TECHNICAL MEMORANDUM (NASA) 88

AREA NAVIGATION IMPLEMENTATION FOR A MICROCOMPUTER-BASED LORAN-C RECEIVER

This report describes the development of an area navigation program and the implementation of this software on a microcomputer-based Loran-C receiver to provide high-quality, practical area navigation information for general aviation. This software provides range and bearing angle to a selected waypoint, cross-track error, course deviation indication (CDI), ground speed and estimated time of arrival at the waypoint. The range/bearing calculation, using an elliptical earth model, provides very good accuracy; the error does not exceed more than 0.012 nm (range) or 0.09° (bearing) for a maximum range to 530 nm. α-β filtering is applied in order to reduce the random noise on Loran-C raw data and in the ground speed calculation. Due to the α-β filtering, the ground speed calculation has good stability for constant or low-accelerative flight. The execution time of this software is approximately 0.2 second. Flight testing was done with a prototype Loran-C front-end receiver, with the Loran-C area navigation softwar demonstrating the ability to provide navigation for the pilot to any point in the Loran-C coverage area in true area navigation fashion without line-of-sight and range restriction typical of VOR area navigation.

by

Fujiko Oguri

Avionics Engineering Center
Department of Electrical and Computer Engineering
Ohio University
Athens, Ohio 45701

August 1983

Prepared for

NASA Langley Research Center
Hampton, Virginia 23665
Contract NGR 36-009-017

(NASA-CR-173048) AREA NAVIGATION IMPLEMENTATION FOR A MICROCOMPUTER-BASED LORAN-C RECEIVER (Ohio Univ.) 150 p HC A07/MF A01 CSCL 17G

N83-33867

Unclas
G3/04 15130

TABLE OF CONTENTS

ORIGINAL PAGE IS OF POOR QUALITY

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (CONTINUED)

ORIGINAL PAGE IS
OF POOR QUALITY

LIST OF FIGURES (CONTINUED)

LIST OF FIGURES (CONTINUED)

LIST OF TABLES

I. INTRODUCTION AND SUMMARY

This paper describes specific engineering work which has been done to make Loran-C a more useful and practical navigation system for general aviation. This work, in particular, deals with development of new software, and implementation of this software on a (MOS6502) microcomputer to provide high quality practical area navigation information directly to the pilot. Flight tests have been performed specifically to examine the efficacy of this new software. Final results were exceptionally good and clearly demonstrate the merits of this new Loran-C area navigation system.

LORAN-C (Long Range Navigation) is a hyperbolic, radio navigation system that has been in operation since 1958 [1]. It uses ground waves at low frequencies to provide positional information, not restricted to line-of-sight [2]. This system can be used in nearly all weather conditions to obtain position accuracies which are essentially independent of altitude. As of 1983, it is not yet a complete navigation system in the United States. In the midwest, which constitutes one third of the U.S. land area, coverage is deficient for some flight operations.

The VOR/DME (VHF Omni-directional Range/Distance Measuring Equipment) navigation system is well known as the contemporary, short-range navigation system which covers the whole United States with over 1000 stations, but this system is sensitive to siting and terrain, and has limits for low altitude coverage because VOR is a line-of-sight system. By relieving these shortcomings, Loran is considered to be a possible supplement for VOR/DME system [3].

The hyperbolic lines of position associated with Loran-C present a problem for the pilot. Historically, the hyperbolic lines of position do not convert readily to a meaningful display without comparatively high airborne equipment cost. However, the present availability of microprocessors makes low-cost airborne coordinate conversion equipment feasible. Contemporary technology provides for light-weight small-volume equipment with a low power drain for small aircraft. Thus, automatic Loran-C navigation can be made practical for general aviation users simply by making use of a microcomputer.

In this paper, work is described indicating rather elementary mechanizations that can provide the pilot very useful navigation at all altitudes. This development of software provides Area (Random) Navigation (RNAV) information from Time Differences (TDs) in raw form using an elliptical earth model and a spherical model. It is prepared for the microcomputer based Loran-C receiver which was developed at the Ohio University Avionics Engineering Center. In order to compute navigational information, a microcomputer(MOS6502) and a mathematical chip (Am9511A) were combined with the Ohio University Loran-C receiver. Final data in the report reveals that this software indeed provides accurate information with reasonable operation times.

II. PROBLEM DESCRIPTION

The purpose of an air navigation system is to provide position information to a pilot in all weather conditions. In order to achieve this goal, certain operational factors, such as accuracy, coverage, integrity and reliability must be considered.

A. Present-Day Navigation System.

1. VOR/DME system. Early air navigation relied heavily on good visibility and pilot skill. However, increased interest in aviation as a viable transportation system necessitated developing navigation equipment that would provide guidance in all weather conditions.

By the 1930s, radio navigation systems were being used routinely. The VOR (Very High Frequency Omnidirectional Range) was developed by the end of the Second World War, and shortly thereafter, was put into use across the U.S. VOR was accepted as the international standard for air navigation by the International Civil Aviation Organization (ICAO) in 1949. VOR transmissions, which range in frequency from 108 to 118MHz, provide signals which the airborne receiver uses to define an angular bearing with respect to the transmitting station [4].

In the 1960's, DME (Distance Measuring Equipment) was added to VOR, as a part of the colocated TACAN (Tactical Air Navigation) ground-radio beacon system [5]. DME determines the distance from the VORTAC to the aircraft, therefore, a VOR-DME station (or VORTAC) provides magnetic bearing and the distance to the station for the pilot (figure 2-1). The VOR/DME system is relatively easy to use and easy to visualize in a navigation sense; hence, it has become a useful, practical system for civil aviation users.

While the VOR/DME system satisfies most of today's enroute navigational requirements, it has some notable disadvantages. One of these is a result of the fact that VHF propagation is essentially line-of-sight, meaning that there must be no obstructions between the transmitter and the receiver. Thus, low-altitude aircraft cannot receive signals from behind mountains, or in valleys. Consequently, even with more than 1000 VOR stations operating in the United States, complete low-altitude coverage is not provided. Further, there is a high cost associated with the maintenance of 1000 stations. Moreover, low-altitude aircraft like helicopters may need additional systems to fulfill their navigational requirements.

2. RNAV using VOR/DME. The definition of Area/Random Navigation (RNAV) according to Advisory Circular 90-45A, is as follows [6];

> 'A method of navigation that permits aircraft operations on any desired course within the coverage of station referenced navigation signals or within the limits of self-contained system capability.'

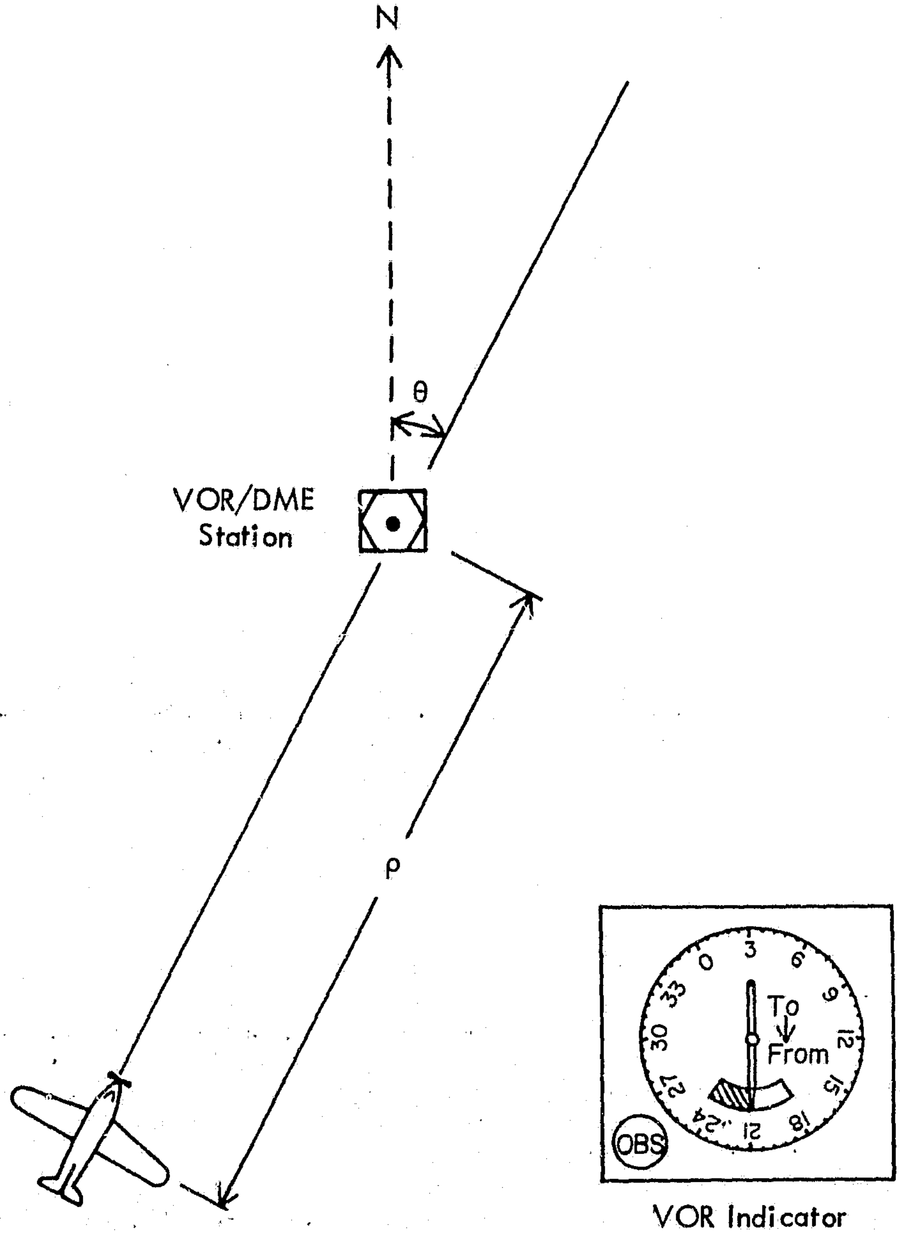


Figure 2-1. VOR/DME Navigation System.

RNAV equipment (primarily VOR/DME dependent systems) is now available on the market which enables an aircraft to fly directly to any destination within VOR/DME signal coverage area. A discussion of the computations involved with RNAV is given in appendix A. Numerous equipments are currently marketed to perform these coordinate translations, conversion and rotations (e.g., King Radio Corp. KNS-81 Navigation System).

Although a large number of RNAV routes have been established by the FAA, the 'line-of-sight' problem has not been solved for low-altitude navigation.

3. Other systems. There are other navigation systems, such as, Non-Directional Beacons (NDB), Omega, inertial navigation, Global Positioning System (GPS) and Loran-C.

The NDB, sometimes called ADF (Automatic Direction Finding), system operates in the 200 to 1600 KHz bands to indicate the direction of a selected ground station with respect to aircraft heading as depicted in figure 2-2. This system is inexpensive but does not provide the guidance afforded by VOR. For example, if there is crosswind, the aircraft may drift with respect to the desired path [7]. A comparison between VOR and ADF in the presence of a crosswind is shown in figure 2-3. And also, ADF is only as accurate as the magnetic compass.

Omega is a very-low-frequency (VLF) hyperbolic navigation system which offers nearly worldwide coverage with eight stations. Position determination is based on a comparison of phase values obtained from the signal of three or more transmitting stations (figure 2-4). Inaccuracy can be caused by ionospheric changes and conductivity of the ground. Therefore, the system needs propagation-phase-correction for proper phase stability. Lane skipping also causes inaccuracy. Consequently, it is a suitable aviation aid primarily for long-range, oceanic navigation [8], or for updating inertial systems.

The inertial system is a completely self-contained navigation system based on the measurement of aircraft acceleration using accelerometers and gyros. It provides position, velocity, altitude and heading at all latitudes in all weather [9]. However, because the position and velocity information degrades as a function of time elapsed and the airborne system expense is high, this method, used alone, is unsuitable for the general aviation applications.

GPS is a radio-navigation system using satellites in space. Extremely accurate three-dimensional position and velocity information can be obtained from the system worldwide and unaffected by weather conditions. The position determination is based on the measurement of the transit time of microwave signals from four or more satellites. Three satellites permit solving of the user position equations in three dimensions, and the fourth satellite estimates the user's clock error [10] (figure 2-5). GPS is still being tested and is expensive; however, the

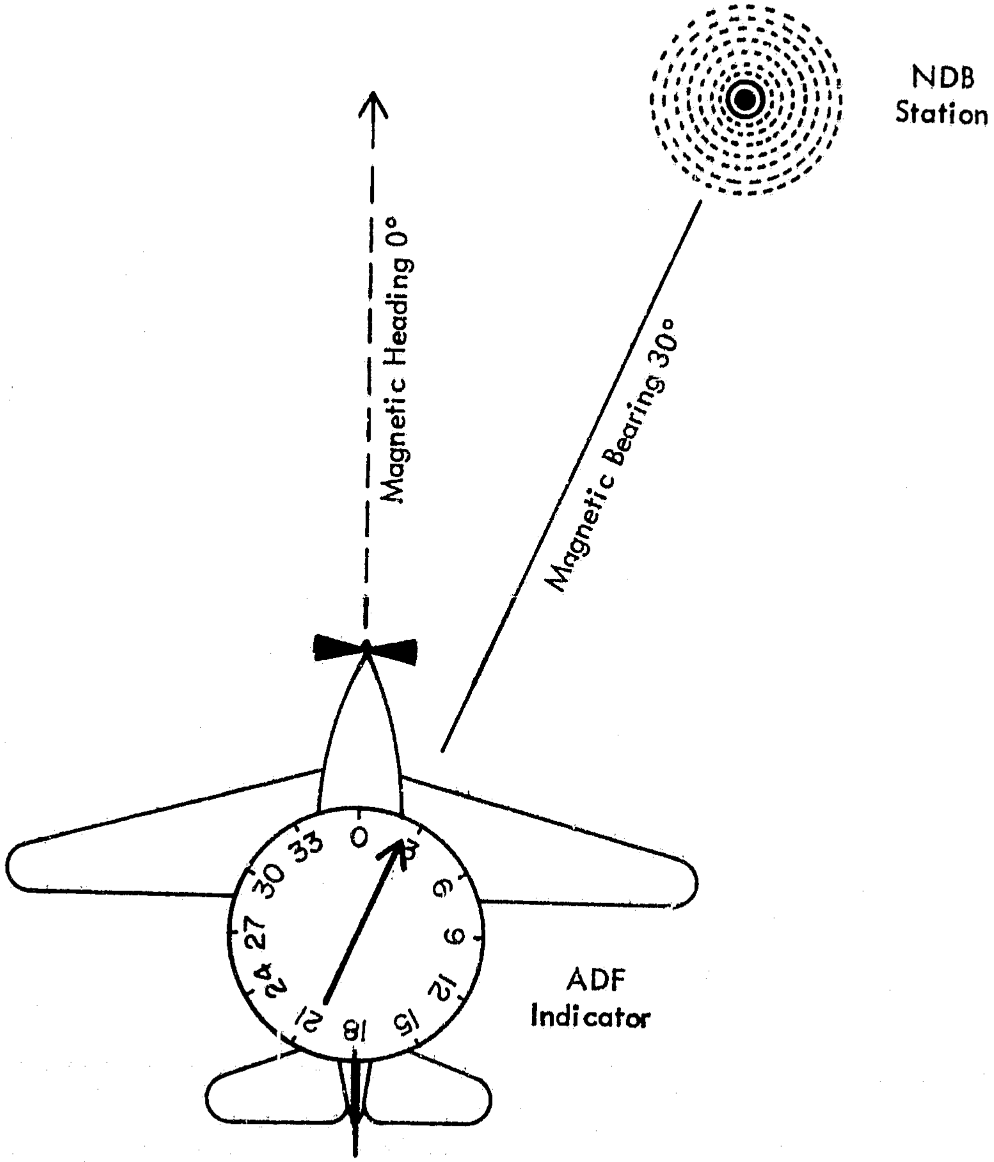


Figure 2-2. NDB (ADF) Navigation System.

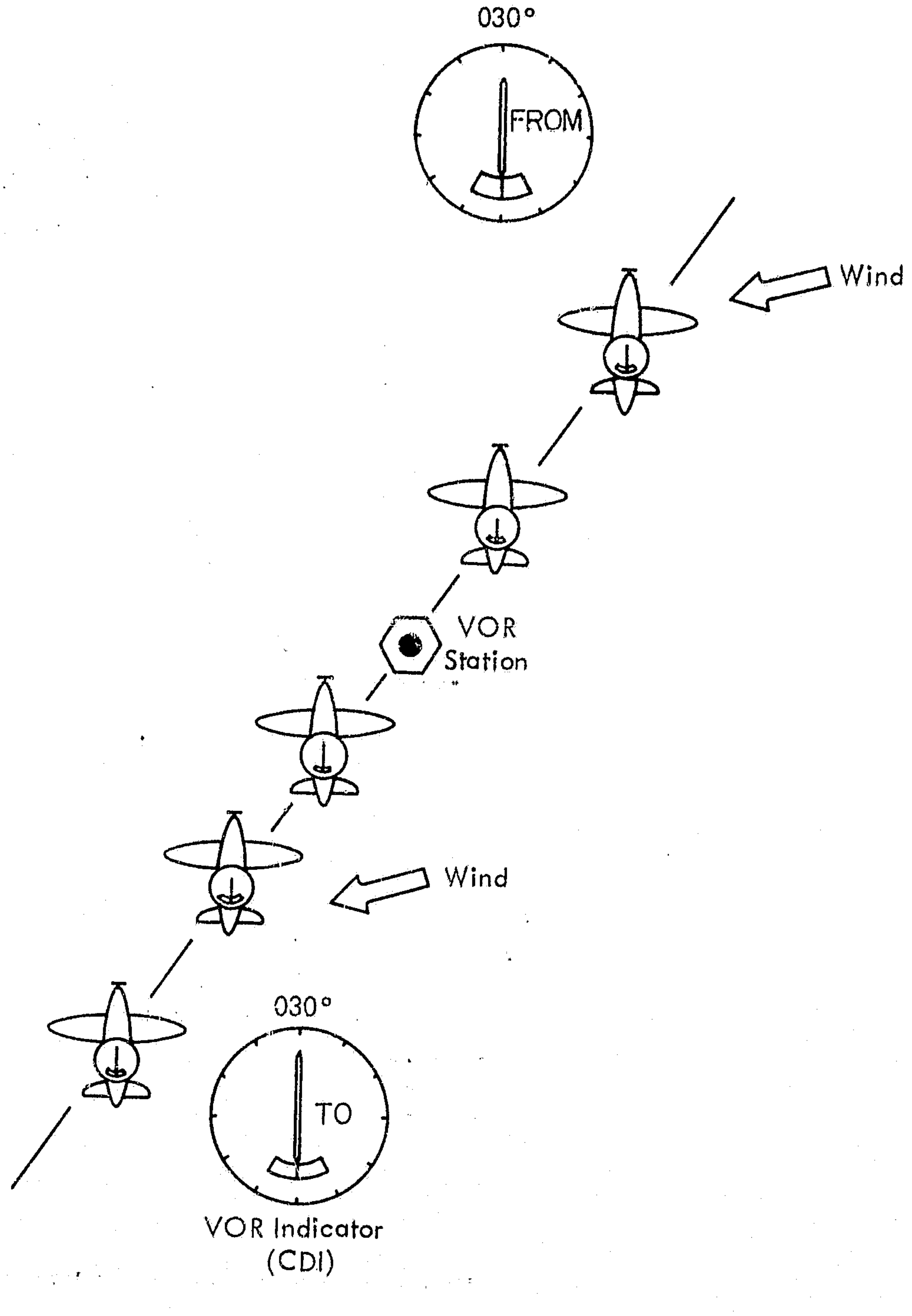


Figure 2-3a. Flying To/From Station with Cross-Wind. VOR with wind correction.

