AD-A033 495 JOHNS HOPKINS UNIV LAUREL MD APPLIED PHYSICS LAB F/G 9/5

HIGH-RESOLUTION CLOCK CONTRO_ FOR THE TIP SATELLITE.(U)

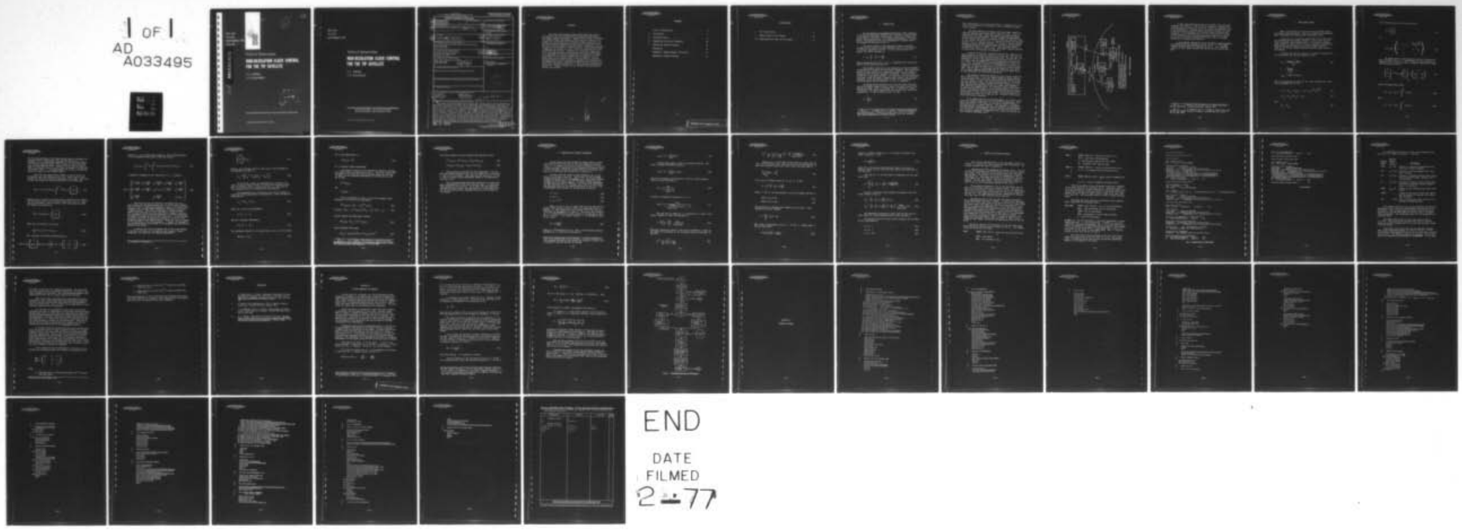
SEP 76 R E JENKINS, A D GOLDFINGER N00017-72-C-4401

UNCLASSIFIED APL/JHU/TG-1301 NL

1 OF 1

AD

A033495



END

DATE

FILMED

2-77

APL/JHU
TG 1301
SEPTEMBER 1976
Copy No. 73

Technical Memorandum

HIGH-RESOLUTION CLOCK CONTROL FOR THE TIP SATELLITE

R. E. JENKINS
A. D. GOLDFINGER

THE JOHNS HOPKINS UNIVERSITY ■ APPLIED PHYSICS LABORATORY

APL/JHU
TG 1301
SEPTEMBER 1976

Technical Memorandum

HIGH-RESOLUTION CLOCK CONTROL FOR THE TIP SATELLITE

R. E. JENKINS
A. D. GOLDFINGER

THE JOHNS HOPKINS UNIVERSITY ■ APPLIED PHYSICS LABORATORY
Johns Hopkins Road, Laurel, Maryland 20810
Operating under Contract N00017-72-C-4401 with the Department of the Navy

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

PLEASE FOLD BACK IF NOT NEEDED FOR BIBLIOGRAPHIC PURPOSES

REPORT DOCUMENTATION PAGE

1. REPORT NUMBER: APL/JHU/TG-1301

2. GOVT ACCESSION NO:

3. RECIPIENT'S CATALOG NUMBER:

4. TITLE (and Subtitle): HIGH-RESOLUTION CLOCK CONTROL FOR THE TIP SATELLITE.

5. TYPE OF REPORT & PERIOD COVERED: Technical Memo[redacted].

6. PERFORMING ORG. REPORT NUMBER: TG 1301

7. AUTHOR(s): R. E. Jenkins [redacted] A. D. Goldfinger

8. CONTRACT OR GRANT NUMBER(s): N00017-72-C-4401

9. PERFORMING ORGANIZATION NAME & ADDRESS:
The Johns Hopkins University Applied Physics Laboratory
Johns Hopkins Rd.
Laurel, MD 20810

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: S2V

11. CONTROLLING OFFICE NAME & ADDRESS:
Strategic Systems Project Office
1931 Jefferson Davis Highway
Arlington, VA 20376

12. REPORT DATE: Sep[redacted]976

13. NUMBER OF PAGES: 47

14. MONITORING AGENCY NAME & ADDRESS:
Naval Plant Representative Office
Johns Hopkins Rd
Laurel, MD 20810

(12) 46p.

15. SECURITY CLASS. (of this report): Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: --

16. DISTRIBUTION STATEMENT (of this Report):
Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report): --

18. SUPPLEMENTARY NOTES:

19. KEY WORDS (Continue on reverse side if necessary and identify by block number):

TIP satellite	Kalman filter
satellite clock	oscillator control
clock control	software program

10 to the 13th Power

20. ABSTRACT (Continue on reverse side if necessary and identify by block number):

The Transit Improvement Program (TIP) satellite clock is controlled by the IPS (Incrementally Programmable Synthesizer) and flight computer subsystems. Together they provide a synthesis of frequency offset and frequency drift that can be used to compensate for such errors in the satellite crystal oscillator. To do this, the ground software must estimate the oscillator offset and drift and then compute the proper control parameters to be injected into the satellite to steer the clock. The system provides a resolution of control of 1 part in 10^{13} in frequency and 1 part in 10^{13} per day in drift. Estimation of oscillator offset and drift from high-resolution pseudo-random noise epoch measurements is accomplished by a discrete Kalman filter, based upon a three-state model with continuous random walks in frequency and frequency drift as the driving noise in the oscillator. A ground software program has been provided to implement the Kalman filter and compute the control parameters required to steer the satellite clock to the reference ground clock. The programs are written in Fortran IV. Complete listings of the software and operating procedures are provided.

DD FORM 1 JAN 73 1473

Unclassified 031 650

SECURITY CLASSIFICATION OF THIS PAGE

LB

ABSTRACT

The Transit Improvement Program (TIP) satellite clock is controlled by the IPS (Incrementally Programmable Synthesizer) and flight computer subsystems. Together they provide a synthesis of frequency offset and frequency drift that can be used to compensate for such errors in the satellite crystal oscillator. To do this, the ground software must estimate the oscillator offset and drift and then compute the proper control parameters to be injected into the satellite to steer the clock. The system provides a resolution of control of 1 part in 10^{13} in frequency and 1 part in 10^{13} per day in drift. Estimation of oscillator offset and drift from high-resolution pseudo-random noise epoch measurements is accomplished by a discrete Kalman filter, based upon a three-state model with continuous random walks in frequency and frequency drift as the driving noise in the oscillator. A ground software program has been provided to implement the Kalman filter and compute the control parameters required to steer the satellite clock to the reference ground clock. The programs are written in Fortran IV. Complete listings of the software and operating procedures are provided.

CONTENTS

ILLUSTRATIONS

1. INTRODUCTION

The Incrementally Programmable Synthesizer (IPS) subsystem on the Transit Improvement Program (TIP) satellite series provides a new method of satellite clock control, heretofore untried with quartz crystal oscillators. The subsystem synthesizes a frequency offset, the magnitude of which is programmable by digital input. The hardware is described in Ref. 1.

The IPS is driven by the high-quality quartz oscillator (5 MHz), and it outputs an offset frequency that is controlled by two digital registers (A and B). If f_0 is the oscillator frequency, the output frequency is

$$f = f_0 \left[1 - \frac{1}{B} \left(1 + \frac{1}{A}\right)\right] \quad . \tag{1}$$

Thus by manipulation of the A and B registers, the output frequency can be controlled directly.

The hardware clock on the old navigation satellites (OSCAR's) is maintained by counting the oscillator frequency. The traditional method of epoch control is to selectively delete or add counts to compensate for oscillator frequency variations. This is usually called the "clock delete system." The counter on the TIP satellites is driven by the IPS output frequency rather than by the oscillator frequency. With this system, epoch control can be maintained by direct frequency control using the A and B registers.

By making small adjustments in A, the IPS output frequency can be set slightly high or low to "steer" the epoch error to zero between settings. Also, by allowing the A register to change continually, the crystal aging drift and flicker noise can be compensated for. It may be seen from Eq. (1) that

$$\dot{f} = \frac{f_0}{A^2 B} \dot{A} \quad . \tag{2}$$

Ref. 1. L. Rueger and A. G. Bates, "Frequency Synthesizer for Normalizing the Frequency and Time Scale of Crystal Clocks," 28th Annual Symposium on Frequency Control, 1974, pp. 395-400.

