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

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

CONTRACT NAS1-15764
June 1982

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

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

CONTRACT NAS1-15764
June 1982
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.
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.

Use of commercial products or names of manufacturers in this report does not constitute official endorsement of such products or manufacturers, either expressed or implied, by the National Aeronautics and Space Administration.
# Table of Contents

## Preface

1. Introduction
   1.1 Background and Goals
   1.2 Organization of the Report

2. Aircraft and Data Acquisition System Preparation
   2.1 Aircraft System Modifications
   2.2 Instrumentation and Data Recording System
   2.3 Software Development

3. Theoretical Aspects of Experiment Design and Flight Path Reconstruction
   3.1 Aircraft Dynamic Model
   3.2 Candidate SPIFR Configurations via the Output Command Algorithm
   3.3 Experimental Matrix Design
   3.4 Implementation of SPIFR Configurations via the Implicit-Model-Following Algorithm
   3.5 Effects of Navigational Accuracy and the "Learning Curve" Effect on Mission Planning
   3.6 Optimal Smoothing of Flight Test Records and Flight Path Reconstruction

4. Preliminary Flights

5. Conclusion
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>APPENDIX B: PROGRAM LISTINGS</td>
<td>54</td>
</tr>
<tr>
<td>APPENDIX C: COMPUTER SYSTEMS FOR PREPROCESSING AND POST-FLIGHT DATA REDUCTION</td>
<td>130</td>
</tr>
<tr>
<td>APPENDIX D: INTEGRATION OF DISTANCE MEASURING EQUIPMENT (DME) INTO THE DATA COLLECTION SYSTEM</td>
<td>138</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>153</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>Avionics Research Aircraft, Navion N5113K</td>
<td>4</td>
</tr>
<tr>
<td>2-2</td>
<td>Overview of the ARA In-Flight Simulator System</td>
<td>5</td>
</tr>
<tr>
<td>2-3</td>
<td>Cockpit Displays of the ARA. Modular SPIFR Evaluation Pilot Panel at Left</td>
<td>7</td>
</tr>
<tr>
<td>2-4</td>
<td>SPIFR Digital Data Recording System</td>
<td>9</td>
</tr>
<tr>
<td>3-1</td>
<td>Block Diagram for Implementation of a SPIFR Configuration on the ARA In-Flight Simulator</td>
<td>23</td>
</tr>
<tr>
<td>3-2</td>
<td>SPIFR Flight Path, Variant I</td>
<td>25</td>
</tr>
<tr>
<td>3-3</td>
<td>SPIFR Flight Path, Variant II</td>
<td>25</td>
</tr>
<tr>
<td>3-4</td>
<td>SPIFR Flight Path, Variant III</td>
<td>25</td>
</tr>
<tr>
<td>3-5</td>
<td>SPIFR Flight Path, Variant IV</td>
<td>25</td>
</tr>
<tr>
<td>3-6</td>
<td>Ground Station Engagement in the VOR/VOR or the DME/DME Modes</td>
<td>26</td>
</tr>
<tr>
<td>3-7</td>
<td>Examples of Application of the Optimal Flight Path Reconstruction Algorithm to the Climbing Turn Pseudo-Flight-Test Data</td>
<td>36</td>
</tr>
<tr>
<td>4-1</td>
<td>Knee-Pad Versions of the Performance and Workload PORs and of the Evaluation Sheet</td>
<td>41</td>
</tr>
<tr>
<td>C-1</td>
<td>Data Reduction Procedure</td>
<td>130</td>
</tr>
<tr>
<td>D-1</td>
<td>DME Tuning Via NAV/COM</td>
<td>140</td>
</tr>
<tr>
<td>D-2</td>
<td>Serial Data Word Format</td>
<td>140</td>
</tr>
<tr>
<td>D-3</td>
<td>Bit Format</td>
<td>142</td>
</tr>
<tr>
<td>D-4</td>
<td>DME Tuning Electrical Interface</td>
<td>144</td>
</tr>
<tr>
<td>D-5</td>
<td>DME - Microprocessor Electrical Interface</td>
<td>146</td>
</tr>
<tr>
<td>D-6</td>
<td>DME Interface Block Diagram</td>
<td>147</td>
</tr>
<tr>
<td>No.</td>
<td>Table Description</td>
<td>Page</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>2-1</td>
<td>Input Assignments for SPIFR Digital Data Recording System</td>
<td>10</td>
</tr>
<tr>
<td>2-2</td>
<td>Output Assignments for SPIFR Digital Data Recording System</td>
<td>11</td>
</tr>
<tr>
<td>3-1</td>
<td>Experimental Matrix for First SPIFR Flight-Test Series</td>
<td>21</td>
</tr>
</tbody>
</table>
# LIST OF SYMBOLS

<table>
<thead>
<tr>
<th>Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbf{a}_B$</td>
<td>acceleration vector in body axes, &quot;g&quot;</td>
</tr>
<tr>
<td>$a_x, a_y, a_z$</td>
<td>cartesian components of $\mathbf{a}_B$, &quot;g&quot;</td>
</tr>
<tr>
<td>$C$</td>
<td>implicit model following gain matrix</td>
</tr>
<tr>
<td>$F$</td>
<td>system dynamics matrix</td>
</tr>
<tr>
<td>$f$</td>
<td>nonlinear functions for vehicle equations of motion</td>
</tr>
<tr>
<td>$G$</td>
<td>control effects matrix</td>
</tr>
<tr>
<td>$g$</td>
<td>gravitational acceleration, ft/sec$^2$</td>
</tr>
<tr>
<td>$H$</td>
<td>observation matrix</td>
</tr>
<tr>
<td>$h$</td>
<td>transformation matrix</td>
</tr>
<tr>
<td>$h$</td>
<td>altitude, ft</td>
</tr>
<tr>
<td>$h$</td>
<td>nonlinear measurement functions</td>
</tr>
<tr>
<td>$I$</td>
<td>identity matrix</td>
</tr>
<tr>
<td>$K$</td>
<td>Kalman gain matrix</td>
</tr>
<tr>
<td>$L$</td>
<td>transformation matrix</td>
</tr>
<tr>
<td>$M(\cdot)$</td>
<td>pitch moment stability-and-control derivative</td>
</tr>
<tr>
<td>$n(\cdot)$</td>
<td>random noise associated with the (\cdot) variable</td>
</tr>
<tr>
<td>$P$</td>
<td>state covariance matrix</td>
</tr>
<tr>
<td>$p$</td>
<td>roll rate, deg/sec</td>
</tr>
<tr>
<td>$Q$</td>
<td>process noise covariance matrix</td>
</tr>
<tr>
<td>$q$</td>
<td>pitch rate, deg/sec</td>
</tr>
<tr>
<td>$R$</td>
<td>measurement noise covariance matrix</td>
</tr>
<tr>
<td>$r$</td>
<td>yaw rate, deg/sec</td>
</tr>
<tr>
<td>$S$</td>
<td>intermediate command output matrix</td>
</tr>
<tr>
<td>$s$</td>
<td>Laplace transform variable</td>
</tr>
<tr>
<td>$T$</td>
<td>duration of flight segment to be reconstructed, sec</td>
</tr>
</tbody>
</table>
t  time, sec
u  x-axis velocity, ft/sec
\vec{u}  control vector
V_{\text{air}}  airspeed vector, ft/sec
v  y-axis velocity, ft/sec
\vec{v}  measurement noise
w  z-axis velocity, ft/sec
\vec{w}  process noise vector
W_W  wind vector in inertial frame, ft/sec
X( )  (aerodynamic + thrust)-force along the x-axis derivative
x_I  axial position in inertial frame, ft
X  state vector
y_I  lateral position in inertial frame, ft
\vec{y}  command vector
Z( )  (aerodynamic + thrust)-force along the z-axis derivative
z_I  vertical position in inertial frame, ft
\vec{z}  observation vector

Variables (Greek)

\alpha  angle of attack, deg
\beta  angle of sideslip, deg
\delta E  elevator deflection, deg
\delta F  flap deflection, deg
\delta T  throttle deflection, percent
\Theta  pitch attitude angle, deg
\Sigma  summation
\Phi  state transition matrix
\phi  roll attitude angle, deg
\psi  yaw attitude angle, deg
\omega  body angular rate skew matrix
\vec{\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

(·) derivative of quantity with respect to time
( _) vector quantity
δ( )/δ( ) partial derivative of one variable with respect to another
Δ( )  perturbation variable
( )\*  steady-state variable
( )^T  transpose of a vector or matrix
( )^{-1}  inverse of a matrix
( )^#  estimated value of a variable
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 ($\alpha$, $\beta$), 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 (pseudo-randomly) 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 mathema-
tics processor in a Multibus™ architecture. As currently con-
figured, the SPIFR microcomputer contains 48K bytes of RAM (ran-
dom 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 pro-
cessing 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>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital Inputs</td>
<td>1. DME Distance</td>
<td>2. VOR#1 Frequency</td>
<td>3. VOR#2 Frequency</td>
<td>4. DME Frequency</td>
<td>5. Time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>----------------</td>
<td>----------------</td>
<td>-------------</td>
<td>----------------</td>
<td>----------------</td>
<td>--------------</td>
<td>----------------------</td>
<td>----------------------</td>
<td>----------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>-------------------------</td>
<td>---------------------</td>
<td>----------------------</td>
<td>-------------------</td>
<td>-----------------</td>
<td>-------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>----------------</td>
<td>----------------</td>
<td>-------------</td>
<td>----------------</td>
<td>----------------</td>
<td>--------------</td>
<td>----------------------</td>
<td>----------------------</td>
<td>----------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>-------------------------</td>
<td>---------------------</td>
<td>----------------------</td>
<td>-------------------</td>
<td>-----------------</td>
<td>-------------------</td>
</tr>
</tbody>
</table>
TABLE 2-2
Output Assignments For
SPIFR Digital Data Recording System

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

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

\[ \dot{x} = f(x, u) \]  

(3-1)

where \( x \) is the state vector and \( u \) is the control vector. The state vector \( x \) 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 \((\phi, \theta, \psi)\). 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:

\[ \dot{V}_{air} = a_B + \dot{\omega}V_{air} + H^B_{II} \]  

(3-2)
The airspeed, expressed in body axes, is:

\[ V_{\text{air}} = [u \ v \ w]^T \]  

(3-3)

Acceleration, expressed in body axes, is:

\[ a_B = [a_x \ a_y \ a_z]^T \]  

(3-4)

The angular rate cross-product-equivalent matrix \( \dot{\omega} \) is defined as:

\[ \dot{\omega} = \begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix} \]  

(3-5)

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

\[ g_I = [0 \ 0 \ g]^T \]  

(3-6)

The transformation matrix \( H^B_I \) from inertial (I) to body (B) axes, with \([\psi \ \theta \ \phi]\) Euler rotations in the specified order, is:

\[ H^B_I = \begin{bmatrix} c\psi c\theta & s\psi c\theta & -s\theta \\ c\psi s\phi - s\psi c\phi & s\psi s\phi c\theta + c\psi c\phi & c\phi s\theta \\ c\psi s\phi + s\psi c\phi & s\psi c\phi - c\psi s\phi & c\phi c\theta \end{bmatrix} \]  

(3-7)

where
\[ s(\ ) \triangleq \sin(\ ) \quad \text{(3-8)} \]
\[ c(\ ) \triangleq \cos(\ ) \]

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

\[
\begin{bmatrix}
\dot{\phi} \\
\dot{\theta} \\
\dot{\psi}
\end{bmatrix} = \mathbf{L}_B^I \begin{bmatrix} p \\ q \\ r \end{bmatrix} \quad \text{(3-9)}
\]

where

\[
\mathbf{L}_B^I \triangleq \frac{1}{c\theta} \begin{bmatrix} c\theta & s\theta \phi & -s\phi \\ 0 & c\phi & -c\theta \phi \\ 0 & s\phi & c\phi \end{bmatrix}
\quad \text{(3-10)}
\]

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

\[
\dot{\mathbf{x}}_I = \mathbf{H}_B^I V_{\text{air}} + \mathbf{W}_I 
\quad \text{(3-11)}
\]

where \( \mathbf{x}_I \) is the position vector expressed in inertial axes:

\[
\mathbf{x}_I \triangleq [x_I \ y_I \ z_I]^T
\quad \text{(3-12)}
\]

Based on the orthonormality of \( \mathbf{H}_I^B \) in eq. (3-7):

\[
\mathbf{H}_B^I = (\mathbf{H}_I^B)^{-1} = (\mathbf{H}_I^B)^T
\quad \text{(3-13)}
\]

16
The following relationships constitute the algebraic part of the model, yielding the output or the measurement models. The airspeed absolute value is:

\[ |V_{\text{air}}| = (u^2 + v^2 + w^2)^{1/2} \]  

(3-14)

The angle of attack is given by:

\[ \alpha = \tan^{-1}\left(\frac{w}{u}\right) \]  

(3-15)

The sideslip angle is:

\[ \beta = \tan^{-1}\left(\frac{v}{u}\right) \]  

(3-16)

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:

\[ h = -z_I \]  

(3-17)

The acceleration vector \( \dot{a}_B \) 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:

\[
\begin{bmatrix}
\Delta u \\
\Delta w \\
\Delta q
\end{bmatrix}
= \begin{bmatrix} X_u & X_w & X_q \\ Z_u & Z_w & Z_q \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta w \\ \Delta q \end{bmatrix}
+ \begin{bmatrix} X_{\delta E} & X_{\delta T} & X_{\delta F} \\ Z_{\delta E} & Z_{\delta T} & Z_{\delta F} \end{bmatrix} \begin{bmatrix} \Delta \delta E \\ \Delta \delta T \\ \Delta \delta F \end{bmatrix}
\]  

(3-18)

The control vector here is:

\[
\begin{bmatrix}
\Delta u \\
\Delta w \\
\Delta q
\end{bmatrix}
\]

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

\[ \beta = \sin^{-1}\left(\frac{v}{|V_{\text{air}}|}\right) \]
\[ u = [\Delta \delta E \, \Delta \delta T \, \Delta \delta F]^T \]  

(3-19)

where \( \Delta \delta E \) is the elevator deflection, \( \Delta \delta T \) is the throttle travel and \( \Delta \delta F \) 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:

\[ \Delta q = M_u \Delta u + M_w \Delta w + M_q \Delta q + M_{\delta E} \Delta \delta E + M_{\delta T} \Delta \delta T + M_{\delta F} \Delta \delta F \]  

(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:

\[
\begin{bmatrix}
\Delta u \\
\Delta w \\
\Delta q \\
\Delta \theta
\end{bmatrix} =
\begin{bmatrix}
X_u & X_w & -w+X_q & -g\Theta \\
Z_u & Z_w & u+Z_q & -g\Theta \\
M_u & M_w & M_q & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
\Delta u \\
\Delta w \\
\Delta q \\
\Delta \theta
\end{bmatrix}
+
\begin{bmatrix}
X_{\delta E} & X_{\delta T} & X_{\delta F} \\
Z_{\delta E} & Z_{\delta T} & Z_{\delta F} \\
M_{\delta E} & M_{\delta T} & M_{\delta F} \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\Delta \delta E \\
\Delta \delta T \\
\Delta \delta q \\
\Delta \delta F
\end{bmatrix}
\]  

(3-21)

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

\[ \dot{x} = F \Delta x + G \Delta u \]  

(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 $\Delta V^*$ or in flight path angle $\Delta \gamma^*$.

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

$$\Delta y = H_x \Delta x + H_u \Delta u$$

(3-23)

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

$$0 = F \Delta x^* + G \Delta u^*$$

$$\Delta y_{\text{comm}} = H_x \Delta x^* + H_u \Delta u^*$$

(3-24)

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

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.
\[ \Delta x^* = -F^{-1}S \Delta y_{comm} \]  
(3-25)

\[ \Delta u^* = S \Delta y_{comm} \]  
(3-26)

where:

\[ S = \Delta \left(-H_x F^{-1}G + H_u\right)^{-1} \]  
(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) \( X_u, Z_u, Z_a, M_u \)

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 \( \Delta x^* \) and \( \Delta u^* \). 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.

<table>
<thead>
<tr>
<th>Parameter to be varied</th>
<th>Test pilot 1</th>
<th>Test pilot 2</th>
<th>GA pilot</th>
<th>Number of mission runs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nominal</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>$X_u$</td>
<td>2±</td>
<td>2±</td>
<td>3++</td>
<td>7</td>
</tr>
<tr>
<td>$Z_u$</td>
<td>2±</td>
<td>2±</td>
<td>3++</td>
<td>7</td>
</tr>
<tr>
<td>$M_u$</td>
<td>2±</td>
<td>2±</td>
<td>3++</td>
<td>7</td>
</tr>
<tr>
<td>$Z_{\alpha}$</td>
<td>2±</td>
<td>2±</td>
<td>3++</td>
<td>7</td>
</tr>
<tr>
<td>$M_{\delta T}$</td>
<td>2±</td>
<td>2±</td>
<td>3++</td>
<td>7</td>
</tr>
<tr>
<td>$Z_{\delta E}$</td>
<td>2±</td>
<td>2±</td>
<td>2±</td>
<td>6</td>
</tr>
<tr>
<td>$Z_{\delta T}$</td>
<td>2±</td>
<td>2±</td>
<td>2±</td>
<td>6</td>
</tr>
</tbody>
</table>

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),

$$\dot{x}_{ARA} = F_{ARA}x_{ARA} + G_{ARA}u_{ARA}$$

(3-28)

$$\dot{x}_M = F_Mx_M + G_Mu_M$$

(3-29)

Our objective is to obtain the control vector $u_{ARA}'$, which will make the ARA respond as the required configuration. The perfect model following condition is:
\[ \Delta x_{\text{ARA}} = \Delta x_{\text{M}} \] (3-30)

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

\[ \Delta u_{\text{ARA}} = G_{\text{ARA}}^\# [(F_{\text{M}} - F_{\text{ARA}}) \Delta x_{\text{ARA}} + G_{\text{M}} \Delta u_{\text{M}}] \]
\[ = C_B \Delta x_{\text{ARA}} + C_F \Delta u_{\text{M}} \] (3-31)

where:

\[ G_{\text{ARA}}^\# \triangleq [G_{\text{ARA}}^T G_{\text{ARA}}]^{-1} G_{\text{ARA}}^T \] (3-32)

Eq. (3-30) renders:

\[ \Delta x_{\text{ARA}} = \Delta x_{\text{M}} \] (3-33)

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

\[ \Delta u_{\text{M}} \]
\[ \text{PILOT} \]
\[ C_F \]
\[ \Delta u_{\text{ARA}} \]
\[ G_{\text{ARA}} \]
\[ \frac{1}{L} \]
\[ C_B \]
\[ F_{\text{ARA}} \]
\[ \Delta x_{\text{ARA}} \]
\[ \Delta x_{\text{M}} = \Delta x_{\text{ARA}} \]

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.
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.

\[
\sigma_\psi \equiv \sigma_{\text{VOR}} = 1.9^\circ \\
\sigma_R = \sigma_{\text{DME}} = 0.15\% \text{ range or } 0.1 \text{ mile: whichever is larger}
\]  

(3-34)

These navigational errors are with respect to a single ground station. The position errors associated with a navigational mode 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 and assuming that the two navigational-
stations' errors are statistically uncorrelated, we obtain:

\[
\sigma = \frac{1}{s\theta} \left[ \sigma_1^2 + \sigma_2^2 \right]^{1/2}
\]

(3-35)

The \(1/s\theta\) term reflects the GDOP effect. For angles between radials in the vicinity of \(\theta = 0^\circ\) or \(\theta = 180^\circ\), 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,

\[
\sigma = \frac{a\sigma}{\sqrt{2s^2\psi}} (1+s^2\theta)^{1/2}
\]

(3-36)

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

\[
\sigma = \sqrt{2} \sigma^*_R/s\theta
\]

(3-37)