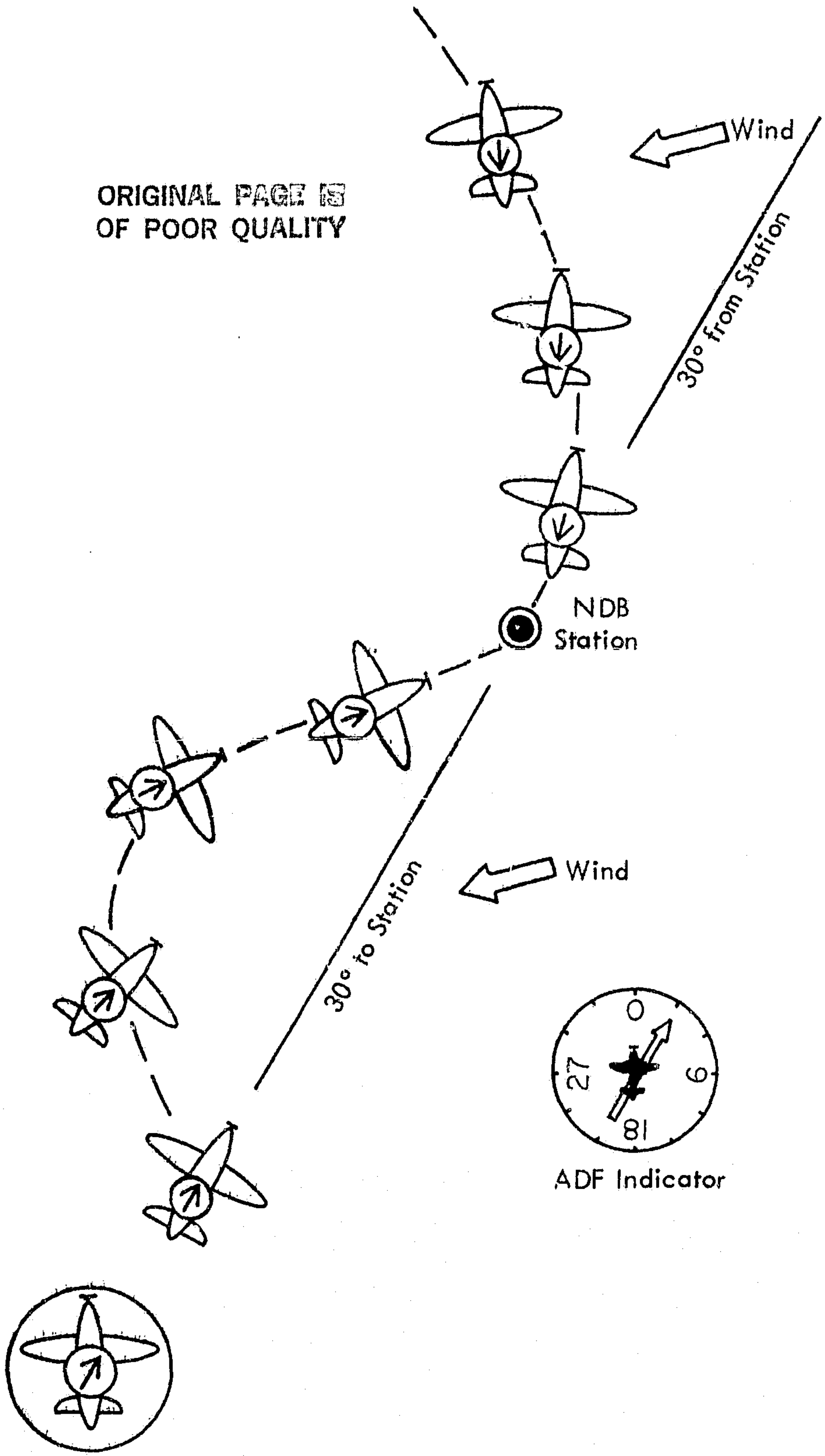


Figure 2-3b. Flying To/From Station with Cross-Wind. ADF without wind correction.

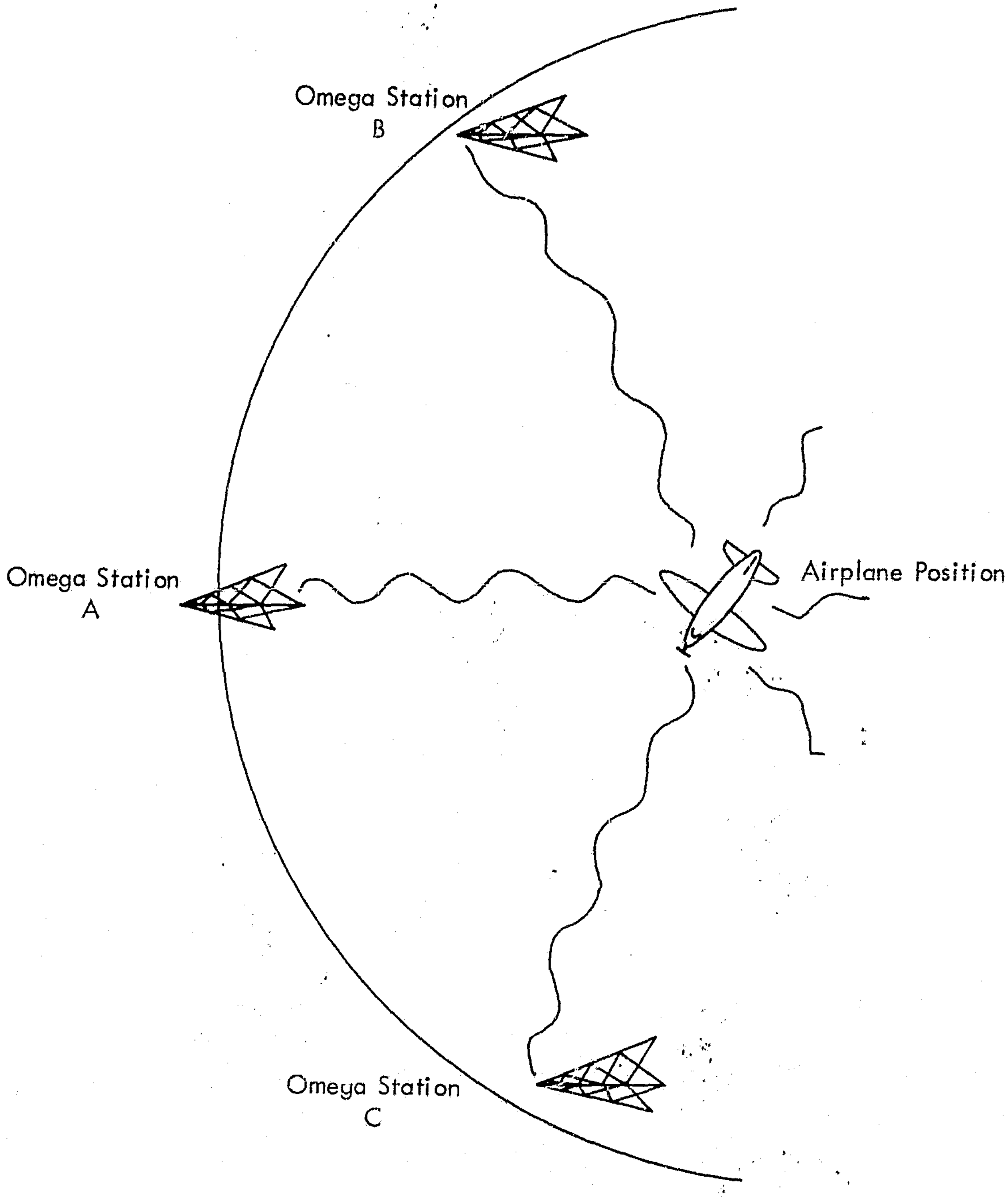


Figure 2-4. Omega Navigation System.

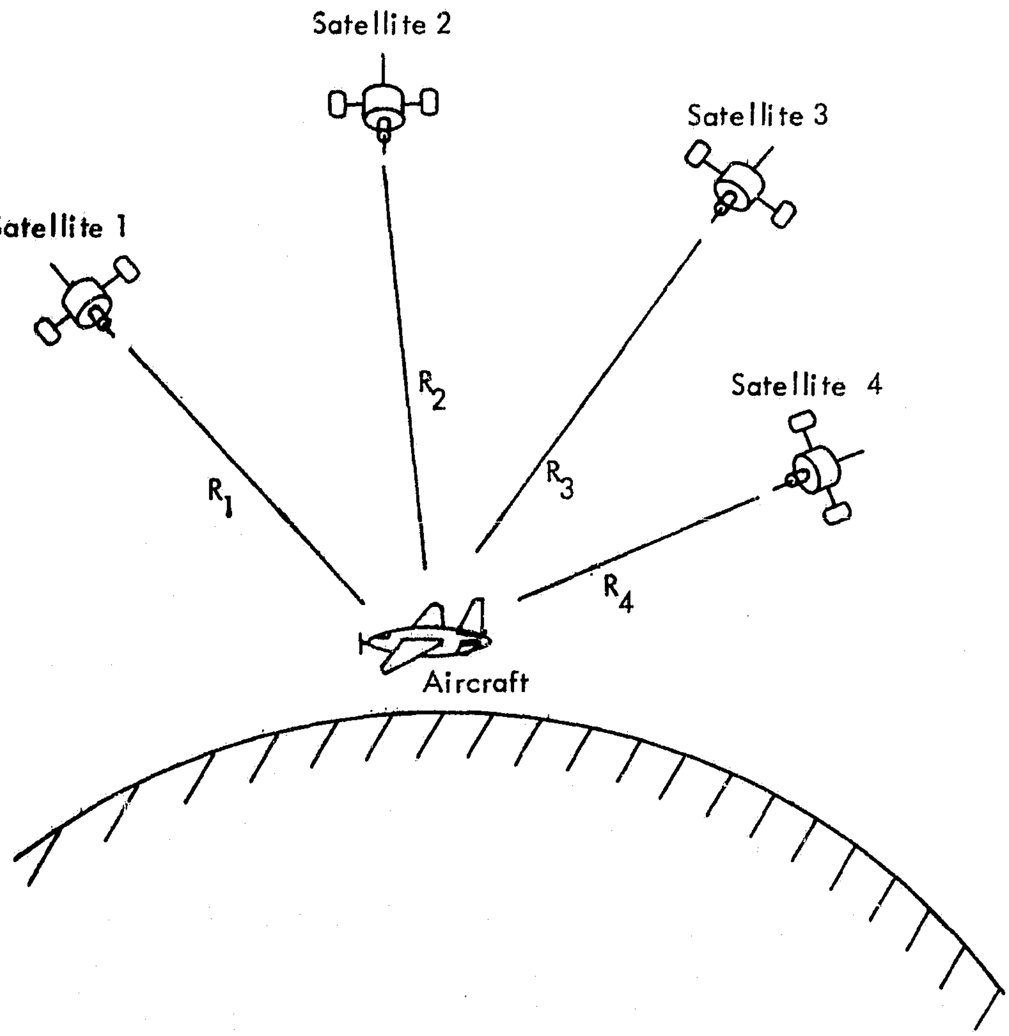


Figure 2-5. GPS Navigation System.

implementation will begin in 1986 and will be completed approximately two years later [11]. It is regarded by some as a future navigation system which has a capability to be a single universal system.

The Loran-C system is now being evaluated as a supplemental system, and possibly, as a replacement for the contemporary VOR/DME system, which is described in the next section. It is the Loran-C system that is addressed in this paper.

B. Low-Altitude Navigation Using Loran-C.

Loran-C system uses a pulsed, low-frequency (LF) signal, resulting in a hyperbolic navigation system. The intrinsic stability of LF, and the time difference measurement of pulsed signals, provide reasonable accuracy (about 2-D r.m.s. system error of ±0.3nm).

The relatively low propagation path losses of LF ground waves, and the resulting long station-to-station separation provides a wide coverage area, including low altitude coverage in mountains and valleys. Thus, about 40 stations are sufficient to cover the entire continental United States [12]. Figure 2-6 shows the present Loran-C coverage area.

Three or four Loran-C radio navigation transmitting stations are constructed to form chains. Two sets of hyperbolic lines give position information in a hyperbolic coordinate system, and this information can be readily used as input to an RNAV computer. As a result, if the Loran-C user chooses, he can perform great-circle navigation between any two points within the coverage area of the entire Loran-C system.

There are certain problems associated with the use of Loran-C. Geometric Dilution of Position (GDOP) causes inaccuracies typical of any hyperbolic system. Inaccuracy arises when the crossing angle of two lines of position is small or when the aircraft position is near the baseline extension for a master-secondary pair. Operating too close to the transmitter also causes instability on time difference (TD) readings [13]. However, these problems can be avoided by provisionally using another station, or avoiding passage near a station.

There are other problems due to interference caused by natural causes and by man-made sources. Static and lightning are examples of natural interference. However, static noise reduction from 20dB to 50dB in the Loran-C frequency band is achieved by implementation of static dischargers for aircraft [14]. Man-made sources such as powerline carrier systems and noise sources near airports cause some interference. According to some measurements, interference was found when the carrier frequency of powerline is synchronous with the frequency of the Loran-C pulse spectrum, and solutions to this problem are still under consideration. Interference measurements near major airports did not show significant interference to Loran-C receivers [15].

The effect of station outages is another problem; reliability of Loran-C stations must be increased for practical, safe air navigation [16].

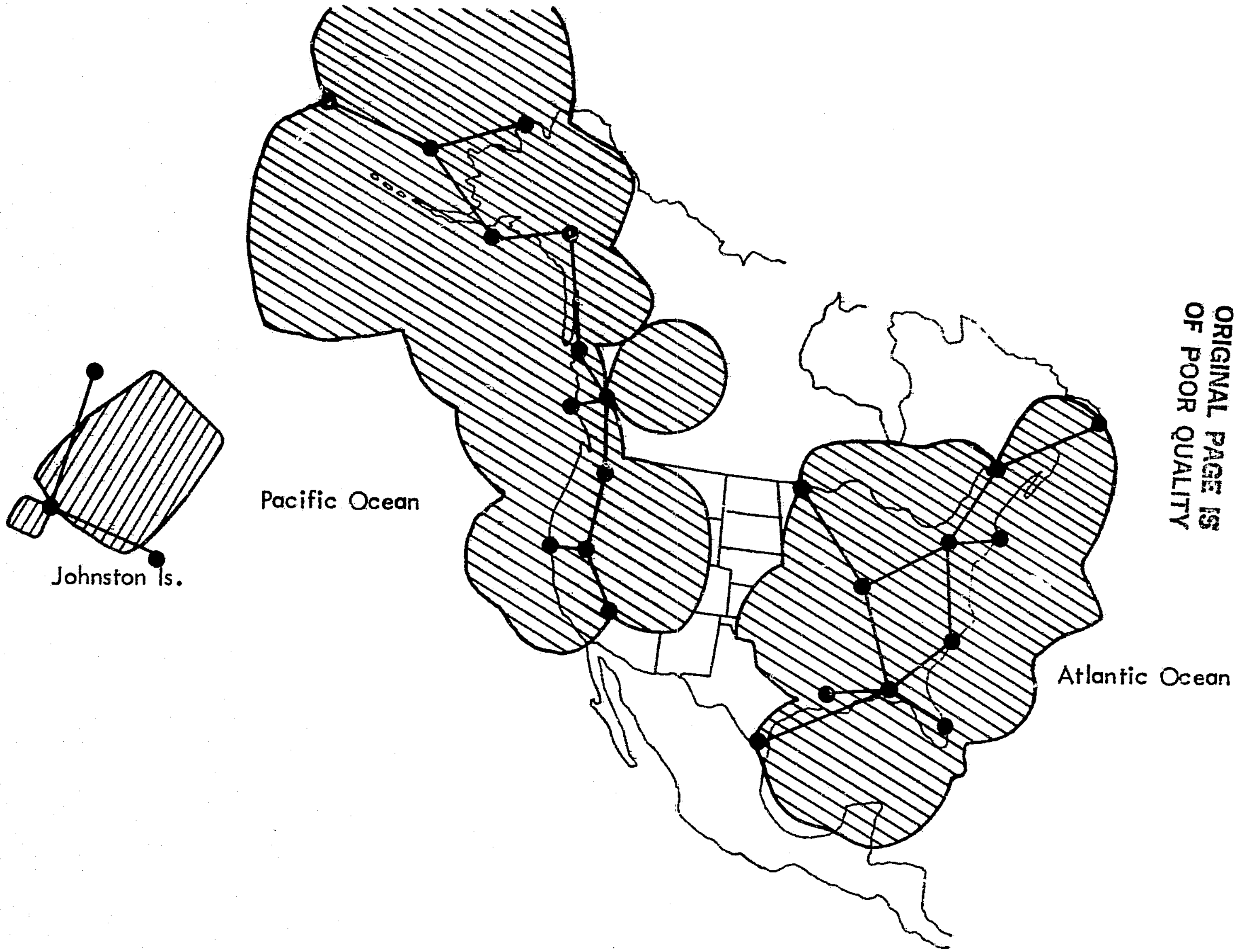


Figure 2-6. Present Loran-C Coverage Area in the United States.

Early Loran-C receivers were configured for military use. Since Loran-C requires a relatively complex signal processing system for airborne use, they cost $40-50,000, which made them impractical for general aviation. This processing can be achieved with a low-cost microcomputer as is presented here. This allows a high-performance Loran-C receiver to be in the the cost and size range that is attractive for general aviation usage.

III. LORAN-C NAVIGATION

A. Background.

During World War II, Loran-A was developed as a pioneering long-range radio navigation system at the Radiation Laboratory of the Massachusetts Institute of Technology (MIT), and operated under the U.S. Coast Guard to satisfy wartime need. Loran-A operated at 1600kHz and provided position information to receivers aboard military and commercial ships and planes. Beginning in 1977, it has been phased out with the shut down of the final chain in 1980.

After the war, the Department of Defense developed a new generation of radio navigation system, called Loran-C, operating at 100kHz. Loran-C provides improved accuracy and increased area of coverage. The Loran-C system began to replace Loran-A in the late 1950s. The Department of Defense predicted that the Loran-C system would cover all of the coastal waters and the entire U. S. by the end of this decade [17].

B. Low Frequency System.

The Loran-C system uses time measurement with ground waves at low frequency (LF). LF signals, which are the most suitable for time measurement accuracy, have predictable ground wave propagation conditions though they are subject to skywave interference at long ranges.

Although very low frequency (VLF) can also be used for long range air navigation (e.g., navigation with the international Omega system), propagation, mainly by sky wave or the waveguide mode, depends on ionospheric conditions which vary with time of day and season. Though the cyclic redundancy of the transmitted signals cause a lane identification problem, there are adjustments which can be used to successfully overcome it.

Medium frequency (MF) and high frequency (HF) signals meet the requirement for the time measurement but, unfortunately, suffer high propagation losses over land and loss of propagation due to natural and man-made physical characteristics whose size is a significant fraction of a wave length [18].

Hence 100kHz low frequency was selected for Loran-C to take advantage of the stable propagation and long range of this frequency band [19].

C. Loran-C Time Difference.

The Loran-C chain contains a master station and two to four secondary stations. The transmitting stations of the chain transmit groups of pulses forming a Group Repetition Interval (GRI). Each station has its own GRI in the range of 49900 μs to 99900 μs. The master station transmits a nine-pulse group spaced 1000 μsec with 2000 μsec between the

ORIGINAL PAGE IS
OF POOR QUALITY

eighth and ninth pulses. Secondary stations transmit an eight-pulse group with pulses spaced 1000 μsec. The secondary stations transmit after a coding delay and a baseline delay that is specific for each secondary in the chain (figure 3-1).

Measuring the time differences (TDs) between arrival of pulse sets from different stations can be used to locate the receiver's position. The hyperbolic navigation system operates on the Loran-C TD readings because narrow bandwidths at low frequencies do not allow high enough data rates to transmit signals with a higher information content than the presence or absence of the pulse at a given time.

The TD value is found by measuring the difference in time of arrival of a set of pulses from two stations. A constant TD number traces a hyperbolic line (figure 3-2). The hyperbolic equation (3-2) is constructed as follows,

$$d_A = \sqrt{(x+c)^2 + y^2}$$

$$d_B = \sqrt{(x-c)^2 + y^2}$$

$$d_A - d_B = 2a \tag{3-1}$$

where d_A and d_B are the distances from each station to the receiver. Put d_A and d_B in (equation 3-1) and rearrange the equation

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \tag{3-2}$$

where $c = \sqrt{a^2 + b^2}$; point a is the intersection point between the hyperbola and the X-axis, and b is the conjugate axis length.

Each hyperbolic locus of points traces a line of position (LOP). Therefore, a second master secondary pair must be used to trace a second LOP, and the crossing point of two or more LOPs define the receiver's location (figure 3-3).

D. Computation of Time-Differences.

The master station transmits a signal in all directions, once per GRI, then secondary stations transmit, after the transmission time of the signal from the master station to the secondary station (baseline time) plus a coding delay. The delay avoids overlapping signals from secondary stations anywhere in the coverage region (figure 3-4).

