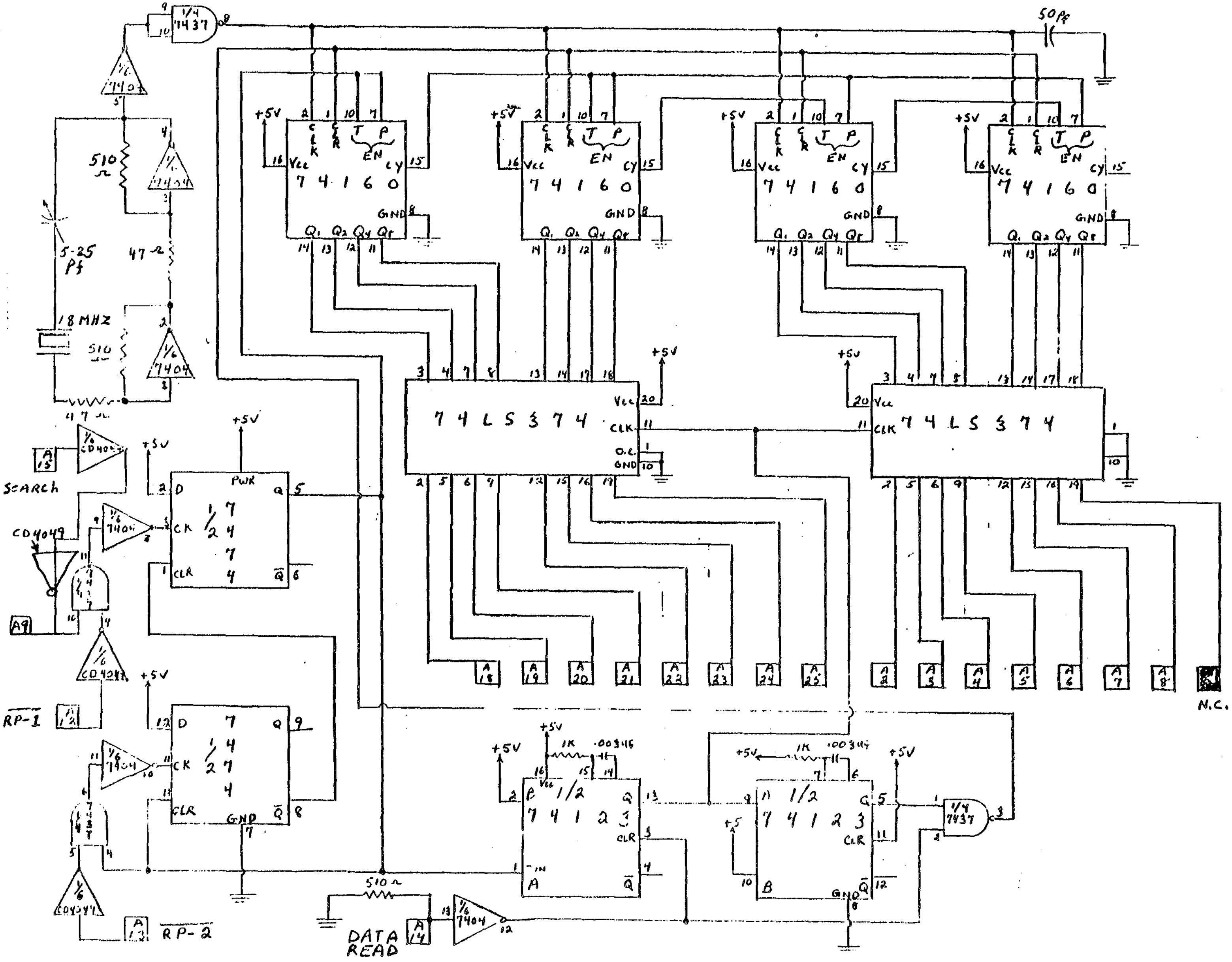


Figure D-5. DME - Microprocessor Electrical Interface.