Numerical values of eq. (3-34) and dependence of \(\sigma_{\text{VOR/VOR}}\) 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 \(\theta = 30^\circ\) 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),

\[ \dot{x} = f(x,u) + w \]  \hspace{1cm} (3-38)

and the measurement equation,

\[ z = h(x,u) + v \]  \hspace{1cm} (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 w, which is referred to in the literature as "process noise". The vector 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 \( a_B \) are measured directly. Equations (3-2) to (3-17) constitute a kinematic model, as they do not reflect the dynamic mechanism by which \( a_B \) 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 $a_B$ -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:

\[
\begin{align*}
   \dot{p} &= n_p^* \\
   \dot{q} &= n_q^* \\
   \dot{r} &= n_r^* \\
   \dot{\phi} &= p + \tan \theta (s\phi q + c\phi r) \\
   \dot{\theta} &= c\phi q - s\phi r \\
   \dot{\psi} &= \frac{1}{c\phi} (s\phi q + c\phi r)
\end{align*}
\]  
(3-40)

The state vector for model A is:

\[
x_A = [p \ q \ r \ \phi \ \theta \ \psi]^T
\]  
(3-42)

The process noise vector (with $n_p^*$, $n_q^*$ and $n_r^*$ random and unknown angular acceleration inputs) is:

\[
w_A = [n_p^* \ n_q^* \ n_r^* \ 0 \ 0 \ 0]^T
\]  
(3-43)
The measurement noise vector for model A is:

\[ \mathbf{v}_A = [n_p \ n_q \ n_r \ n_\phi \ n_\theta \ n_\psi]^T \]  \hspace{1cm} (3-44)

The measurement vector \( \mathbf{z}_A \) in (3-41) contains the measured values of \( \mathbf{x}_A \). Thus, the observation matrix \( \mathbf{H}_A \) 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,

\[ \hat{p}(t), \hat{q}(t), \hat{r}(t); \hat{\phi}(t), \hat{\theta}(t), \hat{\psi}(t) \]  \hspace{1cm} (3-45)

and the associated matrices,

\[ \hat{\mathbf{H}}_B(t), \hat{\mathbf{H}}_I(t), \hat{\mathbf{w}}(t) \]  \hspace{1cm} (3-46)

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

\[ \mathbf{x}_B = [\mathbf{x}_I \ \mathbf{y}_I \ \mathbf{z}_I \ \mathbf{u} \ \mathbf{v} \ \mathbf{w}]^T \]  \hspace{1cm} (3-47)

and the 6-component input vector is,

\[ \mathbf{u}_B = [w_x \ w_y \ w_z \ (a_x - s\theta g)(a_y + c\theta \phi g)(a_x + c\phi g)]^T \]  \hspace{1cm} (3-48)

The input vector contains components of the true wind and accelerations, \( w_I \) and \( a_B \). 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.
Unlike the first step, in which the $x_A$ components have been smoothed optimally, this step reconstructs the $x_B$ components with eq. (3-2) and (3-11) constituting the state model B:

$$w_B = [0 0 0 n_x n_y n_z]^T$$

(3-49)

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

$$\begin{align*}
|V_{\text{air}}| &= (u^2 + v^2 + w^2)^{1/2} + n_v \\
\alpha &= \tan^{-1}\left(\frac{w}{u}\right) + n_\alpha \\
\beta &= \tan^{-1}\left(\frac{v}{u}\right) + n_\beta \\
h &= -2z + n_h \\
x_{\text{DME}, Si} &= \left[(x_i - x_{Si})^2 + (y_i - y_{Si})^2 + (z_i - z_{Si})^2\right]^{1/2} + n_{\text{DME}} \\
\theta_{\text{VOR}, Si} &= \tan^{-1}\left(\frac{y_i - y_{Si}}{x_i - x_{Si}}\right) + n_{\text{VOR}}
\end{align*}$$

(3-51)
where $s_i$ symbolizes the distance $r_{DMEi}$ or angle $\theta_{VORi}$ being measured with respect to navigational station $i$. The measurement noise vector is:

$$v_B = [n_v \ n_\alpha \ n_\beta \ n_h \ n_{DME1} \ n_{DME2} \ \ldots \ n_{VOR1} \ n_{VOR2} \ \ldots]^T \quad (3-52)$$

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 $\hat{x}$ and of the state covariance matrix $P$ between measurements, for forward filtering uses

$$\dot{x}(t) = f[\hat{x}(t), u(t)] \quad (3-53)$$
\[ P_k(-) = \phi_k^{-1} P_{k-1}(+) \phi_k^{-1} T + Q_{k-1} \] (3-54)

where \( Q \) is the process noise covariance matrix and \( \phi \) is the transition matrix obtained after local linearization of eq. (3-38) into:

\[ \dot{\mathbf{x}} = F \mathbf{x} + G \mathbf{u} + L \mathbf{w} \] (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 forward filtering is,

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

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

\[ z_k = H_k \mathbf{x} + \nu_k \] (3-57)

State and covariance updates account for measurements as

\[ \hat{\mathbf{x}}_k(+) = \hat{\mathbf{x}}(t) + K_k [z_k - H_k \hat{\mathbf{x}}(t)] \] (3-58)

\[ P_k(+) = [I - K_k H_k] P_k(-) \] (3-59)

where \( \hat{\mathbf{x}}(t) \) 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 $\hat{x}_k(-) = \hat{x}(t)$ and after the measurement update $\hat{x}_k(+)$. The smoother algorithm uses this information as input and running backwards in time produces the improved estimates of the states $\hat{x}_{k/n}$ and of the covariance matrix $P_{k/n}$. The first step is the computation of the state matrix $F_k$:

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

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

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

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

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

$$P_{k/n} = P_k(+) + A_k[P_{k+1/n} - P_{k+1}(-)]A_k^T$$ (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.
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 reconstruction algorithm to the climbing turn pseudo-flight-test data.
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 reconstruction algorithm to the climbing turn pseudo-flight-test data.
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 acquisition 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
a) Performance POR Scale

Figure 4-1: Knee-pad Versions of the Performance and Workload PORs and of the Evaluation Sheet.
### Workload Rating Scale

<table>
<thead>
<tr>
<th>WORKLOAD CATEGORY</th>
<th>DESCRIPTION OF CATEGORY</th>
<th>RATING</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Low levels of workload, such that all tasks are accomplished prompt-ly. Idle periods exist between tasks.</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Moderate levels of workload indicate that probability of error or omission is low, but improvements are desirable.</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Very high levels of workload indicate that the probability of errors or omissions is high</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>Impossible to perform all operational tasks properly</td>
<td>10</td>
</tr>
</tbody>
</table>

**b) Workload POR Scale**

Figure 4-1: Knee-pad Versions of the Performance and Workload PORs and of the Evaluation Sheet.
### EVALUATION SHEET

<table>
<thead>
<tr>
<th>MISSION VARIANT #</th>
<th>CONFIGURATION #</th>
<th>PILOT</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SPEED RETRIMMING</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHR</td>
</tr>
<tr>
<td>WORKLOAD</td>
</tr>
<tr>
<td>COMMENTS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HOLDING PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHR</td>
</tr>
<tr>
<td>WORKLOAD</td>
</tr>
<tr>
<td>COMMENTS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GLIDE SLOPE TRACKING</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHR</td>
</tr>
<tr>
<td>WORKLOAD</td>
</tr>
<tr>
<td>COMMENTS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OVERALL MISSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHR</td>
</tr>
<tr>
<td>WORKLOAD</td>
</tr>
<tr>
<td>COMMENTS</td>
</tr>
</tbody>
</table>

---

c) Evaluation Sheet.

Figure 4-1: Knee-pad Versions of the Performance and Workload PORs and of the Evaluation 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 accommodate 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 on-board 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 $w_i$ 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 $x_B$. A reasonable low-frequency wind model may consist of a constant component and a linear variation with altitude,

$$w_i(h) = w_{i_o} + \frac{\partial w_i}{\partial z_i} (z_i - z_{i_o})$$  \hspace{1cm} (A-1)

where $w_{i_o}$ is a constant wind vector at reference altitude $-z_{i_o}$ and $\frac{\partial w_i}{\partial z_i}$ is a vector of slope of wind variation with altitude. Adjoining eq. (A-1) to eq. (3-50), we obtain:
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:

\[
\begin{bmatrix}
\dot{x}_I \\
\dot{y}_I \\
\dot{z}_I
\end{bmatrix} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \cdot \left[ \begin{array}{c}
w_{x_I} + \frac{\partial w_x}{\partial z_I} (z_I - z_{I_0}) \\
w_{y_I} + \frac{\partial w_y}{\partial z_I} (z_I - z_{I_0}) \\
w_{z_I} + \frac{\partial w_z}{\partial z_I} (z_I - z_{I_0})
\end{array} \right] + \begin{bmatrix} \hat{H}_B(t) \\
\omega(t) \\
\hat{a}
\end{bmatrix} + \begin{bmatrix} \hat{\omega}_x \\
\hat{\omega}_y \\
\hat{\omega}_z
\end{bmatrix} * \begin{bmatrix} a_x - s \hat{\phi} g \\
\alpha_y + c \hat{\phi} s \hat{\phi} g \\
\hat{a}_z + c \hat{\phi} c \hat{\phi} g
\end{bmatrix} 
\]
\[
\begin{bmatrix}
\dot{u} \\
\dot{v} \\
\dot{w}
\end{bmatrix} = \hat{\omega}(t) 
\begin{bmatrix}
u \\
v \\
w
\end{bmatrix} 
+ \begin{bmatrix}
a_x - s\phi g \\
a_y + c\phi s\phi g \\
a_z + c\phi c\phi g
\end{bmatrix} 
+ \begin{bmatrix}
a_x \\
a_y \\
a_z
\end{bmatrix} 
+ H_{uvw} \hat{H}\left(\frac{\partial \hat{w}}{\partial z}\right)
\]
(A-5)

\[
\begin{bmatrix}
\dot{\hat{w}}_{xI_0} \\
\dot{\hat{w}}_{yI_0} \\
\dot{\hat{w}}_{zI_0}
\end{bmatrix} = \begin{bmatrix}0 \\
0 \\
0
\end{bmatrix}
\]
(A-6)

\[
\begin{bmatrix}
\frac{\partial \hat{w}}{\partial z} \\
\frac{\partial \hat{w}}{\partial z} \\
\frac{\partial \hat{w}}{\partial z}
\end{bmatrix} = \begin{bmatrix}0 \\
0 \\
0
\end{bmatrix}
\]
(A-7)

\[
H_{uvw} \triangleq \begin{bmatrix}\hat{H}_{B_{31}} & \hat{H}_{B_{32}} & \hat{H}_{B_{33}}\end{bmatrix} \begin{bmatrix}u \\
v \\
w
\end{bmatrix}
\]
(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:

\[
\begin{bmatrix}
\dot{p} \\
\dot{q} \\
\dot{r} \\
\dot{\phi} \\
\dot{\theta} \\
\dot{\psi}
\end{bmatrix} =
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1 & \tan \theta \phi & \tan \theta \phi \cos \psi - \tan \phi \sin \psi (q \sin \phi + r \cos \phi) / c^2 \phi & 0 & 0 & 0 \\
0 & \cos \phi & -\sin \phi & -q \sin \phi - r \cos \phi & 0 & 0 \\
0 & \frac{s \cos \phi}{c \phi} & \frac{c \phi}{c \phi} & q \phi / c \phi - r s \phi / c \phi / c \theta (q \sin \phi + r \cos \phi) s \phi / c^2 \phi & 0 & 0
\end{bmatrix}
\begin{bmatrix}
p \\
qu \\
r \\
\phi \\
\theta \\
\psi
\end{bmatrix} +
\begin{bmatrix}
p' \\
qu' \\
r' \\
q\phi' \\
0 \\
0
\end{bmatrix}
\]

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

\[
\begin{bmatrix}
\dot{x}_B \\
\dot{y}_B \\
\dot{z}_B \\
\dot{u} \\
\dot{v} \\
\dot{w}
\end{bmatrix} =
\begin{bmatrix}
0 \\
\hat{H}_B(t) \dot{x}_B \\
\hat{x}_B \\
\hat{w}(t) \\
\hat{w}(t) \\
\hat{w}(t)
\end{bmatrix} + [I] \begin{bmatrix}
x_B \\
y_B \\
z_B \\
u \\
v \\
w
\end{bmatrix} +
\begin{bmatrix}
0 \\
na_x \\
na_y \\
na_z
\end{bmatrix}
\]

\[\text{(A-10)}\]
The linearized version of measurement model B [eq. (3-51)]:

\[
\begin{bmatrix}
X_{\text{air}}'
\end{bmatrix} =
\begin{bmatrix}
0 & 0 & 0 & \frac{u}{a_0^{1/2}} & \frac{v}{a_0^{1/2}} & \frac{w}{a_0^{1/2}} \\
0 & 0 & 0 & -\frac{w}{a_0^{1/2}} & 0 & \frac{u}{a_0^{1/2}} \\
0 & 0 & 0 & -\frac{v}{a_0^{1/2}} & \frac{u}{a_0^{1/2}} & 0 \\
0 & 0 & 0 & -1 & 0 & 0 \\
\frac{x_1-x_{s1}}{a_{31}^{1/2}} & \frac{y_1-y_{s1}}{a_{31}^{1/2}} & \frac{z_1-z_{s1}}{a_{31}^{1/2}} & 0 & 0 & 0 \\
\frac{x_2-x_{s2}}{a_{32}^{1/2}} & \frac{y_2-y_{s2}}{a_{32}^{1/2}} & \frac{z_2-z_{s2}}{a_{32}^{1/2}} & 0 & 0 & 0 \\
\frac{(y_1-y_{s1})}{a_{31}^{1/2}} & \frac{(z_1-z_{s1})^2}{a_{31}^{1/2}} & 0 & 0 & 0 \\
\frac{(y_2-y_{s2})}{a_{32}^{1/2}} & \frac{(z_2-z_{s2})^2}{a_{32}^{1/2}} & 0 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
x_1 \\
y_1 \\
z_1 \\
x_2 \\
y_2 \\
z_2 \\
\end{bmatrix}
\]

(A-11)

with:

\[
a_0 = u^2 + v^2 + w^2
\]

\[
a_{31} = (x_1-x_{s1})^2 + (y_1-y_{s1})^2 + (z_1-z_{s1})^2
\]

(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:
\[
\begin{bmatrix}
F_{11} & F_{12} & F_{13} & F_{14} \\
F_{21} & F_{22} & F_{23} & F_{24} \\
F_{31} & F_{32} & F_{33} & F_{34} \\
F_{41} & F_{42} & F_{43} & F_{44}
\end{bmatrix}
\begin{bmatrix}
\dot{x}_I \\
\dot{y}_I \\
\dot{z}_I \\
\dot{w}
\end{bmatrix}
=
\begin{bmatrix}
F_{11} & F_{12} & F_{13} & F_{14} \\
F_{21} & F_{22} & F_{23} & F_{24} \\
F_{31} & F_{32} & F_{33} & F_{34} \\
F_{41} & F_{42} & F_{43} & F_{44}
\end{bmatrix}
\begin{bmatrix}
x_I \\
y_I \\
z_I \\
u
\end{bmatrix}
+
\begin{bmatrix}
\frac{\partial w_x}{\partial z_I} \\
\frac{\partial w_y}{\partial z_I} \\
\frac{\partial w_z}{\partial z_I}
\end{bmatrix}
\begin{bmatrix}
x_B \\
F_B \\
x_B
\end{bmatrix}
\]

\[(A-13)\]
where each $F_{ij}$ stands for a 3x3 matrix:

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

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

$$
F_{14} = (z_I - z_I^0) [I]
$$
(A-16)

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

$$
F_{12} = \hat{H}_B^I (t)
$$
(A-18)

$$
F_{22} = \begin{bmatrix}
\hat{H}_B^{31} \left( \hat{H}_I \frac{\partial w_I}{\partial z_I} \right) & \hat{H}_B^{32} \left( \hat{H}_I \frac{\partial w_I}{\partial z_I} \right) & \hat{H}_B^{33} \left( \hat{H}_I \frac{\partial w_I}{\partial z_I} \right) \\
-\hat{H}_B^{31} \left( \hat{H}_I \frac{\partial w_I}{\partial z_I} \right) & \hat{H}_B^{32} \left( \hat{H}_I \frac{\partial w_I}{\partial z_I} \right) & \hat{H}_B^{33} \left( \hat{H}_I \frac{\partial w_I}{\partial z_I} \right) \\
\hat{H}_B^{31} \left( \hat{H}_I \frac{\partial w_I}{\partial z_I} \right) & -\hat{H}_B^{32} \left( \hat{H}_I \frac{\partial w_I}{\partial z_I} \right) & \hat{H}_B^{33} \left( \hat{H}_I \frac{\partial w_I}{\partial z_I} \right)
\end{bmatrix}
$$
(A-19)

$$
F_{24} = H_{uvw} \ast \hat{H}_I^B (t)
$$
(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: $38 \times 26 + 36 = 1024$). 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 $\frac{8 \times 128 - 36}{38} = 26$ 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)

1: Version 8 SPIFER 18-V.41
2: ADC READ, PUSHBUTTONS, BLINKING LIGHTS WORKING
3: DISPLAY MEMORY INCLUDED
4: TAPES INSTALLED
5: DATA AVERAGING IMPLEMENTED
6: SLOW CHANNELS AVERAGED AT CNT = 4
7: DUE TO BE IMPLEMENTED AS SLOW CHANNEL
8: ABC CHANNEL ZERO READ TWICE TO HILL OPEN RAM SLOT
9: LEFT BY MOVING DUE TO SLOW CHANNEL
10: STILL AVERAGING OVER 12 FAST CHANNELS
11: HAVE A BUFFER OF 38 WORDS (16 BITS WIDE)
12: INSTEAD OF 32 WORDS.
13: HAVE FOUR LIGHTS INSTEAD OF TAPES
14: LIGHTS IS IN SYM WITH LIGHTS.
15: THE LIGHTS CODING IS ALL NEW.
16: 
17: 
18: GOS 0001H ;ADDRESS OF A TO D CONVERTER
19: JMP EQU O300H
20: JCRE EQU O360H
21: INT75 EQU OFF00H
22: LITERAL EQU OC80H
23: LITERAL EQU OC90H
24: LITERAL EQU O0EH
25: LITERAL EQU O0EH
26: LITERAL EQU O0EH
27: LITERAL EQU O0EH
28: LITERAL EQU O0EH
29: LITERAL EQU O0EH
30: LITERAL EQU O0EH
31: CALL INITPORTA ;A255 PORT A OF J2 ON 8004
32: CALL INITISRT ;INITIALISE THE UART FOR PILOT TERMINAL
33: CALL INITTERM ;SET UP SEMAPHORES ON ALL TEN BUFFERS
34: CALL INITTERM
35: CALL FILMEM ;FILL MEMORY WITH INTEGERS 0-10239
36: CALL INITMEM ;INITIALISE EMPTYING POINTERS
37: JMP TEN BUFFERS EACH 2048
38: TO TAPE, 256 BYTES AT A TIME
39: OVER AND OVER AGAIN.
40: THE BUFFERS ARE DESIGNATED BY
41: THE DIGITS 0 TO 9.
42: ; THE POINTS TO THE LOCATION
43: ; TO BE EMPTIED.
44: 
45: 
46: CALL ALIGN ;INITIALISE TYSW
47: CALL OR
48: ; THE PAIR POINT TO LOCATION TO BE EMPTIED
49: CALL STINT ;DISABLE THE INTERRUPTS
50: BB:
51: CALL TSETPTT ;WAIT UNTIL BUFFER FULL
52: CALL WRITE ;SUM UP ANOTHER 256 BYTES TO TAPE
53: ; UPDATE THE INDICES USED
54: ; TO KEEP TRACK OF EMPTYING THE TEN BUFFERS.
55: ; THERE ARE THREE OF THEM.
56: ; TWO POINTERS AND A COUNTER.
57: 
58: JMP BB
59: BB:
60: LDA FLAG
61: CPI 0
62: JMP BB
63: MVC A.10
STA FLAG
MVI A, "I"
CALL BD
JMP BB