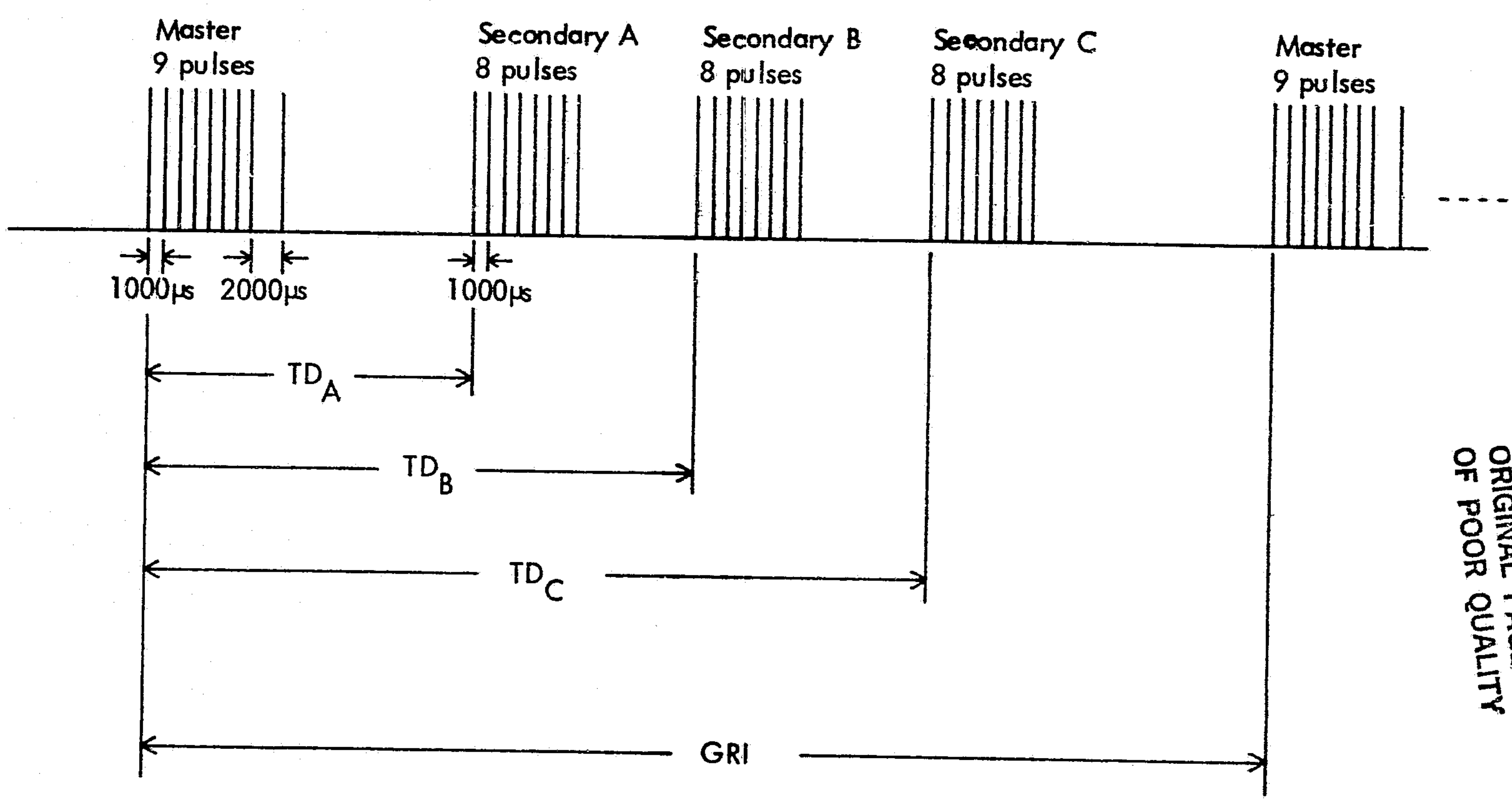


Figure 3-1. Loran-C Transmitted Signal Format. TD is a time difference between the master and each of the secondaries.

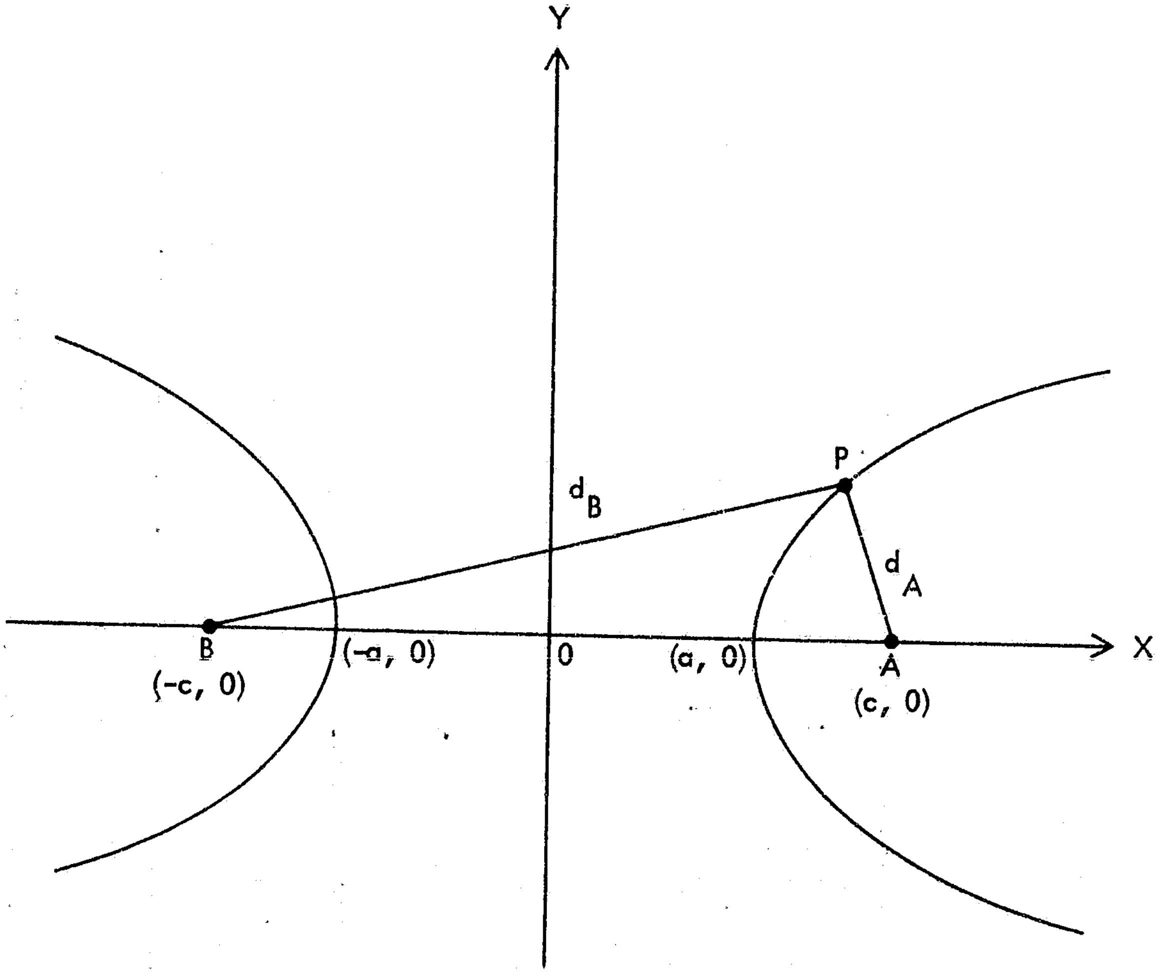


Figure 3-2. The TD Values Received to Point P From the Loran-C Stations.

Hyperbolic line between M and A ————

Hyperbolic line between M and B — — —

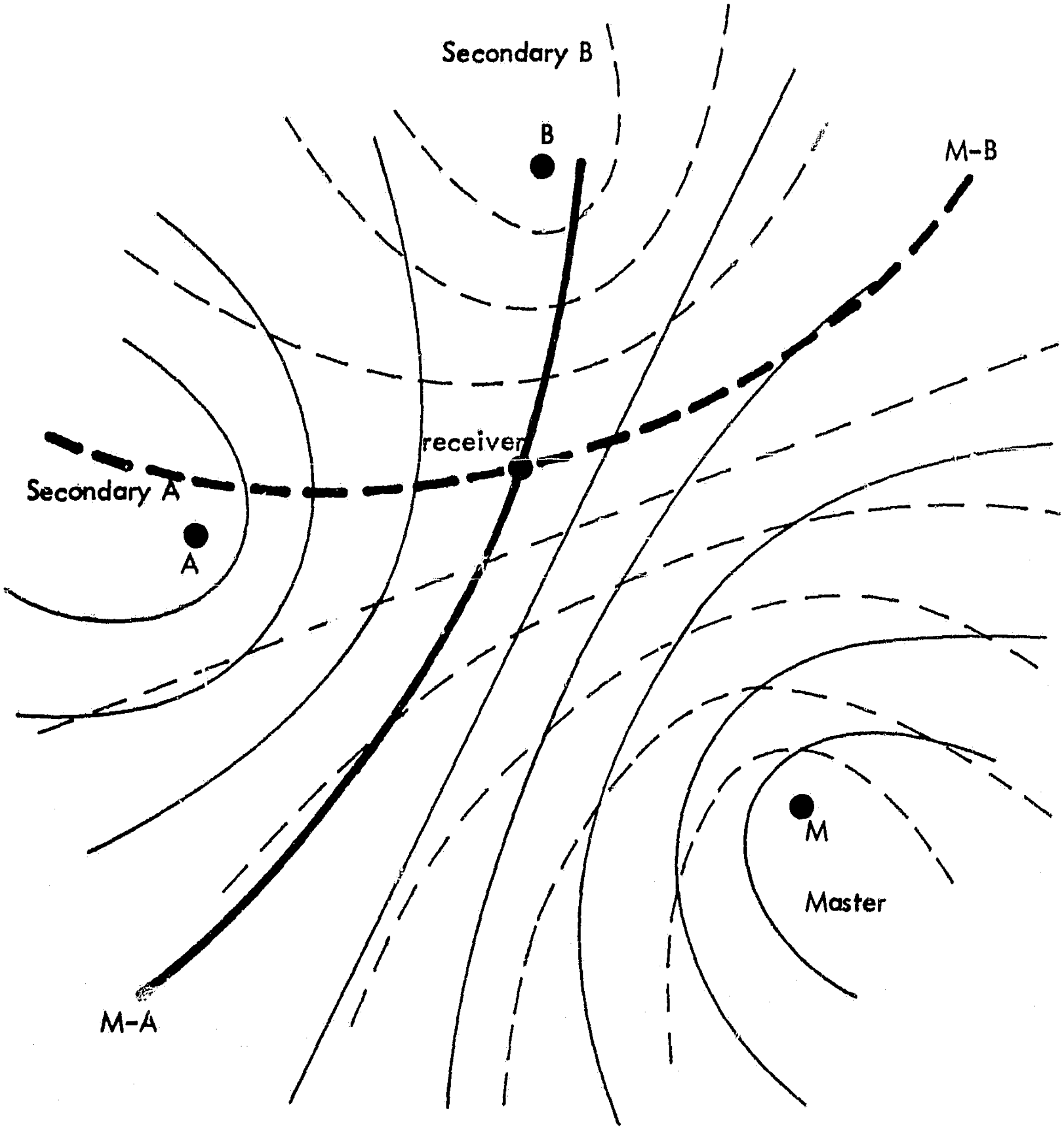


Figure 3-3. TDs Received from the Various Master-Secondary Pairs Define LOPs Which All Intersect at the Receiver's Position.

ORIGINAL PAGE IS
OF POOR QUALITY

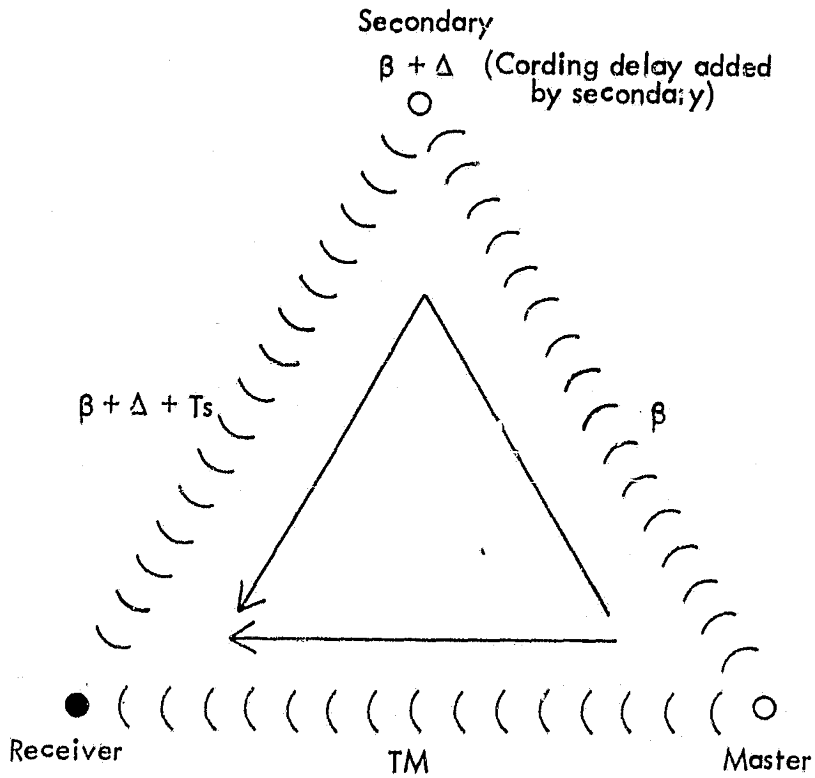


Figure 3-4. The TD Reading at the Receiver.

The receiver gets two signals from the master and the secondary, separated by a TD. This TD in mathematical form is:

$$TD = \quad + \Delta + T_s - T_M$$

where TD = the time difference in arrival of the master and secondary signals

β = the one-way baseline time between the master and secondary

Δ = the secondary coding delay inserted by the secondary

T_s = the one-way baseline time between the secondary station and the receiver

T_M = the one-way baseline time between the receiver and the master station

The baseline time β and the coding delay Δ are generally known quantities which are set up during the installation of the Loran-C chain. These two quantities are published in the data for each chain by the U. S. Coast Guard [20]. The two baseline times to the receiver, T_s and T_M, have unknown values and must be calculated in order to be applied to the TD equation.

There are two factors involved for the computation of the value of travel time from one point to another. In calculating the accurate baseline distance between the two points one should consider the non-spherical nature of the earth, and the corrections to the velocity of the signal which are required when it passes through the medium along the baseline. The velocity of propagation is mostly affected by the conductivity of the earth and the dielectric constant of the air [21].

The baseline length between two points on the earth will be provided by using the elliptical method (Chapter IV). This method uses an oblate spheroid model of the earth to approximate the geodesic. According to the testing (Chapter IV), this usually is adequate for Loran-C work.

If the Loran-C signal travels over the baseline path with a constant velocity, independent of the adjacent medium, the above method may be used to calculate time differences. As a matter of fact, the Loran-C signal travels over a surface which has inhomogeneous conductivity and dielectric constants, and also has irregular terrain [22].

An attempt to resolve the problem of predicting the signal phase delay is very difficult because of the nonspherical and the irregular nature of the surface impedance of the earth. An integral equation model of an inhomogeneous and irregular earth, was proposed by Samaddar [23], to predict the secondary phase delays. This calculation is complex and does not obtain a valuable phase factor correction. More general impedance models are typically applied by the Defense Mapping Agency.

E. Converting Time-Difference.

The time difference measurement, which was discussed in the previous chapter, does not directly provide generalized position coordinates such as latitude and longitude to the Loran-C users. Although the time differences contain position information, they should be converted into position coordinate systems.

The classic conversion method is through the use of charts, maps and tables. This method is adequate for low velocity craft like a ship, but not for high velocity aircraft. Therefore, the Loran-C receiver should have an automatic conversion from TD to the actual position coordinate system.

There is no simple relationship between the Loran-C lines of position (LOP) and a geocentric grid coordinate system (figure 3-5)[24]. However, there are many ways for the conversion, and basically, there are two main methods. One requires iterative calculations that lead to the final result, and the other uses a direct or closed-form solution to do the conversion.

The iterative TD-to-position method assumes the receiver's position first, then compares the position determined by the received TDs and the assumed receiver position. Before the comparison, the TDs for the assumed position are calculated and then the assumed position is regulated to minimize the error in the TDs comparison. This mechanism prepares two tables. A table of TD values is generated encircling the covered region of the three sets of Loran-C stations and the other table is generated to the corresponding positions for the TD table. After receiving a pair of Loran-C TDs, a linear interpolation process is applied between the two tables to find the position. This is repeated until the difference between the two sets of TDs becomes small for position estimation [25].

There are other ways which relate received TD values to assumed TD values. Those ways calculate TD errors from a comparison between measured TDs and assumed TDs, and move the assumed position according to TD errors until errors become acceptable.

The non-iterative, or closed form solution, method provides the actual position of the receiver by setting the received TDs as parameters in spherical equations. Then the unknown values which represent the receiver's position are calculated by solving the equations. It is generally complicated to find the relationships between position on a Loran-C hyperbolic grid (geocentric grid) and spherical angles. Besides, the non-spherical nature of the earth and the non-constant propagation attributes of the Loran-C signals, as discussed in Chapter III.C, require corrections depending on local conditions. For a non-iterative solution, a general model which covers a large area is needed to make corrections. However, the non-iterative solution can be simple if exact solution is not necessary.

For the Ohio University Loran-C receiver, a non-iterative, explicit solution presented by Razin [26] was adopted for TD-to-position conversion. The TD-to-position conversion software for the microcomputer was developed by Fischer [27].

F. Area Navigation.

Since the Loran-C system provides position information at any point inside the wide-coverage area, this position information can be used for Area(or Random) Navigation(RNAV) which provides range, bearing angle, ground speed, cross track error and estimated time of arrival based upon arbitrarily-positioned waypoints. It is very convenient for a pilot to have this information in order to fly on a desired course to a selected waypoint. Area navigation with charts and maps is possible, but inconvenient for aviation use because the pilot needs updated information every second during the flight.

The Loran-C system has automatic area navigation capability. The VOR/DME-based RNAV system also provides this information, but this system needs a VOR receiver, DME plus RNAV computer. The Loran-C receiver developed at the Ohio University Avionics Engineering Center computes all navigational information with one microprocessor. The portion of this computation for area navigation is discussed in the next chapter.

IV. COMPUTATION FOR AREA NAVIGATION

In order to provide sufficient area navigation information for a pilot, proper computation should be made by the software of a microcomputer-based Loran-C receiver.

It is very important to include the computation of range and bearing angle in the area navigation software because calculations of other RNAV information requires range and bearing angle factors. Range and bearing angle errors, especially, are very critical for a ground speed calculation because the ground speed calculation deals with small range and bearing differences. This will be discussed in more detail later on in the chapter.

A. Range and Bearing Angle.

The computation of range and bearing angle between two points on earth is not simple because the shape of the earth is an irregular ellipsoid, as was mentioned in the previous chapter. It is not necessary to perform exact calculations, but there is a certain accuracy which is mandated for practical area navigation. On the other hand, the capacity of memory and execution time for the microcomputer must be adequate for the microprocessor-based Loran-C receiver.

In the previous work by Fischer, his software provides range bearing angle using a simplified elliptical model after the TD-to-position conversion. Three mathematical models including the simplified elliptical model are compared for range/bearing angle calculations to determine which model is suitable for RNAV calculations.

1. Spherical model. If the earth is considered a perfect sphere, the spherical model is the proper model for the calculation of range/bearing angle between two points on the earth. The great-circle calculations are used for this model.

Referring to figure 4-1, R and W are two points on the earth's surface; R is the receiver point and W is the waypoint. The angles X and Y at R and W of the great circle passing through the two places and the distance D between R and W along the great circle can be calculated as follows [28]:

From Napier's Anglogier:

$$\tan\tfrac{1}{2}(Y-X) = \frac{\sin\tfrac{1}{2}(\phi_W-\phi_R)}{\tan\tfrac{1}{2}\Delta\lambda\ \cos\tfrac{1}{2}(\phi_W+\phi_R)} \qquad (4\text{-}1)$$

and

$$\tan\tfrac{1}{2}(Y+X) = \frac{\cos\tfrac{1}{2}(\phi_W-\phi_R)}{\tan\tfrac{1}{2}\Delta\lambda\ \sin\tfrac{1}{2}(\phi_W+\phi_R)} \qquad (4\text{-}2)$$

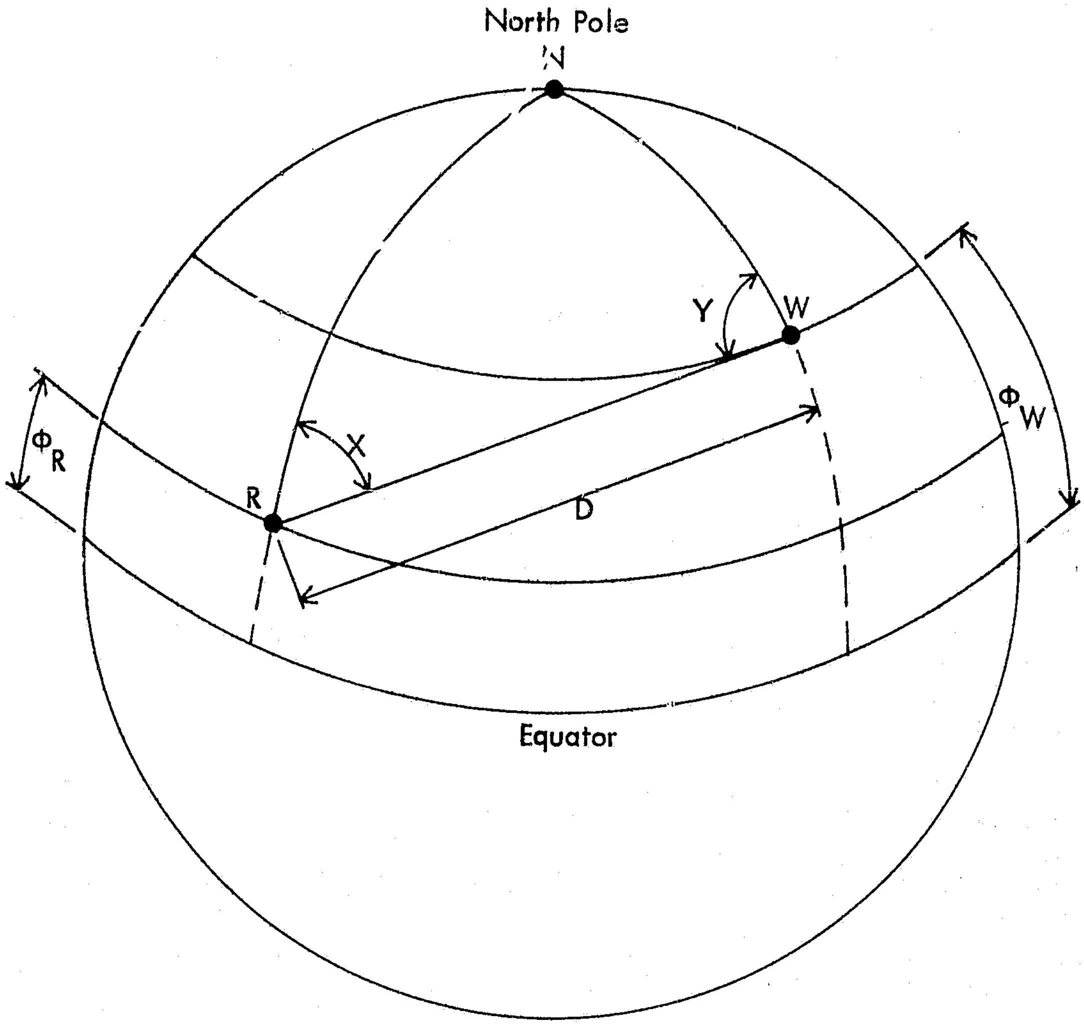


Figure 4-1. Spherical Model.

where ϕ_R = the latitude of the receiver.
ϕ_W = the latitude of the waypoint.
$\Delta\lambda$ = the difference of longitude between the receiver and the waypoint.