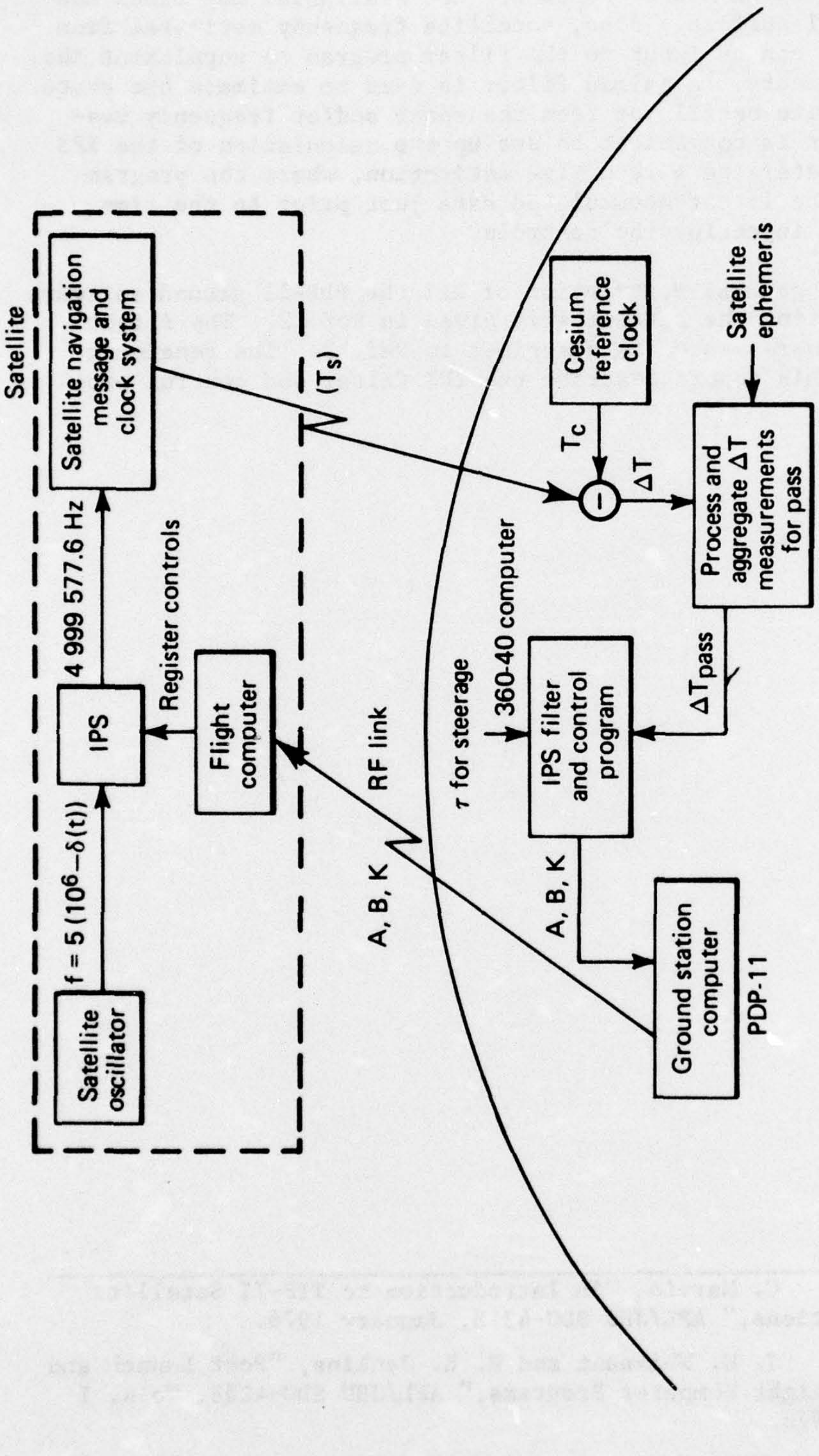
This relationship can be used effectively to compensate for oscillator drift because of the presence on TIP of a general-purpose flight computer.

The IPS hardware is interfaced with the flight computer so that the computer can exercise direct control on the A and B registers in real time. With this system, very-high-resolution clock control can be achieved. The IPS hardware as implemented has an inherent resolution of frequency control as large as 1 part in 10^{11}, set by the size of the A register (14 bits) and the magnitude of the B register contents. By using the flight computer program to manage the IPS registers in real time, the resolution of frequency control is 1 part in 10^{13}, and the drift control resolution is 1 part in 10^{13} per day. The flight computer program for IPS control is described in Appendix A.

Another TIP satellite subsystem provides the capability for high-resolution clock epoch transfers to the ground. This is the pseudorandom noise (PRN) time pulse modulation. The PRN code, consisting of a special fast phase-modulation pattern either 2^{12} or 2^{15} chips long, is transmitted in synchronism with the satellite U.T. clock. By using dual-frequency transmission to correct for first-order ionospheric effects, the PRN modulation provides the capability for epoch recovery with a precision of several nanoseconds. The accuracy of recovering the mean satellite clock epoch for a single combined doppler-PRN pass is limited to about 20 ns by the uncertainty of the satellite position.

The complete closed-loop system for clock control using the IPS is shown in Fig. 1. In this typical feedback system observations are made of the epoch error and controls are applied to null the error signal to zero. The interesting features of the system are that the loop is closed through the ground software, and the time constants for applying controls are rather slow compared to most real-time feedback loops. For practical reasons that have to do with the satellite being in view of ground stations, the controls can be applied at most once per day with the current Transit ground system. Also, for the same reason, epoch measurements can be received only about four times per day. Therefore this loop is quite slow by most standards.

It is assumed that the epoch measurements, δt_c, input to the filter and control program are not the raw PRN measurements. They must be processed to the extent of providing a single-parameter, mean-epoch error at closest approach. The parameter is estimated by simultaneously navigating the receiver position and determining the clock error. The navigated (rather than true) receiver position is used to determine propagation time, since this removes the main satellite ephemeris errors.



T_s is satellite "fiducial" time recovered from satellite message
T_C is cesium-driven reference time
ΔT_{pass} is mean satellite clock error for a single pass
A, B, K are IPS control parameters
τ is desired time delay for steering clock to zero

Fig. 1 IPS Clock Control

If PRN epoch measurements are not available, any clock measurements will suffice. Also, satellite frequency estimates from tracking runs can be input to the filter program to supplement the clock measurements. A Kalman filter is used to estimate the state of the satellite oscillator from the epoch and/or frequency measurements. It is convenient to set up the calculation of the IPS control parameters as a recursive estimation, where the program is run with the latest accumulated data just prior to the time scheduled for injecting the controls.

A good general description of all the PDP-11 ground software for communicating the satellite is given in Ref. 2. The flight computer software system is described in Ref. 3. The remaining sections of this report describe the IPS filter and control program.

Ref. 2. C. Marvin, "An Introduction to TIP-II Satellite Digital Operations," APL/JHU SDO-4318, January 1976.

Ref. 3. J. M. Whisnant and R. E. Jenkins, "Post Launch and Operational Flight Computer Programs," APL/JHU SDO-4268, Vols. I and II, May 1976.

2. THE KALMAN FILTER

Many of the results in this section are standard results from linear filtering theory. They are included for completeness and to aid persons not familiar with the Kalman filter.

In the filter, we model the IPS-driven clock output as being determined by three state variables: epoch error (δt), IPS output frequency, and frequency drift. The satellite hardware clock counts 2^{13} cycles of the IPS output frequency divided by 12. When the IPS output frequency is exactly nominal (exactly 4 999 577.60 Hz) the counter underflows precisely 6103 times in two minutes.

We define the last two states to be relative frequency offset from nominal and relative frequency drift:

$$\delta f = \frac{f_{actual} - f_{nom}}{f_{nom}} , \quad (3)$$

$$\dot{f} = \frac{\dot{f}_{actual}}{f_{nom}} ,$$

$$f_{nom} = 4\,999\,577.60 \text{ Hz} .$$

Then the propagation equations of the state variables as a function of elapsed time, τ, are

$$\delta t_{\tau_1} = \delta t_{\tau_0} + \delta f_{\tau_0} (\tau_1 - \tau_0) + \dot{f}_{\tau_0} \frac{(\tau_1 - \tau_0)^2}{2} , \quad (4)$$

$$\delta f_{\tau_1} = \delta f_{\tau_0} + \dot{f}_{\tau_0} (\tau_1 - \tau_0) , \quad (5)$$

and

$$\dot{f}_{\tau_1} = \dot{f}_{\tau_0} . \quad (6)$$

The transition matrix for the state vector

$$\bar{x} = \begin{pmatrix} \delta t \\ \delta f \\ \dot{f} \end{pmatrix} \tag{7}$$

is

$$\phi(\tau_1, \tau_0) \equiv \begin{pmatrix} 1 & \tau_1 - \tau_0 & \dfrac{(\tau_1 - \tau_0)^2}{2} \\ 0 & 1 & \tau_1 - \tau_0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{8}$$

We assume that the above system is driven by integrated white Gaussian noise (i.e., random walks) in both frequency and frequency drift, so that the state propagation equation for small time intervals is given by

$$\begin{pmatrix} \delta t \\ \delta f \\ \dot{f} \end{pmatrix}_{\tau_1} = \phi(\tau_1, \tau_0) \begin{pmatrix} \delta t \\ \delta f \\ \dot{f} \end{pmatrix}_{\tau_0} + \begin{pmatrix} 0 \\ n_2(\tau_1 - \tau_0) \\ n_3(\tau_1 - \tau_0) \end{pmatrix} \tag{9}$$

where the random walk values

$$n_2 \cdot (\tau_1 - \tau_0) = \int_{\tau_0}^{\tau_1} \nu_2(\tau)\, d_\tau \tag{10}$$

and

$$n_3 \cdot (\tau_1 - \tau_0) = \int_{\tau_0}^{\tau_1} \nu_3(\tau)\, d_\tau \tag{11}$$

are the time integrals of the white driving noise in frequency and frequency drift, respectively, over the time interval $\tau_1 - \tau_0$. Equation (9) is an approximate imbedding of our continuous system in a discrete time step model. This is valid as long as the time step is infinitesimally short. Equation (9) has been included to make it intuitively clearer how the noise is assumed to be driving the system. When $\tau_1 - \tau_0$ becomes larger, the ν_2 and ν_3 noises mix among the states in a more complex manner.