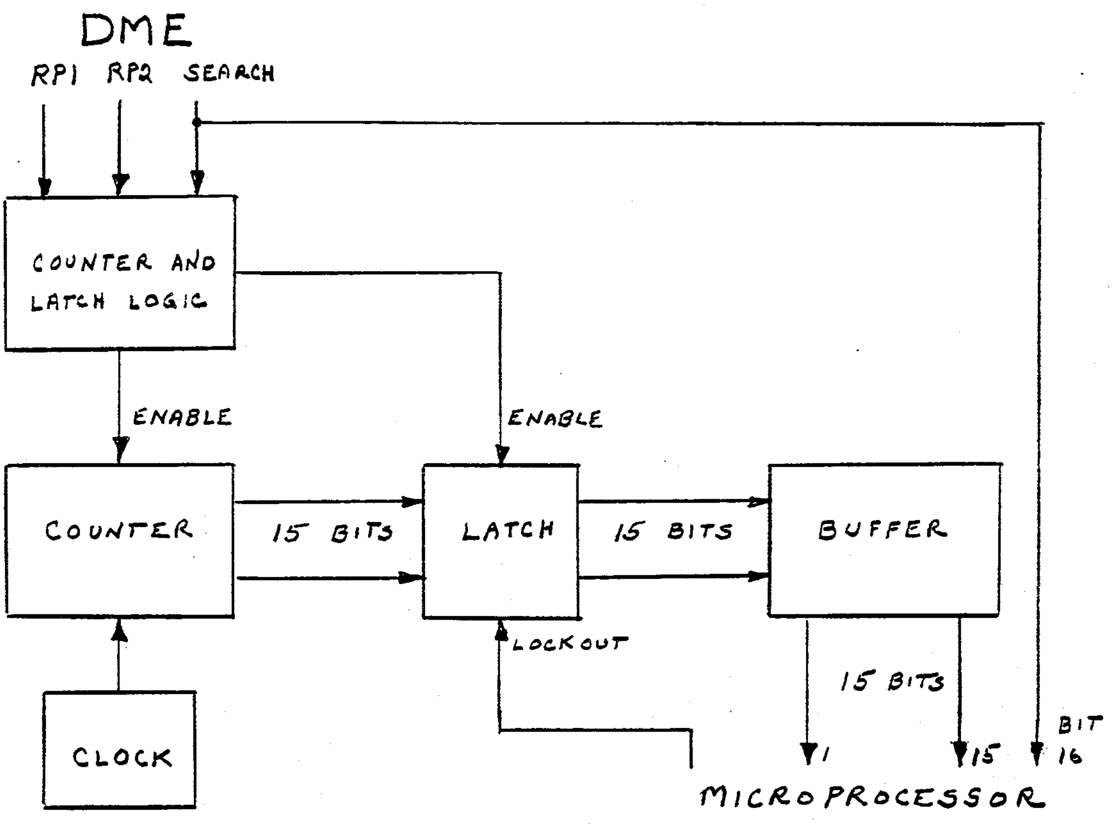


Figure D-6. DME Interface Block Diagram.

operation. The status of SEARCH is used to enable the counter as a precaution, although the absence of pulses RP1 and RP2 would preclude counter operation. The counter is started from a previously cleared (zero) value by the RP1 pulse from the DME. As noted previously, the counter rate is determined by the 18 MHZ referecne clock. The count is stopped by the RP2 pulse.

Two other events occur after receipt of RP2. After a very short delay, the count is transferred to the buffer via the latch control; when this operation is completed, the counter is reset or cleared. This chain of events continues to cycle as long as the signal DATA READ is not asserted by the microprocessor data collection system. Counter and buffer updates will take place at a 21 Hz rate during normal DME operation. When the microprocessor generates a DATA READ request, transfer of counter information to the buffer is inhibited. This signal is maintained by the microprocessor until the buffer has been read. This mode of operation guarantees that some data will be available so that the microprocessor will not "hang" in a wait state. The data in the buffer will normally be valid distance information measured within the last .05 sec of receipt of DATA READ. The signals RP1 and RP2 are not the raw pulses used by the DME interrogating a ground station; rather, they are generated by a sophisticated LSI chip. Corrections for delays in turnaround at the ground station and within the Bendix unit are applied so that the (RP2 - RP1) difference has no bias for true zero distance. Upon loss of the DME station, the (RP2 - RP1) pulses will continue to be sent by the LSI chip for up to 10 sec. A correction also is made to maintain the same rate of change (groundspeed) as observed prior to station signal loss. The correction is 80% of the preobserved groundspeed to prevent a "backing up" indication on the pilot's indicator when the signal is reacquired. The consequences of the above and other effects are discussed in the following overview section.