The bearing angle X is found using (4-1) and (4-2),

$$X = \tfrac{1}{2}(Y+X) - \tfrac{1}{2}(Y-X)$$

$$= \tan^{-1} \frac{\cos\frac{1}{2}(\phi_W-\phi_R)}{\tan\frac{1}{2}\Delta\lambda \ \sin\frac{1}{2}(\phi_W+\phi_R)} - \tan^{-1} \frac{\sin\frac{1}{2}(\phi_W-\phi_R)}{\tan\frac{1}{2}\Delta\lambda \ \cos\frac{1}{2}(\phi_W+\phi_R)}$$

The distance D (in nautical miles) along the great circle between R and W is given as follows:

$$\tan\tfrac{1}{2}d = \frac{\tan\frac{1}{2}(\phi_W-\phi_R)\sin\frac{1}{2}(Y+X)}{\sin\frac{1}{2}(Y-X)}$$

$$D = d \times 60.0 \text{ (in nautical miles)}$$

2. Simplified Elliptical Model. The simplified elliptical model which was applied in Fischer's software might be one of the compromise solutions for the range/bearing angle computations. This model uses mid-latitude approximation. Suppose the earth is approximated by an ellipsoid with major (equatorial) radius, a=3443.9174 nm, and minor (polar) radius, b=3432.3707 nm. The radius of curvature of the earth, R, may be computed for the mid-region of the coverage for the particular Loran-C stations. Referring to figure 4-2 [29], ϕ_R and ϕ_W are same as the previous model, and λ_R and λ_W are the longitude of the receiver and the waypoint [30]. The bearing angle to the waypoint can be calculated as follows:

$$B = \tan^{-1}\left[\frac{(\lambda_R-\lambda_W)\cos(\frac{\phi_R+\phi_W}{2})}{\phi_W-\phi_R}\right]$$

and the distance to the waypoint is:

$$D = R\sqrt{(\lambda_R-\lambda_W)^2\cos^2(\frac{\phi_R+\phi_W}{2}) + (\phi_W-\phi_R)^2}$$

R is computed with mid-latitude approximation and is stored as a constant number. R is calculated after choosing the midpoint of the coverage region:

$$R = \frac{a^2\sin^2\phi_M + b^2\cos^2\phi_M}{b}$$

where ϕ_M is the latitude of the midpoint.

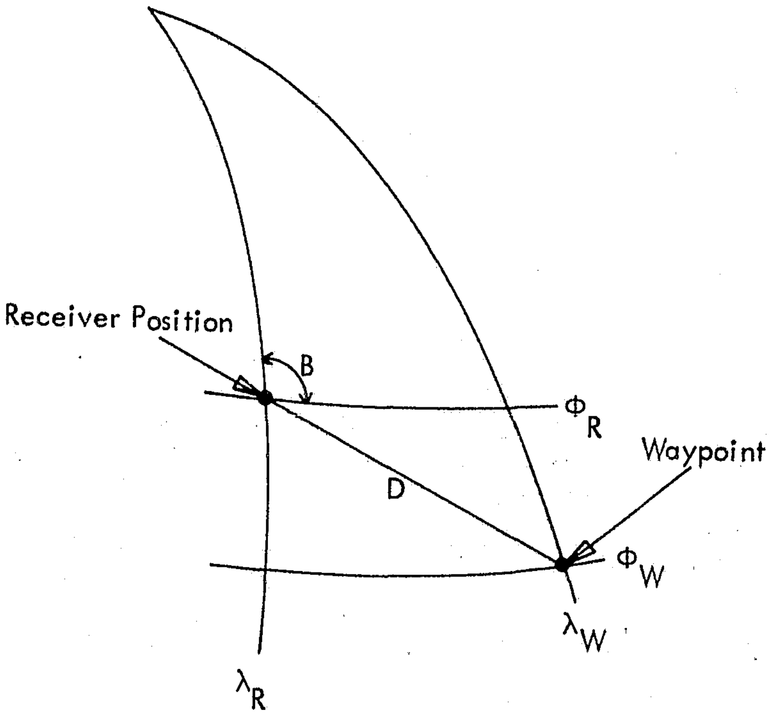


Figure 4-2. Simplified Elliptical Model.

3. Elliptical Model. The elliptical model can be expected to be the model which provides more accurate data. The calculations of this model are more complicated than other models.

This elliptical model projects the point on the ellipsoid onto the sphere circumscribing the elliptical earth model, because the coordinate system uses the sherical earth model. After points are projected onto the sphere whose radius is the earth's major equatorial radius, a, oblique triangle equations are used. For the range computation, the difference between an arc on the sphere and an arc on the ellipsoid is considered [31]. Figure 4-3 shows that the point on the sphere has a parametric latitude by projecting the latitude of a point on the earth onto the sphere. ϕ_R, ϕ_W and $\Delta\lambda$ are defined as before. The parametric latitude is defined as:

$$\tan \beta_R = (1-f)\tan\phi_R$$

$$\tan \beta_W = (1-f)\tan\phi_W$$

where $f = (a-b)/b = 0.00335278$: the flattening of the ellipsoid.

The generalized direction cosines of the projected point are:

$$C1 = \cos\beta_W \sin(\Delta\lambda)$$

$$C2 = \cos\beta_R \sin\beta_W - \sin\beta_R \cos\beta_W \cos(\Delta\lambda)$$

$$C3 = \sin\beta_R \sin\beta_W + \cos\beta_R \cos\beta_W \cos(\Delta\lambda)$$

The bearing angle at the receiver of the geodesic arc from receiver to waypoint, measured from north, is (figure 4-4):

$$\tan\psi = \frac{C1}{C2}$$

The approximate angle from receiver to waypoint, in a plane through the center of the ellipsoid is:

$$\tan\theta = \frac{C2\cos\psi + C1\sin\psi}{C3}$$

The geodesic arc length between the receiver and the waypoint can be calculated as follows:

$$D = a[\theta - \frac{f}{4}(mu + nv)]$$

where $m = (\sin\beta_R + \sin\beta_W)^2$

$$n = \left(\frac{\sin\beta_R - \sin\beta_W}{\sin\theta} \right)^2$$

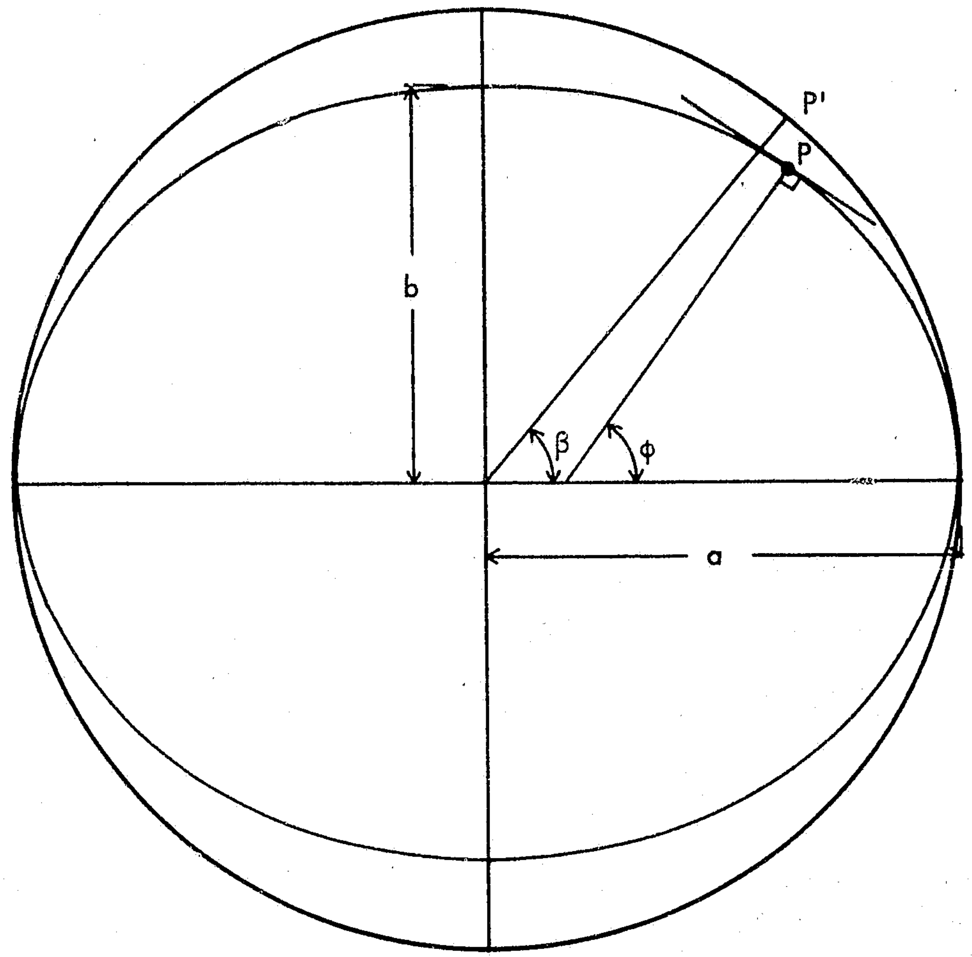


Figure 4-3. Circumscribing Sphere Around the Elliptical Earth Model.

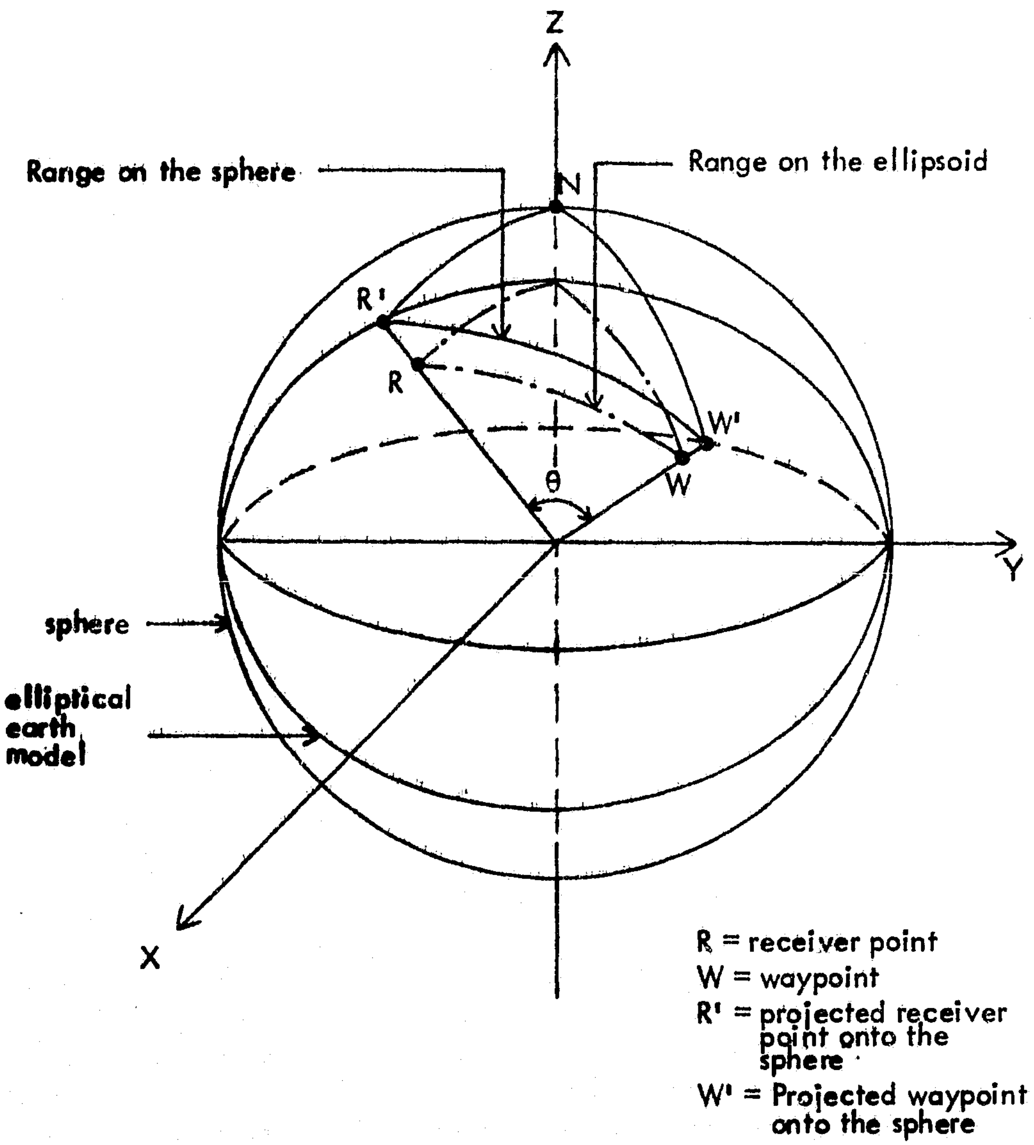


Figure 4-4. Elliptical Model.

$$u = \left(\frac{1 - \cos\theta}{\sin\theta} \right)\left(\frac{\theta - \sin\theta}{\sin\theta} \right)$$

$$v = (1 + \cos\theta)(\theta + \sin\theta)$$

4. Comparison Among Three Models. There are three factors concerning the comparison, such as accuracy, computation time and memory capacity.

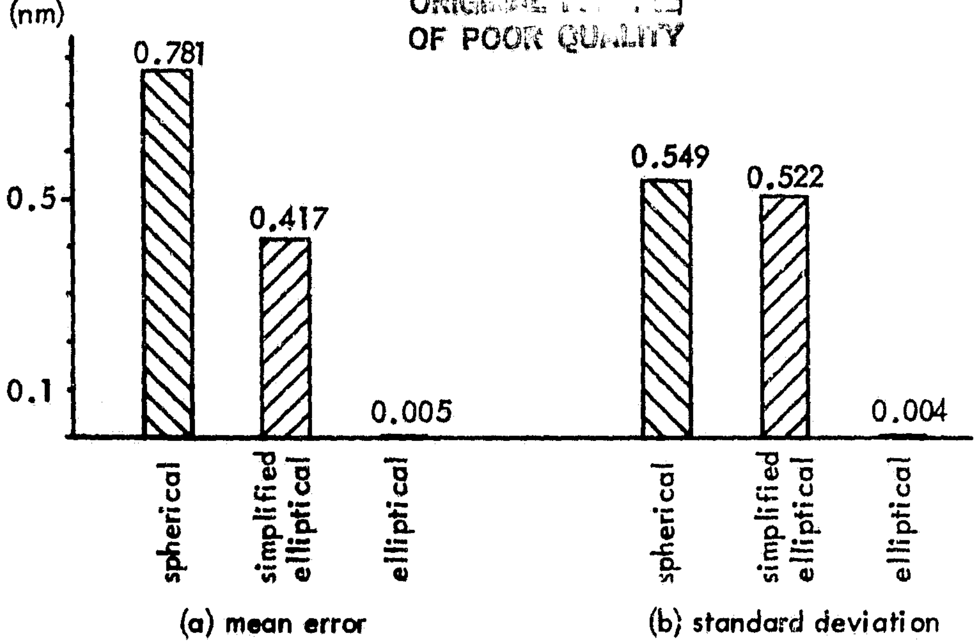
The accuracies of the three models are shown in figure 4-5 (and numerically in table 4-1). The data (coordinates, range/bearing) of numbers 1 to 5 were taken from reference [32], and the data of numbers 6 to 9 were taken from reference [33]. These data are known to have high accuracy (much less than 0.0005nm error). The results were computed using a Fortran-IV program (appendix B) on an IBM4341. As the figure shows, the accuracy of the elliptical model is much better than two other models. The errors from the model are less than 0.013nm (about 79 feet), even the distance between the receiver and the waypoint becomes greater than 500nm. The simplified elliptical model and the spherical model have enough accuracies for an area navigation application. Comparing the two models, the simplified model shows better accuracy on range computation, and the spherical model shows better accuracy on bearing computation.

Execution time depends on numbers of mathematical operations, especially since trigonometric functions consume much computer time. Table 4-2 shows the numbers of mathematical operations in each model. The simplified elliptical model needs the least execution time among the three, and the elliptical model needs much more execution time than the other two. An amount of memory space is directly proportional to execution time in this case.

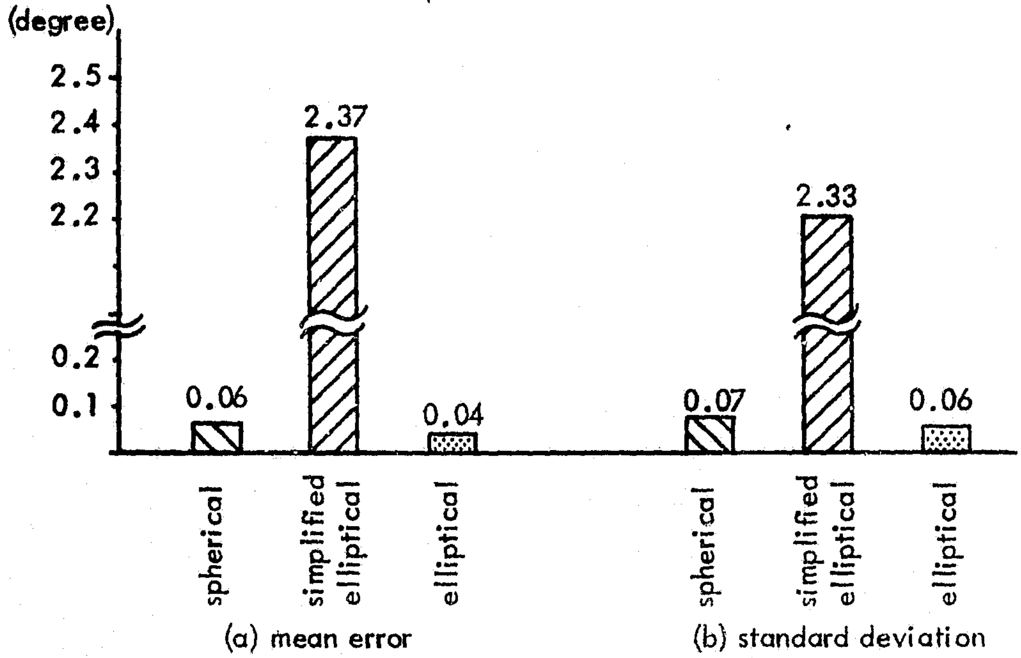
Choosing an optimum model among them depends mostly on the type of receiver. The elliptical model is applied to the Ohio University receiver because the execution time and the amount of memory space can accept this model and the accuracy is great with respect to the ground speed calculation.

B. Other Navigational Information.

1. Cross-Track Error. It is very important for a pilot to know whether the aircraft is on course or not. If it is off course, by how much? Cross-track error (CTE) indicates the position error measured on the perpendicular from the desired track to the actual position of the aircraft, and cross-track error bearing (CTEB) indicates the angular difference between the desired track and actual track (figure 4-6). Since the error of the spherical model is small for short distances, the spherical trigonometry is adequate for the CTE calculation which provides small distances (generally less than 10nm). CTE can be calculated



A. Range (nm)



B. Bearing Angle (degree)

Figure 4-5. Accuracy Comparison among Three Models.