; FILL: DB 0

PUSH PSW
; BASE ADDRESS FOR FILLING IS BUFF
LXI D,BUFF
; X:0 starts with 0
FILL: MOV A,L
; LOWER BYTE INTO ACC
STAX D
; STORE IT TO MEMORY
INK D
; MOVE TO NEXT BYTE LOCATION
MOV A,H
; UPPER BYTE INTO ACC
STAX D
; STORE IT TO MEMORY
INK D
; POINT TO NEXT BYTE LOCATION
INK H
; SIZE OF INTEGER UP BY 1
MOV A,H
; CHECK FOR AN UPPER Bound
CALL SETEMPTY
; FILL I:
; Repeat if below upper bound
PUSH H
PUSH B
CALL SETEMPTY
; FOR DUMPING NEXT 2048 BLOCK
; SET I:
; MAKE CURRENT BUFFER EMPT
CALL SETINDEX
; SET UP INDICES FOR DUMPING NEXT 2048 BLOCK
; WE ARE NOW BEYOND Tenth BUFFER
; NOTE EMIDSB HAS RANGE 0 TO 9
; SWITCH TO BUFFER 0
MVI A,0
; EMIDSB
DAA
STA EMIDSB
; EMIDSB
CP 10
CPI 10
; LOCATION OF THE CURRENT 2048 BLOCK
; BEING EMPTIED

SETINDEX: MOV HL,BUFFER

LLA004 F5 196 PUSH PSW
LLA005 FS 197 PUSH H
LLA006 3E00 198 MVI A,0
LLA008 324888 199 STA B4888
LLA00B 210004 200 LIX H,2048
LLA00E 226688 201 SHLD B2MAIN
LLA011 3E4C 202 MVI A,1'
LLA023 326768 203 STA SSM
LLA024 210090 204 LIX H,BUFF
LLA029 227468 205 SHLD BEURBUF
LLA02C 110090 206 LIX H,BUFF
LLA02F 81 207 POP H
LLA030 F1 208 POP PSW
LLA031 C9 209 RET
ll 210 LMDK5:
ll 211 PUSH H
ll 212 213 THE (= IE - 256)
ll 214 215
ll 216 MOV H.D
ll 217 MOV L.E
ll 218 MOV B,256
ll 219 STA A
ll 220 MOV B,H
ll 221 MOV L.L
ll 222 MOV IE
ll 223 RET
ll 224 LCTU:
ll 225 CALL IE
ll 226 MOV A,IE
ll 227 CALL DO
ll 228 MOV A,'I
ll 229 CALL DO
ll 230 MOV A,'I'
ll 231 CALL DO
ll 232 MOV A,'I'
ll 233 CALL DO
ll 234 MOV A,'I'
ll 235 CALL DO
ll 236 MOV A,'I'
ll 237 CALL DO
ll 238 MOV A,'I'
ll 239 CALL DO
ll 240 CALL AN
ll 241 CALL IE
ll 242 CALL IE
ll 243 STA SAV
ll 244 CALL IE
ll 245 STA SAV+1
ll 246 RET
ll 247 LCTU:
ll 248 LOAD TEMPE
ll 249 MOV A,IE
ll 250 CALL DO
ll 251 MOV A,'I'
ll 252 CALL DO
ll 253 MOV A,'I'
ll 254 CALL DO
ll 255 MOV A,'I'
ll 256 CALL DO
ll 257 MOV A,'I'
ll 258 CALL DO
ll 259 MOV A,'I'
ll 260 CALL DO
ll 261 MOV A,'I'

59
CALL BD
MVI A, '6'
CALL BD
MVI A, '4'
CALL BD
MVI A, 'E'
CALL BD
MVI A, '2'
CALL BD
MVI A, '5'
CALL BD
MVI A, 'C'
CALL BD
MVI A, '4'
CALL BD
MVI A, 'p'
CALL BD
MVI A, '2'
CALL BD
MVI A, '5'
CALL BD
MVI A, 'C'
CALL BD
MVI A, '4'
CALL BD
MVI A, 'p'
CALL BD
MVI A, '2'
CALL BD
MVI A, '5'

; READ S OR F
; READ CR
; EITHER S OR F
; RETURN Z SET ON 'S', Z RESET ON 'F'

; ADDRESS OF RCTU
; WRITE FILEMARK COMMAND IS ASCII CHAR 5
L8197 CM162 328 CALL BO
L817D 356 329 MVI A, '6'
L817E CM162 330 CALL BO
L8201 327 331 MVI A, '4'
L8203 CM162 332 CALL BO
L8206 382 333 MVI A, '2'
L8210 CM162 334 CALL BO
L8212 CM162 335 CALL BI
L820E 324b 336 STA SAV
L8211 FFA 337 CPI ACX
L8213 3A 338 JZ NEXT
L8216 C2581 339 JMP RCTUM
L824E 349 NEXT:
L8219 010001 341 LXI B,256
L821C 37482 342 CALL SEND
L821F 3311 343 MVI A,DC1
L8221 CM162 344 CALL BO
L8224 CM5762 345 CALL BI ; READ S OR F
L8227 D42581 346 JC RCTUM
L8228 325b8 347 STA SAVE
L8229 CM5762 348 CALL BI ; READ CR
L8230 D42581 349 JC RCTUM
L8233 325b8 350 STA SAVE+1
L8236 34258a 351 LDA SAVE ; EITHER S OR F
L8239 FES 352 CPI "S" ; RETURN WITH Z SET ON S, Z RESET ON F
L823B 3A 353 AZ
L823C E957848 354 JED TEMPREG
L8240 3C 355 RET
L8241 F5 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 ; 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; }
U25C3 08 394 RXI B
U25C5 78 395 RXI A,B
U25C6 61 396 DMA C
U25F6 CAG102 397 JZ TIMEOUT
U262C B5E5 398 IN DB251 A SET USAR STATUS
U2634 6A02 399 ANI DB251A CHECK RECEIVER READY
U263A 5A5312 400 JZ B1 1
U2639 D5E2 401 IN DB251 A GET CHAR
U2638 3F 402 STC
U263C 3F 403 ORC
U263A 61 404 POP B
U263F 00 405 RET
U2670 37 406 TIMEOUT: NOP
U2671 C1 408 POP B
U2672 09 409 RET
U2673 76 410 HLX
U2674 1A 412 LDX D
U2675 C4162 413 CALL BD
U2678 13 414 INX D
U267F 08 415 INX B
U267A 78 416 MOV A,B
U267B 61 417 DMA C
U267C D7462 418 J1 SER SEND
U267F 09 419 RET
U2680 F5 421 PUSH PSW
U2681 E1 422 PUSH H
U2682 21FC59 423 LDI H,START
U2685 3E1B 424 MVI A,ESC
U2687 CD4162 425 CALL BD
U268A 3E5E 426 MVI A, ESC
U268C CD4162 427 CALL BD
U268F 3E11 428 MVI A, ESC
U2691 CD4162 429 CALL BD
U2694 CD5782 430 CALL BI RATE ESC
U2697 CD5782 431 CALL BI RATE BACKSLASH
U269A CD5782 432 CALL BI RATE 0
U2697 CD5782 433 CALL CO
U269A CD5782 434 CALL BI RATE 1
U269A CD5782 435 CALL CO
U2696 CD5782 436 CALL BI RATE 2
U269F CD5782 437 CALL CO
U269C CD5782 438 CALL BI RATE 3
U269F CD5782 439 CALL CO
U269C CD5782 440 CALL BI RATE 4
U269B CD5782 441 CALL CO
U269C CD5782 442 CALL BI RATE 5
U269B CD5782 443 CALL CO
U269C CD5782 444 CALL BI RATE 6
U269C CD5782 445 CALL CO
U269A CD5782 446 CALL BI SET CR
U269C CD5782 447 CALL CO
U269E E1 448 POP H
U269F FI 449 POP PSW
U26A 09 450 RET
U26A 3F 451 LITUNT:
U26A E2 452 PUSH PSW
U26A E2 453 PUSH H
U26A E2 454 MVI A,ESC
U26A 91 CD5782 455 CALL CO
U26A D2 456 MVI A, ESC
U26A 91 CD5782 457 CALL CO
U26A 3D70 458 MVI A, ESC
U26A 3D88 459 CALL CO
L68226 DB 3E31 460 MVI A,'t'
L6822D C6088 461 CALL CD
L6822E 3E31 462 MVI A,'z'
L6822F C6088 463 CALL CD
L68238 3E11 464 MVI A,00C1
L68239 C6088 465 CALL CD
L6823E C5762 466 CALL BI ;ESC READ
L68240 C5762 467 CALL BI ;BACKSLASH READ
L68243 C5762 468 CALL BI ;P READ
L68246 C5762 469 CALL BI ;DEVICE CORE DIGIT READ
L68249 C5762 470 CALL BI ;BYTE 0 READ
L6824E C5762 471 CALL BI ;BYTE 1 READ
L68252 C5762 472 CALL BI ;BYTE 2 READ
L68256 DB 08 473 CALL BI ;CR READ
L68257 DB 0A 474 POP H
L68258 FB 475 POP PSW
L68259 C9 476 RET
L6827D 477 ; DELAY ONE MILLISECOND
L6827E 478 ;
L6827F 479 ;
L68280 480 DUMS:
L68283 08 481 PUSH B
L68287 482 PUSH PSW
L6828C 019800 483 LXI B,0132
L68291 08 484 DUMB: SCI B
L68295 7B 485 MOV A,B
L68296 F1 486 PLA C
L6829B C2043 487 JAZ DUMB0
L6829F FF 488 POP PSW
L682A3 C1 489 POP B
L682A5 C9 490 RET
L682A8 FF 491 ESC EDU 1BH
L682A9 492 CR EDU 0DH
L682AA 493 SR23A EDU 0EH
L682AB 494 DA23A EDU 0EH
L682AC 495 RO23A EDU 02H
L682AD 496 RSGYA EDU 01H
L682AE 497 LF EDU 0AH
L682AF 498 ENQ EDU 02H
L682B0 499 ACK EDU 0AH
L682B2 C9 500 DCL EDU 1NH
L682B5 4F 501 cmtrz: PUSH PSW
L682B7 3E34 502 MVI A,074H
L682B9 3E35 503 OUT 0DH
L682BA 3E30 504 MVI A,070H
L682BB 3E32 505 OUT 0DH
L682BD 3E37 506 MVI A,073H
L682BE 3E31 507 OUT 0DH
L682BF 3E33 508 XTHL
L682C0 4E 509 XML
L682C3 F1 510 POP PSW
L682C5 C9 511 RET
L682C7 3E34 512 cmtrz: PUSH PSW
L682C9 3E35 513 MVI A,070H
L682CA 3E32 514 OUT 0DH
L682CB 3E0A 515 MVI A,0AH
L682CE 3E32 516 OUT 00CH
L682D0 3E00 517 MVI A,00H
L682D2 3E00 518 OUT 00CH
L682D4 EB 519 XML
L682D5 3E 520 XML
L682D8 FE 521 POP PSW
L682D9 C9 522 RET
L682E0 3E34 523 STMT:
L682E2 4F 524 PUSH PSW
L682E3 E5 525 PUSH H
CALL DISPLAY ;IF THERE IS TWO DIGIT PAIR READY, DISPLAY MEMORY
CALL URFPS ;THERE ARE TWO BUFFER POINTERS AND COUNTER
CALL UCLE ;TO UPDATE
CALL FLIP ;TOGGLE LITE 3
CALL AVG
CALL UENTS
CALL RENTS
CALL SENTS
CALL AFR
CALL OED
CALL OEDM
CALL NIG
CALL INTERRUPT
CALL NIG
CALL RMS
CALL PIP ;MAKE OUTPUT OF FIVE VOLTS
CALL DIP
CALL OEDM
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHANNEL
CALL AOC
CALL REAL ADC CHA
UMF012 790
STAX D

UMF013 791
IN D

UMF014 792
POP PSW

UMF015 793
RET

UMF016 794
;

UMF017 795
;

UMF018 796
U84F1 I3

UMF019 797
;

UMF020 798
;

UMF021 799
;

UMF022 800
;

UMF023 801
THERE ARE TWO POINTERS

UMF024 802
AND ONE COUNTER

UMF025 803
TO BE UPDATED WHEN CURRENT

UMF026 804
BUFFER IS FULL.

UMF027 805
WHICH IS FROM 0 TO 9

UMF028 806
AND INDICATES THE CURRENT BUFFER

UMF029 807
BEING FILLED

UMF030 808
;

UMF031 809
SAVE POINTS TO THE BYTE OF

UMF032 810
THE CURRENT BUFFER TO BE FILLED.

UMF033 811
;

UMF034 812
REMAIN CONTAINS THE NUMBER

UMF035 813
OF UNFILLED BYTES IN CURRENT BUFFER

UMF036 814
;

UMF037 815
PUSH PSW

UMF038 816
PUSH H

UMF039 817
PUSH B

UMF040 818
I8D REMAIN

UMF041 819
LXI P, BYPERS

UMF042 820
JRA A

UMF043 821
ISBC B

UMF044 822
I8D REMAIN

UMF045 823
REM ail = REMAIN - BYPERS

UMF046 824
;

UMF047 825
IS REMAIN > BYPERS

UMF048 826
JRA A

UMF049 827
ISBC B

UMF050 828
JP ROOM

UMF051 829
;

UMF052 830
;

UMF053 831
;

UMF054 832
;

UMF055 833
;

UMF056 834
;

UMF057 835
CAL l SETFULL

UMF058 836
MAX CURRENT BUFFER FULL

UMF059 837
;

UMF060 838
;

UMF061 839
;

UMF062 840
LDA BYPIDO

UMF063 841
JMP A

UMF064 842
STA PIDO

UMF065 843
CP1 10

UMF066 844
JSR 5W

UMF067 845
;

UMF068 846
;

UMF069 847
MV1 A, 0

UMF070 848
STA PIDO

UMF071 849
LXI B, BYPIDO

UMF072 850
SHED CURBFU

UMF073 851
JSF ROOM

UMF074 852
;

UMF075 853
I8L CURBFU

UMF076 854
LXI B, 2048

UMF077 855
; INCREMENT BY 2048

68
U3534 227688 850 SHI.D ILJL DATED BASE ADDRESS SAVETH E L THE DE PAIR

U3537 C1 857 MOV D,H ; AND PUT INTO
U3538 50 858 MOV E,L
U3539 C9 859 ;

U3539 C1 860 ROR:
U3536 E1 861 POP B
U3538 F1 862 POP H
U353C C9 863 POP PSH

U353D F5 864 INS:
U353E 3053A8 867 LDA CNT
U3541 EC 868 INR A
U3542 3253A8 869 STA CNT
U3543 F1 870 POP PSH
U3544 C9 871 RET

U3547 F5 872 RINTs:
U3548 3053A8 874 LDA CNT
U3549 FD68 875 CPI 10
U354A 2253A8 876 JNZ NEXT
U3550 3060 877 MVI A,0
U3552 3253A8 878 STA CNT
U3553 C9 879 MVI:

U3554 F1 880 POP PSH
U3555 C9 881 RET

U3556 C9 882 UCLK:
U3557 F5 883 PUSH H
U3558 2653A8 884 LCLD TIM
U3559 23 885 INX H
U355C 2253A8 886 SHLD TIM
U355F E1 887 POP H
U3564 C9 888 RET

U3565 C9 889 ADO:
U3566 322130 890 STA ADRH+01H
U3567 3611 891 MVI A,01
U3568 360000 892 STA ADRH+00H
U3569 360000 893 BUSY LDA ADRH+00H
U356A 07 894 RLC
U356B 026945 895 JNC BUSY
U356C 260430 896 LCLD ADRH+00H
U356D C9 897 RET

U356E 2600 898 CEC:
U356F 6F 899 MVI L,A
U3575 2660 901 MVI H,0
U3577 C9 902 RET

U3578 6F 903 SED:
U357C EC 904 MVI L,A
U357E C7 906 MVI H,A
U3574 C9 907 RET

U3578 2600 908 IMM:
U3579 2660 909 MVI L,0
U357A 2660 910 MVI H,0
U3582 C9 911 RET

U3582 C9 912 DIG:
U3582 59 913 MVI L,34
U3582 51 914 MVI H,0
U3583 38B4 915 IN ALTPORT
U3585 6F 916 MVI L,A
U3586 3660 917 MVI H,0
U3588 C9 918 RET

U3589 39 919 DIGI:
U3589 20 920 MVI L,25
U3589 21 921 MVI H,0

69
L8589 265268 922 LHI LIGHTS
L858C C9 923 RET
L858D C9 924 BICODE:
L858D DBUS 925 IN PUSHPRT
L858F AF 926 MOV L1,A
L8590 2600 927 MOV H1,0
L8591 2572 928 MOV L1,36
L8592 C9 929 MOV H1,0
L8592 C9 930 BICODE:
L8593 265268 932 LHI TD
L8596 C9 933 RET
L8597 934 STATE:
L8597 935 ; THIS MACHINE HAS TWO STATES
L8597 936 ; STATE 0 THE MACHINE STAYS IN THIS STATE FOR 30 SECS.
L8597 937 ; STATE 1 THE MACHINE STAYS IN STATE 1 UNTIL THE PROPER
L8597 938 ; PUSH BUTTON IS DEPRESSED (LIGHT X GEE WITH FB X, ETC)
L8597 939 ; THE CORRESPONDING LIGHT IS TURNED OFF AND THE
L8597 940 ; MACHINE SHIFTS TO STATE 0.
L8599 FS 944 PUSH PSM
L859A FS 945 PUSH H
L8599 CS 946 PUSH B
L8599 SA 947 LDA ST
L8597 B7 948 ORA A ;SET THE FLAGS
L8598 C3C185 949 JNZ STATE1
L859A 212C01 950 STATE0:
L859A AF 951 LXI H, THE THER ;LOAD THE NUMBER OF TIMER CLICKS NEEDED FOR 30 SEC
L859A AF 952 LDA A ;CLEAR CARRY
L859A ED4588 953 LDX PTIM
L859A ED42 954 DBX B
L859A C38763 955 JNZ BPM
L859A C38405 956 CALL TOHL
L859A C3D866 957 CALL TOHS ;STATE=1
L859A C3D865 958 JMP BACK
L859B 959 BPM:
L859B 265068 960 LHI PTIM
L859A 23 961 INX H
L859A 225068 962 INX PTIM
L859A C3D845 963 JMP BACK
L859C 964 STATE1:
L859C C3D586 965 CALL FB ;RETURN PUSHPBTN STATUS IN Z FLAG
L859A C3D845 966 JNZ BACK
L859D 967 PTIM:
L859D C3FB65 968 CALL TOFF ;TURN OFF LITE
L859A C3D584 969 CALL TOFFS ;STATE=0
L859A C3D1066 970 CALL OPTIM ;PTIM = 0
L859A C3D586 971 CALL SUPPLET ;SWITCH TO NEXT PUSHPBTN + LITE SET
L859A C3D845 972 JMP BACK
L859C 973 BACK:
L859A C1 974 POP B
L859A E1 975 POP H
L859A F1 976 POP PSM
L859A C9 977 RET
L859A FS 978 PUSHP PSM
L859A 3A5668 980 LDA BINARY
L859A FE00 981 CPI 0
L859A C38765 982 JNZ ON2
L859A 3A5838 983 LDA LIGHTS
L859A EAFE 984 ANI O188
L859A 8D05 985 OUT LITEPORT ;BIT 0 PORT B 5 VOLTS
L859A 3235088 986 STA LIGHTS
L859A F1 987 POP PSM

70
144582 C9 998 RET
14457F 3856B8 999 LDA LIGHTS
14457F 3856B8 999 ANI 0FH ;
144574 B3C5 992 OUT LITEPORT ;BIT 1 PORT B 5 VOLTS
14457A 325288 993 STA LIGHTS
14457F F1 994 POP PSW
14457F C9 995 RET
14457F F5 996 PUSH PSW
14457F 3856B8 998 LDA BINARY
14457F FE00 999 CPI 0
144601 C21086 1000 JNZ QFF2
144604 3856B8 1001 LDA LIGHTS
144607 F601 1002 ORI 0FH
144609 B3C5 1003 OUT LITEPORT ;BIT 0 PORT B 0 VOLTS
144609 325288 1004 STA LIGHTS
144608 F1 1005 POP PSW
14460F C9 1006 RET
14460F F5 1007 QFF2
144610 3856B8 1008 LDA LIGHTS ;
144613 F602 1009 ORI 02H
144615 B3C5 1010 OUT LITEPORT ;BIT 0 PORT B 0 VOLTS
144617 325288 1011 STA LIGHTS
14461A F1 1012 POP PSW
14461B C9 1013 RET
14461C E5 1015 PUSH H
14461D 210000 1016 LXI H, 0
144620 325288 1017 SWLD PTIN ;CLEAR PTIN
144623 E1 1018 POP H
144624 C9 1019 RET
144625 3856B8 1021 LDA BINARY
144628 F600 1022 CPI 0
14462A C21086 1023 JNZ QF2
144629 D882 1024 IN PUSHPORT ;PUSHBUTTON STATUS
14462F 325288 1025 STA MODES
144632 E601 1026 ANI 0FH ;BIT 0 PORT B
144634 C9 1027 RET ;STATUS RETURNED IN Z FLAG
144635 D882 1028 PR2
144637 325288 1029 IN PUSHPORT
14463A E602 1030 STA MODES
14463E E602 1031 ANI 02H ;BIT 1 PORT B
144640 C9 1032 RET
144642 F5 1033 TMGE
144643 E801 1035 MVI A, 01H ;
144646 3245B8 1036 STA ST ;STATE=1
14464A F1 1037 POP PSW
14464A C9 1038 RET
14464F F5 1039 TOFF5
14464F 3856B8 1040 MVI A, 0
14464F 3245B8 1041 STA ST
14464F F1 1043 POP PSW
14464C C9 1044 RET
14464F F1 1045 FLIP2 ;TURG ELIGHT 3, TURG LIGH 4 TOGETHER
14464F F5 1046 ;
14464F 3856B8 1048 LDA LIGHTS
144651 E604 1049 XRI 04H ;FLIP BIT 2
144653 E608 1050 XRI 06H ;FLIP BIT 3
144657 325288 1051 STA LIGHTS
14465A B3C5 1052 OUT LITEPORT
14465A F1 1053 POP PSW
REf U1054

SLITe:

IF YOU ARE USING PB1 AND LITE1,

USE PB1 AND LITE1.