Since our time steps will be finite, we need a better approximation than that afforded by Eq. (9). We get this by integrating Eq. (9) over a span of infinitesimal time steps. When this is done, and using the transition property of ϕ, we find that

$$\bar{x}(\tau_1) = \phi(\tau_1,\tau_0)\,\bar{x}(\tau_0) + \int_{\tau_0}^{\tau_1} d\lambda\,\phi(\tau_1,\lambda) \begin{pmatrix} 0 \\ \nu_2(\lambda) \\ \nu_3(\lambda) \end{pmatrix} . \tag{12}$$

Equation (12) is usually called the "matrix superposition integral," and the second term is the accumulated noise from τ_0 to τ_1. Equation (12) describes a forced, linear, first-order system in which the dynamics are modeled as:

$$\dot{\bar{x}}(\tau) = F(\tau)\,\bar{x}(\tau) + \begin{pmatrix} 0 \\ \nu_2(\tau) \\ \nu_3(\tau) \end{pmatrix} \tag{13a}$$

where $F(\tau)$ is related to $\phi(\tau,\tau_0)$ by

$$\frac{d}{dt}\,\phi(\tau,\tau_0) = F(\tau)\,\phi(\tau,\tau_0) \quad . \tag{13b}$$

The covariance of the white driving noise is

$$q(\lambda,\mu) \equiv \left\langle \begin{pmatrix} 0 \\ \nu_2(\lambda) \\ \nu_3(\lambda) \end{pmatrix} [0 \;\; \nu_2(\mu) \;\; \nu_3(\mu)] \right\rangle = \delta(\lambda - \mu) \begin{pmatrix} 0 & 0 & 0 \\ 0 & \sigma_2^{\,2} & 0 \\ 0 & 0 & \sigma_3^{\,3} \end{pmatrix} \tag{14}$$

where $\delta(\lambda - \mu)$ is a Dirac delta function. The covariance matrix of the accumulated noise after a long time step is

$$Q(\tau_1,\tau_0) = \int_{\tau_0}^{\tau_1} d\lambda \int_{\tau_0}^{\tau_1} d\mu \; \phi(\tau_1,\lambda)\, q(\lambda,\mu)\, \phi^+(\tau_1,\mu) \quad . \tag{15}$$

An explicit evaluation of Eq. (14) with $\Delta\tau = \tau_1 - \tau_0$ gives:

$$Q(\Delta\tau) = \begin{bmatrix} \sigma_2^2 \frac{(\Delta\tau)^3}{3} + \sigma_3^2 \frac{(\Delta\tau)^5}{20} & \sigma_2^2 \frac{(\Delta\tau)^2}{2} + \sigma_3^2 \frac{(\Delta\tau)^4}{8} & \sigma_3^2 \frac{(\Delta\tau)^3}{6} \\ \sigma_2^2 \frac{(\Delta\tau)^2}{2} + \sigma_3^2 \frac{(\Delta\tau)^4}{8} & \sigma_2^2 (\Delta\tau) + \sigma_3^2 \frac{(\Delta\tau)^3}{3} & \sigma_3^2 \frac{(\Delta\tau)^2}{2} \\ \sigma_3^2 \frac{(\Delta\tau)^3}{6} & \sigma_3^2 \frac{(\Delta\tau)^2}{2} & \sigma_3^2 (\Delta\tau) \end{bmatrix} \quad . \tag{16}$$

Equations (12) and (16) describe the modeled IPS output state during the time spans between changes in the control parameters A, B, and K.† When these parameters change, the frequency and frequency drift undergo sudden changes not described by the state transition matrix. This does not represent a serious problem. Given a state estimate at a time just before a change in the control parameters, we can obtain an estimate of the state immediately following the change, as we will see in Section 3. In the process, of course, there will be significant changes in δf^* and $\dot{f}^*$, the estimated frequency and frequency drift. There will also be changes in the covariance of the state estimate, Σ. However the changes in the covariance matrix can be shown to be negligible and they will be ignored in the filter.

In running the filter, we assume that we have some estimate of the state at a time τ_L following the last change in control parameters, and based upon all previous measurements:

†K is defined in Appendix A.

$$\begin{pmatrix} \delta t^* \\ \delta f^* \\ \dot{f}^* \end{pmatrix} = \bar{x}_{\tau_L}^{\ *} \ . \tag{17}$$

Further, the covariance matrix of the error in this estimate is also generated at time τ_L:

$$\Sigma_{\tau_L} = \left\langle \left(\bar{x}_{\tau_L}^{\ *} - \bar{x}_{\tau_L}\right)\left(\bar{x}_{\tau_L}^{\ *} - \bar{x}_{\tau_L}\right)^+ \right\rangle \ . \tag{18}$$

We are given a number of measurements of the epoch error and/or frequency offset at times τ_i, and we must produce an estimate of the output state at some future time, τ_I, when new control parameters are to be injected into the satellite.

The measurements are corrupted by noise that we assume is uncorrelated from measurement to measurement, so that the measured quantities are

$$z_{\tau_L} = H\,\bar{x}_{\tau_i} + v(\tau_i) \tag{19}$$

where for a time error measurement,

$$H = (1 \quad 0 \quad 0) \tag{20}$$

and for a frequency measurement,

$$H = (0 \quad 1 \quad 0) \ . \tag{21}$$

The covariance "matrix" of the scalar noise, $v(\tau_i)$, is the variance

$$R[v(\tau_i)] = \sigma_{\delta t}^{\ 2} \tag{22}$$

for a time measurement, or

$$R[v(\tau_i)] = \sigma_f^2 \tag{23}$$

for a frequency offset measurement.

The optimal solution of this estimation problem is provided by a Kalman filter, as described in Ref. 4. The filter starts with estimates of the state and of the covariance matrix at some time τ_L, given all the previous measurements up to some time τ_P; that is, we know

$$\overline{x}^* (\tau_L|\tau_P)$$

and

$$\Sigma(\tau_L|\tau_P) \quad .$$

Given a measurement at time τ_i, we first propagate these estimates to the time of the measurement:

$$\overline{x}^* (\tau_i|\tau_P) = \phi(\tau_i - \tau_L)\, \overline{x}^* (\tau_L|\tau_P) \tag{24}$$

$$\Sigma(\tau_i|\tau_P) = \phi(\tau_i - \tau_L)\, \Sigma(\tau_L|\tau_P)\, \phi^+(\tau_i - \tau_L) + Q(\tau_i - \tau_L) \quad . \tag{25}$$

We then compute the measurement residual

$$\overline{Z}(\tau_i|\tau_P) = \overline{Z}_{\tau_i} - H\, \overline{x}^* (\tau_i|\tau_P) \tag{26}$$

and the optimal filter gain

$$K(\tau_i) = \Sigma(\tau_i|\tau_P)\, H^+[R + H\, \Sigma(\tau_i|\tau_P)\, H^+]^{-1} \tag{27}$$

Ref. 4. R. E. Kalman, "New Methods in Wiener Filtering," Proceedings of First Symposium on Engineering Applications of Random Function Theory and Proabability, J. Wiley, 1963.

and finally update the state estimate and covariance matrix

$$\overline{x}^* (\tau_i|\tau_i) = \overline{x}^* (\tau_i|\tau_p) + K(\tau_i)\ \overline{Z}(\tau_i|\tau_p) \tag{28}$$

$$\Sigma(\tau_i|\tau_i) = \Sigma(\tau_i|\tau_p) - K(\tau_i)\ H\ \Sigma(\tau_i|\tau_p) \quad . \tag{29}$$

This process is continued for each measurement. We note that the matrix inversion in Eq. (27) is trivial in our case since R and $H\ \Sigma\ H^+$ are scalars. Note also that the values for H and R used to process a given measurement depend on the measurement type (δt or δf).

After the measurements have been processed, the state estimate is propagated up to the next injection time, τ_L, using equations similar to (24) and (25). To do this, the next injection time must be known; hence, the data should not be processed until the IPS injection has been scheduled so that τ_I can be input to the program.

3. COMPUTATION OF CONTROL PARAMETERS

Having passed the data through the Kalman filter, we have an estimate of the clock state at the next scheduled injection time, τ_I. To compute the required control parameters for that injection, we require estimates of the frequency and frequency drift of the oscillator. These are obtained from the clock state estimate at τ_I along with the values of A, B, and K at τ_I.