ORIGINAL PAGE IS
OF POOR QUALITY

No.	Receiver Point N.LAT/W.LONG	To Waypoint N.LAT/W.LONG	Range / Bearing (nm/degree)			
			Range/Bear (nm/degree)	Spherical Model	Simplified Elliptical Model	Elliptical Model
1	40 30 37.8 17 19 43.3	40 00 0.0 18 00 0.0	43.448 225.43	43.394 225.32	43.474 225.10	43.448 225.38
2	9 59 48.3 16 31 55.9	10 00 0.0 18 00 0.0	86.897 270.26	86.731 270.26	86.610 270.13	86.896 270.26
3	69 48 5.7 9 37 28.6	70 00 0.0 18 00 0.0	173.794 277.87	172.964 277.87	173.731 273.94	173.787 277.86
4	13 04 12.6 14 51 13.3	10 00 0.0 18 00 0.0	260.690 225.63	261.028 225.44	260.709 225.12	260.696 225.54
5	73 35 9.2 3 26 35.1	70 00 0.0 18 00 0.0	347.588 238.84	345.989 238.82	349.081 231.75	347.574 238.83
6	42 42 50.6 76 49 33.9	41 15 11.9 69 58 39.1	318.621 103.57	317.656 103.70	318.232 106.01	318.618 103.67
7	42 42 50.6 76 49 33.9	46 48 27.2 67 55 37.7	452.416 54.05	451.301 53.98	452.528 57.06	452.408 54.03
8	42 42 50.6 76 49 33.9	39 51 7.5 87 29 12.1	511.412 253.91	509.972 253.92	511.110 250.34	511.410 253.05
9	42 42 50.6 76 49 33.9	34 03 46.0 77 54 46.8	521.045 185.92	521.573 185.97	522.125 185.02	521.051 185.98

Table 4-1. Numerical Comparison Among Three Models (Fortran Simulation).

No.	Receiver Point N.LAT/W.LONG	To Waypoint N.LAT/W.LONG	Range/Bear (nm/degree)	Error in Range/Bearing (nm/degree)		
				Spherical Model	Simplified Elliptical Model	Elliptical Model
1	40 30 37.8 17 19 43.3	40 00 0.0 18 00 0.0	43.448 225.43	0.054 0.11	0.026 0.33	0.0 0.05
2	9 59 48.3 16 31 55.9	10 00 0.0 18 00 0.0	86.897 270.26	0.166 0.0	0.287 0.13	0.001 0.0
3	69 48 5.7 9 37 28.6	70 00 0.0 18 00 0.0	173.794 277.87	0.830 0.0	0.063 3.93	0.007 0.0
4	13 04 12.6 14 51 13.3	10 00 0.0 18 00 0.0	260.690 225.63	0.338 0.19	0.019 0.51	0.006 0.10
5	73 35 9.2 3 26 35.1	70 00 0.0 18 00 0.0	347.588 238.84	1.597 0.02	1.493 7.09	0.014 0.01
6	42 42 50.6 76 49 33.9	41 15 11.9 69 58 39.1	318.621 103.57	0.965 0.13	0.389 2.44	0.003 0.1
7	42 42 50.6 76 49 33.9	46 48 27.2 67 55 37.7	452.416 54.05	1.115 0.06	0.112 3.01	0.008 0.02
8	42 42 50.6 76 49 33.9	39 51 7.5 87 29 12.1	511.412 253.91	1.440 0.01	0.302 3.57	0.002 0.04
9	42 42 50.6 76 49 33.9	34 03 46.0 77 54 46.8	521.045 185.92	0.528 0.05	1.080 0.30	0.006 0.06

Table 4-1. Continued.

Operation \ Type of Model	Spherical	Simplified Elliptical	Elliptical
Addition and Subtraction	4	4	12
Multiplication, Division and Square Root	11	7	18
Trigonometric Function	10	2	16

Table 4-2. Number of Mathematical Operations in each Model.

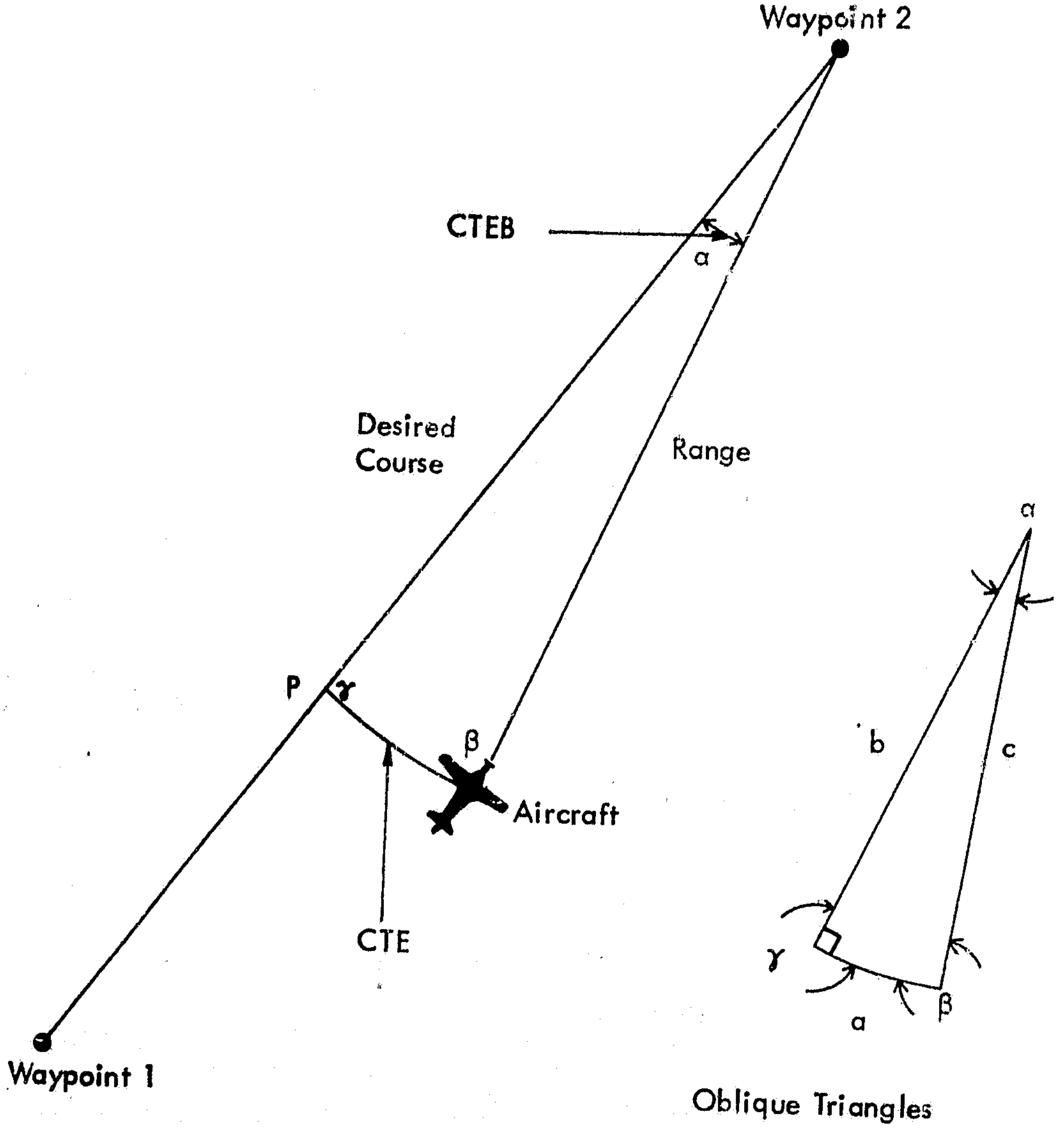


Figure 4-6. Cross-Track Error (CTE).

as follows:

Since waypoint 2, the aircraft position and point P form a right spherical triangle,

$$\sin a = \sin c \ \sin \alpha$$

where $a = \frac{CTE}{r}$, $c = \frac{Range}{r}$, $\alpha = CTEB$

where r is a midpoint radius between airplane and waypoint.

$$r = \sqrt{a^2\cos^2\beta_1 + b^2\sin^2\beta_1}$$

$$\phi_1 = \frac{\phi_1 + \phi_2}{2}$$

$$\beta_1 = \tan^{-1} [(1-f)\tan\phi_1]$$

ϕ_1 and ϕ_2 are the latitude of waypoints, 1 and 2 respectively. r may be set as a constant number such as $r= \sqrt{a^2 + b^2} = 3438.1489$ for the computer use because of the small-angle approximation.

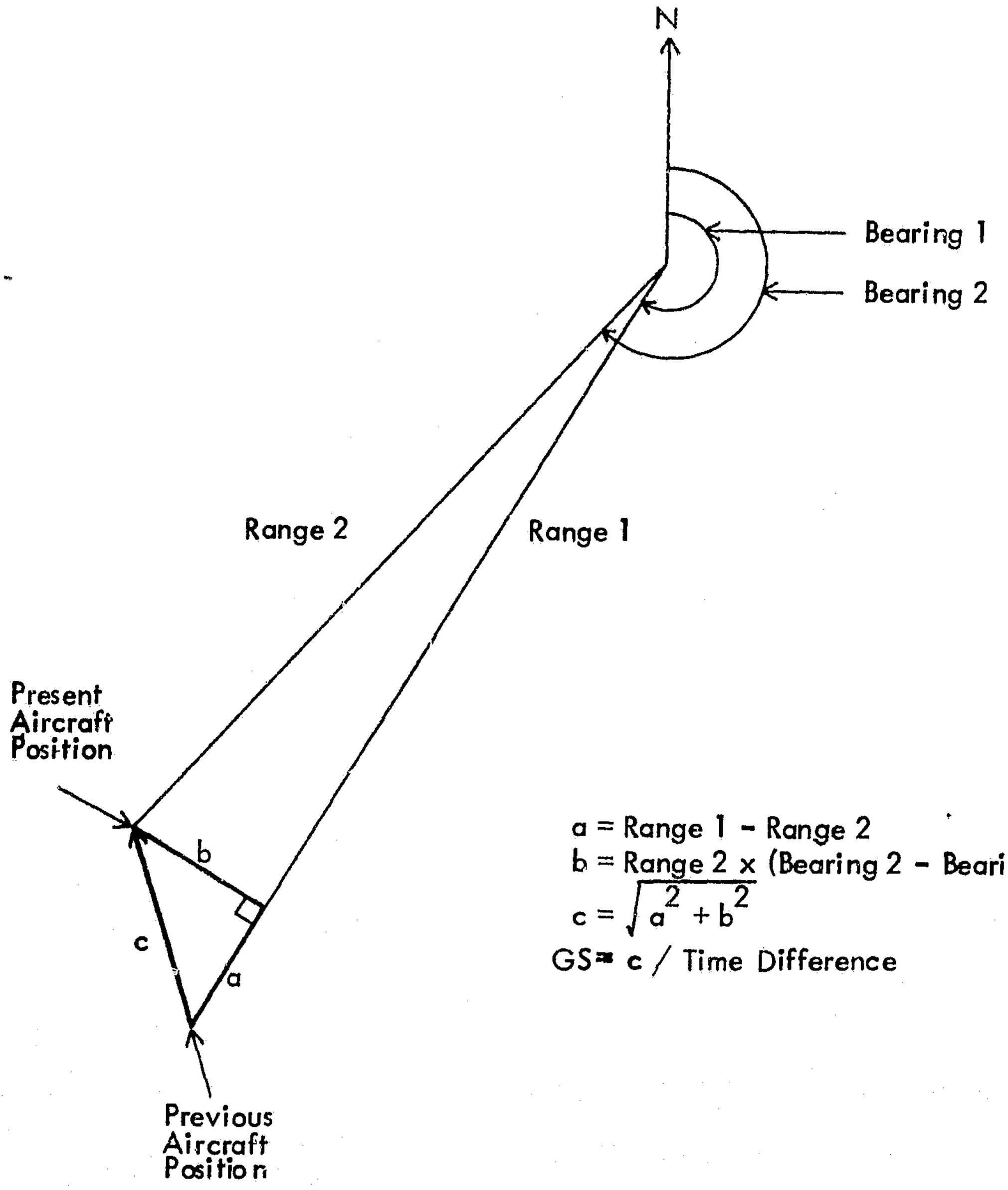
Cross-track error includes airborne equipment, ground equipment, and flight technical error (FTE).

2. Ground Speed and Estimated Time of Arrival. Theoretically, the calculation for ground speed (GS) between two points is simple. Figure 4-7 is the diagram used to determine the ground speed calculation.

$$GS = \sqrt{\frac{(\text{Range Difference})^2+(\text{Bearing Difference})^2\text{Range}^2}{\text{Cycle Time}}} \qquad (4-3)$$

However, as an application for Loran-C area navigation, the computation cycle is about one and one-half second, and range difference is less than 0.1nm, so that a small range error causes a large ground speed error. For example, if the range error is 0.1nm per 1.5 second then the ground speed error becomes 240nm/hour. Since time differences (TDs) have random noise errors, an accurate ground speed cannot be obtained by simply using equation (4-3). To solve this problem two processes are made: process 1 calculates the GS using the present point and a point which occurred sixteen cycles previously, then uses a recursive filter (α-β filter) on the calculated GS. Process 2 uses the same filter on TD values and calculates GS between the present point and a point which occurred four cycles previously. The flowcharts of the two processes are shown in figure 4-8.

Process 1 computes GS after sixteen data points are collected in a memory table, and replaces the old data of the 16 cycles by the present



a = Range 1 - Range 2
b = Range 2 x (Bearing 2 - Bearing 1)
$c = \sqrt{a^2 + b^2}$
GS = c / Time Difference

Figure 4-7. Ground Speed (GS).

ORIGINAL PAGE IS
OF POOR QUALITY

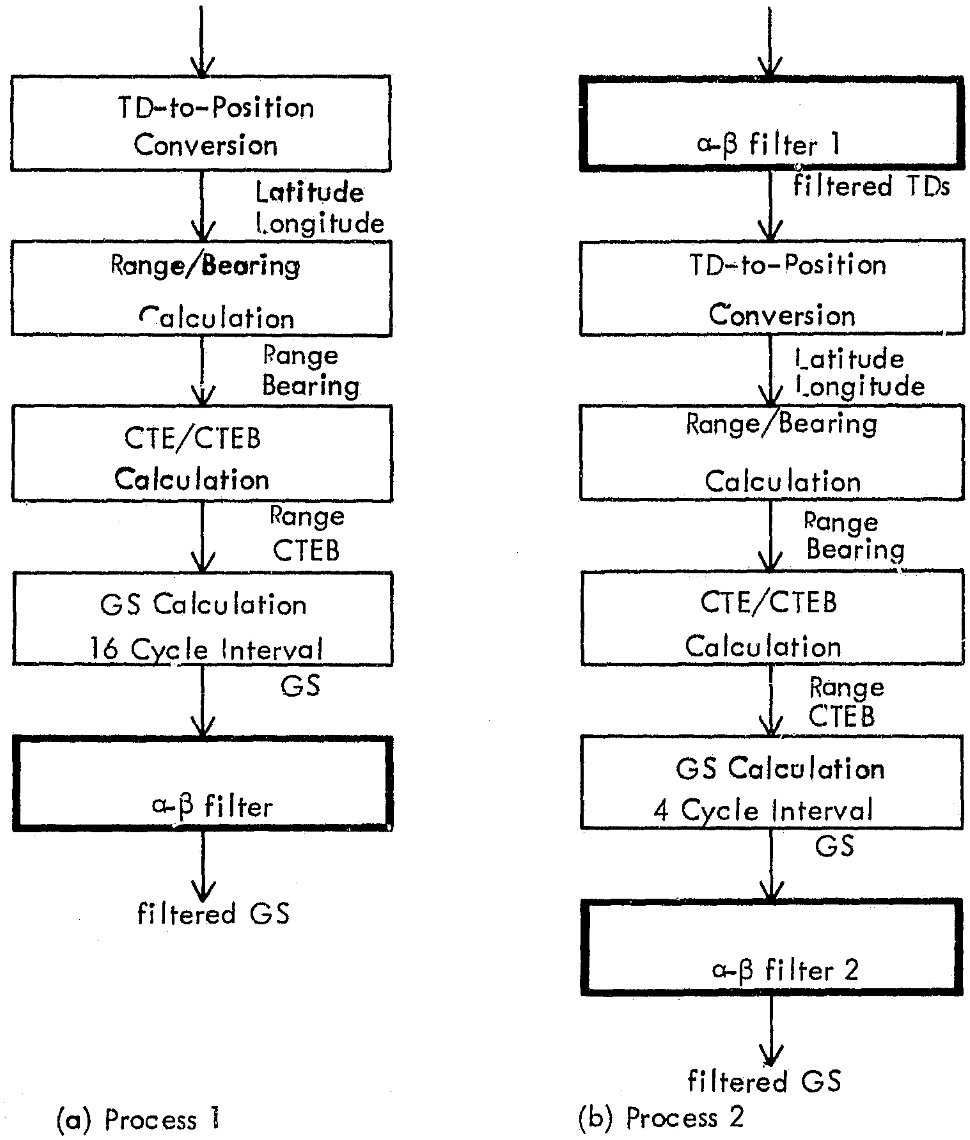


Figure 4-8. Process-1 (One α-β filter) and Process-2 (two α-β filters) for Ground Speed Calculation.

data. Computing the GS between the present point and the oldest 16 cycles just minimizes the random noise errors. More cycles might reduce random noise errors; however, they consume more memory space and cause large errors on turns because the distance between a straight line and a curved line becomes larger. In order to eliminate random errors, the recursive filter loop [34] is added after the GS computation. This filter is a valuable aid in data smoothing and prediction, and is easily implemented on a microprocessor system.

Initial condition(n=0) is:

$$ACP(0) = 0$$
$$GSP(1) = GSO(0)$$

Inside the loop(n≥1):

$$GSS(n) = GSP(n) + \alpha[GSO(n) - GSP(n)]$$
$$ACS(n) = ACP(n) + \beta[GSO(n) - GSP(n)]/T$$
$$GS(n) = GSS(n) + T\ ACS(n)$$

For the next calculation:

$$GSP(n+1) = GS(n)$$
$$ACP(n+1) = ACS(n)$$

where GSP is predicted ground speed
GSO is ground speed observed by Loran-C receiver
GSS is smoothed ground speed
ACP is predicted acceleration
ACS is smoothed acceleration
α is first-order gain
β is second-order gain
T is period of loop

To determine values of α and β, a damping ratio ρ and an effective time T_f should be considered.
A damping ratio ρ;

$$\rho^2 = \frac{\alpha^2}{4\beta}$$

For a compromise between rise time, overshoot, and ringing of the transient response unity damping $\rho=1$ was chosen.

$$T_f = \frac{0.72T}{\alpha}$$

An effective time decides the response time. The step response becomes within e^{-2}(13.5%) of the final value at $t=2T_f$. Random errors can be eliminated when the effective time becomes greater than 30 seconds ($\alpha=0.033$, $\beta=0.00028$) for the Ohio University Loran-C use. This long effective time and each 16-cycle old data which is used for the GS calculation makes the GS response too slow for aircraft speed.

Process 2 was made to speed up the slow response. This process uses the same filter twice on TD values and GS. Although TD values have a hyperbolic geometry, a small TD change (less than 1 microsecond per cycle) can be considered a linear change. The purpose of filtering on TDs minimizes random errors on position information and avoids a long response time of the filter on calculated GS. Also, four cycles are used instead of sixteen cycles for the GS calculation to reduce errors on turns. The effective time of the filter on TDs should be short enough to provide adequate response for position information. The T_f=6sec.(α=0.167, β=0.007) on TD values and T_f=12sec.(α=0.08, β=0.0017) on GS are chosen for Ohio University Loran-C use. Process 2 also minimizes memory space. Although the response is still slow for accelerated flight, it is adequate for most flight conditions.

Estimated time of arrival (ETA) can be provided after the calculations of range and ground speed. The equation is as follows:

$$ETA = \frac{Range}{GS}$$

C. A Scheme for Microcomputer Use.

The area navigation (RNAV) program can be executed after TDs and coordinates of position are computed. The program which measures two TDs with Loran-C phase-locked loop operation and the other program which does TD-to-geographic conversion were already available for the Ohio University Loran-C [35,36]. For the ground speed computation, an α-β filter was added to the TD-to-geographic conversion program. The flow chart of these programs is shown in figure 4-9. These three programs form a loop to provide navigation outputs every short period of time GRIx14, (about 1.4 second when GRI=9960μs). The TD measurement program is the main program and jumps to subprogram 1 (the TD-to-geographic program) after two TDs are measured. The TD-to-geographic program calculates the coordinates of the position by reading two TDs and continuing to subprogram 2 (RNAV program). The RNAV program gets input data from the TD-togeographic program (coordinates of the position) and other input data from the user (coordinates of the waypoints) and calculates RNAV informations.