IF YOU ARE USING PB1 AND LITE1,

USE PB1 AND LITE1.

IF YOU USE PB1 AND LITE1,

USE PB1 AND LITE1.

PUSH PSN

LDA BDRY

IIR1OlH

STA BDRY

STABDRY

RET

SET DIGIT COUNT PNTDIG

IF THERE IS A

IF THERE IS

RETURN WITH Z SET

RETURN WITH Z SET

IF Z SET, NEXT

IF Z SET, NEXT

IF Z SET

SET DIGIT IF WAITING

SET DIGIT IF WAITING

IF Z SET, NEXT

IF Z SET

CONVERT ASCII DIGIT TO BINARY DIGIT

CONVERT ASCII DIGIT TO BINARY DIGIT

SAVE DIGIT IN D

SAVE DIGIT IN D

SAVE DIGIT IN D

SAVE DIGIT IN D

MOVE FIRST INPUT DIGIT IN ACC

MOVE FIRST INPUT DIGIT IN ACC

MOVE FIRST INPUT DIGIT IN ACC
73
LL4600 87 1186 ADD A  ;FORM BYTE OFFSET IN ACC
LL460A  DA0  1187 SUI  Z  ;VALUE OF 1 IS AN OFFSET OF ZERO
LL460A 2600 1188
LL460E  4F  1189 MVI H,0  ;MOVE ACC OFFSET
LL4611  D140578  1190 MDV L,A  ;ML HAS ACC OFFSET
LL4615  09  1191 LIRD SAVD16  ;BASE ADDRESS OF CURRENT STORAGE IN BC
LL4618  0D  1192 DAD B  ;ADDED TO OFFSET TO FORM POINTER TO
LL461A  18  1193  ;MEMORY LOCATION YOU WANT TO SEE
LL461C  11  1194

LL461E  CK4A07  1195 CALL CR1F
LL4623  CKF7A6  1196 CALL HEMP2  ;
LL4627  11  1197
LL462B  12  1198

LL462F  C1  1199 ;CALL CR1F
LL4633  E1  1200 ;LDA TWOBIG
LL4637  F1  1201 ;CALL HEMP
LL463B  00  1202 ;CALL CR1F
LL463F  0D  1203 ;LXI H,SAVING
LL4643  04  1204 ;CALL HEMP2
LL4647  05  1205 ;CALL CR1F
LL464B  3B00 1206 MVI A,R  ;ZERO
LL464F  324C88  1207 STA TWOBIG  ;TWOBIG
LL4653  1208
LL4657  1209
LL465B  1210 DISPLAY;

LL465F  C1  1211 POP B
LL4663  E1  1212 POP H
LL4667  F1  1213 POP PSW
LL466B  C9  1214 RET
LL466F  1215 ;DUMP THE CONTENTS OF MEMORY IN HEX
LL4673  1216 ;H HELD STARTING ADDRESS OF DUMP
LL4677  1217 ;A REG IS USED IN HEMP

LL467F  F5  1218 HEMP2: PUSH PSW
LL4683  E5  1219 PUSH H
LL4687  23  1220 INX H
LL468B  7E  1221 MDV A,N
LL468F  CK9F07  1222 CALL HEMP
LL4693  EB  1223 DCK H
LL4697  7E  1224 MDV A,N
LL469B  CK9F07  1225 CALL HEMP
LL469F  CK9A07  1226 CALL CR1F
LL46A3  E1  1227 POP H
LL46A7  F1  1228 POP PSW
LL46AB  C9  1229 RET

LL46AF  1230  ;HEMP USES THE A REG

LL46B3  F5  1231 HEMP: PUSH PSW  ;SAME ACC
LL46B7  F5  1232 PUSH PSW  ;TWICE
LL46BB  E60F  1233 ANI OFH  ;ISOLATE THE HIGH ORDER MOBILE
LL46BF  0F  1234 RRC
LL46C3  0F  1235 RRC
LL46C7  0F  1236 RRC
LL46CA  0F  1237 RRC  ;HL DIG SHIFITED RIGHT BY 4
LL46D3  CK2207  1238 CALL H1MHE  ;PRINT HIGH ORDER DIGIT
LL46D7  CK4888  1239 CALL CO  ;PRINT ASCII FORM OF HIGH DIGIT
LL46DC  F1  1240 POP PSW  ;RESTORE THE ACC TO VALUE AT ENTRY
LL46E0  E60F  1241 ANI OFH  ;ISOLATE THE LOW ORDER MOBILE
LL46E4  CK2207  1242 CALL H1MHE  ;
LL46E8  CK4888  1243 CALL CO  ;
LL46EB  F1  1244 POP PSW  ;
LL46F1  C9  1245 RET
LL46F5  0F  1246 BIDMHE: AB1 3OH  ;
LL46FC  FE9A  1247 CPI 3AH  ;
LL46FF  08  1248 RC
LL46F9  07  1249 AB1 7H
LL46FB  C9  1250 RET

LL46FD  1251  ;CR1F USES ONLY THE AB1G

74
1272 CALL CO
1273 POP    PSW
1274 RET
1275 IEP:
1276 ;VALIDATE:
1277 LET THE FILLER CATCH UP WITH EMPTIER
1278 PUSH PSW
1279 PUSH B
1280 LDA  FILLONE
1281 CPI   0
1282 JZ    VI
1283 V2: LDA FMICRO
1284 CPI   0
1285 V1: MVI A'w'
1286 CALL CO
1287 JZ    V2
1288 ONE BUFFER HAS NOT YET BEEN FILLED
1289 ONE BUFFER HAS BEEN FILLED
1290 MVI A,l
1291 STA    FILLONE
1292 VI: MAKE SURE FILLER STAYS AHEAD OF EMPTIER
1293 LDA   FMICRO
1294 MVI B, A
1295 LDA   FMICRO
1296 CMP   B
1297 LDA   FMICRO
1298 ADI   10
1299 CALL CO
1300 JZ    VI
1301 POP    B
1302 POP    PSW
1303 RET
1304 INITI:
1305 INITIALISE:
1306 MARK ALL BUFFERS EMPT:
1307 PUSH PSW
1308 PUSH B
1309 PUSH H
1310 LTI H,SEMIPHERE
1311 MVI B,10
1312 S1: MVI H, 'E'
1313 INX H
1314 BXR B
1315 JNZ S1
1316 POP    H
1317 POP    B
1318 RET
; I WAIT UNTIL BUFFER EMPTY (INFINITE LOOP!

.LW748 F1 1318 POP PSW
.LW749 C9 1319 RET
.LL 1320 TSTFULL: U
.LW74A F5 1321 PUSH PSW
.LW74B C5 1322 PUSH B
.LW74C ES 1323 PUSH H
.LW74D 216A88 1324 LDX H$SIMPHERE
.LW74E 36888 1325 LDA $SIMPHERE
.LW753 0600 1326 MVI B,0
.LW755 4F 1327 MOV C,A
.LW756 09 1328 SAI B ;(HL)= SIMPHERE($PMCHAR)
.LW757 7E 1329 FULL1: MVI A,H ;(A)= SIMPHERE($PMCHAR)
.LW758 F846 1330 CPI 'F' ;SEE IF BUFFER FULL
.LW759 CCA187 1331 CZ ORA $HBUF
.LW75A E1 1332 ;WAIT TIL EMPTY
.LW75B E1 1333 POP H
.LW75C C1 1334 POP B
.LW75D F1 1335 POP PSW
.LW760 C9 1336 RET
.LL 1337 DUR1:
.LW761 3E4F 1338 MVI A,'0'
.LW762 C9088 1339 CALL CD
.LW763 3E4E 1340 MVI A,'v'
.LW764 C9088 1341 CALL CD
.LW765 3E888 1342 LDA $PMCHR
.LW766 C41 1343 ADI 'A'
.LW767 0600 1344 CALL CD
.LW768 C50000 1345 AP 0
.LL 1346 TSTEMPTY: U
.LW769 F5 1347 PUSH PSW
.LW770 C5 1348 PUSH B
.LW771 ES 1349 PUSH H
.LW772 216A88 1350 LDX H$SIMPHERE
.LW773 36888 1351 LDA $SIMPHERE
.LW774 0600 1352 MVI B,0
.LW775 4F 1353 MOV C,A
.LW776 09 1354 SAI B ;(HL)= SIMPHERE($PMCHR)
.LW777 F845 1355 CPI 'E' ;SEE IF BUFFER EMPTY
.LW778 F5 1356 PUSH PSW ;SAVE Z FLAG
.LW779 36888 1357 LDA $PMCHR
.LW77A C30 1358 ADI 'V' ;CONVERT TO ASCII
.LL 1359 DUR2: CALL CD
.LW77B F1 1360 POP PSW ;RESTORE Z FLAG
.LW77C 0600 1361 JT EMPTY1: U
.LW77D E1 1362 POP H
.LW77E C1 1363 POP B
.LW77F F1 1364 POP PSW
.LW782 C9 1365 RET
.LL 1366 SETFULL: U
.LW783 F5 1367 PUSH PSW
.LW784 C5 1368 PUSH B
.LW785 ES 1369 PUSH H
.LW787 216A88 1370 LDX H$SIMPHERE
.LW788 36888 1371 LDA $SIMPHERE
.LW789 0600 1372 MVI B,0
.LW78A 4F 1373 MOV C,A
.LW78B 09 1374 SAI B ;(HL)= SIMPHERE($PMCHR)
.LW78C 3E46 1375 MVI A,'F' ;MARK IT FULL
.LW78D 77 1376 MOV H,A ;(SIMPHERE($PMCHR))= 'F'
.LW78E E1 1377 MOV H,0
.LW78F C1 1378 POP H
.LW790 F1 1379 POP B
.LW791 C9 1380 POP PSW
.LW792 0600 1381 MOV H,0
.LW793 4F 1382 MOV C,A
.LW794 09 1383 RET

76
1384 SETEMPTY: \#MARK BUFFER EMPTY

1385

1386 PUSH P SW

1387 PUSH B

1388 PUSH H

1389 LXI H,SEMAPHORE

1390 LDA EMICHB

1391 MOV B,0

1392 MOV C,A \#(DC)=(EMICHB)

1393 DAD B \#(SEMAPHORE)=(EMICHB)

1394 MOV A,\#0 \#MARK IT EMPTY

1395 CALL SETDIG \#ZERO DIGIT COUNTER, CNTDIG

1396 CALL CNTDIG

1397 CALL CNTDIG

1398 CALL CNTDIG

1399 CALL CNTDIG

1400 INTIRE: \#SAVE STATUS

1401 STA FILLER \#FILLER=0, MEANS ALL BUFFERS ARE EMPTY

1402 \#OF DATA.

1403 MOV A,\#68H \#ORIGaddir

1404 MOV A,\#00H \#ORIGdirb

1405 MOV A,\#00H \#ORIGdirc

1406 MOV A,\#00H \#ORIGdird

1407 MOV A,\#00H \#ORIGdirx

1408 MOV A,\#00H \#ORIGdifu

1409 MOV A,\#00H \#ORIGdirf

1410 MOV A,\#00H \#ORIGdirg

1411 MOV A,\#00H \#ORIGdirh

1412 MOV A,\#00H \#ORIGdire

1413 MOV A,\#00H \#ORIGdirj

1414 MOV A,\#00H \#ORIGdirk

1415 MOV A,\#00H \#ORIGdirl

1416 MOV A,\#00H \#ORIGdirm

1417 MOV A,\#00H \#ORIGdirn

1418 MOV A,\#00H \#ORIGdiro

1419 MOV A,\#00H \#ORIGdirp

1420 MOV A,\#00H \#ORIGdirq

1421 MOV A,\#00H \#ORIGdirr

1422 MOV A,\#00H \#ORIGdirs

1423 MOV A,\#00H \#ORIGdirs

1424 MOV A,\#00H \#ORIGdirf

1425 MOV A,\#00H \#ORIGdirg

1426 MOV A,\#00H \#ORIGdirh

1427 MOV A,\#00H \#ORIGdiri

1428 MOV A,\#00H \#ORIGdirj

1429 MOV A,\#00H \#ORIGdirk

1430 MOV A,\#00H \#ORIGdirl

1431 MOV A,\#00H \#ORIGdirm

1432 MOV A,\#00H \#ORIGdirn

1433 MOV A,\#00H \#ORIGdiro

1434 MOV A,\#00H \#ORIGdirp

1435 MOV A,\#00H \#ORIGdirq

1436 MOV A,\#00H \#ORIGdirr

1437 MOV A,\#00H \#ORIGdirs

1438 MOV A,\#00H \#ORIGdirf

1439 MOV A,\#00H \#ORIGdirg

1440 MOV A,\#00H \#ORIGdirh

1441 MOV A,\#00H \#ORIGdiri

1442 MOV A,\#00H \#ORIGdirj

1443 MOV A,\#00H \#ORIGdirk

1444 MOV A,\#00H \#ORIGdirl

1445 MOV A,\#00H \#ORIGdirm

1446 MOV A,\#00H \#ORIGdirn

1447 MOV A,\#00H \#ORIGdiro

1448 MOV A,\#00H \#ORIGdirp

1449 MOV A,\#00H \#ORIGdirq

1450 MOV A,\#00H \#ORIGdirr

1451 MOV A,\#00H \#ORIGdirs

1452 MOV A,\#00H \#ORIGdirf

1453 MOV A,\#00H \#ORIGdirg

1454 MOV A,\#00H \#ORIGdirh

1455 MOV A,\#00H \#ORIGdiri

1456 MOV A,\#00H \#ORIGdirj

1457 MOV A,\#00H \#ORIGdirk

1458 MOV A,\#00H \#ORIGdirl

1459 MOV A,\#00H \#ORIGdirm

1460 MOV A,\#00H \#ORIGdirn

1461 MOV A,\#00H \#ORIGdiro

1462 MOV A,\#00H \#ORIGdirp

1463 MOV A,\#00H \#ORIGdirq

1464 MOV A,\#00H \#ORIGdirr

1465 MOV A,\#00H \#ORIGdirs

1466 MOV A,\#00H \#ORIGdirf

1467 MOV A,\#00H \#ORIGdirg

1468 MOV A,\#00H \#ORIGdirh

1469 MOV A,\#00H \#ORIGdiri

1470 MOV A,\#00H \#ORIGdirj

1471 MOV A,\#00H \#ORIGdirk

1472 MOV A,\#00H \#ORIGdirl

1473 MOV A,\#00H \#ORIGdirm

1474 MOV A,\#00H \#ORIGdirn

1475 MOV A,\#00H \#ORIGdiro

1476 MOV A,\#00H \#ORIGdirp

1477 MOV A,\#00H \#ORIGdirq

1478 MOV A,\#00H \#ORIGdirr

1479 MOV A,\#00H \#ORIGdirs

1480 MOV A,\#00H \#ORIGdirf

1481 MOV A,\#00H \#ORIGdirg

1482 MOV A,\#00H \#ORIGdirh

1483 MOV A,\#00H \#ORIGdiri

1484 MOV A,\#00H \#ORIGdirj

1485 MOV A,\#00H \#ORIGdirk

1486 MOV A,\#00H \#ORIGdirl

1487 MOV A,\#00H \#ORIGdirm

1488 MOV A,\#00H \#ORIGdirn
WRITE TO DDRAM, RESTORE CHAR TO A REG

SEND IT

READ!

TO

READ!

TO

READ!

TO

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!