We start from the values of these parameters at τ_L, the time of our previous injection. (Notice that τ_I from the last program run becomes τ_L for the current run.) The values are given in their "external form," that is, the actual numbers that are transmitted to the flight computer. In effect, these parameters are then converted into an "internal form," and it is this internal form that occurs in the control equations (1) and (2).† The transformation from external to internal form is given by

$$A \rightarrow A + 1 \tag{30}$$

$$B \rightarrow B + 1 \tag{31}$$

$$K \rightarrow K \cdot 2^{-31} \tag{32}$$

Since K and B do not change with time once they are injected, their internal values at time τ_L are maintained until τ_I. However, A does change with time as shown in Eq. (2), and its value must be propagated up to time τ_I. The actual time variation in A is controlled by the flight computer (see Appendix A, Eqs. (A-1) to (A-3)). In the flight computer, A is incremented every Δt by a value

$$\Delta A = K\left(\frac{A}{128}\right)^2 \tag{33}$$

where $\Delta t = 128 \cdot 120/6103 \approx 2.52$ s. Hence, the difference equation satisfied by A with 2.52-s intervals is

†This is an idiosyncrasy of the hardware. To avoid confusion we have treated the internal form inside the software so that it can be ignored by persons running the program.

$$A_{n+1} = A_n + \frac{K}{(128)^2} A_n^{\ 2} \ . \tag{34}$$

We have been unable to solve this equation exactly. However, a similar difference equation,

$$A_{n+1} = A_n + \frac{K}{(128)^2} A_n A_{n+1} \ , \tag{35}$$

should be an adequate approximation to (34) for these purposes. This is exactly solvable as

$$A_n = \frac{A_0}{1 - \frac{K}{(128)^2} A_0 n} \tag{36}$$

so that our propagation equation for A is

$$A(\tau_I) = \frac{A(\tau_L)}{1 - \frac{K \cdot 6103}{(120)(128)^3} A(\tau_L)(\tau_I - \tau_L)} \ . \tag{37}$$

The time rate of change of A is manifest as a drift in the output frequency so that (see Appendix A)

$$\dot{f} = \frac{f_0}{B}\frac{\dot{A}}{A^2} = \frac{f_0 K}{BA^2} \cdot \frac{6103}{(120)(128)^3} \ . \tag{38}$$

Therefore, having the values of the control parameters at hand, we can arrive at an estimate of the oscillator state by using Eqs. (1) and (38):

$$f_0^{\ *} = \frac{f^*}{\left[1 - \frac{1}{B}\left(1 + \frac{1}{A}\right)\right]} \tag{39}$$

$$\dot{f}_0^{\,*} = \frac{1}{\left[1 - \frac{1}{B}\left(1 + \frac{1}{A}\right)\right]} \left(\dot{f}^* - \frac{f^*}{B} \frac{K\ (6103)}{(120)(128)^3}\right) . \tag{40}$$

Values of A, B, and K must now be chosen to correct the output to zero frequency drift and to provide an output frequency, f_s, that will steer the epoch error back to zero over some time, T:

$$\frac{f_s - f_{nom}}{f_{nom}} = - \frac{\delta t^*}{T} . \tag{41}$$

To do this, we require values of A and B so that

$$f_s = f_0^{\,*} \left[1 - \frac{1}{B}\left(1 + \frac{1}{A}\right)\right] \tag{42}$$

where A and B are constrained to lie in the ranges (see Ref. 1)

$$6301 \le A \le 13\ 470 \tag{43}$$

$$6669 \le B \le 12\ 428 . \tag{44}$$

The algorithm for choosing these numbers is as follows. First, compute the control constant:

$$C = \frac{f_s}{f_0^{\,*}} = \frac{1}{B}\left(1 + \frac{1}{A}\right) . \tag{45}$$

Next, find a (noninteger) value of B so that C obtains when A has the value 6301:

$$B = \left(\frac{1}{1 - C}\right)\left(1 + \frac{1}{6301}\right) . \tag{46}$$

Choose as candidate values of B the integers straddling this value, computing A from

$$A = \frac{B}{[B] - B} \tag{47}$$

where [B] is the integer either greater than or less than B. Finally, choose the A, B pair that fits within the constraints of (43) and (44).

The value of K is then chosen to compensate for the estimated drift:

$$K = \frac{\dot{f}_0^*}{f_0^*}\left[1 - \frac{1}{B}\left(1 + \frac{1}{A}\right)\right] B \frac{(128)^3(120)}{6103} \quad . \tag{48}$$

Finally, the output state estimate is updated to the corrected values at time τ_I:

$$f_{\tau_I}^* = \left[1 - \frac{1}{B}\left(1 + \frac{1}{A}\right)\right] f_0^* \approx f_s \tag{49}$$

$$\dot{f}_{\tau_I}^* = \left[1 - \frac{1}{B}\left(1 + \frac{1}{A}\right)\right] \dot{f}_0^* + \frac{K}{B} f_0^* \frac{6103}{(120)(128)^3} \approx 0 \quad . \tag{50}$$

The approximate equalities in Eqs. (49) and (50) are not exact since the control parameters have finite precision.

The external form of the new control parameter can be found by the transformation

$$A \rightarrow A - 1 \tag{51}$$

$$B \rightarrow B - 1 \tag{52}$$

$$K \rightarrow K \cdot 2^{31} \quad . \tag{53}$$

4. RUNNING THE SOFTWARE PROGRAM

The software implementation of the algorithms, written in Fortran IV, consists of a main program and nine subroutines. The program listings are shown in Appendix B.

The subroutine FILT implements the Kalman filter. The subroutine PRED propagates the state estimate up to the input injection time. These routines call matrix multiplication routines MM3 and MMT3. Calculation of the control parameters is done by subroutine IPS. Card input and output are performed by the routines INPUT, TASK, and OUTPUT, which were isolated to simplify changes. For the same reason, a separate BLOCK DATA routine is used to set the noise parameters assumed by the filter.

The program is designed to run on the IBM 360-40, and inputs are via punched cards in the present version. One group of input cards (the "state" cards) is punched out by the program each time it is run, and these cards become the input state cards for the next run. The remaining input cards, containing the measurement data, must be newly prepared before each run.

The use of cards is one area that the Naval Astronautics Group (NAVASTROGRU) may wish to change. It may be more convenient to use one or more disk files as the input and output stream, where the state output card images are written by the program onto the disk to be read by the subsequent run. Also, the data on some of the measurement cards could come directly from the program that processes the satellite clock data. It may eventually prove convenient to have this program write a disk file for the measurement card images that would be read by the IPS control program. In order to effect this type of change, all that is required is to change the subroutines INPUT, TASK, or OUTPUT to read or write a different tape number, and define the appropriate files in the JCL setup.

The first cards in the input deck are the six state cards that describe the previous state estimate at time τ_L. The formats of the six cards are as follows:

CARD 1 FORMAT (I15, F15.8) - gives the last injection time, τ_L

IDAYL - day number

SECL - U.T. seconds of day

CARD 2	FORMAT (3D20.13) - gives clock state at τ_L
	DELT - epoch error (microseconds)
	F - IPS frequency (ppm offset from 5 MHz)
	FD - frequency drift (ppm per day)
CARDS 3, 4, 5	FORMAT (3D20.13) - gives covariance matrix at τ_L
	P - 3 × 3 covariance matrix (internal units)
CARD 6	FORMAT (D20.13, 2I10) - gives control parameters at τ_L
	A, B, K - control parameters (external form)

With the exception of the covariance matrix, all variables on the state cards are in "external" units. Internally, epoch error is converted to seconds, frequency to fractional offset from the nominal frequency, frequency drift to fractional offset per second, and A, B, and K to their internal forms as per Eqs. (30), (31), and (32). The covariance matrix is input in these internal units.

Following the state cards are the series of task (measurement) cards with the following format:

TASK CARD	FORMAT (2I15, 2F15.8) - one for each measurement
	MEAS - flag indicating type of measurement
	IDAYM - day of measurement
	SECM - measurement of time of day (seconds)
	Z - value of measurement

If MEAS = 1, the card represents a time error measurement at epoch IDAYM, SECM; and Z is in microseconds. If MEAS = 2, the card represents a frequency measurement, and Z is in ppm offset from 5 MHz. Following all the measurements (which must be time ordered) is a final card in which MEAS = 0, indicating that a control injection is to take place at time IDAYM, SECM. (The Z value is ignored on this card and may be blank.)

The card output from the program is the set of six state cards to be used as input the next time it is run. The printed output is generated by the main routine and the IPS subroutine. Figure 2 is an example of the output.

```
IPS CONTROL PROGRAM
********************************

INITIAL VALUES:

LAST COMPUTED EPOCH:
DAY=  27 SECONDS=  43200.000000

ESTIMATED STATE VARIABLES:
TIME ERROR=        -0.849 MICROSECONDS
FREQUENCY=  -84.479990173 PARTS PER MILLION OFFSET
FREQUENCY DRIFT=    -0.000000032 PARTS PER MILLION OFFSET PER DAY
COVARIANCE MATRIX (INTERNAL UNITS):
 0.99943356779240D-13 0.27651262141080D-17 0.58795074042050D-23
 0.27651262141080D-17 0.92976585719320D-22 0.20077820633380D-27
 0.58795074042050D-23 0.20077820633380D-27 0.31131407855160D-32
IPS CONTROL PARAMETERS:
A=    6505.7391118830000  B=   11838  K=     569

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

TIME MEASUREMENT AT EPOCH:
DAY=  28 SECONDS=      0.0
TIME ERROR=       -0.423 MICROSECONDS

FILTER GAINS:
K1=   0.99843284156270 00  K2=    0.15333692910970-04  K3=    0.30544602247720-10

FILTERED STATE ESTIMATE:
TIME ERROR=        -0.423 MICROSECONDS
FREQUENCY=  -84.479990159 PARTS PER MILLION OFFSET
FREQUENCY DRIFT=    -0.000000027 PARTS PER MILLION OFFSET PER DAY

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

INJECTION AT EPOCH:
DAY=  28 HOUR=   12 MINUTES=     0 SECONDS=       0.0

STATE ESTIMATE:
TIME ERROR=         0.002 MICROSECONDS
FREQUENCY=  -84.479990173 PARTS PER MILLION OFFSET
FREQUENCY DRIFT=    -0.000000027 PARTS PER MILLION OFFSET PER DAY

VALUES OF IPS PARAMETERS AT INJECTION TIME:
A=    6529.3290508640810 B=   11838  K=     569

OSCILLATOR STATE AT INJECTION TIME:
FREQUENCY=   -0.000462241 PARTS PER MILLION OFFSET
FREQUENCY DRIFT=    -0.000046925 PARTS PER MILLION OFFSET PER DAY

STEERING FOR        0.0  MICROSECONDS TIME ERROR
AT    86400.000 SECONDS PAST INJECTION TIME

DESIRED OUTPUT FREQUENCY:
FREQUENCY=  -84.480000022 PARTS PER MILLION OFFSET

POSSIBLE VALUES OF IPS PARAMETERS:
A=    6524.3602224463300 B=   11838 K=     569
A=    4205.4919357869050 B=    11839 K=     569
```