Since the RNAV provides a cross-track error, the desired course is calculated first, and then the GS calculation starts four cycles after the cross-track error calculation, because the GS calculation has a four-cycle interval. More detail will be shown in the next chapter.

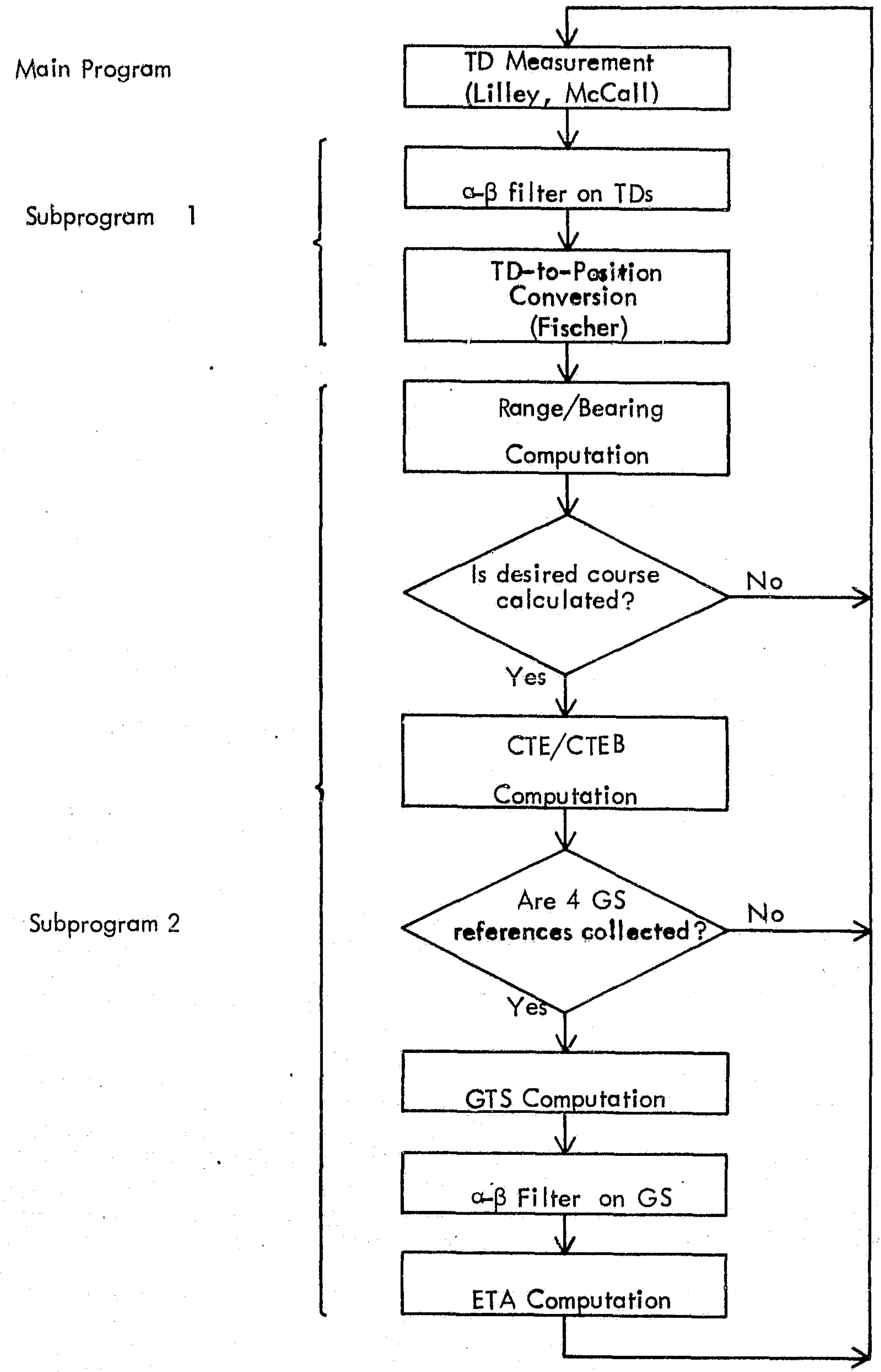


Figure 4-9. Flowchart of Navigation Program for Ohio University Loran-C Receiver

V. THE MICROCOMPUTER SYSTEM

A. System Design.

The navigation programs are software additions for the microcomputer-based Loran-C receiver. The total system design is the main constraint for the design of the software.

1. Hardware. The configuration in figure 5-1 shows that the receiver utilizes a whip antenna, a wide-band preamplifier/coupler, an AGC processor, an RF frond-end processor and tracking loop hardware/software to receive Loran-C signals and to measure TDs [37]. The MOS Technology 6502 Super-Jolt does all of the microprocessor work. The software is written in 6502 assembler language (figure 5-2 [38]). A pilot enters control functions and input data to the receiver from the panel-mounted keyboard or the hand-held ASCII terminal. All navigational information is displayed on the CRT screen for the pilot, and can also be recorded on a digital cassette unit for data reduction purposes.

The TD-to-position and RNAV calculations require a numerical range of approximately 10^{-7} to 10^{6}, therefore, it is necesary to use a floating-point format. Besides, these calculations involve multi-precision addition, subtraction, multiplication, division and trigonometric functions. The 6502 microprocessor has only an 8-bit data bus, so that the processor needs a large amount of memory and rapid access. For many applications of the microprocessor, including the Loran-C application, the access time and memory amount are limited, therefore, it is desired to use an external device to support the microprocessor for these calculations.

The Am9511A by Advanced Micro Devices is a peripheral mathematics processor which does the necessary floating-point calculations. It is designed to be used in microprocessor systems which have an eight-bit data bus. It can handle 16-bit and 32-bit fixed point arithmetic, 32-bit floating-point arithmetic and trigonometric functions using a stack-oriented operand storage (sixteen 8-bit words). Hence, this device can provide useful support for the microprocessor's calculations. A listing of the instruction set for the Am9511A is shown in figure 5-3 [39].

To interface the Am9511A to the Jolt microcomputer system, it is necessary to use additional hardware in order to handle device selection and data transfer. An M6820 peripheral interface adapter(PIA) is used as the additional hardware for the hand-shaking between the Jolt and the 9511. The M6820 consists of two eight-bit ports and several other registers used to interface to peripheral devices. The overall design of the microcomputer system is shown in figure 5-4.

2. Interfacing Software. Particular software is required in order to initialize the hardware interface and the 9511, to write a single floating-point (32-bit) number, to read a floating-point (32-bit) number,

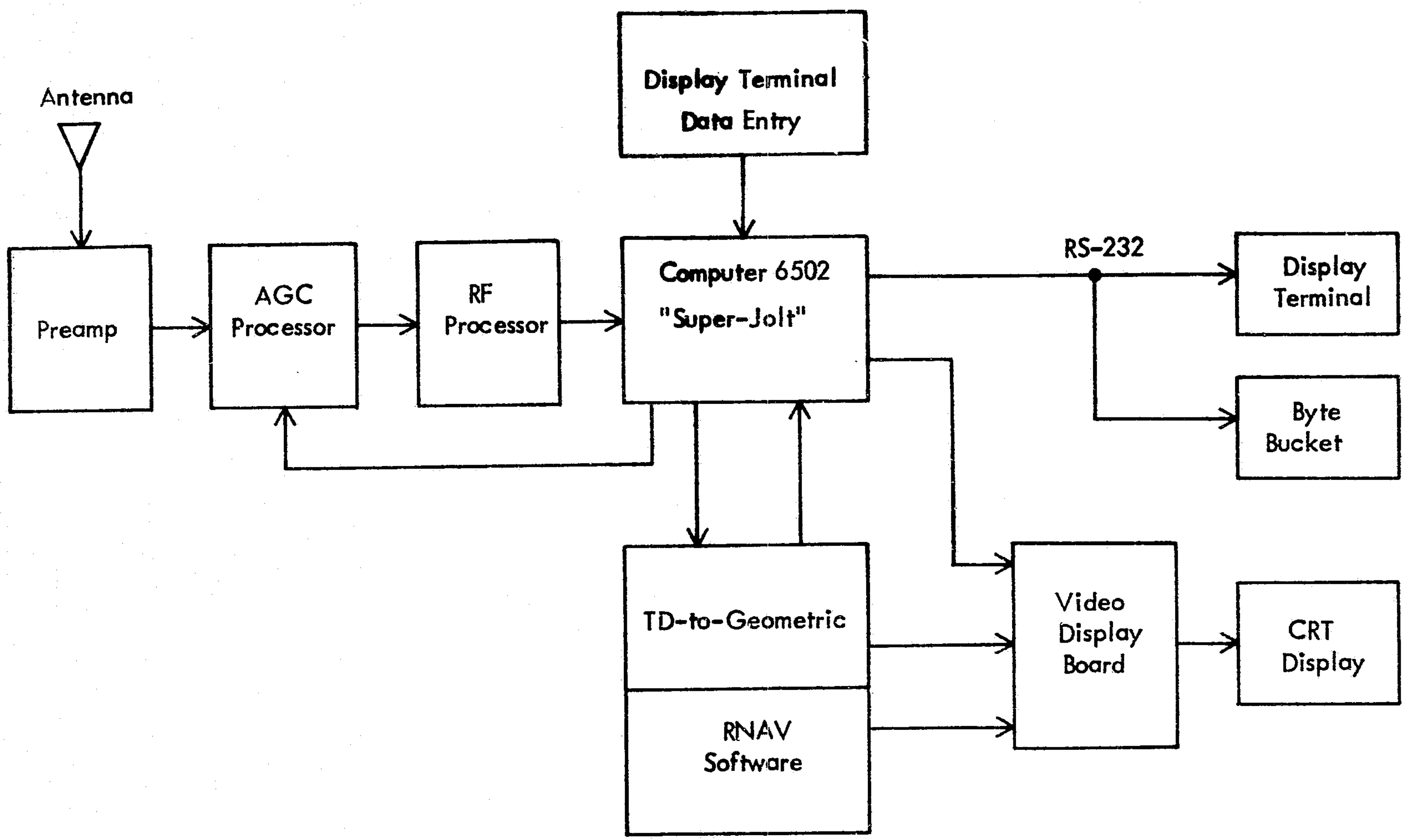


Figure 5-1. Block Diagram of Total System Ohio University Loran-C Receiver.

ORIGINAL PAGE IS
OF POOR QUALITY

ADC	Add with carry	JSR	Jump to subroutine
AND	Logical AND	LDA	Load accumulator
ASL	Arithmetic shift left	LDX	Load X
BCC	Branch if carry clear	LDY	Load Y
BCS	Branch if carry set	LSR	Logical shift right
BEQ	Branch if result = 0	NOP	No operation
BIT	Test bit	ORA	Logical OR
BMI	Branch if minus	PHA	Push A
BNE	Branch if not equal to 0	PHP	Push P status
BPL	Branch if plus	PLA	Pull A
BRK	Break	PLP	Pull P status
BVC	Branch if overflow clear	ROL	Rotate left
BVS	Branch if overflow set	ROR	Rotate right
CLC	Clear carry	RTI	Return from interrupt
CLD	Clear decimal flag	RTS	Return from subroutine
CLI	Clear interrupt disable	SBC	Subtract with carry
CLV	Clear overflow	SEC	Set carry
CMP	Compare to accumulator	SED	Set decimal
CPX	Compare to X	SEI	Set interrupt disable
CPY	Compare to Y	STA	Store accumulator
DEC	Decrement memory	STX	Store X
DEX	Decrement X	STY	Store Y
DEY	Decrement Y	TAX	Transfer A to X
EOR	Exclusive OR	TAY	Transfer A to Y
INC	Increment memory	TSX	Transfer SP to X
INX	Increment X	TXA	Transfer X to A
INY	Increment Y	TXS	Transfer X to SP
JMP	Jump	TYA	Transfer Y to A

Figure 5-2. Instruction Set of MOS Technology 6502.

ORIGINAL PAGE IS
OF POOR QUALITY

Command Mnemonic	Hex Code (sr = 1)	Hex Code (sr = 0)	Execution Cycles	Summary Description
16-BIT FIXED-POINT OPERATIONS				
SADD	EC	6C	16-18	Add TOS to NOS. Result to NOS. Pop Stack.
SSUB	ED	6D	30-32	Subtract TOS from NOS. Result to NOS. Pop Stack.
SMUL	EE	6E	84-94	Multiply NOS by TOS. Lower result to NOS. Pop Stack.
SMUU	F6	76	80-98	Multiply NOS by TOS. Upper result to NOS. Pop Stack.
SDIV	EF	6F	84-94	Divide NOS by TOS. Result to NOS. Pop Stack.
32-BIT FIXED-POINT OPERATIONS				
DADD	AC	2C	20-22	Add TOS to NOS. Result to NOS. Pop Stack.
DSUB	AD	2D	38-40	Subtract TOS from NOS. Result to NOS. Pop Stack.
DMUL	AE	2E	194-210	Multiply NOS by TOS. Lower result to NOS. Pop Stack.
DMUU	B6	36	182-218	Multiply NOS by TOS. Upper result to NOS. Pop Stack.
DDIV	AF	2F	196-210	Divide NOS by TOS. Result to NOS. Pop Stack.
32-BIT FLOATING-POINT PRIMARY OPERATIONS				
FADD	90	10	54-368	Add TOS to NOS. Result to NOS. Pop Stack.
FSUB	91	11	70-370	Subtract TOS from NOS. Result to NOS. Pop Stack.
FMUL	92	12	146-168	Multiply NOS by TOS. Result to NOS. Pop Stack.
FDIV	93	13	154-184	Divide NOS by TOS. Result to NOS. Pop Stack.
32-BIT FLOATING-POINT DERIVED OPERATIONS				
SQRT	81	01	782-870	Square Root of TOS. Result to TOS.
SIN	82	02	3796-4808	Sine of TOS. Result to TOS.
COS	83	03	3840-4878	Cosine of TOS. Result to TOS.
TAN	84	04	4894-5886	Tangent of TOS. Result to TOS.
ASIN	85	05	6230-7938	Inverse Sine of TOS. Result to TOS.
ACOS	86	06	6304-8284	Inverse Cosine of TOS. Result to TOS.
ATAN	87	07	4992-6536	Inverse Tangent of TOS. Result to TOS.
LOG	88	08	4474-7132	Common Logarithm of TOS. Result to TOS.
LN	89	09	4298-6956	Natural Logarithm of TOS. Result to TOS.
EXP	8A	0A	3794-4878	e raised to power in TOS. Result to TOS.
PWR	8B	0B	8290-12032	NOS raised to power in TOS. Result to NOS. Pop Stack.
DATA AND STACK MANIPULATION OPERATIONS				
NOP	80	00	4	No Operation. Clear or set SVREQ.
FIXS	9F	1F	90-214	Convert TOS from floating point format to fixed point format.
FIXD	9E	1E	90-336	
FLTS	9D	1D	62-156	Convert TOS from fixed point format to floating point format.
FLTD	9C	1C	56-342	
CHSS	F4	74	22-24	Change sign of fixed point operand on TOS.
CHSD	B4	34	26-28	
CHSF	95	15	16-20	Change sign of floating point operand on TOS.
PTOS	F7	77	16	Push stack. Duplicate NOS in TOS.
PTOD	B7	37	20	
PTOF	97	17	20	
POPS	F8	78	10	Pop stack. Old NOS becomes new TOS. Old TOS rotates to bottom.
POPD	B8	38	12	
POPF	98	18	12	
XCHS	F9	79	18	Exchange TOS and NOS
XCHD	B9	39	26	
XCHF	99	19	26	
PUPI	9A	1A	16	Push floating point constant π onto TOS. Previous TOS becomes NOS.

Figure 5-3. Instruction Set of Am9511A.

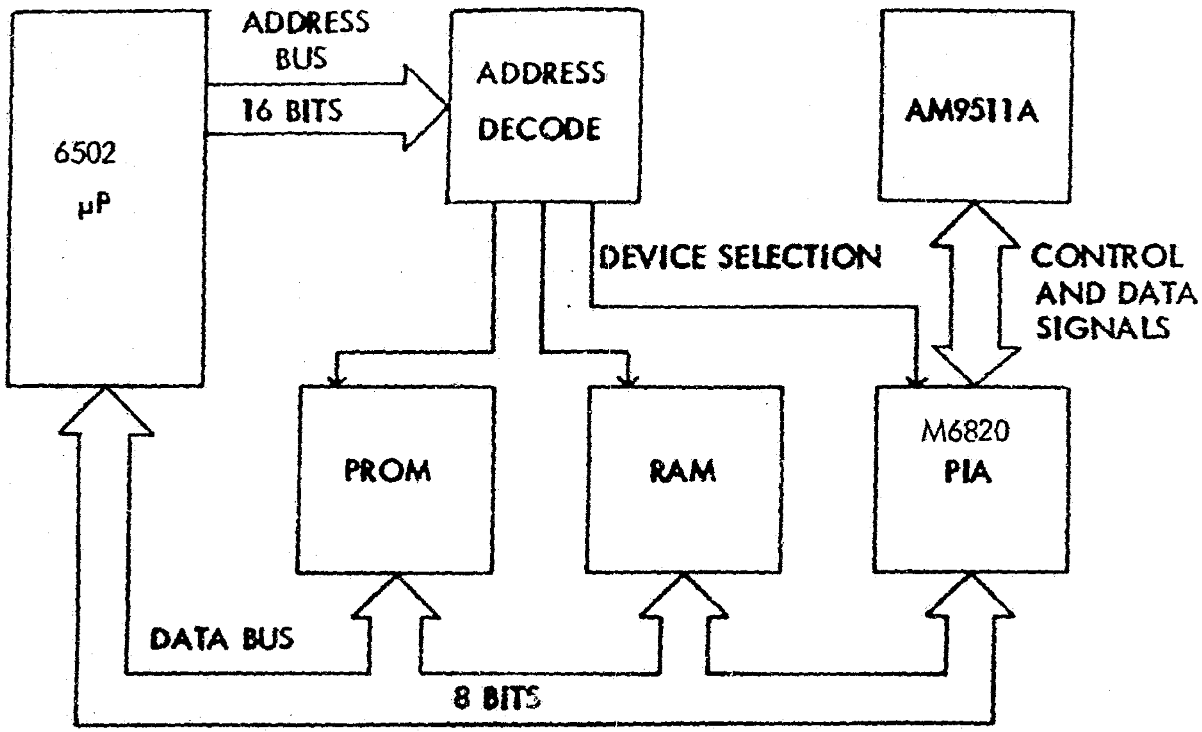


Figure 5-4. Block Diagram of Microcomputer Navigational System.

to send an eight-bit word to the 9511 representing a command to be executed, and to read the 9511 eight-bit status register. Four subroutines were developed for the interface by Fischer.

These four subroutines are: "PINT", "PUSH", "POP" and "CMND". "PINT" initializes the M6820 & PIA and also initializes RAM locations for the scratch-pad use. "PUSH" is used to copy a four-byte number from read/write memory onto the stack of the 9511, and "POP" is used to do the opposite. "CMND" is used to command the 9511 to perform a given function. A part of the "CMND" subroutine checks the status register to determine the final outcome of the completed command. Figure 5-5 shows a set of logic flow diagrams illustrating steps of the contral program which execute to communicate with the 9511.

B. Navigational Programs.

1. Relationships Among Navigational Programs. There are three navigational programs for the Loran-C receiver, as was mentioned in Chapter IV. The main program "LORPROM5" does Loran-C tracking phase-locked-loop operation providing two TDs. The execution time of this program is ten group repetition intervals(GRI) because the program computes two TDs per one GRI and calculates each average TD with ten TD references.

After two TDs are calculated, ten GRIs later, the main program jumps to subprogram 1, "COORD2" which filters two TD values and provides coordinates of the position using a TD-to-position conversion. The filter included in "COORD2" is the α-β filter with an effective time constant of 6 seconds. This time gives acceptable response for position information, considering typical aircraft speeds.

Subprogram-2, "RNAV", which provides the area navigation information, can be executed after "LORPROM5" and "COORD2" are executed. Two sets of input data are needed for subprogram 2; one is the set of coordinates of position from "COORD2" and the other is a set of coordinates of a selected waypoint from the user's waypoint table.

The execution time of subprogram "COORD2" and subprogram "RNAV" is three to four GRIs depending upon the content of the calculation. After the execution of "RNAV", the program process goes back to the main program "LORPROM5" and repeats the same process. Hence, the outer loop including "LORPROM5", "COORD2" and "RNAV" forms the Loran-C software to provide navigational information, such as TDs, coordinates of aircraft position, range and bearing angle, CTE/CTEB, GS and ETA, every thirteen to fourteen GRIs (every 1.29 to 1.39 seconds at 9960 GRI). Figure 5-6 shows this scheme for Loran-C navigation software, and figure 5-7 shows the address map for these programs. The main program (LORPROM5) is placed on 2K erasable, programmable, read-only memory (EPROM). The subprogram (COORD2 and RNAV) is placed on three 2K EPROM and uses two pages (256 byte per page) of general read/write storage for results of intermediate calculations. The page zero (in RAM) is for temporary variables and flags.