READ!
1516 PAKCH; DB 0 ;FILL WHICH BUFFER
1517 TIM: DB 0
1518 CHN: DB 0
1519 FILLING: DB 0
1520 FLAG: DB 0
1521 SAVE: DB 0
1522 SAVE: DB 0
1523 EDNAM: DB 0
1524 EDNCH: DB 0 ;EMPTY WHICH BUFFER
1525 TS: DB 0
1526 SEPHORES: DB 'E' SEPHORES ALL 'E'
1527 CLEARBUF: DB 0
1528 CURRBUF: DB BUFFER
1529 TENDER: DB 0
1530 AVG:
1531 PUSH PMF
1532 PUSH H
1533 PUSH D
1534 PUSH D
1535 PUSH I
1536 PUSH Y
1537 LDA
1538 CPI 0
1539 JNI AVG1
1540 JNI AVG1
1541 CALL CO
1542 CALL PART1
1543 PDM
1544 PDM
1545 POP D
1546 POP B
1547 POP H
1548 POP PSW
1549 RET
1550 AVG1: CPI 9
1551 JNI AVG2
1552 PDM
1553 PDM
1554 PDM
1555 POP D
1556 POP B
1557 POP H
1558 POP PSW
1559 RET
1560 AVG2: CALL PART2
1561 PMR
1562 PMR
1563 POP D
1564 POP B
1565 POP H
1566 POP PSW
1567 RET
1568 PART1
1569 LXI H,AC2
1570 LXI B,AC2
1571 CALL CLEAR
1572 PART2:
1573 JNI AVG1
1574 CALL CO
1575 CALL CLR2
1576 XRA A
1577 STA ITER
1578 TP1:
1579 CALL PORTA}
1580 LXI PMTHA16
U6eI!4 FD21a&eA 1581
Lm
Ua8G8 CDBF89 1582 CALL LD2BCDE ;(IDE)\Rightarrow=A16<Im)
1rJIU. IIDE ;IUf'
a:DE Ra;IStStS
U1MCB CD4Jl89 1585 CALL DSTACI ;PUr(BCIlE)
32 BIT STACI
UMD9 CD4D89 1590 CALL DSTACI ;(BCIlEh=RSJI.T
OF
32BIT =~
U88DC CIl7189
1591 CALL FIXAIlD
U8&DF CIl5F89
1592 CALL UDSTta ;(BCDE) \Rightarrow=A22ITER)...
U8906 FD21888A 1597 CALL FIXAIlD
U88F4 FDC 1603 CPI 12
U88F6 C22088 1604 JNZ TP1
U88E6 10000 1605 CALL PORTION
U88F9 C9 1606 RET
U88F4 AF 1607 PORT3:
U88FB 32888A 1608 STA ITER
U88E7 B0215788 1610 TP2:
U8802 F21888A 1612 LIT A
U8806 CDBF89 1613 CALL LD2BCDE \Rightarrow=(A16(ITER))
U8809 CD4089 1614 CALL DSTACK \Rightarrow=(BCDE) OMT 32 BIT STACK OF T5511
U880C B021468A 1615 LIT A2Z
U8810 F21888A 1616 LITY ITER
U8814 CDBF89 1617 CALL LD2BCDE \Rightarrow=(A22(ITER))=BCDE)
U8817 CD4089 1618 CALL DSTACK \Rightarrow=(BCDE) OMT 32 BIT STACK OF T5511
U881A CD7189 1619 CALL FIXAIlD \Rightarrow=A22 32 BIT INTEGERS TOGETHER
U881D 110000 1620 LIT 5;10
U8820 010000 1621 LIT 5;0
U8823 CD4089 1622 CALL DSTACK
U8826 CD7829 1623 CALL FIXDIV \Rightarrow=DIVIDE BY TEN
U8829 CD4089 1624 CALL UDSTSTACK \Rightarrow=(BCDE) HAS AVERAGE OF TEN VALUES
U882C B0215788 1625 LIT PMTA16
U8830 F21888A 1626 LITY ITER
U8834 CD4089 1627 CALL LD2BCDE \Rightarrow=(A16(ITER))
U8837 CD888A 1628 LDA ITER \Rightarrow=CHECK ITERATION COUNT
U883A CD888A 1629 LBA a ...
U883B CD888A 1630 STA ITER
U883E FDC 1631 CPI 12
U8840 CD2588 1632 JNZ TP2
U8845 C9 1633 RET
U8844 F5 1634 TPSRSTY: PUSH PSM
U8845 B00S 1635 TPSRSTY1: IN TPSRSTY1; TPS511CRTREPORT \Rightarrow=INPUT THE STATUS WORD
U8847 B7 1636 TPSRSTY1: IN TPS511CRTREPORT \Rightarrow=INPUT THE STATUS WORD
U8848 FM589 1637 VBA A \Rightarrow=SET UP FLAGS
U884B P1 1638 JN TPSRSTY1 \Rightarrow=BIT 7 SET MEANS TPS511 IS BUSY
U884C C9 1639 POP PSM
U884D F5 1640 RET
U8854 CD4489 1641 DISTACK: PUSH PSM
U8855 7B 1642 CALL TPSRSTY \Rightarrow=HOLD UNTIL TPS511 NOT BUSY
U8852 7D4 1643 OUT TPS511CRTREPORT \Rightarrow=NETWORK TO STACK
U8854 7A 1644 OUT TPS511CRTREPORT \Rightarrow=NETWORK TO STACK
; SIGNED AND UNSIGNED VALUE TO STACK OF 19511

1647 OUT TPS11DATAPORT ; NEXT SIG BYTE TO STACK OF TPS11
1648 MOV A,C
1649 OUT TPS11DATAPORT ; MSB BYTE TO STACK OF TPS11
1650 MOV A,B
1651 OUT TPS11DATAPORT ; NEXT BYTE FROM STACK OF TPS11

1654 LUSSTACK: PUSH PSM ; WAIT UNTIL TPS11 NOT BUSY
1655 CALL TPSBUSY
1656 IN TPS11DATAPORT ; MSB FROM STACK OF TPS11
1657 MOV B.A
1658 IN TPS11DATAPORT ; NEXT BYTE FROM STACK OF TPS11
1659 MOV C.A
1660 IN TPS11DATAPORT ; NEXT BYTE FROM STACK OF TPS11
1661 MOV D.A
1662 IN TPS11DATAPORT ; NEXT BYTE FROM STACK OF TPS11
1663 MOV E.A
1664 IN TPS11DATAPORT ; NEXT BYTE FROM STACK OF TPS11
1665 POP PSM
1666 RET

1668 FIXADD: PUSH PSM
1669 CALL TPSBUSY
1670 MOV A,02CH ; 32 BIT FIX POINT ADD
1671 OUT OSH
1672 CALL STAT
1673 POP PSM
1674 RET

1676 FIXDIV: PUSH PSM
1677 CALL TPSBUSY
1678 MOV A,02CH ; 32 BIT FIX POINT DIVIDE
1679 OUT OSH
1680 CALL STAT
1681 POP PSM
1682 RET

1683 LIMBCHR:
1684
1685 ; THIS SUBROUTINE LOADS A 32 BIT ARRAY ELEMENT INTO
1686 ; THE REGISTERS BCE. 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 ; LXII ITER
1692 ; CALL LIMBCHR
1693 F5 PUSH PSM

1694 E5 PUSH H
1695
1696 F5EDD0 MOVAY L,0
1697 F5EEO MOVAY H,1 ; (HL)=CONTENTS(ITER)
1698
1699 29 DAD H
1700 29 DAD H ; (HL)+=CONTENTS(ITER)
1701
1702 85 PUSHX
1703 C1 POP B ; (BC):=(IX)
1704
1705 09 DAD B ; (HL)+=ASR(ITER)
1706
1707 06
1708 06
1709 5E MOV B,M
1709 23 INX H
1710 5E MOV B,M
1711 23 INX H
1712 4E MOV C,M

81
L898F 23 1713  IDX H  
L898F 46 1714  MDS B,M  \(\text{BCD} \times 4\) BYTES OF A32(I) 
L8990 E1 1715  
L8991 F1 1716  POP H  
L8992 C9 1717  POP PSW  
L8993 F5 1718  RET  
L8994 E5 1719  \(*\)  
L8995 C5 1720  \(*\)  
L8996 65 1721  \(*\)  
L8997 F8A00 1722  MOVY L,0  
L8998 F8A01 1723  MOVY H,1  \(\text{HL} = \text{CONTENTS(ITER)}\)  
L8999 29 1724  DAD H  
L899A 29 1725  DAD H  \(\text{HL} = \text{CONTENTS(ITER)}\)  
L899B D8 1726  PUSHX  
L899C C1 1727  POP B  \(\text{BC} = \text{IX}\)  
L899D 09 1728  DAD B  \(\text{HL} = \text{A32(ITER)}\)  
L899E D1 1729  POP D  
L899F 73 1730  MOV M,E  
L89A0 23 1731  DEX H  
L89A1 72 1732  MOV M,D  
L89A2 23 1733  DEX H  
L89A3 C1 1734  POP B  
L89A4 71 1735  MOV M,C  
L89A5 23 1736  DEX H  
L89A6 70 1737  MOV M,B  
L89A7 E1 1738  POP H  
L89A8 F1 1739  POP PSW  
L89A9 C9 1740  RET  
L89A0  
L89A1  
L89A2  
L89A3  
L89A4  
L89A5  
L89A6  
L89A7  
L89A8  
L89A9  
L89AA  
L89AB  
L89AC  
L89AD  
L89AE  
L89AF F5 1741  \(*\)  
L89B0 E5 1742  \(*\)  
L89B1 D8A00 1743  MOVX L,0  
L89B2 D8A01 1744  MOVX H,1  \(\text{HL} = \text{ADDR(A16)}\)  
L89B3 EB 1745  \(*\)  
L89B4 CB 1746  \(*\)  

82
L89CB FB6401 1779 MOVY H,1 \( (HL)=\text{CONTENTS} \text{(ITER)} \)
L89CE 29 1780 DAD H \( (HL)+2=\text{CONTENTS} \text{(ITER)} \)
L89CF 19 1782 DAD D \( (HL)+2=\text{ADDR} \text{(A16)}+2=\text{CONTENTS} \text{(ITER)} \)
L89D0 5E 1784 1785
L89D1 23 1787 JXH H
L89D2 5A 1788 MOV D,H \( (DE)=\text{LOADED WITH 16 BIT INTEGER} \)
L89D9 010000 1789 1790
L89D3 1791 LXI B,0 \( (BC) \text{ SET TO 0} \)
L89D6 E1 1792 1793
L89D7 F1 1794 POP PSW
L89D8 C9 1795 RET
L89D9 1796 STBCE:
L89DA 73 1797 1798
L89DB 010400 1799 1800
L89DC 5E 1799 1800
L89DD 73 1801 SF 1802 ;THIS SUBROUTINE LOADS THE REGISTERS INTO A 16 BIT ARRAY ELEMENT.
L89DE 29 1803 1804
L89DF 010401 1804 1805
L89E0 73 1805 1806
L89E1 E1 1806 1807
L89E2 F1 1807 1808
L89E3 C9 1808 1809
L89E4 1809 1810
L89E5 FB4A00 1810 1811
L89E6 086401 1811 1812
L89E7 1812 1813
L89E8 086400 1813 1814
L89E9 1814 1815
L89EA FB4A00 1815 1816
L89EB 086401 1816 1817
L89EC 29 1817 1818
L89ED 73 1818 1819
L89EE E1 1819 1820
L89EF 72 1820 1821
L89F0 83 1821 1822
L89F1 1822 1823
L89F2 C9 1823 1824
L89F3 83 1824 1825
L89F4 1825 1826
L89F5 83 1826 1827
L89F6 1827 1828
L89F7 83 1828 1829
L89F8 1829 1830
L89F9 C1 1830 1831
L89FA C1 1831 1832
L89FB E1 1832 1833
L89FC F1 1833 1834
L89FD C9 1834 1835
L89FE 1835 1836
L89FF 1836 1837
L8A0 010400 1837 1838
L8A1 086401 1838 1839
L8A2 086400 1839 1840
L8A3 73 1840 1841
L8A4 C1 1841 1842
L8A5 C1 1842 1843
L8A6 83 1843 1844
L8A7 1844 83

\( R16 \) is how many bytes to zero
L849 F5 1845
L849 F6 1846
L849 F7 1847
L849 F8 1848
L849 F9 1849
L849 FA 1850
L849 FB 1851
L849 FC 1852
L849 FD 1853
L849 FE 1854
L849 FF 1855
L850 00 1856
L850 01 1857
L850 02 1858
L850 03 1859
L850 04 1860
L850 05 1861
L850 06 1862
L850 07 1863
L850 08 1864
L850 09 1865
L850 0A 1866
L850 0B 1867
L850 0C 1868
L850 0D 1869
L850 0E 1870
L850 0F 1871
L850 10 1872
L850 11 1873
L850 12 1874
L850 13 1875
L850 14 1876
L850 15 1877
L850 16 1878
L850 17 1879
L850 18 1880
L850 19 1881
L850 1A 1882
L850 1B 1883
L850 1C 1884
L850 1D 1885
L850 1E 1886
L850 1F 1887
L850 20 1888
L850 21 1889
L850 22 1890
L850 23 1891
L850 24 1892
L850 25 1893
L850 26 1894
L850 27 1895
L850 28 1896
L850 29 1897
L850 2A 1898
L850 2B 1899
L850 2C 1900
L850 2D 1901
L850 2E 1902
L850 2F 1903
L850 30 1904
L850 31 1905
L850 32 1906
L850 33 1907
L850 34 1908
L850 35 1909
L850 36 1910

84
CALLING SEQUENCE

1911 : LXI D; DONE
1912 : CALL MSG
1913 : DIX D; done
1914 : DB LTH; DONE
1915 : LTH eq + (DONE+1)

1916 : PULL
1917 : PULL
1918 : PULL
1919 : PULL
1920 : PULL
1921 : PULL
1922 : PULL
1923 : PULL
1924 : PULL
1925 : PULL
1926 : PULL
1927 : PULL
1928 : PULL
1929 : PULL
1930 : PULL
1931 : PULL
1932 : PULL
1933 : PULL
1934 : PULL
1935 : PULL
1936 : PULL
1937 : PULL
1938 : PULL
1939 : PULL
1940 : PULL
1941 : PULL
1942 : PULL
1943 : PULL
1944 : PULL
1945 : PULL
1946 : PULL
1947 : PULL
1948 : PULL
1949 : PULL
1950 : PULL
1951 : PULL
1952 : PULL
1953 : PULL
1954 : PULL
1955 : PULL
1956 : PULL
1957 : PULL
1958 : PULL
1959 : PULL
1960 : PULL
1961 : PULL
1962 : PULL
1963 : PULL
1964 : PULL
1965 : PULL
1966 : PULL
1967 : PULL
1968 : PULL
1969 : PULL
1970 : PULL
1971 : PULL
1972 : PULL
1973 : PULL
1974 : PULL
1975 : PULL
1976 : PULL
1977 : PULL
1978 : PULL
1979 : PULL
1980 : PULL
1981 : PULL
1982 : PULL
1983 : PULL
1984 : PULL
1985 : PULL
1986 : PULL
1987 : PULL
1988 : PULL
1989 : PULL
1990 : PULL

85
FILE: GTRAJ FORTRAN A1 PRINCETON UNIVERSITY TIME-SHARING SYSTEM

COMMON/COM/C(2000)
COMMON/DEGREE/DUMMY2(5)
COMMON/RK/DUMMY4(112)
COMMON/RUNOUT/DUMMY8(1)

235 CONTINUE
CALL MYINIT
CALL MYRUN
GOTO 235
END
SUBROUTINE MYINIT
CALL INIT
RETURN
END
SUBROUTINE MYRUN
COMMON/COM/C(2000)
COMMON/RUNCUT/DAN
COMMON/SINAV/XST(7),YST(7),ZST(7),DIREQ(7),ISTDME,ISTVCR
1,R1(40),NOYES(40),SIGMA(40)
COMMON/DOUBLE/DSEED,DDS
DOUBLE PRECISION DSEED,DDS
EQUIVALENCE (C(203),TIME),(C(207),NSTEP)
EQUIVALENCE (C(211),TIMPR),(C(212),TIMPLT),(C(213),TIMTTY)
EQUIVALENCE (C(241),J1),(C(244),TIMSOF)

C***************TEMPORARILY:
DO 259 I=1,40
  259 R1(I)=0.
C
DAN=0.
C
CALL ERROR
C
242 CONTINUE
IF (TIME.LT.TIMSOF) GOTO 82
DAN=2.
CALL OUTPT
CALL FINT
GOTO 100
C
82 CONTINUE
C
C***NOISE HAS TO BE DRAWN FROM RANDOM GEN FOR EACH CHANN EVERY DT AND
C***BE AVAILABLE FOR SUBROUTINE OUTPT AT WHATEVER REQUIRED RECORDING
C***INSTANTS.
C*****CALL GGNPM(DSEED,30,R1)
  DSEED=DSEED+DDS
C
CALL LOGIC
C***R.K. LOOP
C
DO 246 J=1,4
CALL ROTAT
CALL DYN
C
IF (TIME.LE.0.) CALL OUTPT
CALL RKG
246 CONTINUE
C
NSTEP = NSTEP + 1
IF (TIME.LE.TIMPR. AND. TIME.LE.TIMPL. AND. TIME.LE.TIMITY) GOTO 242
CALL OUTPT
GOTO 242
100 CONTINUE
RETURN
END
BLOCK DATA
COMMON / DEGREE / ZETA (50), PSI0J, PHI0J, WPOJ (50), WBOJ (50)
COMMON / PROG / TACCX (50), TACCY (50), TACZZ (50), TPRCG (50),
11WC (50), WRC (50), WC (50), NTIME, TSWITCH (15), ISWITCH
COMMON / COM / C (2000)
COMMON / ALBET / XBETA, ZBETA, XALPHA, YALPHA
COMMON / SINAV / XST (7),YST (7), ZST (7), DIREQ (7), ISTDME, ISTDV
1R1 (40), NOYES (40), SIGMA (40)
COMMON / DEDS / DSEED, DDS
DOUBLE PRECISION DSEED, DDS
C
EQUIVALENCE (C (202), NRAJE)
EQUIVALENCE (C (214), DTPR), (C (215), DTFY), (C (216), DTTY)
EQUIVALENCE (C (220), IPR), (C (244), TIMSOF)
EQUIVALENCE (C (341), XE0), (C (347), XEO), (C (348), YEO)
EQUIVALENCE (C (277), USPEDO), (C (278), VSPEDO), (C (279), WSPEDO)
C
DATA H0/116./, TIMSOF/60./, TETO/10./, PSI0J/0./, PHI0J/12.7/, 1NTIME/15/, USPEDO/125., WSPEDO/0., WSPEDO/12./
2,TPROP/0.1.,60.1.,61.1.,101.1.,160.1.,161.1.,43.600.1/
6Q0/2.6.200.2.6.4400.,/,
7WPOJ/2.2.2.5,200.2.5,4400.,/
9WBOJ/2.2.2.86,200.2.2.2.86,4400.,/
ETA0J/2.193,2.096,2.193,2.,206.422.2.,/
CTA0J/40.0.,10.0.,/
DTA0J/2.1.,003.2.,994.2.,1.003.2.985.422.1./
DATA NREAE/20/, DTPR/400./, DIPL/400./, DTTY/1./,
1IPR/1/
2, XBETA/0./, ZBETA/0./, XALPHA/0./, YALPHA/0./
3,XST/7*120000./
4,YST/7*120000./
5,ZST/7*00./
6,DIREQ/7*32./
7, ISTDME/1./, ISTDV/1./
DATA DSEED/1.0D0/, DDS/1.0D0/, NOYES/40*0/, SIGMA/40*0./
DATA ISWITCH/1./, TSWITCH/15*600./
1, XEO/0./, YEO/0./
END
SUBROUTINE INPT
INTEGER FILE (6), GO
LOGICAL SOF

DIMENSION NAME(3), LNAME(4)

COMMON/COM/C(2000)
COMMON/DEGREE/TETO, QO(50), PSIOJ, PHI0J, WPOJ(50), WROJ(50)
COMMON/PROG/TACCX(50), TACCY(50), TACCC(50), I2PROG(50),
WPC(50), WRC(50), WQC(50), NTIME, TSWTCH(15), TSWTCH
COMMON/ALBE1/XBETA, ZBETA, XALPHA, YALPHA
COMMON/STNAV/XST(7), YST(7), ZST(7), DIREQ(7), ISTDME, ISTVOR
COMMON/PRCG/1ACCX(50), TACCY(50), TACCZ(50), PHIOJ(50),
P1C(50), WRC(50), WQC(50), NTIME, TSWTCH(15), ISWTCH
COMMON/ALBE1/XBETA, ZBETA, XALPHA, YALPHA
COMMON/STNAV/XST(7), YST(7), ZST(7), DIREQ(7), ISTDME, ISTVOR

EQUIVALENCE (C(202), NRATE)
EQUIVALENCE (C(214), DTPR), (C(215), DTFLT), (C(216), DTTY)
EQUIVALENCE (C(220), IPR), (C(243), IF2), (C(244), TIMSOF)
EQUIVALENCE (C(341), H0), (C(347), XEO), (C(348), YEO)
EQUIVALENCE (C(277), WSPEDO), (C(278), VSPED0), (C(279), WSPE E0)

NAMELIST/INE/FILE
NAMELIST/INCN/H0, TETO, QO, TIMSOF, PSIOJ, PHI0J, WPOJ, WROJ,
TACCX, TACCY, TACCC, NTIME, TPROG, USPED0, VSPED0, WSPED0,
2, XBETA, ZBETA, XALPHA, YALPHA
3, XST, YST, ZST, DIREQ, ISTDME, ISTVOR
4, TSWTCH, XEC, YEO

EQUIVALENCE (C(277), USPEDO), (C(278), VSPEDO), (C(279), WSPE E0)

DATA NAME/4HINCN, 4HPARM, 4H, 4H, 4H, 4H, 4H, 4H

CONTINUE
PRINT 502
502 FORMAT (1H4, 'TO CONTINUE ENTER F, TO STOP ENTER T')
READ 503, SCF
503 FORMAT (1L1)
IF (SCF) STOP
PRINT 500
500 FORMAT (1H4, '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

IF (IF1.EQ.5) PRINT 501, LNAME
501 FORMAT (1H4, 'ENTER NAMELISTS ', A4, A4, 'IN THIS ORDER'), (/)
IF (IF3.NE.0) READ(IF1, INCN)
IF (IF4.NE.0) READ(IF1, PARM)

PRINT 504
504 FORMAT (1H4, 'TO RUN ENTER 1, TO MODIFY INPUT'), DATA ENTER 0)
READ 505, 60

89
FILE: GT RAJ
FORTRAN A1 PRINCETON UNIVERSITY TIME-SHARING SYSTEM

505 FORMAT(I1)
1F(G0.,NE.1)GCTO 235
RETURN
END

SUBROUTINE INIT
DIMENSION IPL(100), IPD(100)
COMMON/COM/C(2000)
COMMON/DEGREE/TETO,Q0(50), PSIOJ, PHI0J, W0J(50), W0J(50)
COMMON/PROG/TACCX(50), TACCY(50), TACCZ(50), TPROG(50), WPC(50),
WRB(50), WQC(50), NTIME, TSWITCH(15), TSWITCH
COMMON/RK/ARK(4), BBK(4), CRK(4), QRK(100)

EQUIVALENCE (C(201), N)
EQUIVALENCE (C(202), NRATE), (C(203), TIME), (C(204), TIME)
EQUIVALENCE (C(205), DT), (C(207), NSTEE)
EQUIVALENCE (C(211), TIMPR), (C(212), TIMPLT), (C(213), TIMITY)
EQUIVALENCE (C(214), DTPR), (C(215), DTEL), (C(216), DTNT)
EQUIVALENCE (C(217), LINE), (C(218), NPRINT), (C(219), NPRINT)
EQUIVALENCE (C(222), NSQ2)
EQUIVALENCE (C(242), CRAD), (C(370), GRAV1), (C(319), THET0)
EQUIVALENCE (C(318), PHI0), (C(320), PSIO)

GRAV1=32.17

ARK(1) = .5
ARK(2) = 1. - 1./SQRT(2.).
ARK(3) = 1. + 1./SQRT(2.).
ARK(4) = 1./6.
BBK(1) = 2.
BBK(2) = 1.
BBK(3) = 1.
BBK(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.