D-3. DME/DATA COLLECTION OVERVIEW

The purpose of the DME system is to provide the pilot with slant range distance information from the aircraft to a selected DME ground facility. The system transmits interrogation signals in the form of pulse pairs to the selected ground station. The DME ground facility receives the interrogation signal and returns a reply signal (again a pulse pair) for each interrogation received. Multiple aircraft may interrogate the DME ground station.

The DME system operates in the frequency range of 978 MHZ to 1212 MHz. There are 200 DME channels which are paired with VHF NAV frequencies between 108.00 MHz and 117.95 MHz (100 "X" channels and 100 "Y" channels). For example DME channel 85X is paired with NAV frequency 113.80 MHz. The aircraft transmits the interrogation pulses at 1109 MHZ and receives the reply offset by 63 MHz at 1172 MHz. (Some X channels are offset below the transmission frequency.) On the .05 spacing VHF frequencies such as 113.85 MHz (paired DME channel 85Y) the same transmission frequency is used but the reply is offset opposite to that used for the X channel (1046 MHz). Since some electrical processing delay will take place from receipt of interrogation signal to reply signal, all replys are adjusted to a specific delay to permit accurate measurement of the elapsed time by the airborne distance measuring circuits. This delay is 50 µsec for "X" stations and 56 µsec for Y stations (measured between the first pulse of interrogation to the first pulse of reply).

The interrogation pulse pairs are spaced at 12 µsec for X channels and 36 µsec for Y channels. The reply pulse pairs are 12 μsec and 30 μsec for X and Y respectively.

The DME ground facility continuously transmits a nominal 2700 pulse pairs per second squitter signal with a 1350 Hz identification morse code signal at 30-second intervals. The 1350 Hz identification signal consists of groups of evenly spaced pulse pairs. The ground station provides a reply pulse pair that replaces a squitter pulse pair 50 μsec after receiving an interrogation. The identification signal is available to the pilot as an audio tone to verify tuning and station selection.

When the DME is first tuned to a ground station, it must determine which reply pulses are to its interrogation pulses as opposed to those meant for other aircraft. Old model DME equipment frequently took 20 seconds or longer to achieve a lock-on and track. The Bendix DM-2031A specification for lock-on is less than 1 second. During the search period the interrogation rate is increased to 140 pulse pairs/sec to improve the detection time. This is reduced to 21 interrogations per sec during track. All DME units also use a variable pulse repetition rate to prevent synchronization of distance replies between other DME aircraft interrogators. A random jitter of the interrogation rate about the nominal of ± 1% is used in the Bendix DM-2031A.

The specification measurement accuracy of the Bendix DM-2031A is ± 0.1 nautical mile or .15 percent, whichever is larger. The minimum indication on the pilot's display of the Bendix DME system is 0.1 nautical mile. The output resolution of the signal to the indicator (RP1 - RP2) pulses is 0.05 nautical mile.

A possible source of error, both at the ground station and in the aircraft processing circuits, is proper pulse delay processing. The DME ground station error specification is ± 0.1 nautical miles indicating that the pulse delay (50 μsec on channel X) is within 1.2 μsec. This type of error, at a given ground station, and airborne unit should be predictable and could be removed from the data. Determination of this error is predicated on range measurement of multiple DME stations by the aircraft. A small dynamic error occurs with the data collection system since the measurement time may be in error by the update period (approximately 0.05 sec). In the implementation discussed here, an error due to signal loss is possible. Time difference information can continue up to 10 seconds after signal loss as mentioned previously. With the present scheme of sequencing stations every 2 seconds, the memory circuit is only partially charged, and it is unlikely that a memory generated signal will be obtained.

REFERENCES

1. Forsyth, D.L. and Shaughnessy, J.D., "Single Pilot IFR Operating Problems Determined from Accident Data Analysis", NASA TM 78773, September 1979.

2. Federal Aviation Regulations, Part 23, "Airworthiness Standards: Normal, Utility and Aerobatic Category Airplanes", Department of Transportation, Federal Aviation Administration, December 1969.