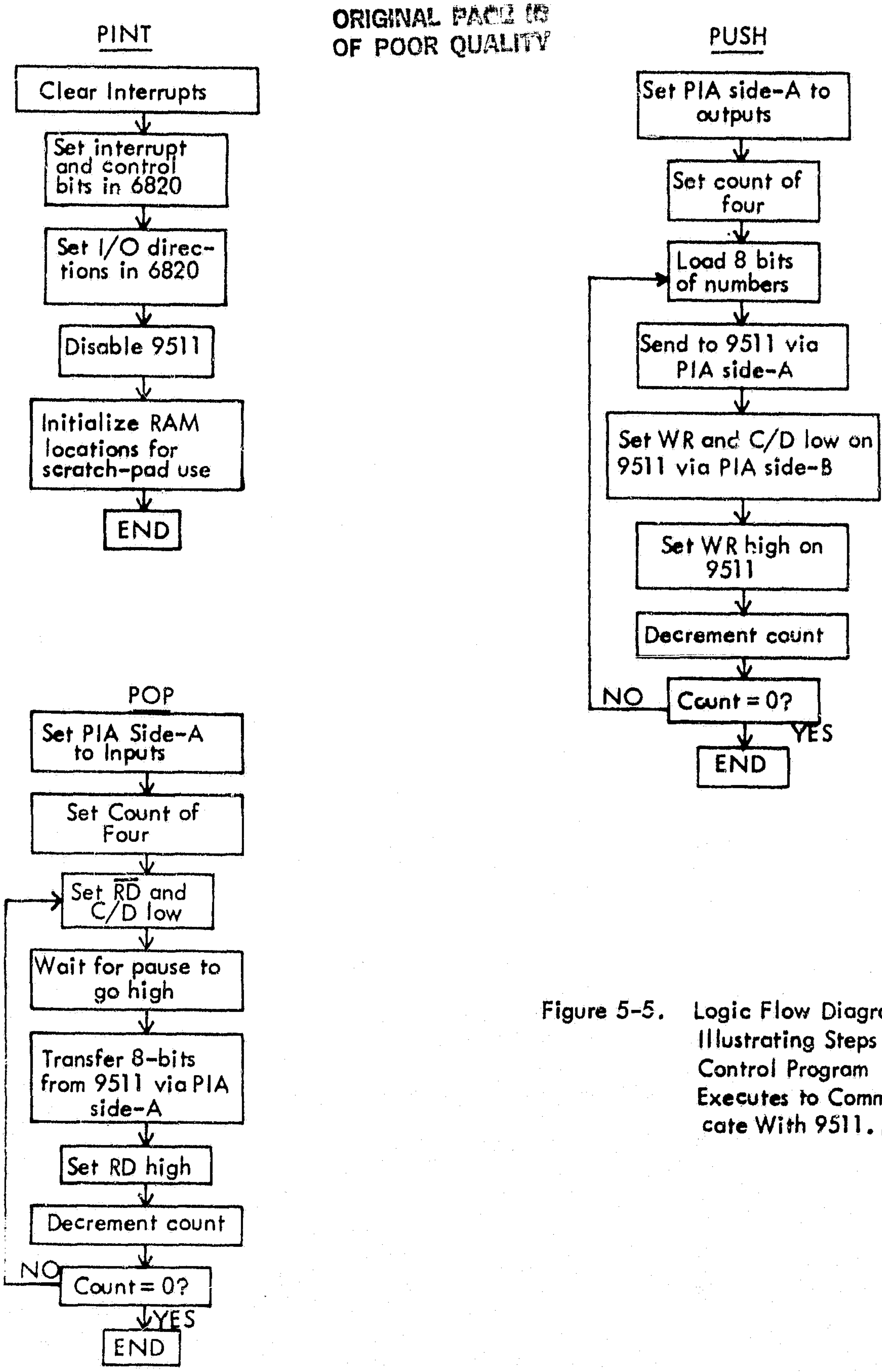


Figure 5-5. Logic Flow Diagrams Illustrating Steps Control Program Executes to Communicate With 9511. [40]

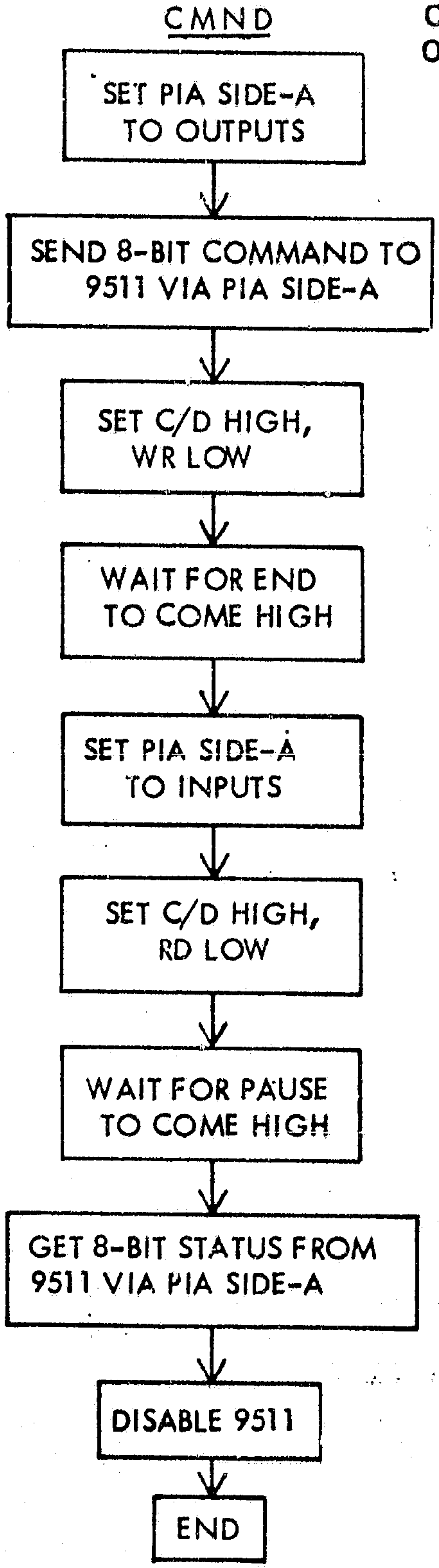


Figure 5-5. Continued.

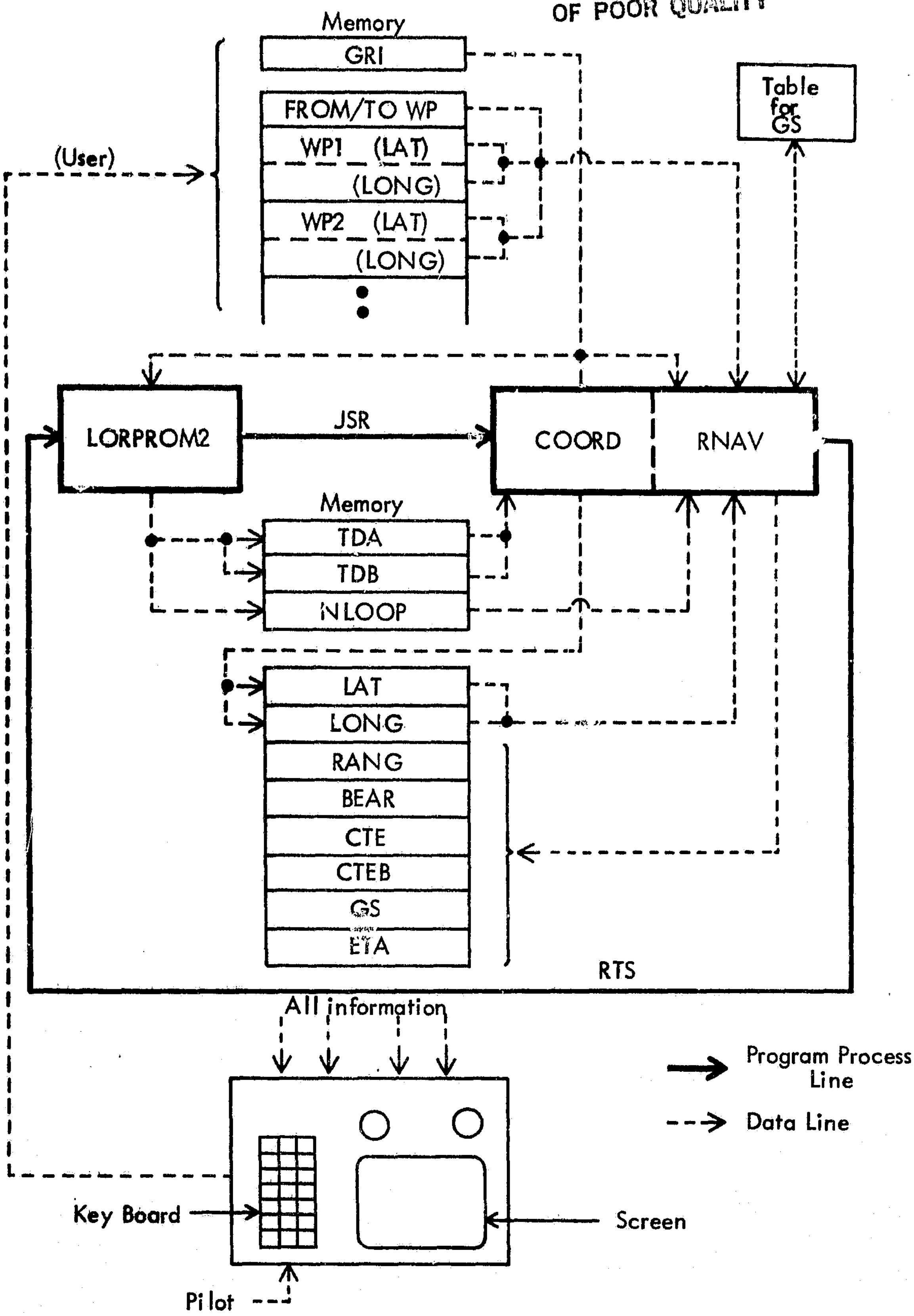


Figure 5-6. Process of Loran-C Navigation Programs.

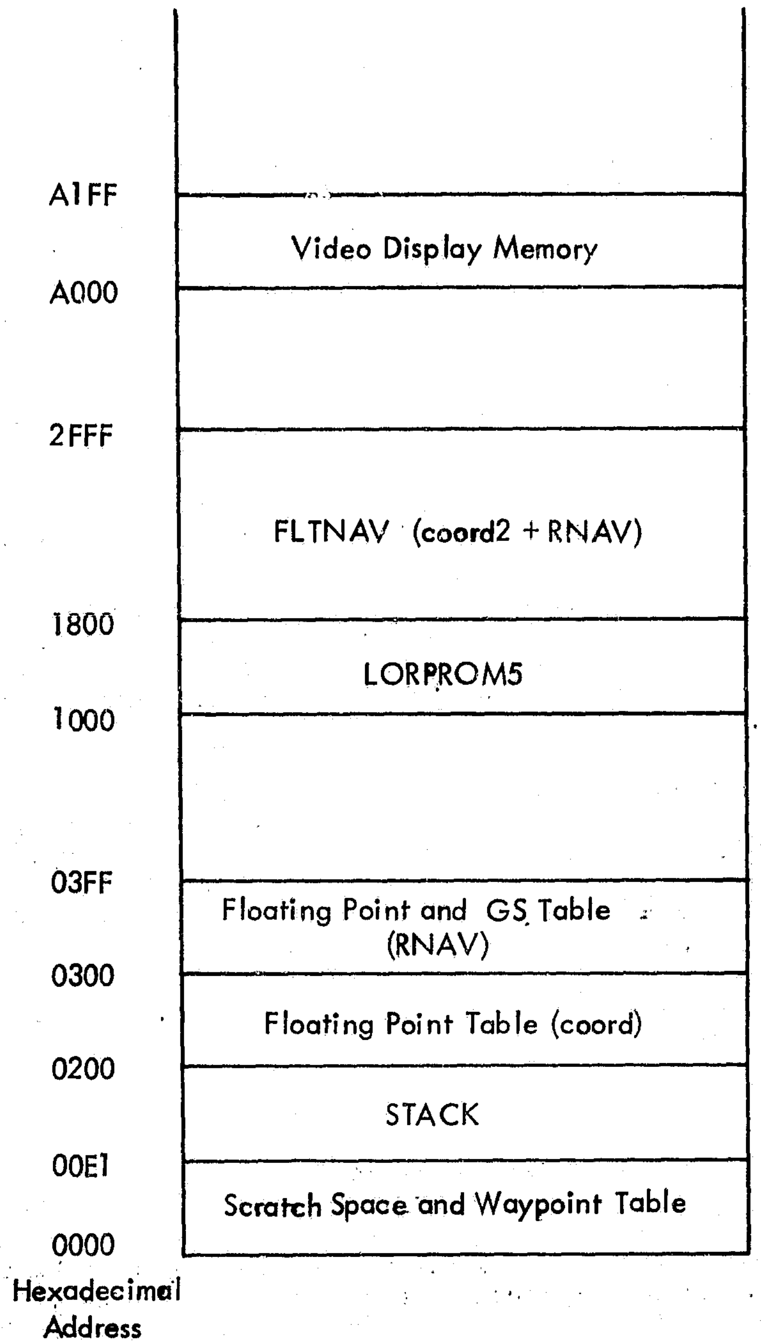


Figure 5-7. Memory Map of Loran-C Navigation Software.

2. RNAV Program. The RNAV program is the main subject of this paper. Figure 5-8 shows the flow chart of the whole program. There are four parts in this program.

The first part of the RNAV program takes care of finding selected waypoints from the user's waypoint table, displaying waypoint numbers and converting the degree-minute-second format (BCD) to the floating-point format in radian units.

The flow chart for the waypoint conversion is shown in figure 5-9. The coordinates of the waypoints and desired waypoint numbers are input to the waypoint table from the keyboard by the user. The input has a certain format in the waypoint table as a part of figure 5-10 indicates. Since the 9511 is used for calculations, it is necessary to convert each element for navigational calculations into 32-bit floating-point format.

An example of waypoint conversion is shown in figure 5-10. Suppose the user chooses waypoint No. 3 whose latitude is 125° 53' 41". A four-byte waypoint register, three-byte temporary register and accumulator are used for this conversion, and the final result is stored into the four-byte waypoint register. One of the commands available with the 9511 is to convert a fixed-point (integer) number to floating-point (step 14 in figure 5-10). The fixed-point number must be in binary because it is not possible to represent a fraction using this command. Therefore, step 14 must come before step 15. This waypoint conversion is not repeated until the user changes waypoint numbers for a desired course change.

The second part of the RNAV program calculates range and bearing angle. For the first loop, which includes all programs (LORPROM5, COORD2 and RNAV) after the user chooses the desired waypoints, this part calculates range and bearing angle of the desired course. From the second loop, it starts calculating range and bearing angle between the position of the aircraft and a TO waypoint.

After range and bearing angle are calculated, the third part calculates CTEB and CTE.

Calculation of CTEB is as follows:

CTEB = (Present Bearing Angle)
-(Bearing Angle of The Desired Course)

However, when the calculation includes a transition between 0° and 360°, a certain correction is necessary. For example, when a present bearing angle is 358° and a bearing angle of the desired course is 1°, the CTEB becomes 357°. The resultant number, which should be 3°, can be obtained by subtracting this CTEB of 357° from 360°.

It is very convenient for the software to show whether the course

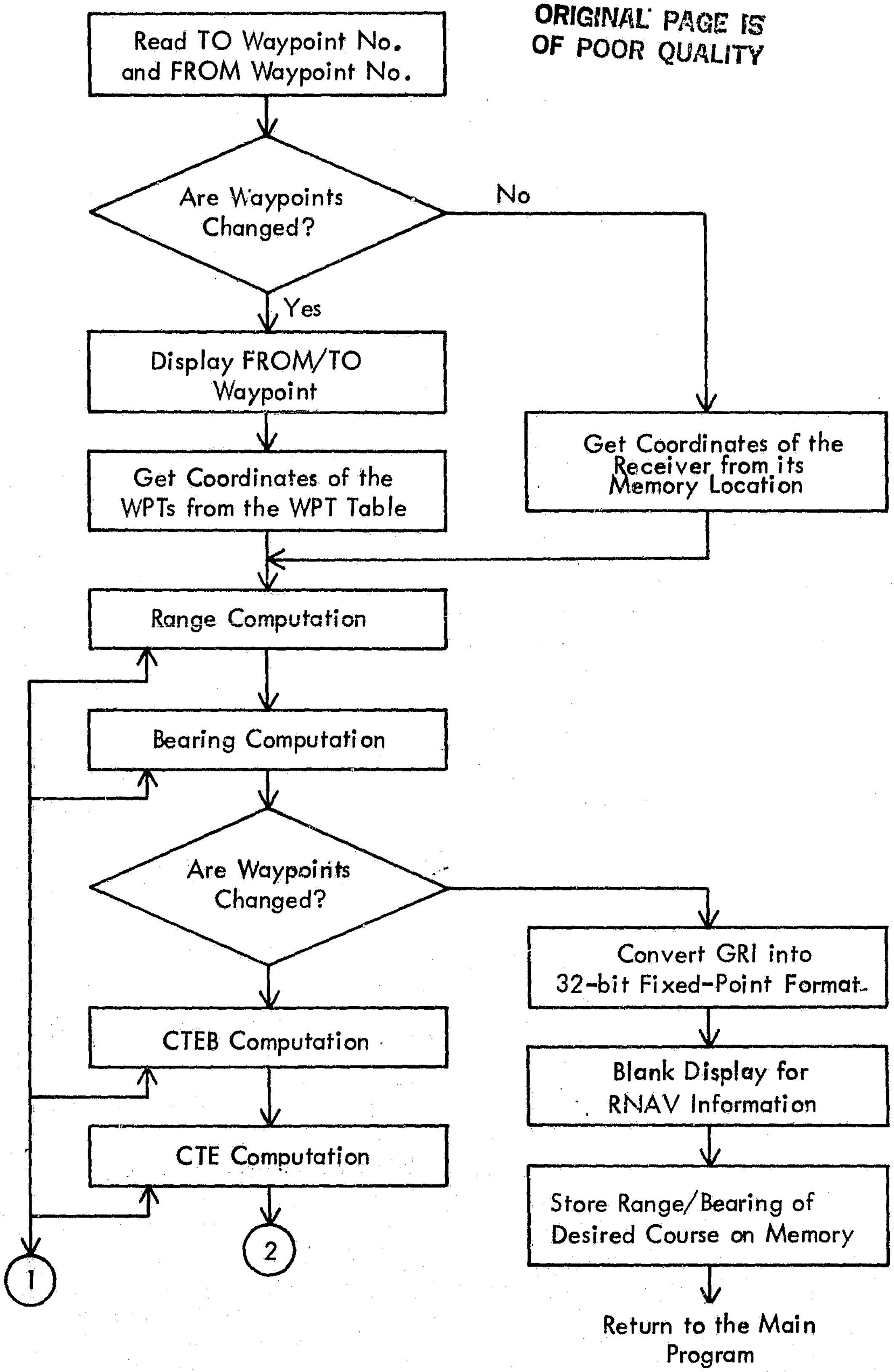


Figure 5-8. Flow Chart of RNAV Program.