NSQ2=1

PI=4.*ATAN(1.)
CRAD=180. / PI
THET0=TETO/CRAD
PSIO=PSIOJ/CRAD
PHIO=PHIOJ/CRAD
DO 246 I=1, NTIME
WPC(I)= W0J(I)/CRAD
WQC(I)= Q0(I)/CRAD
246 WRC(I)= WROJ(I)/CRAD

IPI(1)=203
IPD(1)=204
N=1
C(1) = IPL(1)
C(101)=IPD(1)
TIME=0.
TIMED=1.
NSTEP=0
NRAT=NRA1
DT=1./FLOAT(NRAT)

C
NPRINT=0
NPIOT=0
LINK=60
TIMPR=DTPR-.5*DT
TIMPLT=DTPLT-.5*DT
TIMTTY=DTTTY-.5*DT
IF(DTTY.GT.50.) TIMTTY=1000.

C
RETURN
END
SUBROUTINE ERROR
PRINT 507
507 FORMAT(1H,'RUNNING NOW',/) RETURN
END
SUBROUTINE DYNAMI
DIMENSION IPL(100),IPD(100)
COMMON/COM/C(2000)
EQUIVALENCE (C(201),N)
EQUIVALENCE (C(205),DT)
EQUIVALENCE (C(311),TET), (C(319),THET0), (C(312),PHI)
EQUIVALENCE (C(318),PHIO), (C(310),PSI), (C(320),PSIO)
EQUIVALENCE (C(334),XE), (C(335),YE), (C(336),ZE)
EQUIVALENCE (C(341),HO), (C(343),HM), (C(347),XEO), (C(348),YEO)
EQUIVALENCE (C(271),USPEED), (C(277),USPEED0), (C(272),VSPEED)
EQUIVALENCE (C(278),WSPEDO), (C(273),WSPEED), (C(279),WSPEED)

C
N=N+1
IPL(N)=310
IPD(N)=314
N=N+1
IPL(N)=311
IPD(N)=315
N=N+1
IPL(N)=312
IPD(N)=316
N=N+1
IPL(N)=271
IPD(N)=274
N=N+1
IPL(N)=272
IPD(N)=275
N=N+1
IPL(N)=273
IPD(N)=276
N=N+1
IPL(N)=334
IPD(N)=337
N = N + 1
IFL(N) = 335
IPD(N) = 338
N = N + 1
IFL(N) = 336
IPD(N) = 339

C
TET = THETO
PSI = PSI0
PHI = PHI0
XE = XE0
YE = YE0
ZE = HO
HM = HO

C
USPEED = USPEED0
VSPEED = VSPED0
WSPEED = WSPEED0

C
DO 111 I = 2, N
C(I) = IPL(I)
111 C(100 + I) = IPD(I)

C
RETURN
END

SUBROUTINE LOGIC
COMMON/CCH/C(2000)
COMMON/PROG/TACCX(50), TACCY(50), TACCZ(50), TPEOG(50),
1WPC(50), WRC(50), WQC(50), N TIME, 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(TSWTCH.EQ.1) GOTO 12
IF(TSWTCH.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(SLANKS BETWEEN VALUES),

92
C*** AND A SEPARATE PROGRAM CONVERTS IT TO A TIME-VECTOR-RECORD FILE.

C

WQ = SQUINT (TIME, TPROG, WQC, NTIME, 1)
WP = SQUINT (TIME, TPROG, WPC, NTIME, 1)
WR = SQUINT (TIME, TPROG, WRC, NTIME, 1)
ACCY = SQUINT (TIME, TPROG, TACCY, NTIME, 1)
ACCA = SQUINT (TIME, TPROG, TACCA, NTIME, 1)
ACCZ = SQUINT (TIME, TPROG, TACCZ, NTIME, 1)
RETURN
END
SUBROUTINE DYNAM

COMMON/COM/C(2000)
COMMON/REC/A3(2000,20)

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), SNF)
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), VSPED), (C(273), WSPEED)
EQUIVALENCE (C(274), DUSPEED), (C(275), DVSPEED), (C(276), DWSPEED)
EQUIVALENCE (C(370), GRAV1)

C

DPSI = (WB*CSF+WQ*SNF)/CST
DTET = WQ*CSF-WR*SNF
DPHI = WP*DPSI*SNT
CALL DBTEL (USPEED, VSPEED, WSPEED, CEB11, CEB12, CEB13, CEB21, CEB22, CEB23, CEB31, CEB32, CEB33, VX, VY, VZ)
XED = VX
ZED = VZ
YED = VY
DUSPEED = -WQ*WSPEED+WR*VSPEED-GRAV1*SNT + ACCX*GRAV1
DVSPEED = -WB*USPEED+WP*VSPEED+GRAV1*CST*SNF+ACCY*GRAV1-ASSUME COORD.
DWSPEED = 0.

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), NPLT), (C(220), IEB), (C(243), IF2), (C(242), CRAD)
EQUIVALENCE (C(212), TIMPLT)

C

ENTRY FIN
WRITE (IF2, 516)
516 FORMAT (1H5, "TIME IS OVER", //)

93
FILE: GTRAJ1 FORTRAN A1 PRINCETON UNIVERSITY TIME-SHARING SYSTEM

C
IP(NPLOT.LT.2)G0TO 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 (1F2,517)NPLOT

517 FORMAT (1H ,115)

518 CONTINUE
RETURN

C

SUBROUTINE CUTPT

COMMON/COM/C(2000)
COMMON/RUNCUT/DAN
COMMON/REC/A3(2000,20)
DIMENSION RME(7),VOR(7)

COMMON/ALPHA,XBETA,2BETA,XALPHA,YALPHA
COMMON/STNAV/XST(7),YST(7),DIREQ(7),ISTDME,ISTVOR

1,R1(40),NOYES(40),SIGMA(40)

EQUIVALENCE (C(336),ZE)
EQUIVALENCE (C(203),TIME),(C(211),TIMEP),(C(212),TIMPLT)
EQUIVALENCE (C(213),TIMT),(C(214),DTR),(C(215),DPLT)
EQUIVALENCE (C(216),DTTY),(C(217),LINE),(C(218),NPRINT)
EQUIVALENCE (C(219),NPLLOT),(C(242),CRAD),(C(243),IPLT)
EQUIVALENCE (C(302),WQ),(C(321),VX),(C(323),VZ),(C(334),XE)
EQUIVALENCE (C(343),HN),(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

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)
THEPJ=CRAD*TET+NOYES(5)*SIGMA(5)*R1(5)
PHIJ=CRAD*PHI+NOYES(6)*SIGMA(6)*R1(6)
VAIR=SQR(T(USPEED**2+VSPEED**2+WSPEED**2)+NOYES(7)*SIGMA(7)*R1(7)
VBP=VSPEED+WP*XBETA-WP*ZBETA
USPD=USPEED
BETA=ATAN2(WRP,USPD)
BETAJ=CRAD*BETA+NOYES(8)*SIGMA(8)*R1(9)
WQR=USPEED*Q*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 RDMJ=SQR((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
RUNIT=X=CO5(REQ)
FILE: GTBAJ1 FORTRAN A1 PRINCETON UNIVERSITY TIME-SHARING SYSTEM

RUNITY=SIN(REQ)
VCOS=ABS((XVCR*RUNITX+YVOR*RUNITY)/SQR(T(XVOR**2+YVOR**2)))
RVCR=ARCOS(VCOS)
RVOR=CRAD*RVCRI+NOYES(10+ISTDME+1)*SIGMA(10+ISTDME+1)^1
1R1(10+ISTDME+1)
302 VOR(I)=SATP(RVOR,10.)
VXY=SQR(T(VX*VX+VY*VY)
VZN=-VZ
GAMV=ATAN2(VZN,VXY)
GAMVJ=CRAD*GAMV
GAMH=ATAN2(VY,VX)
GAMHJ=CRAD*GAMH
ACCX=ACCX+NOYES(10+ISTDME+ISTVOR+1)*SIGMA(10+ISTDME+ISTVOR+1)*
1R1(10+ISTDME+ISTVOR+1)
ACCY=ACCY+NOYES(10+ISTDME+ISTVOR+2)*SIGMA(10+ISTDME+ISTVOR+2)*
1R1(10+ISTDME+ISTVOR+2)
ACCZ=ACCZ+NOYES(10+ISTDME+ISTVOR+3)*SIGMA(10+ISTDME+ISTVOR+3)*
1R1(10+ISTDME+ISTVOR+3)
C***NOW,IF TRIGGERED-CONTAMINATION BY NOISE;IF NOT-BYPASSED.
C
C***OUTPUT OPTIONS:
C*** 1,SHORT (TERMINAL) PRINTOUT;
C*** 2, LONG PRINTOUT OF FIRST BUNCH,SECOND OR BOTH;
C*** 3,CREATION OF 1SEC-INTERVAL-FILE OF DATA(SAME AS FLIGHT FILE)-
C*** - TO BE PRINTED, PLOTTED OR FURTHER PROCESSED.
C
IF(TIME.GT.0.AND.DAY.LT.1.)GOTO 879
PRINT 508 WRITE(IF2,511)TIME,VAIR,GAMVJ,GAMHJ,HM,XE,YE,PSIJ,THETJ,
1PHIJ,VOR(1),RDME(1),ALPHAJ,BETAJ
879 CONTINUE
IF(TIME.LE.TIMPR)GOTO 241
IF(LINE.NE.60)GO TO 259
WRITE(IF2,508)
508 FORMAT(1H1,* TIME VAIR GAMVJ GAMEJ HM XE YE PSIJ THETJ PHIJ VOR(1) RDME(1) ALPHA J BETAJ)
NPRINT=NPRINT+1
LINE=1
259 CONTINUE
WRITE(IF2,511)TIME,VAIR, GAMVJ, GAMHJ, HM, XE, YE, PSIJ, THETJ, PHIJ, VOR(I),RDME(I),ALPHAJ,BETAJ
511 FORMAT(1H1,* TIME TIMPR+ITPR
LINE=LINE+1
241 IF(TIME.LE.TMPLT.OR.NPLOT.GE.2000)GOTO 247
NPLOT=NPLOT+1
A3(NPLOT,1)=TIME
A3(NPLOT,2)=PHIJ
A3(NPLOT,3)=WQJ
A3(NPLOT,4)=WRJ
A3(NPLOT,5)=ACCX
A3(NPLOT,6)=ACCY
A3(NPLOT,7)=ACCZ
A3(NPLOT,8)=HM
95
A3(NPLOT,9)=VAVR
A3(NPLOT,10)=PSIJ
A3(NPLOT,11)=THETJ
A3(NPLOT,12)=PHIJ
A3(NPLOT,13)=VOR(1)
A3(NPLOT,14)=EME(1)
A3(NPLOT,15)=ALPHAJ
A3(NPLOT,16)=BETAJ

TIME=TIME+DT
247 IF(TIME.LE.TIMTTY)GOTO 240

IF(TIME.LT.(1.5*DTTTY))PRINT 510

510 FORMAT(1H1,' TIME VAVR GAMVJ GAMHJ XE YE GTF 1
HM')
PRINT 513,TIME,VAVR,GAMVJ,GAMHJ,XE,YE,HM

513 FORMAT(1H7F10.1)
TIMTTY=TIMTTY+DTTTY

240 CONTINUE

C
RETURN
END

FUNCTION SATF(X11,XM11)
SATF=SIGN(MIN(ABS(X11),XM11),X11)
RETURN
END

SUBROUTINE BCTAT
COMMON/COM/C(2000)

EQUIVALENCE (C(311),TET), (C(351),CEE11), (C(353),CEE13)
EQUIVALENCE (C(357),CEB31), (C(359),CEB33)
EQUIVALENCE (C(441),CST), (C(442),SNF)
EQUIVALENCE (C(310),PSI), (C(312),PHI), (C(352),CEE12)
EQUIVALENCE (C(354),CEB21), (C(355),CEB22), (C(356),CEB23)
EQUIVALENCE (C(358),CEB32), (C(443),CSF), (C(444),SNP)
EQUIVALENCE (C(439),CSF), (C(440),SNF)

C
SNT=SIN(TET)
CST=COS(TET)
SNF=SIN(PHI)
CSF=COS(PHI)

CEE11=CST*CSP
CEE13=-SNT
CEE31=+SNT*CSF*CSP+SNF*SNP
CEB33=CST*CSF
CEE12=CST*SNP
CEB21=SNF*SNT*CSF-CSF*SNP
CEB22=SNF*SNT*SNP+CSF*CSP
CEB23=SNF*CST
CEE32=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
FILE: GTEAJ1 FORTRAN A1 PRINCETON UNIVERSITY TIME-SHARING SYSTEM

ZO = A31 * XI + A33 * ZI + A32 * YI
YO = A21 * XI + A22 * YI + A23 * ZI
RETURN
ENTRY DBTCI (XI, YI, ZI, A11, A12, A13, A21, A22, A23, A31, A32, A33,
1 X0, YO, Z0)
X0 = A11 * XI + A31 * ZI + A21 * YI
ZO = A13 * XI + A33 * ZI + A23 * YI
YO = A12 * XI + A22 * YI + A32 * ZI
RETURN
END
SUBROUTINE BKG
DIMENSION IPL (100), IPD (100)
COMMON/RK/ARK (4), BRK (4), CRK (4), QRK (100)

C EQUIVALENCE (C(201), N)
C EQUIVALENCE (C(205), H), (C(241), J)

C DO 100 I = 1, N
IL = C (I)
ID = C (100 + I)
X1 = C (ID) * H
X2 = (X1 - BRK (J) * QRK (I)) * ARK (J)
C (IL) = C (IL) + X2
100 QRK (I) = QRK (I) + 3. * X2 - CRK (J) * X1
RETURN
END
FUNCTION SQUINT (X, TABX, TABY, NTAB, N)
DIMENSION TABX (NTAB), TABY (NTAB)
IF (NTAB .NE. 1) GO TO 1
SQUINT = TABY (NTAB)
RETURN
1 IF (X - TABX (N)) .LT. 3, 4
2 N = N - 1
2 IF (N) 9, 9, 1
3 SQUINT = TABY (N)
FACTOR = 0.
RETURN
4 IF ((N + 1) .GT. NTAB) GO TO 10
IF (X - T A B X (N + 1)) .LT. 5, 6, 7
6 N = N + 1
GOTO 3
7 N = N + 1
GOTO 4
5 FACTOR = (X - T A B X (N)) / (T A B X (N + 1) - T A B X (N))
SQUINT = (T A B Y (N + 1) - T A B Y (N)) * FACTOR + T A B Y (N)
RETURN
9 PRINT 1000, X, TABX (1)
Y = SQRT (- 1.)
STOP
1000 FORMAT (1H, 10 (" "), 'SQUINT UNDERFLOW - INPUT = ', E15.8, ' LESS THAN IN FIRST ARG. TABLE', ENTRY (' ', E15.8, ' )')
10 PRINT 2000, X, TABX (NTAB)
Y = SQRT (- 1.)
STCP
2000 FORMAT(1H,10('$'), 'SQUIN1 OVERFLOW INPUT =', 'E15.8', ' GREATER THAN LAST ARG. TABLE ENTRY (', 'E15.8', ')')
END
OPTIMAL FLIGHT PATH RECONSTRUCTION (B.3)

FILE: OPTFILT FORTRAN A
PRINCETON UNIVERSITY TIME-SHARING

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,S2ALP,S2BET,S2H,S2DME,S2VOR,S2BDME,S2ZR,PERCNT,OMEGA,AGLOBE
4,OUTL1,OUTL2,S2QA,S2QB
2,EPS2,TLAMO,SMEWO,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,S2ALP,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,TLAMO,SMEWO,S2WNDZ,S2BDME,S2ZR,S2WNDY,KDME2
DIMENSION P(11,11),PO(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),UNKH(11,11),PLUS(11,11)
4,CKALMT(6,11),UNKHT(11,11),STAK1(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(5,0K)
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)
FILE: OPTFILT FORTRAN A

CRAD=180.0D0/PI

DO 17 I=1,NXP

17 XO(I)=X00(I)

KSTEP=0

KSTEP1=0

DO 101 I=1,NXP

DO 102 J=1,NXP

AIDEN(I,J)=0.0D0

102 PO(I,J)=0.0D0

101 CONTINUE

C*****

DO 109 I=1,NXP

109 AIDEN(I,I)=1.0D0

DO 104 I=1,6

DO 105 J=1,6

105 RK(I,J)=0.0D0

104 CONTINUE

RK(1,1)=S2V

RK(2,2)=S2ALP

RK(3,3)=S2BET

RK(4,4)=S2H

RK(5,5)=S2DME

RK(6,6)=S2DME

C***

DO 461 I=1,7

XST7(I)=XST(I)

YST7(I)=YST(I)

ZST7(I)=ZST(I)

461 DIREQ7(I)=DIREQ(I)

ISTV07=ISTV0B

ISTDM7=ISTDME

NRAT7=NRATE

TIMS7=TIMSOF

NXP7=NXP

S2LI7=S2LIAC

C***

IDME2=1

IDME=1

301 CONTINUE

KSTEP=KSTEP+1

KSTEP1=KSTEP1+1

C*

IFLAG=1

C***

IF(KDME(IDME).GT.KSTEP)GOTO 807

C*

DO 817 I=1,NXP7

DO 817 J=1,NXP7

817 IF(I.NE.J)PO(I,J)=0.0D0

IFLAG=5

C*

XST7(1)=XST(NDME1(IDME))

YST7(1)=YST(NDME1(IDME))

ZST7(1)=ZST(NDME1(IDME))

IDME=IDME+1
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)=0.000
IFLAG=5
XST7(2)=XST(NDME2(IDME2))
YST7(2)=YST(NDME2(IDME2))
ZST7(2)=ZST(NDME2(IDME2))
IDME2=IDME2+1
827 CONTINUE