Fig. 2 Sample Output of IPS Program

```
CHOICE OF IPS PARAMETERS:
A=   6524.3602224463300 B=   11838 K=        569

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

STATE ESTIMATE AFTER INJECTION:

LAST COMPUTED EPOCH:
DAY=  28 SECONDS=  43200.000000

ESTIMATED STATE VARIABLES:
TIME ERROR=         0.002 MICROSECONDS
FREQUENCY=   -84.480000022 PARTS PER MILLION OFFSET
FREQUENCY DRIFT=    -0.000000027 PARTS PER MILLION OFFSET PER DAY
COVARIANCE MATRIX (INTERNAL UNITS):
 0.99943525897720-13 0.27651318979830-17 0.58795903161010-23
 0.27651318979830-17 0.92976776747770-22 0.20078099290620-27
 0.58795903161010-23 0.20078099290620-27 0.31131814338360-32
IPS CONTROL PARAMETERS:
A=   6524.3602224463300  B=  11838  K=      569
                                          4-4
********************************
```

Fig. 2 (cont'd)

The BLOCK DATA subroutine contains seven parameters that may need to be changed occasionally. They are:

Program Name	Analytic Notation (Eq.)	Definition
S2MD	σ_f^2 (23)	Variance of frequency measurement error (fractional offset, squared)
S2M	$\sigma_{\delta t}^2$ (22)	Variance of time measurement error (seconds, squared)
Q2	σ_2^2 (14)	Variance of frequency driving noise (fractional frequency offset, squared, over a 1-s interval)
Q3	σ_3^2 (14)	Variance of frequency drift driving noise (fractional offset per second, squared, over a 1-s interval)
FNOM	f_{nom} (3)	Nominal output frequency (ppm offset from 5 MHz)
STEER	—	A nonzero time error to which the output will be steered, if desired; zero otherwise (seconds)
TAU	T (41)	Time in which output time error is to be steered to STEER (seconds)

The compiled values for the listed parameters are based on experimental runs, some with simulated data and some with a real oscillator. Some of the values will depend on how the program is being run, and the BLOCK DATA routine will have to be recompiled accordingly. The recommended values for the parameters are described below.

With PRN recovered epoch errors, the value of S2M is 9×10^{-16}. Using this value, the satellite would be injected every day, and the program would be run once per day with a value of TAU of 86 400.

With epoch errors recovered from the satellite fiducial time in the navigation message, the value of S2M is 4×10^{-10}. In this mode, the program will need to be run less often, possibly once per week. The required frequency of injection will be left open to be operationally determined by NAVASTROGRU. In any case, the value of TAU should be compiled accordingly. The actual measurements entered into the program should be the processed† clock data, averaged over each pass. The corresponding measurement time should then be the time of closest approach for the pass.

Bear in mind that the program runs recursively and chronologically in time. This means that the program should always be run with the new injection time later than the last measurement time in the input deck. Also, if there are no measurements, a new injection time must still be supplied.

If a run has to be repeated for some reason (such as the scheduled injection not being made), the run must be restarted with the old state cards, not with the state cards from the invalid run. This is extremely important because the program assumes that the IPS parameters on the state cards have actually been injected into the satellite at τ_L. This is the only way it can predict the correct clock behavior to compare to the measurements. Therefore to recover from any problem, the program run sequence must be restarted with the state cards for the last valid injection.

If no state cards are available because they have been lost, or if the satellite clock is reset completely because of a flight computer failure, the state cards will have to be generated again to restart the process. This is done by selecting any convenient epoch and supplying the best estimate of the clock state and its covariance matrix at that epoch. For the frequency offset and drift, these values can be obtained by extrapolating from the last known history as long as the oscillator has not lost power. If the oscillator has lost power and is restarted, then a satellite orbit fit will be needed to obtain the estimates.

In the absence of better information to start from the beginning, the covariance matrix can be input as a diagonal matrix:

$$\Sigma = \begin{pmatrix} \varepsilon_1^2 & 0 & 0 \\ 0 & \varepsilon_2^2 & 0 \\ 0 & 0 & \varepsilon_3^2 \end{pmatrix}$$

where

ε_1 = expected error in clock epoch estimate (10^{-3} if clock was just reset)

---

†Corrected for time delays, etc.

ε_2 = expected error in offset (10^{-11} if there is an offset from orbit fit)

ε_3 = expected error in drift (10^{-16} if there is drift from orbit fit).

Once the program is run with valid data, the covariance matrix will eventually stabilize to a range of values determined by the measurement errors and the assumed filter noise parameters.

REFERENCES

1. L. Rueger and A. G. Bates, "Frequency Synthesizer for Normalizing the Frequency and Time Scale of Crystal Clocks," 28th Annual Symposium on Frequency Control, 1974, pp. 395-400.

2. C. Marvin, "An Introduction to TIP-II Satellite Digital Operations," APL/JHU SDO-4318, January 1976.

3. J. M. Whisnant and R. E. Jenkins, "Post Launch and Operational Flight Computer Programs," APL/JHU SDO-4268, Vols. I and II, May 1976.

4. R. E. Kalman, "New Methods in Wiener Filtering," Proceedings of First Symposium on Engineering Applications of Random Function Theory and Probability, J. Wiley, 1963.

Appendix A

FLIGHT COMPUTER IPS CONTROL†

By the nature of its design, the TIP IPS subsystem cannot achieve the frequency or frequency-drift resolution necessary for high-precision time control with fixed settings in the IPS registers. This resolution problem has nothing to do with the stability, phase drift, or temperature sensitivity of the circuitry. Bench tests have shown that the hardware performs excellently. The resolution is limited only by the lengths of the registers.

Better resolution is achieved by using the flight computer to manipulate the IPS registers in real time. Without a great deal of trouble, a relative frequency resolution of 10^{-13} and a relative drift resolution of 10^{-13} per day can be obtained. This is sufficient to maintain 10-ns accuracy with one setting a day. Whether that accuracy can be realized in orbit depends largely on the performance of the oscillator and the precision of the PRN clock epoch error measurements.

Preliminary tests with the engineering model of the TIP oscillator showed a real potential for better than 100 ns control. To exploit the full potential of the system, a flight computer program to control the IPS registers has been written to produce a resolution of 10^{-13}. The program will be loaded with new IPS control parameters periodically (once a day) via the ground station computer. The parameters will be generated from epoch or frequency measurements on the ground using the IPS filter and control program.

Four values are input: A, B, AF, and K. A and B are 16-bit integers. A goes directly to the IPS A register; B goes directly to the B register. AF and K are 32-bit integers.

AF is the fractional part of A and represents the fraction of time that the A register should hold (A + 1).

Effective IPS A = $\boxed{A} \cdot \boxed{AF}$

†This appendix is based on an internal memorandum by R. E. Jenkins, "Flight Computer IPS Control," APL/JHU S1A-88-74, 16 December 1974.

Out of every 128 tocks,† the IPS A register should hold (A + 1) for .AF × 128 tocks and should hold the value A for 128(1 - .AF) tocks. We pick 128 for convenience since multiplication by 128 shifts the decimal point seven places in AF. This means we pick up the first seven bits from AF as the counter for changing the A register.

K determines the secular change in the A register to compensate for crystal aging drift. Every 128 tocks, the value of AF is permanently incremented (or decremented) by AD:

$$AD = KA^2 .$$

Each time AD is added to AF in the active program, the carrys from the most significant bit of AF increment or decrement A by one.

The sign of the drift correction will be carried by K. To simplify the program, it is preferred that the value of AD does not cause a carry in a single 128-tock interval with a reasonably high drift rate. That is, we want the secular change to A to be less than one count in 128 tocks, and let the accumulation of the changes in AF carry into the IPS A register. This sets the maximum correctable drift to about 10^{-8} per day, which is more than enough.

Also, we want the precision in drift correction to be 10^{-13} per day. To achieve the required precision in calculating AD over the full range of values of A (6300 to 13 470) and B (6668 to 12 427), we need to calculate KA^2 and hold AF in 31-bit precision. A practical scaling to do this is

$$AD = K \left(\frac{A}{128} \right)^2 . \qquad \text{(A-1)}$$

With this scaling, K is defined as follows.