3. Sherman, W.L., "A Theoretical Analysis of Airplane Longitudinal Stability and Control as Affected by Wind Shear", NASA TN-D-8496, July 1979.

4. Ellis, D.R., "Flying Qualities of Small General Aviation Airplanes. Review of Recent In-Flight Simulation Experiments and Some Suggested Criteria", FAA-RD-71-118, December 1971.

5. Ellis, D.R. and Griffith, C.L., "A Study of Longitudinal Controllability and Stability Requirements for Small General Aviation Airplanes", FAA-RD-78-113, August 1978.

6. Loschke, P.C. et al., "Handling Qualities of Light Aircraft with Advanced Control Systems and Displays", in NASA Aircraft Safety and Operating Problems, Vol. I, NASA SP-270, May 1971.

7. Roscoe, A.H., ed., "Assessing Pilot Workload", AGARD-AG-233, February 1978.

8. Steinberger, J.M., "In-Flight Simulation of the General Aviation Aircraft Stall", Princeton University, MAE 1451T, August 1979.

9. Stengel, R.F., "Equilibrium Response of Flight Control Systems", Proceedings of the 1980 Joint Automatic Control Conference, San Francisco, August 1980.

10. Erzberger, H., "Analysis and Design of Model Following Control Systems by State Space Techniques", Proceedings of the 1968 Joint Automatic Control Conference, June 1968.

11. Anon., "Approval of Area Navigation Systems for Use in the U.S. National Airspace System", FAA Advisory Circular AC-90-45A, February 1975.

12. Klein V. and Shiess, J.R., "Compatibility Check of Measured Aircraft Responses Using Kinematic Equations and Extended Kalman Filtering", NASA TN D-8514, August 1977.

13. Bach, R.E., "A Variational Technique for Smoothing Flight-Test and Accident Data", AIAA 80-1601, August 1980.

14. Gelb, A. et al., "Applied Optimal Estimation", M.I.T. press, 1974.

15. Cooper, G.E. and Harper, R.P., "The Use of Pilot Rating in the Evaluation of A/C Handling Qualities", NASA TN D-5153, April 1969.

16. Sheridan, T.B., "Mental Workload in Decision and Control", IEEE Conference on Decision and Control, 1979.

1. Report No. NASA CR-165932	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle DEVELOPMENT OF FLYING QUALITIES CRITERIA FOR SINGLE-PILOT INSTRUMENT FLIGHT OPERATIONS: INTERIM REPORT	5. Report Date June 1982 6. Performing Organization Code
7. Author(s) Aharon Bar-Gill, W. Barry Nixon and George E. Miller	8. Performing Organization Report No. MAE-1528
9. Performing Organization Name and Address Princeton University Department of Mechanical & Aerospace Engineering Flight Research Laboratory Princeton, New Jersey 08544	10. Work Unit No. 11. Contract or Grant No. NAS1-15764
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546	13. Type of Report and Period Covered Contractor Report Sept.1979 to May 1981 14. Sponsoring Agency Code

15. Supplementary Notes

Langley Technical Monitor: Terrence S. Abbot
Interim Report

16. Abstract

Research is being conducted to develop flying qualities criteria for Single Pilot Instrument Flight Rule (SPIFR) operations. Significant progress has been made with regard to most of the key issues encompassed in the SPIFR research program. The ARA aircraft has been modified and adapted for SPIFR operations. Aircraft configurations to be flight-tested have been chosen and matched on the ARA in-flight simulator, implementing modern control theory algorithms. Mission planning and experimental matrix design have been completed. Microprocessor software for the onboard data acquisition system has been debugged and flight-tested. Flight-path reconstruction procedure and the associated FORTRAN program are at a final stage of development. Work has begun on algorithms associated with the statistical analysis of flight test results and the SPIFR flying qualities criteria deduction.

17. Key Words (Suggested by Author(s)) Flying qualities criteria;General Aviation; Instrument flight rule;In-flight simulation; Multiple DME scanning;Optimal smoothing; Optimal trajectory reconstruction;Phugoid response;Single pilot operations.	18. Distribution Statement UNCLASSIFIED-UNLIMITED

19. Security Classif. (of this report) UNCLASSIFIED	20. Security Classif. (of this page) UNCLASSIFIED	21. No. of Pages 165	22. Price

End of Document