C* CALLING THE PROPAGATION-BETWEEN-MEASUREMENTS SUBROUTINE
C***
CALL PROP27(TIMS7,PERCNT,OMEGA,AGLOBE,EPS2,S2WNDY,TLAMO
1,SNEW0,S2WIND,S2WNDZ,X0,P0,XST7,YST7,ZST7,DIREQ7,S2ZR,S2BDME
3,OUT1,OUT2,S2QA,S2QB
2,S2L7,S2XYZR,YOUT,P,HB,RES,IFLAG,NDAR7,NXP7,ISTDM7,ISTVO7,KSTEP)
DO 201 I=1,6
DO 202 J=1,NXP
201 CONTINUE
HBT(J,I)=HB(I,J)
202 CONTINUE
CALL VMULFF(P,HBT,NXP,6,NXP,NXP,PHT,NXP,IER1)
CALL VMULFF(HB,PHT,6,NXP,6,NXP,HPHT,6,IER2)
DO 211 I=1,6
DO 212 J=1,6
211 CONTINUE
HPHTR(I,J)=HPHT(I,J)+RK(I,J)
212 CONTINUE
C***
DO 341 I=1,6
341 SUM(I)=SUM(I)+HPHTR(I,I)
C***
CALL LINVP(HPHTR,6,6,HPHTRI,IDGT,WKAREA,IER3)
CALL VMULFF(PHT,HPHTRI,NXP,6,NXP,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,6,CH,NXP,IER6)
DO 241 I=1,NXP
DO 242 J=1,NXP
241 CONTINUE
242 UNMKH(I,J)=AIDEN(I,J)-CH(I,J)
C***
CALL VMULFF(UNMKH,P,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(I,J)=CKALM(I,J)
... CALL VMULFF(P,UNMKHT,NXP,NXP,NXP,NXP,NXP,STABK1,NXP,IER7)
CALL VMULFF(UNMKH,STABK1,NXP,NXP,NXP,NXP,STABK2,NXP,IER8)
CALL VMULFF(RK,CKALMT,6,6,NXP,6,6,STABK3,6,IER9)

101
CALL VMULFP(CKALM,STABK3,NXP,6,NXP,NXP,6,STABK4,NXP,IER10)
DO 282 I=1,NXP
DO 282 J=1,NXP
282 PPLUS(I,J)=STABK2(I,J)+STABK4(I,J)
C***
C***
IF(KSTEP1.LT.5) GOTO 253
KSTEP1=0
DO 251 I=1,NXP
DO 252 J=1,NXP
252 PPLUS(I,J)=.5DO*(PPLUS(I,J)+PPLUS(J,I))
252 CONTINUE
251 CONTINUE
253 CONTINUE
C*****
DO 741 I=1,NXP
741 X0(I)=XPLUS(I)
DO 742 J=1,NXP
742 PO(I,J)=PPLUS(I,J)
C***
KSTEPA=KSTEP
C***
WRITE(12,KSTEPA) (XPLUS(I),I=1,NXP)
1, (XOUT(I),I=1,NXP), ((PPLUS(I,J),I=1,NXP),J=1,NXP)
2, ((P(I,J),I=1,NXP),J=1,NXP)
C***
IF(KSTEP.EQ.NSTEP) GOTO 311
C***
GOTO 301
C***
311 CONTINUE
C***
IF(NSTEP.GT.60) GOTO 1
DO 342 I=1,6
342 PREDCT(I)=DSQRT(SUM(I)/59)
PREDCT(2)=57.3*PREDCT(2)
PREDCT(3)=57.3*PREDCT(3)
PRINT 343, (PREDCT(I),I=1,6)
343 FORMAT(' ',F12.0,2F12.1,JF12.0)
C***
GOTO 1
92 CONTINUE
STOP
END
BLOCK DATA
IMPLICIT REAL*8 (A-H,O-Z)
COMMON/DAT0/S2LIAC,S2XYZR,TIMSOF,X00(11),S2WIND,S2WINDZ,S2WINDT
1,S2V,S2ALF,S2BET,S2H,S2DME,S2VOR,S2BDME,S2ZM,SERCNT,OMEGA,AGLOBE
4,OUTL1,OUTL2,S2QA,S2QB
2,EP52,TLAM0,SMENG,0ST,0ST,0ST,0ST,0ST,0ST,0ST,0ST,0ST,0ST,0ST,0ST
3,KDREQ(20),NXP,NG3T,NSDF,IDDT,IDSTD,IDSTDF,SOF
DIMENSION XST(7),YST(7),ZST(7),DIREQ(7)
LOGICAL SOF
DATA SOF/.FALSE./,NXP/11/,S2LIA/=400/,NRATE/20/,TIMSOF/1.0D0/
1,X00/0.0D0,0.0D0,-116.0D0,125.0D0,0.0D0,12.0D0,5*0.0D0/
2,S2V/6.25D0/,S2ALF/.000076D0/,S2WINI/.10D0/,S2WINDZ/.00001D0/
3,S2BET/.000076D0/,S2H/25.0D0/,S2DME/40000.0D0/,S2VOR/.00004D0/
4,NSTEP/60/,IDGT/3/,S2XYZR/25.0D0/,S2ZR/.10D0/
6,ISTDME/2/,ISTVOR/2/,S2BDME/.000025D0/,S2WNY/-10D0/
7,XST/.703D0,.6985D0,5*.706D0/
8,YST/-1.297D0,-1.2985D0,5*-1.294D0/
9,ZST/7*0.0D0/
A,DIREQ/32.0D0,32.0D0,0.0D0,0.0D0,0.0D0,0.0D0,0.0D0/
B,KDME/1,19*2500/,NDME1/1,2,3,1,4,15*2/,NDME2/2,3,1,4,16*3/
C,OMENA/.0000728D0/,AGLOBE/20940000.0D0/,EPS2/.0067D0/
D,TLAMO/.700D0/,S2EWO/-1.300D0/
DATA PERCENT/.00001D0/
1,KDME2/1,19*2600/
2,OUTL1/20000.0D0/,OUTL2/20000.0D0/,S2QA/.81D0/,S2QB/.81D0/
END
C***STATE AND COVARIANCE MATRIX PROPAGATION BETWEEN MEASUREMENTS.

C

SUBROUTINE PROP27(TIMSOF, PERC1, OMEG1, AGL0B1, EPS1, S2WOY
1, TLAM00, SMEW00, S2WO, S2WZ, XO, P0, XST, YST, ZST, DIREQ, S2ZR1, S2BDM
3, OUT1T, OUTLT2, S2QAT, S2QBT
2, S2LIA, S2XXZ1, XOUT, P, HB, RES, IFLAG7, NRATE, NXP, ISTD1ME, ISTD0R, KSTEP)

IMPLICIT REAL*8(A-H,O-$)

EQUIVALENCE (IC(202), NRATE1), (C(311), TIMSF)
DIMENSION P0(11,11), XI(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)

EQUIVALENCE (C(1257), XST1(1)), (C(1271), YST1(1))
EQUIVALENCE (C(1299), DIREQ1(1)), (C(1285), ZST1(1))
EQUIVALENCE (IC(252), ISTD1ME), (IC(253), ISTD0R)
EQUIVALENCE (IC(259), KSTEP1)
EQUIVALENCE (C(505), OMEGA), (C(507), AGL0B1), (C(509), EPS2)
EQUIVALENCE (C(511), TLAM0), (C(513), SMEW0)
EQUIVALENCE (C(470), S2WIND)

EQUIVALENCE (C(482), WX), (C(484), WY), (C(486), WZ)
EQUIVALENCE (C(468), S2WIND)
COMMON/COM/C(2000)
COMMON/INCOM/IC(500)

EQUIVALENCE (C(312), PERCNT), (C(381), S2ZR), (C(382), S2BDME)
EQUIVALENCE (C(494), BDME1), (C(498), BDME2), (C(474), S2WIND)
EQUIVALENCE (IC(261), IFLAG), (C(502), OUT1T), (C(504), OUTL2)
EQUIVALENCE (C(506), S2QA), (C(509), S2QB)

OUTLT1=OUTLT2
S2QA=S2QAT
S2QB=S2QBT
IFLAG=IFLAG7
S2WIND=S2WOY
S2ZR=S2ZR1
S2BDME=S2BDM
PERCNT=PERC1

OMEGA=OMEG1
AGLOBE=AGLOB1
EPS2=EPS1
TLAM0=TLAM00
SMEW0=SMEW00
KSTEP1=KSTEP
NRATE1=NRATE
TIMSF=TIMSF
S2LIA=S2LIA
S2XYZ=S2XYZR
S2WIND=S2WO
S2WINDZ=S2WZ
NXP1=NXP
DO 61 I=1,NXP1
61 XO(I)=XO(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
ISTV01=ISTVOR
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-S)
CALL INIT
CALL DYNAMI
RETURN
END

SUBROUTINE MYRUN
IMPLICIT REAL*8(A-H,O-S)
COMMON/CROM/C(2000)
COMMON/COM/C(500)
EQUIVALENCE (C(305),TIME),(IC(207),NSTEP)
EQUIVALENCE (IC(241),J1),(C(311),TIMSP)
C
242 CONTINUE
  IF (TIME .. LT. TIMSP) 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
NSTEP=NSTEP+1
GOTO 242
100 CONTINUE
RETURN
END

SUBROUTINE INIT
IMPLICIT REAL*8(A-H,O-S)
REAL*4 Z1(11),XA1(6)
DIMENSION IPL(100),IPD(100)
COMMON/CROM/C(2000)
COMMON/COM/C(500)
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(11))
DIMENSION HB1(8,11),XA(6),Z(11)
EQUIVALENCE (IC(259),KSTEP1)
C
EQUIVALENCE (IC(201),N)
EQUIVALENCE (IC(202),NRATE21),(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)
FILE: OPTPROPG FORTRAN A

PRINCETON UNIVERSITY TIME-SHARING

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), S2BME), (C(474), S2WNY)
2, (IC(261), IFLAG), (C(502), OUTL1), (C(504), OUTL2)
3, (C(506), S2QA), (C(508), S2QB)

C
GRAV1=32.17DO

C
ARK(1)=.5DO
ARK(2)=1.0DO-1.0DO/D2RT(2.0DO)
ARK(3)=1.0DO+1.0DO/D2RT(2.0DO)
ARK(4)=1.0DO/6.0DO
BRK(1)=2.0DO
BRK(2)=1.0DO
BRK(3)=1.0DO
BRK(4)=2.0DO
CRK(1)=ARK(1)
CRK(2)=ARK(2)
CRK(3)=ARK(3)
CRK(4)=ARK(1)
DO 229 I=1,100

229 QRK(I)=0.0DO

C
PI=4.0DO*D2TAN(1.0DO)
CRAD=180.0DO/PI

C
IPL(1)=305
IPD(1)=307
N=1
TIME=0.0DO
TIMEED=1.0DO
NSTEP=0
NRAT=NRATE1

DT=1.0DO/D2FLOAT(NRAT)

C***

C
DO 306 I=1,NXP1
DO 307 J=1,NXP1

307 FB(I,J)=0.0DO

306 CONTINUE.

DO 308 I=1,8
DO 309 J=1,NXP1

309 HB1(I,J)=0.0DO

308 CONTINUE.

HB1(4,3)=-1.0DO

C*

HB1(5,10)=1.0DO
HB1(6,11)=1.0DO
DO 311 I=1,NXP1
DO 312 J=1,NXP1

312 QB(I,J)=0.0DO

311 CONTINUE.
FILE: OPTPROPG FORTRAN A

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 CONSTITUTES 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) (XAI(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)
OONR=Z (8)
OOR2=Z (9)
OONR2=Z (9)
GOTO 838
837 CONTINUE
-- IF (DABS (Z (8)-OONR1).LT.OUTL1) GOTO 839
-- Z (8)=OONR1+(OONR1-OOR1)
839 CONTINUE
OOR1=OONR1
OONR1=Z (8)
IF (DABS (Z (9)-OONR2).LT.OUTL2) GOTO 840
108
FILE: OPTPROPG FORTRAN A

PRINCETON UNIVERSITY TIME-SHARING

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), C3P), (C(339), SNP), (C(341), CST), (C(343), SNT)
1, (C(505), OMEGA), (C(507), AGLOBE), (C(509), EPS2)
2, (C(511), TLAM0)
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), BME1), (C(498), BME2), (C(312), PERCNT), (C(311), TMSF)
7, (C(313), WP), (C(315), WQ), (C(317), WR)
COMMON/FQ/PB(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), TPMTHT(11,11)
EQUIVALENCE (IC(251), NXP1), (C(551), I01(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), VSPEED), (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
**FILE: OPTPROPG FORTRAN**

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)
 201 DO 202 I=2,NXP1
 202 P1(I,2)=P01(I,2)
 202 DO 203 I=3,NXP1
 203 P1(I,3)=P01(I,3)
 203 DO 204 I=4,NXP1
 204 P1(I,4)=P01(I,4)
 204 DO 205 I=5,NXP1
 205 P1(I,5)=P01(I,5)
 205 DO 206 I=6,NXP1
 206 P1(I,6)=P01(I,6)
 206 DO 207 I=7,NXP1
 207 P1(I,7)=P01(I,7)
 207 DO 208 I=8,NXP1
 208 P1(I,8)=P01(I,8)
 208 DO 209 I=9,NXP1
 209 P1(I,9)=P01(I,9)
 209 DO 210 I=10,NXP1
 210 P1(I,10)=P01(I,10)
 210 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

C141=C112*(ZE/AGLOBE+.5*EPS2*COS2T)
C111=(CEB11*USPEED+CEB21*VSPEED+CEB31*WSPEED+WX)/AGLOBE
C211=(CEB12*USPEED+CEB22*VSPEED+CEB32*WSPEED+WT)/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*_.5*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

111
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 + 0.5D0 * EPS2 * SINT * SINT) * OMEGA * OMEGA
C4112 = (C1111 * SINT2 - EPS2 * SINT * COST * COST) * OMEGA * OMEGA
C4113 = (WSPEED * C412 - VSPEED * C411) / AGLOBE

1 - CEB11 * C4111 + CEB13 * C4112 + COST1 * (WX * CEB12 - WY * CEB11) + SINT1 * (WX * CEB13 - WZ * CEB12)

2 WX * CEB13 + WZ * CEB12

C51 = (WSPEED * C411 - VSPEED * C413) / AGLOBE

1 - CEB21 * C4111 + CEB23 * C4112 + COST1 * (WX * CEB22 - WY * CEB21) + SINT1 * (WX * CEB23 + WZ * CEB22)

2 WX * CEB23 + WZ * CEB22

C61 = (VSPEED * C413 - USPEED * C412) / AGLOBE

1 - CEB31 * C4111 + CEB33 * C4112 + COST1 * (WX * CEB32 - WY * CEB31) + SINT1 * (WX * CEB33 - WZ * CEB32)

2 WX * 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

112
C****
.
.
CALL CSTM(PB,PERCENT,TMSP,TMAT,B60,PKDT,SUME,SUME1,NXP1,INTERMS)
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 TMPTAT(I,J) = P1(I,J) + QB(I,J)
CALL VMULFF(TMPTAT,TMATT,NXP1,NXP1,NXP1,NXP1,NXP1,NXP1,PTMATT,NXP1,I1)
CALL VMULFF(TMATT,PTMATT,NXP1,NXP1,NXP1,NXP1,NXP1,NXP1,NXP1,NXP1,P1,NXP1,I2)
C
RETURN
END
SUBROUTINE DINA
IMPLICIT REAL*8(A-H,O-Z)
COMMON/COM/C(2000)
COMMON/INCOM/IC(500)
C****
.
.
EQUIVALENCE (IC(251),NXP1)
COMMON/FQ/PB(11,11),QB(11,11)
EQUIVALENCE (C(494),BDME1),(C(496),BDME2)
1,(C(498),BDME2),(C(500),BDME2)
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)
EQUIVALENCE (C(437), DP76), (C(438), DP86), (C(439), DP96)
EQUIVALENCE (C(440), DP77), (C(441), DP87), (C(442), DP97)
EQUIVALENCE (C(443), DP98), (C(444), DP99)
EQUIVALENCE (C(319), WPD), (C(321), WOD), (C(323), WRD)
DIMENSION DPM(11, 11), FP(11, 11), P1(11, 11)

C***
EQUIVALENCE (C(482), WX), (C(484), WY), (C(486), WZ)
EQUIVALENCE (C(488), DXW), (C(490), DWY), (C(492), DWZ)

C
EQUIVALENCE (C(315), WQ), (C(327), TET), (C(333), DTET)
EQUIVALENCE (C(499), VX), (C(503), VY)
EQUIVALENCE (C(469), XE), (C(473), ZE), (C(475), ZED), (C(479), ZED)
EQUIVALENCE (C(351), CEB11)
EQUIVALENCE (C(355), CEB13), (C(363), CEB31), (C(367), CEB33)
EQUIVALENCE (C(325), PSI), (C(329), PHI), (C(331), DPSI), (C(335), DPHI)
EQUIVALENCE (C(313), WP), (C(317), WR)
EQUIVALENCE (C(501), Y), (C(471), YE), (C(477), YED)
EQUIVALENCE (C(337), CSF), (C(339), SFP), (C(341), CST), (C(343), SNT)
EQUIVALENCE (C(353), CEB12), (C(357), CEB21), (C(359), CEB22)
EQUIVALENCE (C(361), CEB23), (C(365), CEB32)
EQUIVALENCE (C(493), ACCX), (C(495), ACCY), (C(497), ACCZ)
EQUIVALENCE (C(481), USPEED), (C(483), VSPED), (C(485), WSPED)
EQUIVALENCE (C(487), DUSPEED), (C(489), DVSPED), (C(491), DWSPEED)
EQUIVALENCE (C(301), GRAV1)
EQUIVALENCE (C(511), TLM0)
EQUIVALENCE (C(505), OMEGA), (C(507), AGLOBE), (C(509), EPS2)

C***
CALL DBTOI(USPEED, VSPEED, WSPEED, CEB11, CEB12, CEB13
, CEB21, CEB22, CEB23, CEB31, CEB32, CEB33, VX, VY, VZ)

C

DO 101 I=1, NXPI
DO 102 J=1, NXP1

IF (I .LE. J) GOTO 101

102 P1(I, J) = P1(I, J)

101 CONTINUE

TLAM = TLM0 + 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

C141 = C112*(ZE/AGLOBE-.5*EPS2*COS2T)
C1111 = (CEB11*USPEED+CEB21*VSPED+CEB31*WSPEED+WX)/AGLOBE
C211 = (CEB12*USPEED+CEB22*VSPED+CEB32*WSPEED+VX)/AGLOBE
C241 = -C112*YE*TANT/AGLOBE
C341 = C112*EPS2*SINT2
C56 = OMEGA*(CEB11*COST-CEB13*SINT)
C64 = OMEGA*(CEB21*COST-CEB23*SINT)
C45 = OMEGA*(CEB31*COST-CEB33*SINT)
PILE: OPTPROPG FORTRAN A PRINCETON UNIVERSITY TIME-SHARING

XED=VX+WX+C141*C111*AGLOBE
ZED=VZ+WX+C341*C111*AGLOBE
YED=VY+WX+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*SINT2
CENT2=-OMEGA**2*AGLOBE*C11111*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***
DUSPED=-WQ*WSPEED+WR*VSPEED+ACCX*GRAV1
1+C45*VSPEED-C64*WSPEED
2+C47*WX+C48*WY+C49*WZ+CENT1*CEB11+CENT2*CEB13

DVSPEED=-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***
RETURN
END
SUBROUTINE OUTPT
IMPLICIT REAL*8(A-H,O-$)
COMMON/COM/C(2000)
COMMON/INCOM/IC(500)
DIMENSION EDME(7),VOR(7)

C***
DIMENSION IST1(7),YST1(7),ZST1(7),DIREQ1(7)
EQUIVALENCE (C(1257),YST1(1)),(C(1271),YST1(1)),(C(1285),ZST1(1))
EQUIVALENCE (C(1299),DIREQ1(1)),(IC(252),ISTDM1),(IC(253),ISTV01)
EQUIVALENCE (C(993),HB1(1,1))
EQUIVALENCE (C(1235),ZCAL(1))
DIMENSION ZCAL(8),HB1(8,11)

C
EQUIVALENCE (C(309), CRAD)
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), YI), (C(471), YE)
EQUIVALENCE (C(493), ACCX), (C(495), ACCY), (C(497), ACCZ)
EQUIVALENCE (C(481), CSPEED), (C(483), VSPEED), (C(485), WSPEED)
EQUIVALENCE (C(507), AGLOBE), (C(509), EPS2)
EQUIVALENCE (C(511), TLANO), (C(513), SHEWO)