From the formula for the IPS transfer function, to correct for a relative drift, $\dot{f}/f_0$, the required increment in A per Δt is

---

†We have defined the unit of time in the flight computer clock as a "tock," equal to 19.662461 ms. The flight computer maintains a U.T. clock by counting tocks generated by a computer interrupt coming from a hardware frequency counter.

$$AD = -\frac{\dot{f}}{f_0} BA^2 \Delta t \quad . \qquad \text{(A-2)}$$

For Δt = 128 tocks, $\Delta t = 128 \times 120/6103 = 2.516795019$ s. Thus

$$AD = -\frac{\dot{f}}{f_0} B(128)^3 \frac{120}{6103} \left(\frac{A}{128}\right)^2 \quad . \qquad \text{(A-3)}$$

which defines K, a number considerably less than one.

To transmit K to the flight computer, which has 16-bit words, we change it to a double-bit precision, 31-bit binary fraction:

K =	S	KH	KL
	32 ●	16	1

Therefore $K = (\dot{f}/f_0)(B)(128)^3\ 120/6103 \times 2^{31}$ and will be transmitted as a right-adjusted 31-bit integer. The range of this integer will be from 1 to about 210 000 for the worst-case drift. A negative value for K will be carried as a double-precision integer, with bit 32 indicating sign.

With the above scaling, the value of $(A/128)^2$ can be computed and stored in single precision (16 bits), since the maximum value of A is 16 272. This considerably simplifies the calculation of AD in the program.

A detailed description of the IPS management program and how it fits into the overall flight computer software system is given in Ref. 2. Figure A-1 is a simplified flow chart of the program. The program is re-initialized every time new IPS parameters are injected from the ground.