DIMENSION IDME(7), YDME(7), ZDME(7)

TLAN = TLANO * XE / AGLOBE
COST = DCOS (TLAM)
SINT = DSIN (TLAM)
SHEW = DSIN (CYST) / AGLOBE / COST
SINT2 = 2 * SINT * COST
C1111 = 1.0D0 - 5D0 * EPS2 * COS2T - ZE / AGLOBE
AC1111 = AGLOBE * C1111
COS2 = DCOS (SHEW)
SIN2 = DSIN (SHEW)
RX1 = EPS2 * SINT2 * COST * COS2 - C1111 * SINT * COS2 - C1111 * TANT * SINT * YE / AGLOBE
RX2 = EPS2 * SINT2 * COS2 - SINT * SINT * EPS2 * DSINS * YE / AGLOBE
RX3 = EPS2 * SINT2 * SINT + C1111 * COST

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 = USPEED
USPD = USPEED
BETA = DATAN2 (VRP, USPD)
WQR = WSPEED
ALPHA = DATAN2 (WQR, USPD)
DO 301 I = 1, ISTDM1
XDM1(I) = AC1111 * COST * COS2 - DCOS (XST1(I)) * DSIN (YST1(I)) *
1. (-ZST1(I) + AGLOBE * (1.0D0 - 5D0 * EPS2 * DCOS (2 * XST1(I))))
YDME(I) = AC1111 * COST * DSIN - DCOS (XST1(I)) * DSIN (YST1(I)) *
1. (-ZST1(I) + AGLOBE * (1.0D0 - 5D0 * EPS2 * DCOS (2 * XST1(I))))
ZDME(I) = AC1111 * SINT
C***XST, YST ARE LATITUDE AND LONGITUDE
301 RDME(I) = DSQRT (XDM1(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, ISTV01
302 VOR(I) = 1.
C***
ZCAL(1) = VAIR
ZCAL(2) = ALPHA
FILE: OPTPROPG FORTRAN A

PRINCETON UNIVERSITY TIME-SHARING

ZCAL(3) = BETA
ZCAL(4) = -ZE
ZCAL(5) = RDME(1)
ZCAL(6) = RDME(2)
ZCAL(7) = VOR(1)
ZCAL(8) = VOR(2)

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*YDME(1) + CI111*COSS*YDME(1)) / RDME(1)
HB1(6, 2) = (-C1111*SINS*YDME(2) + CI111*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) = USPEED / (VAIR2 - WSPEED**2)
HB1(1, 5) = VSPEED / VAIR
HB1(2, 5) = USPEED / (VAIR2 - VSPEED**2)
HB1(1, 6) = WSPEED / VAIR
HB1(2, 6) = USPEED / (VAIR2 - VSPEED**2)

RETURN

FUNCTION SATP(X11, X11)
IMPLICIT REAL*8(A-H, O-$)
SATP = DSIGN(DMIN1(DABS(X11), XM11), X11)
RETURN

END

SUBROUTINE ROTAT
IMPLICIT REAL*8(A-H, O-$)
COMMON/COF/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), SNP)

SNT = DSIN(TET)
CST = DCOS(TET)
SNP = DSIN(PSI)
CSP = DCOS(PSI)
SNF = DSIN(PHI)
CSF = DCOS(PHI)

CEB11 = CST*CSP
CEB13 = SNT
CEB31 = SNT*CSP*CSP+SNP*SNP
CEB33 = CST * CSF
CEB12 = CST * SNP
CEB21 = SNF * SNT * CSF - CSF * SNP
CEB22 = SNF * SNT * SNP + CSF * CSP
CEB23 = SNF * CST
CEB32 = CSP * SNT * SNP - SNP * CSP
RETURN
END

SUBROUTINE DITOB(XI, YI, ZI, A11, A12, A13, A21, A22, A23, A31, A32, A33, 
XI0, 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 + A23 * YI + A23 * ZI
 RETURN
END

ENTRY DBTOI(XI, YI, ZI, A11, A12, A13, A21, A22, A23, A31, A32, A33, 
XI0, YO, ZO)
 XO = A11 * XI + A31 * ZI + A21 * YI
 ZO = A13 * XI + A33 * ZI + A32 * 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. OD0 * X2 - CRK(J) * X1
 C
 RETURN
 END
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,TLAMO,OMEGA,AGLOBE,EPS2,IDGT,NSTEP,NXP,SOF

LOGICAL GO,SOF

NAMELIST/INP/FILE

INTEGER FILE(2)

NAMELIST/OK/GO

NAMELIST/DAT/SOF,RSTEP,NIP,PERCNT,DT,TLAMO,OMEGA,AGLOBE,EPS2

NAMELIST/DAT/SOF,NSTEP,NXP,PERCNT,DT,IDGT,TLAMO

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 IPK1(6),XMK1(6),XPK(6),XMK(6)

1,PPK1(6,6),PMK1(6,6),PPK(6,6),PMK(6,6)

2,PK(6,6),THAT(6,6),TMATT(6,6),PTMATT(6,6)

3,PINV(6,6),WAREA(60),AK(6,6),AKT(6,6)

4,DELP(6,1),DELP(6,6),DELPES(6,1),XS(6)

5,ARDELPE(6,6),ADPATE(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 6INP FILE= SEND')

READ(5,INP)

INF1=FILE(1)

INF2=FILE(2)

PRINT 2000,FILE

2000 FORMAT(21I10)

PRINT 1004

1004 FORMAT(1H,'TYPE 6DAT DATA= SEND')

READ(INF1,DAT)

IF(SOF)GOTO 92

PRINT 1003

1003 FORMAT(1H,'IF EVERYTHING O.K. TYPE 6OK GO=.TRUE. SEND')

GO=.TRUE.

READ(5,OK)

IF(GO)GOTO 10

GOTO 1

10 CONTINUE

C***

KSTEP=NSTEP

KSTEP1=0

C***

KSTEPA=KSTEP

C*

PI=4.0DO*DATAN(1.0DO)

CRAD=180.0DO/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, ((PK12(I, J), I = 1, 11), J = 1, 11)
C*
DO 431 I = 1, NXP
XPK1(I) = XPK12(I)
XPK(I) = XPK12(I)
DO 432 I = 1, NXP
DO 432 J = 1, NXP
PPK1(I, J) = PPK12(I, J)
C***
301 CONTINUE
KSTEP = KSTEP - 1
IF (KSTEP .LT. 1) GO TO 311
KSTEP1 = KSTEP + 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)
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***PK-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)
SNP=DSIN(PSI)
CSP=DCOS(PSI)
CBB11=CST*CSP
CEB12=CST*SNP
CEB13=-SNT
CEB21=SNP*SNT*CSP-CSP*SHP
CEB22=SNP*SNT*SNP+CSP*CSP
CEB23=-SNT*CST
CEB31=SNP*SNT*CSP+SNF*SHP
CEB32=CSP*SNT*SNP*SNP
CEB33=-CST*CSP
C***

XE=XPK(1)
YE=XPK(2)
ZE=XPK(3)
USPEED=XPK(4)
VSPED=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)
SINT2=SINT/COST
SINT2T=2*SINT*COST
COST2=COST*COST*SINT*SINT
C1111=1.0D0-ZE/AGLOBE-.5D0*EPS2*COST2
C1112=1/C1111
C*
C*
FILE: OPTSMOOT FORTRAN A

PRINCETON UNIVERSITY TIME-SHARING

C141=C112* (Z$E$/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*CINT)
C64=OMEGA*(CEB21*COST-CEB23*CINT)
C45=OMEGA*(CEB31*COST-CEB33*CINT)

C413=OMEGA*(CEB11*SINT+CEB13*COST)
C412=OMEGA*(CEB21*SINT+CEB23*COST)
C411=OMEGA*(CEB31*SINT+CEB33*COST)
C211=-OMEGA**2*AGLOBE*C1111*5*SIN2T
C2NT=-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

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

122
SINT1=SINT*OMEGA/AGLOBE
C4111=(C1111*COS2T+.5D0*EPS2*SIN2T*SIN2T)*OMEGA*OMEGA
C4112=(C1111*SIN2T*COS2T*COST)*OMEGA*OMEGA
C41=(WSPEED*CU12-WSPEED*CU1)*SINT1*
2 WY*CEB13+CEB12
C51=(WSPEED*CU11-WSPEED*CU1)*SINT1*
2 WY*CEB13+CEB12
C61=(WSPEED*CU11*CEB33+CU12*CEB33+CU1)*SINT1*
2 WY*CEB33+CEB32
C65=-C56
C46=-C64
C54=-C45
PK(1,1)=C11
PK(2,1)=C21
PK(3,1)=C31
PK(4,1)=C41
PK(5,1)=C51
PK(6,1)=C61
PK(2,2)=C22
PK(1,3)=C13
PK(2,3)=C23
PK(3,3)=C33
PK(4,3)=C43
PK(5,3)=C53
PK(6,3)=C63
PK(1,4)=CEB11+C14
PK(2,4)=CEB12+C24
PK(3,4)=CEB13+C34
PK(1,5)=CEB21+C15
PK(2,5)=CEB22+C25
PK(3,5)=CEB23+C35
PK(1,6)=CEB31+C16
PK(2,6)=CEB32+C26
PK(3,6)=CEB33+C36
PK(5,4)=-WR+C54
PK(6,4)=-WQ+C64
PK(4,5)=-WR+C45
PK(6,5)=-WP+C65
PK(4,6)=-WQ+C46
PK(5,6)=-WP+C56

CALL CSTM(FK,PERCNT,DT,THAT,B60,FKDT,sume,sume1,NXP,NTERM)

DO 600 I=1,NXP
DO 600 J=1,NXP
600 TMATT(J,I)=TMATT(I,J)

DO 701 I=1,NXP
DO 701 J=1,NXP
701 PMK2(I,J)=PMK1(I,J)
CALL LINVF(PMK2,NXP,NXP,PINV,IDGT,WKAREA,IER1)
CALL VMULPP(PPK,TPATT,NXP,NXP,NXP,NXP,NXP,NXP,PTMATT,NXP,IER2)
CALL VMULPP(PTMATT,PINV,NXP,NXP,NXP,NXP,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 VMULPP(AK,DELX,NXP,NXP,1,NXP,NXP,DELXES,NXP,IER4)
DO 604 I=1,NXP
DO 604 J=1,NXP
604 XS(I)=XPK(I)+DELXES(I,1)
CALL VMULPP(AK,DELX,NXP,NXP,1,NXP,NXP,AKDELP,NXP,IER5)
CALL VMULPP(AKDELP,AK,DELX,NXP,NXP,1,NXP,ADELP,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.LT.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 XM1(I)=XM(I)
C***
   GOTO 301
C***
   311 CONTINUE
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 XBS FOR OUTPUT OR FURTHER PROCESSING

C*

DIMENSION J(15),Z(11)
1, RDME(7), XDME(7), YDME(7), ZDME(7)
COMMON/DAT0/SOP,J,NSTEP,N,NAMES,TLAMO,AGLOBE
1, EPS2,SMEWO,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,TLAMO,AGLOBE
1, EPS2,SMEWO,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 (2110).
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 SUM1(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(TLAMO+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)).
**FILE: OPTSNPR FORTRAN A  PRINCETON UNIVERSITY TIME-SHARING**

```
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(' ',110,12F10.3)
WRITE(12,905)J1, (XS(I),I=7,9), (PS(I,I),I=7,9)
905 FORMAT(' ',110,6F20.3)
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

IDME(I)=AC1111*COST*COSS+(-ZST(I)+AGLOBE*(1-.5*EPS2*COS(2*XST(I))
1)) *COS(XST(I)) *SIN(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(IDME(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(' ',110,6F20.3)
912 FORMAT(' ',110,6F20.3)
```

127
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) = SUM(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(' **', 6P10.3)
PRINT 915, (BIAS(I), I = 1, 4), BIAS(9), BIAS(10), (STDZ(I1), I1 = 1, 4)
1, STDZ(9), STDZ(10)
915 FORMAT(' **', 6P10.3)
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/DATO/SOF,J,NSTEP,N,NAMES,TLAM0,AGLOBE
1 ,EPS2,SMEWO,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/, ANSTED/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/,SMEWO/-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’,‘INDP’,‘HDIG’
5,‘LIGHT’,‘MDSW’,‘TIME’/
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:

```
SPIFR flight records on
DC100A digital data cartridges

Transfer from cartridges to
9-track 1600 BPI magnetic tape
(HP 1000 digital computer)

Data reduction & analysis
(IBM 4341 digital computer)

Data reduction & analysis
(IBM 3033 digital computer)
```

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

%CAT3 T:00004 IS ON CR00005 USING 00012 BLKS R:0000

0001 FTN4.L
0002 PROGRAM CAT9(3,99), VERSION OF 4 JUNE 1981
0003 C
0004 C PROGRAM TO COPY BINARY DATA FROM CASSETTE TO IBM COMPATIBLE
0005 C TAPE DRIVE.
0006 C
0007 C LOADING THE PROGRAM
0008 C :RU_LOADR.*F4X,.*CAT9
0009 C
0010 C RUNNING THE PROGRAM
0011 C :RU,CAT9,P1,P2
0012 C WHERE P1 - IS THE LOGICAL UNIT NUMBER OF YOUR TERMINAL
0013 C P2 - IS THE LOGICAL UNIT NUMBER OF THE MAG TAPE
0014 C
0015 C
0016 C INTEGER IBUFF(128), IMORE, ISTAT, ITLOG, PARMS(5), NBLCK
0017 C EQUIVALENCE (PARMS(1), LUCRT), (PARMS(2), MTLU)
0018 C CALL RMPAR(PARMS)
0019 C NBLCK = 0
0020 C***READ FROM LEFT CARTRIDGE LU 4
0021 C
0022 21 CONTINUE
0023 C
0024 21 CONTINUE
0025 C GET STATUS
0026 C CALL ABREG(ISTAT, ITLOG)
0027 47 FORMAT(I10)
0028 C CHECK FOR END OF FILE
0029 C IF(IAND(ISTAT, 200B) .EQ. 200B) GO TO 22
0030 C CHECK FOR END OF TAPE
0031 C IF(IAND(ISTAT, 400B) .EQ. 400B) GO TO 22
0032 C CHECK FOR END OF DATA
0033 C IF(IAND(ISTAT, 2) .EQ. 2) GO TO 22
0034 C***WRITE TO TAPE
0035 C CALL EXEC(2, 100B+MTLU, IBUFF, 128)
0036 C NBLCK = NBLCK + 1
0037 C WRITE(LUCRT, 31) NBLCK
0038 31 FORMAT(I7)
0039 GOTO 21
0040 22 CONTINUE
0041 C IF(ITLOG, LT, 128) GO TO 41
0042 C CALL EXEC(2, 100B+MTLU, IBUFF, 128)
0043 C NBLCK = NBLCK + 1
0044 C WRITE(LUCRT, 31) NBLCK
0045 31 FORMAT(I7)
0046 41 WRITE(LUCRT, 23)
0047 23 FORMAT("PLUG IN NEXT CARTRIDGE AND TYPE 1 OR IF LAST-TYPE 0")
0048 C READ(LUCRT, IMORE)
0049 C IF(IMORE, EQ, 1) GOTO 21
0050 C WRITE TWO CONSECUTIVE END OF FILE MARKS
0051 C CALL EXEC(3, 0100B+MTLU)
0052 C STOP

132
FILE: RRAWY1 FORTRAN A1 PRINCETON UNIVERSITY TIME-SHARING SYSTEM

COMMON/DAT0,SOF,NB
LOGICAL GO,SCP
NAMELIST/INF/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/OK/SCP,NB

***
1 CONTINUE
PRINT 1002
1002 FORMAT (1H,'TYPE &INF FILE= &END')
READ (5,INF)
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 (SCF) GOTO 92
PRINT 1003
1003 FORMAT (1H,'IF EVERYTHING O.K. TYPE &OK GO=.TRUE.&END')
GO=.TRUE.
READ (5,CK)
IF (GO) GOTO 10
GOTO 1
10 CONTINUE
***
I1=0
I2=0
***
READ (1S, 17) DATA
DO 28 I=1,255,2
SWLOG(I)=DALOG(I+1)
SWLOG(I+1)=DALOG(I)
28 CONTINUE
GOTO 32
***
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
***
DO 31 I=1,128
FILE: RAWY1  FORTRAN A1 PRINCETON UNIVERSITY TIME-SHARING SYSTEM

IF(DATA2(I).NE.DATA1(I))GO TO 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***M=1-(128*8-36);I1=0-8;I2=NC. 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***
   J1=0
   DO 431 K=K1,K2
       IF(K.GT.NB)GOTO 437
   DO 432 J=1,36
   432 A2(K,J)=A3(J7+J)
   J7=J7+38
   431 CONTINUE
   I1=0
   GOTO 99
100 CONTINUE
C***
   PRINT 441
441 FORMAT(1H,11,EOT;DECREASE NB AND RERUN AS A NEW JCB*)
C***
   437 CONTINUE
   KSOF=K-1
27 FORMAT(1H,110)
   PRINT 27,KSOF
   DO 531 J=1,36
531 WRITE(9)(A2(K0,J),K0=1,KSOF)
   ENDFILE 9
C***
   GOTO 1
92 CONTINUE
   STCP
   END
   BLOCK DATA
   COMMON DATO/SOF,NB
   LOGICAL SOF
   DATA SOF/.FALSE. /*,NB/256/
   END
FILE: SPIFY1 FORTRAN A1 PRINCETON UNIVERSITY TIME-SHARING SYSTEM

COMMON/DATC/SOF, VSLOPE, VCONST, PHSLOP, PHCONS, N11, DNSTD
INTEGER FILE(2)
DIMENSION A3(1700,38), PHSLOP(55), PHCONS(55)
INTEGER*2 A2(1700,38)
LOGICAL GO, SOF
NAMELIST/INF/FILE
NAMELIST/OK/GO
NAMELIST/DAT/SOF, VSLOPE, VCONST, PHSLOP, PHCONS, N11, DNSTD
CONTINUE
PRINT 1002
1002 FORMAT (1H , 'TYPE &INP FILE= &END')
READ(5, INP)
INF1=FILE(1)
INF2=FILE(2)
PRINT 2000, FILE
2000 FORMAT (2110)
PRINT 1004
1004 FORMAT (1H , 'TYPE &DAT DATA= &END')
READ(INF1, DAT)
IF (SCF) 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 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 INTERPRETED 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.+VCONS)*PHSLOP(J) +
PHCONS(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
522 CONTINUE
C***
DO 524 I=1, N11
ISIGN=0
IF (A2(I,13) .LT. 0) ISIGN=1
524 A3(I,13)=A2(I,13) +ISIGN*65535
DO 525 I=1, N11
DO 526 J=14, 34
ISIGN=0
525 CONTINUE
C***
FILE: SPIFY1 FORTRAN A1 PRINCETON UNIVERSITY TIME-SHARING SYSTEM

IF (A2(I,J) .LT. 0) ISIGN = 1
523 A3(I,J) = ((A2(I,J) + ISIGN*65535) * VSLOPE/16. + VCONST) * PHSLOP(J) + 1 * PHCONS(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) = 0.5*(A3(I-1,1) + A3(I+1,1))
601 A3(I,13) = 0.5*(A3(I-1,13) + A3(I+1,13))
A3(N11,13) = A3(N11-1,13)

C***
PRA1 = PRATIO; RRA1 = RRATIO
DO 611 I = 1, N11
PRA1 = (A3(I,29) + DPNSTD) / 1013.3
RRA1 = PRA1**.81
A3(I,18) = 1.689*A3(I,18)/SQRT(RRAT)
HCONST = EXP(AIOG(PRAT)/5.256)
A3(I,29) = (1-HCONST)/.00000689
611 CONTINUE

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 SCF
DATA SOF/ .FALSE. /, VSLOPE/.004884/, VCONST/-10./,
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.
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.

Figure D-2. Serial Data Word Format.
The bit format is shown in Figure D-3. Synchronization, 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 oscilloscope 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 later 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, preferably 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.
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 reference 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.
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 replies 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


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.
End of Document