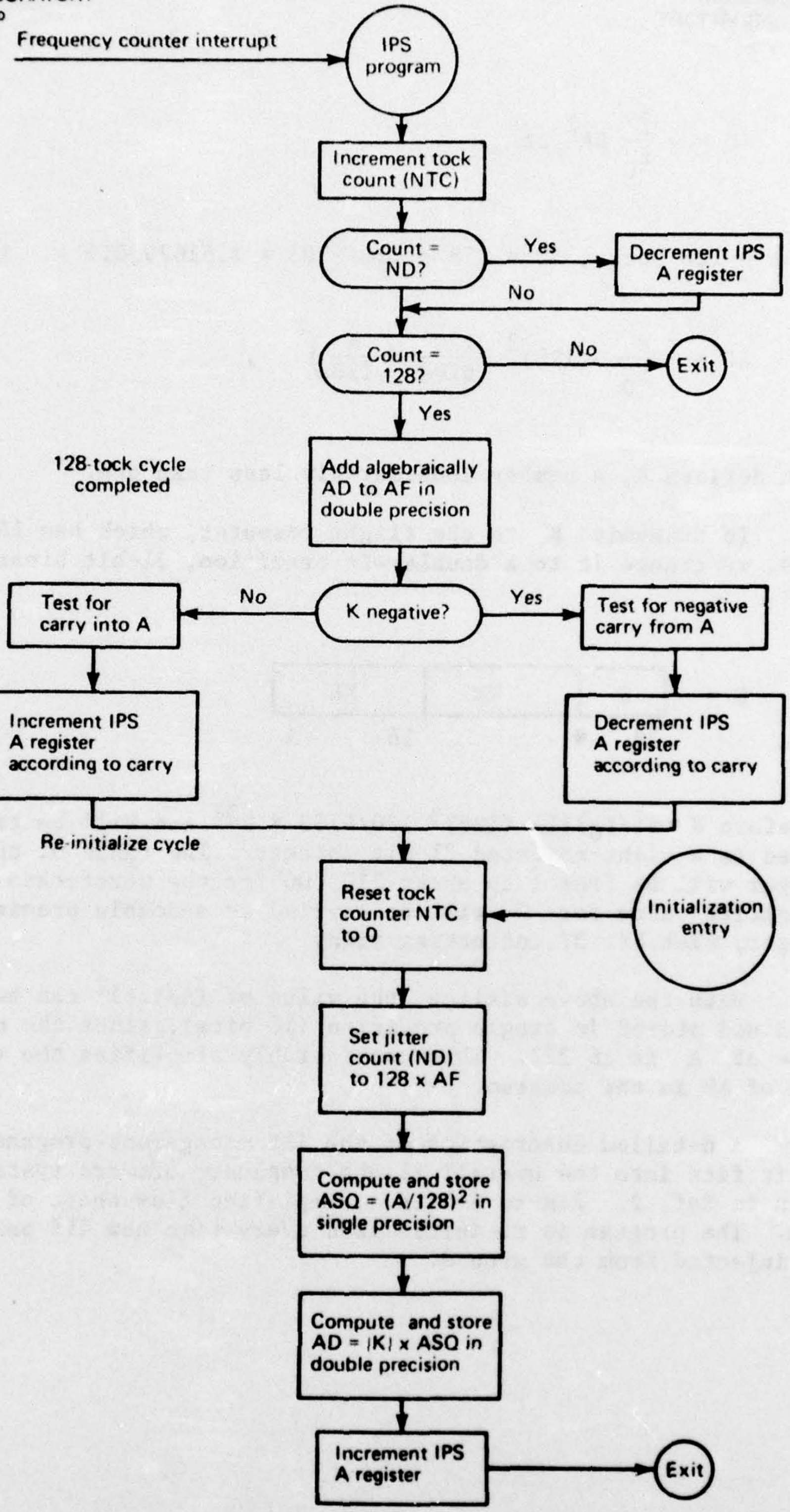


Fig. A-1 Simplified Flow Chart of IPS Program

Appendix B

PROGRAM LISTINGS

```
C
C     IPS CONTROL PROGRAM
C
C     PROGRAMMER: A. GOLDFINGER JHU/APL
C
      DIMENSION X(3),P(3,3)
      DOUBLE PRECISION DELT,F,FD,FC,FNOM,X,P,Z,S2M,Q2,Q3,STEER,TAU,A,Z
      DOUBLE PRECISION TL,TC,SECL,SECM,DELDAY,TM,TI,S2MD
      DOUBLE PRECISION SEC
      COMMON /C1/ S2M,Q2,Q3,FNOM,STEER,TAU,S2MD
   10 FORMAT(20H1IPS CONTROL PROGRAM )
   20 FORMAT(16H0INITIAL VALUES: )
   30 FORMAT(21H0LAST COMPUTED EPOCH: )
   40 FORMAT(6H DAY= ,I3,11H SECONDS= ,F12.6)
   41 FORMAT(6H DAY= ,I3,8H HOUR= ,I3,11H MINUTES= ,I3,
     1      11H SECONDS= ,F12.6)
   50 FORMAT(27H0ESTIMATED STATE VARIABLES:)
   60 FORMAT(13H TIME ERROR= ,F10.3,13H MICROSECONDS)
   70 FORMAT(12H FREQUENCY= ,F14.9,25H PARTS PER MILLION OFFSET)
   80 FORMAT(18H FREQUENCY DRIFT= ,F14.9,
     1    33H PARTS PER MILLION OFFSET PER DAY)
   81 FORMAT(36H COVARIANCE MATRIX (INTERNAL UNITS): )
   82 FORMAT(1H ,3D20.13/1H ,3D20.13/1H ,3D20.13)
   83 FORMAT(24H IPS CONTROL PARAMETERS: )
   84 FORMAT(4H A= ,F20.13,5H  B= ,I6,5H  K= ,I6)
   90 FORMAT(20H0INJECTION AT EPOCH:)
  100 FORMAT(27H0TIME MEASUREMENT AT EPOCH:)
  110 FORMAT(32H0FREQUENCY MEASUREMENT AT EPOCH:)
  120 FORMAT(25H0FILTERED STATE ESTIMATE:)
  130 FORMAT(16H0STATE ESTIMATE:)
  140 FORMAT(32H0STATE ESTIMATE AFTER INJECTION:)
  150 FORMAT(31H0******************************)
C
C     READ IN DATA
C
      CALL INPUT(IDAYL,SECL,DELT,F,FD,P,A,IB,K)
      WRITE(6,10)
      WRITE(6,150)
      WRITE(6,20)
      WRITE(6,30)
      WRITE(6,40) IDAYL,SECL
      WRITE(6,50)
      WRITE(6,60) DELT
      WRITE(6,70) F
      WRITE(6,80) FD
      WRITE(6,81)
      WRITE(6,82) P
      WRITE(6,83)
      WRITE(6,84) A,IB,K
C
C     CONVERT DATA TO INTERNAL FORM
C
      FNOM=5.D0*(1.D6+FNOM)
      X(1)=DELT*1.D-6
      X(2)=5.D0*(1.D6+F)/FNOM-1.D0
      X(3)=5.D0*FD/(86400.D0*FNOM)
      TL=SECL
      FC=X(2)
      TC=SECL
```

```
C
C     FILTER MEASUREMENTS
C
  200 CALL TASK(MEAS,IDAYM,SECM,Z)
      WRITE(6,150)
      IF(MEAS.EQ.0) WRITE(6,90)
      IF(MEAS.EQ.1) WRITE(6,100)
      IF(MEAS.EQ.2) WRITE(6,110)
      IF(MEAS.EQ.0) GO TO 800
      WRITE(6,40) IDAYM,SECM
      IF(MEAS.EQ.1) WRITE(6,60) Z
      IF(MEAS.EQ.2) WRITE(6,70) Z
      IF(MEAS.EQ.1) Z=Z*1.D-6
      IF(MEAS.EQ.2) Z=5.DO*(1.D6+Z)/FNOM-1.DO
      DELDAY=IDAYM-IDAYL
      TM=SECM+86400.DO*DELDAY
      CALL FILT(TL,TM,X,P,Z,MEAS)
      WRITE(6,120)
      DELT=X(1)*1.D6
      F=FNOM*(1.DO+X(2))/5.DO-1.D6
      FD=86400.DO*FNOM*X(3)/5.DO
      WRITE(6,60) DELT
      WRITE(6,70) F
      WRITE(6,80) FD
      TL=TM
      GO TO 200
C
C     PREDICT STATE AT TI
C
  800 IHR=SECM/3600.DO
      SEC=SECM-3600.DO*IHR
      MIN=SEC/60.DO
      SEC=SEC-60.DO*MIN
      WRITE(6,41) IDAYM,IHR,MIN,SEC
      DELDAY=IDAYM-IDAYL
      TI=SECM+86400.DO*DELDAY
      CALL PRED(TL,TI,X,P)
      WRITE(6,130)
      DELT=X(1)*1.D6
      F=FNOM*(1.DO+X(2))/5.DO-1.D6
      FD=86400.DO*FNOM*X(3)/5.DO
      WRITE(6,60) DELT
      WRITE(6,70) F
      WRITE(6,80) FD
C
C     COMPUTE IPS PARAMETERS
C
      DELT=X(1)
      F=X(2)
      FD=X(3)
      CALL IPS(TC,TI,DELT,F,FD,A,IB,K)
      X(1)=DELT
      X(2)=F
      X(3)=FD
C
C     CONVERT DATA TO EXTERNAL FORM
C
      DELT=X(1)*1.D6
      F=FNOM*(1.DO+X(2))/5.DO-1.D6
      FD=86400.DO*FNOM*X(3)/5.DO
```

```
C
C       OUTPUT DATA
C
        WRITE(6,150)
        WRITE(6,140)
        WRITE(6,30)
        WRITE(6,40) IDAYM,SECM
        WRITE(6,50)
        WRITE(6,60) DELT
        WRITE(6,70) F
        WRITE(6,80) FD
        WRITE(6,81)
        WRITE(6,82) P
        WRITE(6,83)
        WRITE(6,84) A,IB,K
        WRITE(6,150)
        CALL OUTPUT(IDAYM,SECM,DELT,F,FD,P,A,IB,K)
        STOP
        END
```

```
      BLOCK DATA
      COMMON /C1/ S2M,Q2,Q3,FNOM,STEER,TAU,S2MD
      DOUBLE PRECISION S2M,Q2,Q3,FNOM,STEER,TAU,S2MD
      DATA S2MD/ 1.0D-22/
      DATA S2M/ 4.0D-10/
      DATA Q2/ 2.00D-29/
      DATA Q3/ 4.0D-40/
      DATA FNOM/-84.48D0/
      DATA STEER/0.0D0/
      DATA TAU/86400.00/
      END

      SUBROUTINE INPUT(IDAYL,S53L,4ECT,6 6D.7 1.98 2'
      DIMENSION P(3,3)
      DOUBLE PRECISION DELT,F,FD,P,A,SECL
C
C     INPUT FORMATS FOLLOW
C
   10 FORMAT(I15,F15.8)
   20 FORMAT(3D20.13)
   30 FORMAT(D20.13,2I10)
C
C     READ IN DATA
C
      READ(5,10) IDAYL,SECL
      READ(5,20) DELT,F,FD
      DO 100 I=1,3
  100 READ(5,20) P(I,1),P(I,2),P(I,3)
      READ(5,30) A,IB,K
      RETURN
      END

      SUBROUTINE TASK(MEAS,IDAYM,SECM,Z)
      DOUBLE PRECISION Z,SECM
C
C     INPUT FORMAT
C
   10 FORMAT(2I15,2F15.8)
C
C     READ DATA
C
      READ(5,10) MEAS,IDAYM,SECM,Z
      RETURN
      END

      SUBROUTINE OUTPUT(IDAYM,SECM,DELT,F,FD,P,A,IB,K)
      DIMENSION P(3,3)
      DOUBLE PRECISION DELT,F,FD,P,A,SECM
C
C     OUTPUT FORMATS FOLLOW
C
   10 FORMAT(I15,F15.8)
   20 FORMAT(3D20.13)
   30 FORMAT(D20.13,2I10)
C
C     WRITE OUT DATA
C
      WRITE(7,10) IDAYM,SECM
```

```
      WRITE(7,20) DELT,F,FD
      DO 100 I=1,3
  100 WRITE(7,20) P(I,1),P(I,2),P(I,3)
      WRITE(7,30) A,IB,K
      RETURN
      END

      SUBROUTINE MM3(A,B,AB)
      DIMENSION A(3,3),B(3,3),AB(3,3)
      DOUBLE PRECISION A,B,AB
      DO 100 I=1,3
      DO 100 J=1,3
      AB(I,J)=0.0D0
      DO 100 K=1,3
  100 AB(I,J)=AB(I,J)+A(I,K)*B(K,J)
      RETURN
      END

      SUBROUTINE MMT3(A,B,ABT)
      DIMENSION A(3,3),B(3,3),ABT(3,3)
      DOUBLE PRECISION A,B,ABT
      DO 100 I=1,3
      DO 100 J=1,3
      ABT(I,J)=0.0D0
      DO 100 K=1,3
  100 ABT(I,J)=ABT(I,J)+A(I,K)*B(J,K)
      RETURN
      END
```

```
      SUBROUTINE FILT(TL,TM,X,P,Z,MEAS)
      COMMON /C1/S2M,Q2,Q3,FNOM,STEER,TAU,S2MD
      DIMENSION X(3),P(3,3),E(3,3),PU(3,3),PHI(3,3),PX(3,3),K(3)
      DOUBLE PRECISION X,P,E,PU,PHI,PX,DENOM,S2M,Q2,Q3,K,ZT,Z,DT,F,FD
      DOUBLE PRECISION FNOM,STEER,TAU,FC,TL,TM,S2MD,TDEL
   10 FORMAT('OFILTER GAINS:')
   20 FORMAT(' K1=  ',D20.13,'  K2=  ',D20.13,'  K3=  ',D20.13)
C
C     GET TRANSITION MATRIX
C
      PHI(1,1)=1.D0
      PHI(1,2)=TM-TL
      PHI(1,3)=0.5D0*(TM-TL)**2
      PHI(2,1)=0.0D0
      PHI(2,2)=1.0D0
      PHI(2,3)=TM-TL
      PHI(3,1)=0.0D0
      PHI(3,2)=0.0D0
      PHI(3,3)=1.0D0
C
C     PROPAGATE OLD COVARIANCE MATRIX
C
      CALL MMT3(P,PHI,PX)
      CALL MM3(PHI,PX,P)
      TDEL=TM-TL
      P(1,1)=P(1,1)+Q2*TDEL**3/3.D0+Q3*TDEL**5/20.D0
      P(1,2)=P(1,2)+Q2*TDEL**2/2.D0+Q3*TDEL**4/8.D0
      P(1,3)=P(1,3)+Q3*TDEL**3/6.D0
      P(2,1)=P(2,1)+Q2*TDEL**2/2.D0+Q3*TDEL**4/8.D0
      P(2,2)=P(2,2)+Q2*TDEL+Q3*TDEL**3/3.D0
      P(2,3)=P(2,3)+Q3*TDEL**2/2.D0
      P(3,1)=P(3,1)+Q3*TDEL**3/6.D0
      P(3,2)=P(3,2)+Q3*TDEL**2/2.D0
      P(3,3)=P(3,3)+Q3*TDEL
C
C     PROPAGATE STATE
C
      DT=X(1)+PHI(1,2)*X(2)+PHI(1,3)*X(3)
      F=X(2)+PHI(2,3)*X(3)
      FD=X(3)
      X(1)=DT
      X(2)=F
      X(3)=FD
C
C     COMPUTE FILTER GAINS
C
      IF(MEAS.EQ.1) GO TO 250
      IF(MEAS.EQ.2) GO TO 270
  250 DENOM=S2M+P(1,1)
      K(1)=P(1,1)/DENOM
      K(2)=P(2,1)/DENOM
      K(3)=P(3,1)/DENOM
      GO TO 300
  270 DENOM=S2MD+P(2,2)
      K(1)=P(1,2)/DENOM
      K(2)=P(2,2)/DENOM
      K(3)=P(3,2)/DENOM
  300 CONTINUE
C
```

```
C     GET MEASUREMENT RESIDUAL
C
      IF(MEAS.EQ.1) GO TO 350
      IF(MEAS.EQ.2) GO TO 370
  350 ZT=Z-X(1)
      GO TO 400
  370 ZT=Z-X(2)
C
C     GET NEW STATE ESTIMATE
C
  400 X(1)=X(1)+K(1)*ZT
      X(2)=X(2)+K(2)*ZT
      X(3)=X(3)+K(3)*ZT
      WRITE(6,10)
      WRITE(6,20) K
C
C     UPDATE COVARIANCE MATRIX
C
      DO 600 I=1,3
      DO 600 J=1,3
  600 E(I,J)=0.0D0
      E(1,1)=1.0D0
      E(2,2)=1.0D0
      E(3,3)=1.0D0
      IF(MEAS.EQ.1) GO TO 650
      IF(MEAS.EQ.2) GO TO 670
  650 E(1,1)=E(1,1)-K(1)
      E(2,1)=E(2,1)-K(2)
      E(3,1)=E(3,1)-K(3)
      GO TO 700
  670 E(1,2)=E(1,2)-K(1)
      E(2,2)=E(2,2)-K(2)
      E(3,2)=E(3,2)-K(3)
  700 CALL MM3(E,P,PU)
      DO 800 I=1,3
      DO 800 J=1,3
  800 P(I,J)=PU(I,J)
      RETURN
      END
```

```
      SUBROUTINE PRED(TM,TI,X,P)
      COMMON /C1/ S2M,Q2,Q3,FNOM,STEER,TAU,S2MD
      DIMENSION X(3),XI(3),P(3,3),PX(3,3),PHI(3,3)
      DOUBLE PRECISION X,XI,P,PX,PHI,TM,TI,S2MD,TDEL
      DOUBLE PRECISION S2M,Q2,Q3,FNOM,STEER,TAU,FC
C
C     GET TRANSITION MATRIX
C
      PHI(1,1)=1.D0
      PHI(1,2)=TI-TM
      PHI(1,3)=0.5D0*(TI-TM)**2
      PHI(2,1)=0.D0
      PHI(2,2)=1.D0
      PHI(2,3)=TI-TM
      PHI(3,1)=0.D0
      PHI(3,2)=0.D0
      PHI(3,3)=1.D0
C
C     PROPAGATE STATE
C
      XI(1)=X(1)+PHI(1,2)*X(2)+PHI(1,3)*X(3)
      XI(2)=X(2)+PHI(2,3)*X(3)
      XI(3)=X(3)
      X(1)=XI(1)
      X(2)=XI(2)
      X(3)=XI(3)
C
C     PROPAGATE TRANSITION MATRIX
C
      CALL MMT3(P,PHI,PX)
      CALL MM3(PHI,PX,P)
      TDEL=TI-TM
      P(1,1)=P(1,1)+Q2*TDEL**3/3.D0+Q3*TDEL**5/20.D0
      P(1,2)=P(1,2)+Q2*TDEL**2/2.D0+Q3*TDEL**4/8.D0
      P(1,3)=P(1,3)+Q3*TDEL**3/6.D0
      P(2,1)=P(2,1)+Q2*TDEL**2/2.D0+Q3*TDEL**4/8.D0
      P(2,2)=P(2,2)+Q2*TDEL+Q3*TDEL**3/3.D0
      P(2,3)=P(2,3)+Q3*TDEL**2/2.D0
      P(3,1)=P(3,1)+Q3*TDEL**3/6.D0
      P(3,2)=P(3,2)+Q3*TDEL**2/2.D0
      P(3,3)=P(3,3)+Q3*TDEL
      RETURN
      END
```

```
      SUBROUTINE IPS(TO,TI,DELT,F,FD,A,IB,K)
      DOUBLE PRECISION DELT,F,FD,FC,FNOM,S2M,Q2,Q3,STEER,TAU
      DOUBLE PRECISION FO,FDO,FOP,FDOP,CIPS,B,EK,EKALT,A,AALT,ENINT,FOUT
      DOUBLE PRECISION TO,TI,TINT,ENINT,FOUTP,AO,S2MD
      COMMON /C1/ S2M,Q2,Q3,FNOM,STEER,TAU,S2MD
   10 FORMAT(44H0VALUES OF IPS PARAMETERS AT INJECTION TIME:)
   15 FORMAT(36H0OSCILLATOR STATE AT INJECTION TIME:)
   20 FORMAT(12H FREQUENCY= ,F14.9,25H PARTS PER MILLION OFFSET)
   30 FORMAT(18H FREQUENCY DRIFT= ,F14.9,
     1    33H PARTS PER MILLION OFFSET PER DAY)
   31 FORMAT(14H0STEERING FOR ,F10.3,24H MICROSECONDS TIME ERROR )
   32 FORMAT(4H AT ,F11.3,28H SECONDS PAST INJECTION TIME)
   35 FORMAT(26H0DESIRED OUTPUT FREQUENCY:)
   50 FORMAT(35H0POSSIBLE VALUES OF IPS PARAMETERS:)
   60 FORMAT(4H A= ,F20.13,5H B=  ,I6,5H K=  ,I6)
   70 FORMAT(26H0NO A,B SOLUTIONS IN RANGE)
   80 FORMAT(26H0CHOICE OF IPS PARAMETERS:)
C
C     CONVERT DATA TO INTERNAL FORM
C
      A=A+1.0D0
      IB=IB+1
      B=IB
      EK=K
      EK=EK*2.0D0**(-31)
C
C     PROPAGATE A TO TI
C
      TINT=TI-TO
      ENINT=TINT/2.516795018D0
      A=A/(1.0D0-ENINT*EK*A/16384.0D0)
      WRITE(6,10)
      AO=A-1.0D0
      IBO=IB-1
      KO=K
      WRITE(6,60) AO,IBO,KO
C
C     GET OSCILLATOR FREQUENCY AT TI
C
      CIPS=1.0D0-(1.0D0+1.0D0/A)/B
      FO=(F+1.D0)/CIPS-1.D0
      FOP=FNOM*(1.D0+FO)/5.D0-1.D6
      WRITE(6,15)
      WRITE(6,20) FOP
C
C     GET OSCILLATOR DRIFT
C
      FDO=(FD-(FO+1.D0)*EK*6103.D0/(B*120.D0*128.D0**3))/CIPS
      FDOP=86400.D0*FNOM*FDO/5.D0
      WRITE(6,30) FDOP
C
C     GET DESIRED OUTPUT FREQUENCY
C            NOTE: DESIRED FDO=0
C
      STEERP=STEER*1.0E6
      WRITE(6,31) STEERP
      WRITE(6,32) TAU
      FOUT=(STEER-DELT)/TAU
      FOUTP=FNOM*(1.D0+FOUT)/5.0D0-1.D6
```

```
      WRITE(6,35)
      WRITE(6,20) FOUTP
C
C     GET IPS PARAMETERS
C
C     INITIAL A AND B SO THAT F=FOUT:
C
      CIPS=(FOUT+1.D0)/(FO+1.D0)
      B=1.D0/(1.D0-CIPS)
      IB=B+B/6301.D0
      A=B/(IB-B)
      IBALT=IB+1
      AALT=B/(IBALT-B)
C
C     GET K SO THAT FDOUT=0
C
      EK=-FDO*IB*128.D0**3*120.D0/((FO+1.D0)*6103.D0)*CIPS
      EKALT=-FDO*IBALT*128.D0**3*120.D0/((FO+1.D0)*6103.D0)*CIPS
C
C     CHOOSE A,B,K
C
      WRITE(6,50)
      AO=A-1.D0
      IBO=IB-1
      KO=EK*2.D0**31
      WRITE(6,60) AO,IBO,KO
      AO=AALT-1.D0
      IBO=IBALT-1
      KO=EKALT*2.D0**31
      WRITE(6,60) AO,IBO,KO
      I=1
      IX=1
      IF((A.LT.6301.).OR.(A.GT.13470.)) I=0
      IF((AALT.LT.6301.).OR.(AALT.GT.13470.)) IX=0
      IF((IB.LT.6669).OR.(IB.GT.12428)) I=0
      IF((IBALT.LT.6669).OR.(IBALT.GT.12428)) IX=0
      IF((I.EQ.0).AND.(IX.EQ.0)) GO TO 200
      IF((I.EQ.0).AND.(IX.NE.0)) GO TO 300
      IF((I.NE.0).AND.(IX.EQ.0)) GO TO 400
      IF(A.LT.AALT) ICH=1
      IF(A.GE.AALT) ICH=2
      GO TO 500
  200 WRITE(6,70)
      GO TO 600
  300 ICH=2
      GO TO 500
  400 ICH=1
      GO TO 500
  500 IF(ICH.EQ.1) GO TO 550
      A=AALT
      IB=IBALT
      EK=EKALT
  550 WRITE(6,80)
      AO=A-1.D0
      IBO=IB-1
      KO=EK*2.D0D**31
      WRITE(6,60) AO,IBO,KO
C
C      GET STATE AFTER INJECTION
C
```

```
      B=IB
      CIPS=1.D0-(1.D0+1.D0/A)/B
      EK=K0*2.0D0**(-31)
      F=(F0+1.D0)*CIPS-1.D0
      FD=FD0*CIPS+(F0+1.D0)*EK*6103.D0/(B*120.D0*128.D0**3)
C
C     CONVERT DATA TO EXTERNAL FORM
C
  600 A=A-1.D0
      IB=IB-1
      EK=EK*2.D0**31
      K=EK
      RETURN
      END
```

INITIAL DISTRIBUTION EXTERNAL TO THE APPLIED PHYSICS LABORATORY*

The work reported in TG 1301 was done under Navy Contract N00017-72-C-4401. This work is related to Task S2V, which is supported by SSPO, SP-243.

ORGANIZATION	LOCATION	ATTENTION	No. of Copies
DEPARTMENT OF DEFENSE			
DDC	Alexandria, VA		12
Department of the Navy			
Strategic Systems Project Office	Arlington, VA	SP-243	1
NAVSEASYSCOM	Washington, DC	SEA-09G3	2
NAVPRO	Laurel, MD		1

Requests for copies of this report from DoD activities and contractors should be directed to DDC, Cameron Station, Alexandria, Virginia 22314 using DDC Form 1 and, if necessary, DDC Form 55.

*Initial distribution of this document within the Applied Physics Laboratory has been made in accordance with a list on file in the APL Technical Publications Group.