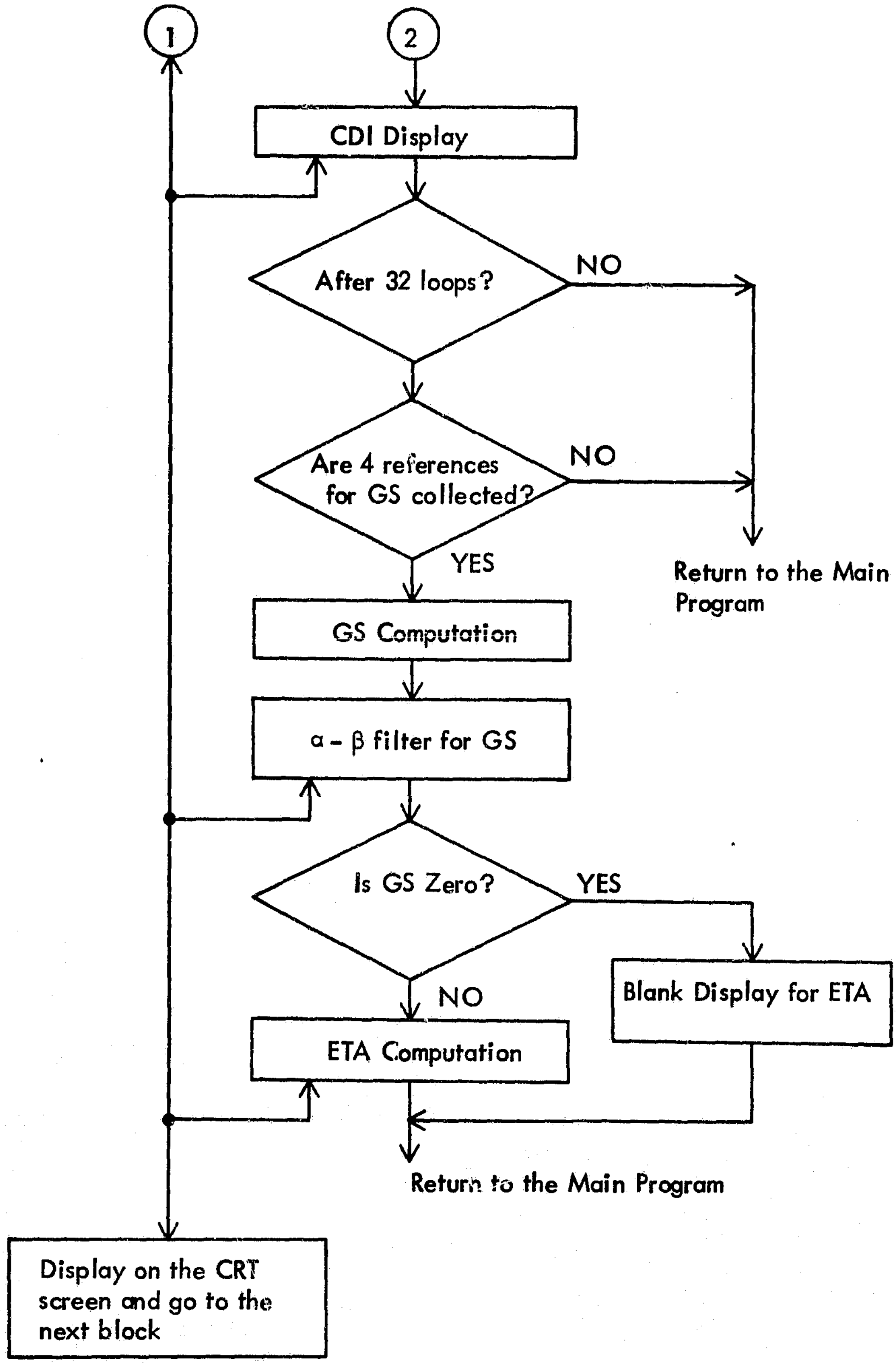


Figure 5-8. Continued.

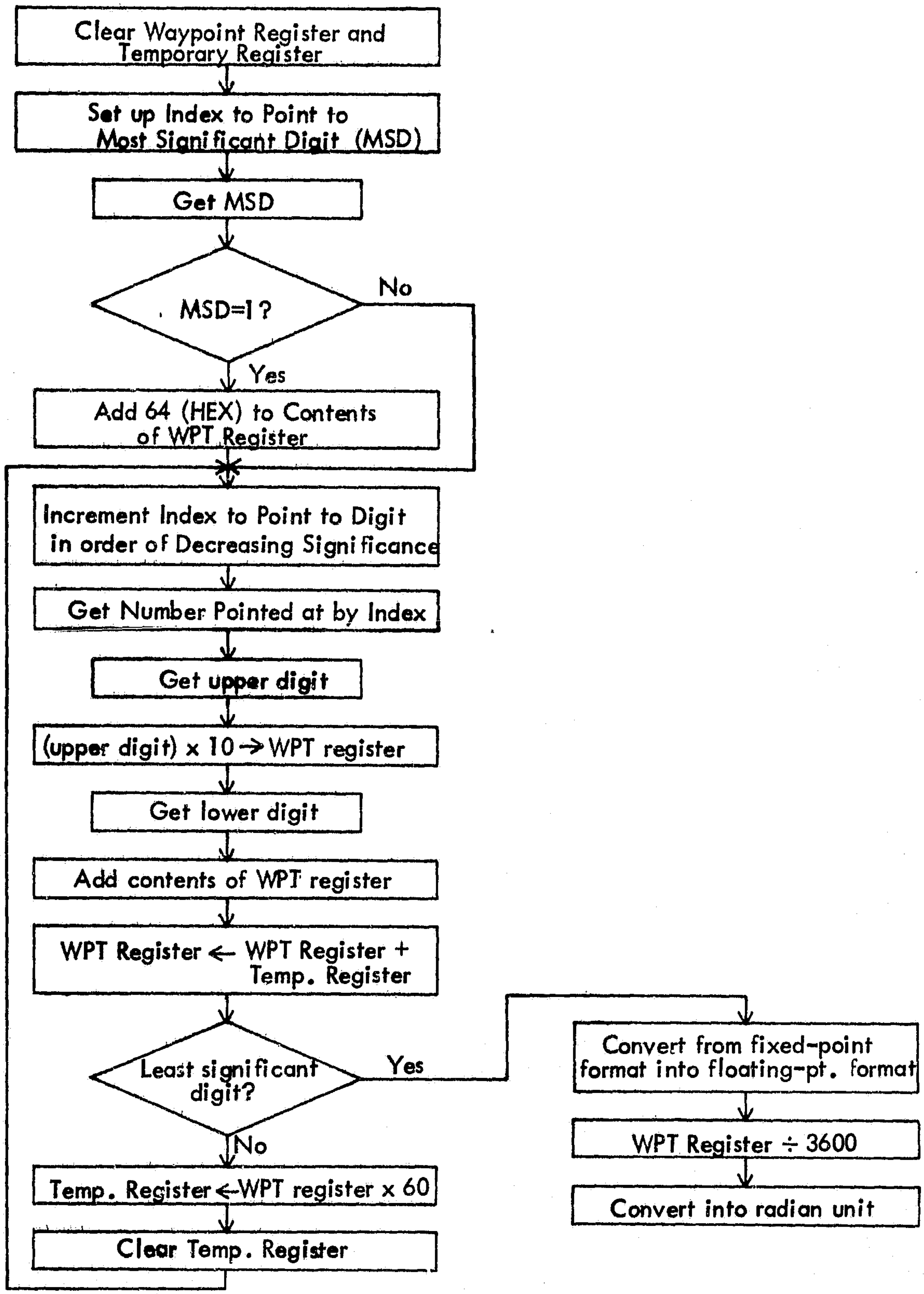


Figure 5-9. Flow Chart of Waypoint Conversion.

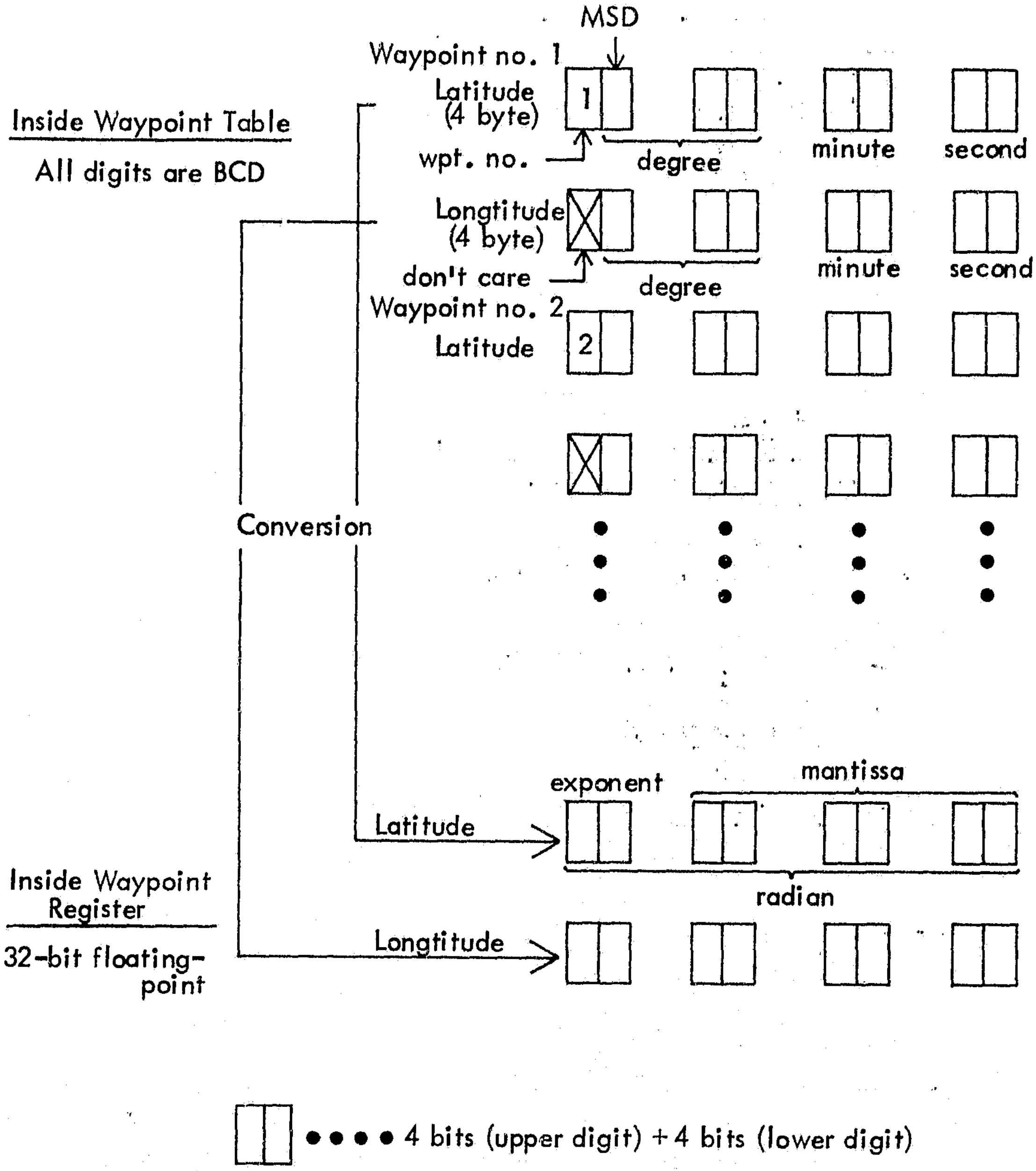


Figure 5-10. Steps of Waypoint Conversion.

Example: | 3 | 1 | | 2 | 5 | | 5 | 3 | | 4 | 1 |

Waypoint No. 3 125° 53' 41" (Latitude)

	WPT Reg. (4 byte)	Temp.Reg. (3 byte)	Accum.
1. Clear Temp. Register and WTP Register	00 00 00 00	00 00 00	
2. Get MSD			01
3. Add 64 (HEX) to WPT Register	00 00 00 64		
4. Get Upper Digit			02
5. Add to Temp. Register		00 00 02	
6. Temp. Register X 10		00 00 14	
7. Get Lower Digit			05
8. Add to WPT Register		00 00 19	
9. Add Temp. Register to WPT Register	00 00 00 7D		
10. WPT Register X 60 ≡ Temp. Register	00 00 1D 4C		
11. Clear Temp. Register		00 00 00	
12. Repeat from 4 to 11 for minute	00 06 EA 3C		
13. Repeat from 4 to 9 for second	00 06 EA 65		
14. Convert from Fixed Point to Floating Point	31 CE 19 DE		
15. WPT Register ÷ 3600	07 FB CA 19		
16. Convert into Radian Unit	02 8C A0 30		

Figure 5-10. Continued.

is to the left or to the right. When the CTEB is between 0° and 90°, the desired course is on the right-side of the airplane, but if the CTEB is between -90° and 0°, then the course is on the opposite (left) side of the airplane. Furthermore, the magnitude of the CTEB exceeds 90° after the airplane passes the To waypoint. In order to find the side of the desired course from any airplane position, the procedure in figure 5-11 was added inside the CTEB calculation routine. Letters "L" and "R" are added to the front of the CTEB display.

The CDI (Course Deviation Indicater) shows the CTE value visually. A zero center is the desired course, and a needle shows CTE (FTE). The Ohio University Loran-C display uses a video board, so that a graphic display is possible. In the program, a position register is set according to a CTE value (see figure 5-12).

The last part of RNAV takes care of the ground speed calculation (figure 5-13) and estimated time of arrival at the waypoint. Since ground speed fluctuates due to any kind of position error, the ground speed calculation needs some delay so that TD values become stable. Usually, TD values need several computation cycles to become stable after the receiver starts tracking Loran-C signals. The 32 loops delay after the first TD measurement was added between the third part and the last part to eliminate the TD transition errors.

After calculating GS, the ground speed goes through a recursive filter loop, as was discussed in Chapter IV. The effective time of the $\alpha-\beta$ filter for ground speed (12 seconds), gives the fastest response without conspicuously noisy data. However, the ground speed has a random noise error range of zero to 30 knot(=nm/hour) when the receiver is at a fixed position; therefore, the display indicates zero ground if the speed is below 30 knots. Although this response is still slow for accelerated flight, it is reasonable for a constant ground speed flight.

All navigational information are displayed on a CRT screen. Figure 5-14 shows a Loran-C receiver display. To display information on the screen, all data must be converted from floating-point format to BCD. A subroutine now available for this conversion was developed by Fischer.

The complete microprocessor program for area navigation is shown in appendix C. This program occupies 2064 bytes of EPROM-2716 and 400 bytes of read/write memory for scratch calculations, a waypoint table and a reference table for GS averaging. Total RNAV calculations require about 0.20 second to complete with the AM9511A and the 6502 running at a clock speed of 1MHz. Figure 5-15 is a photograph of the Loran-C receiver developed by the Ohio University Avionics Engineering Center. The efficacy of the RNAV program was checked using this receiver.

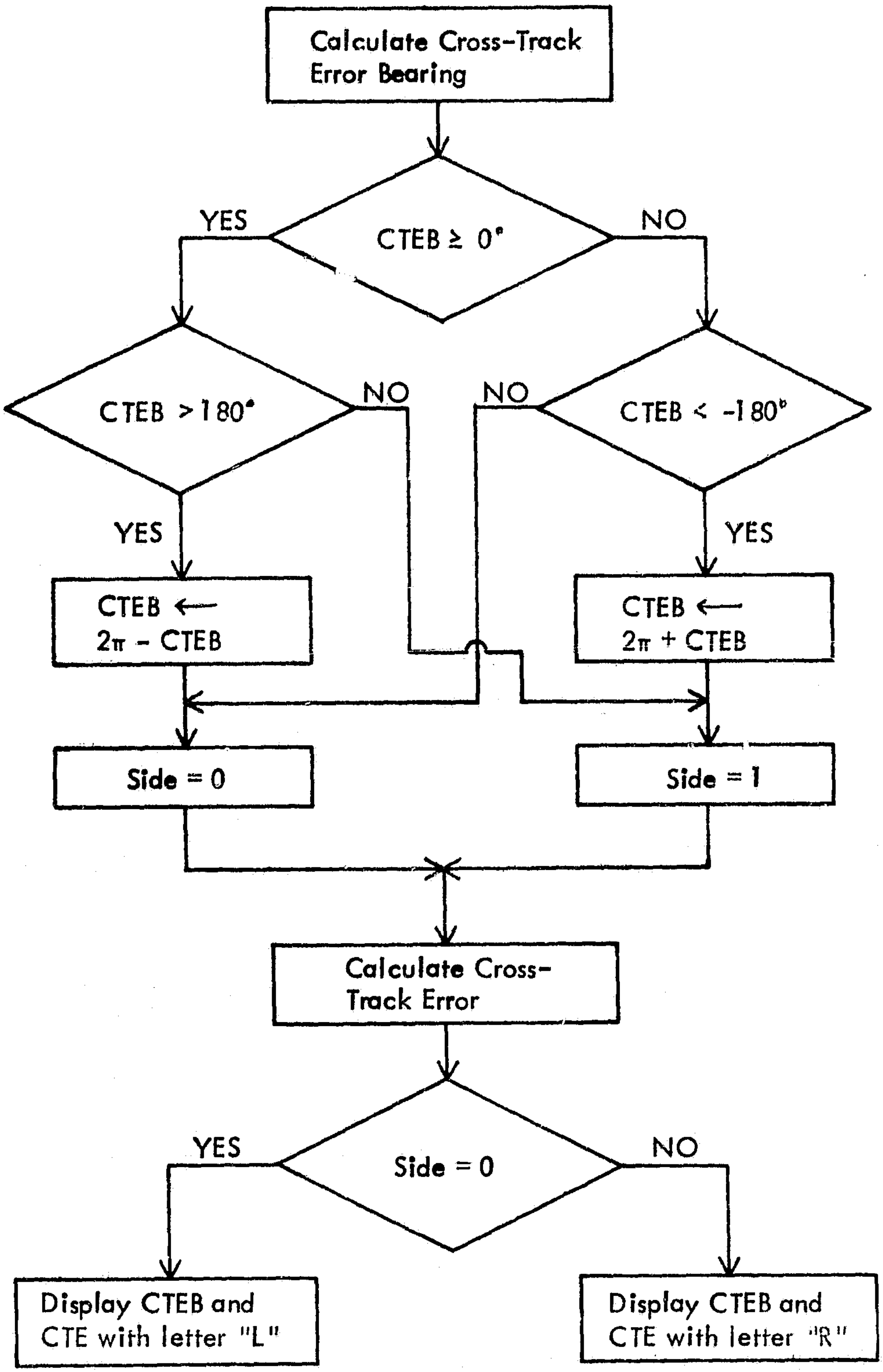


Figure 5-11. Flow Chart of Cross-Track Error and Cross-Track Error Bearing.

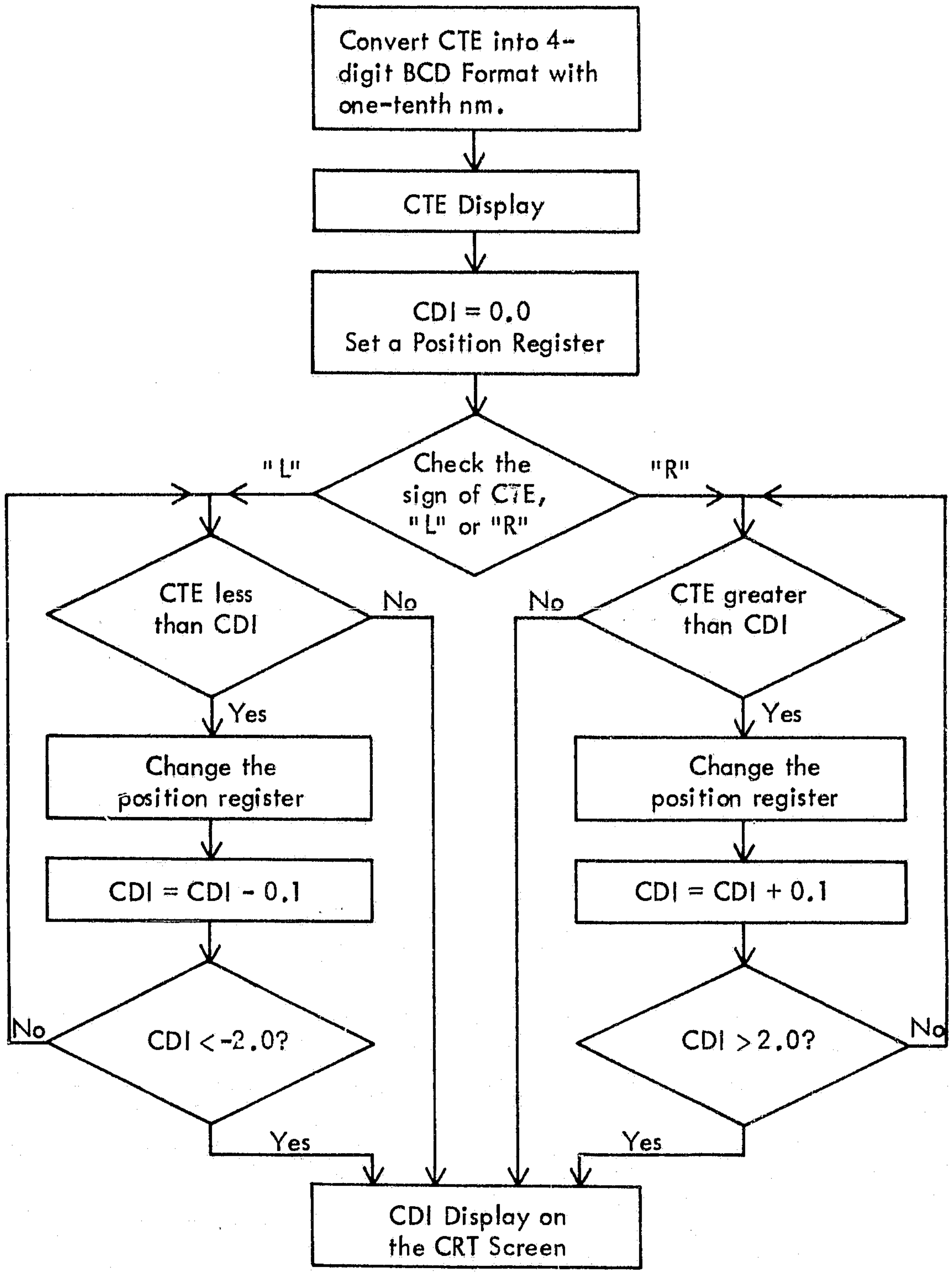


Figure 5-12. CDI Display.

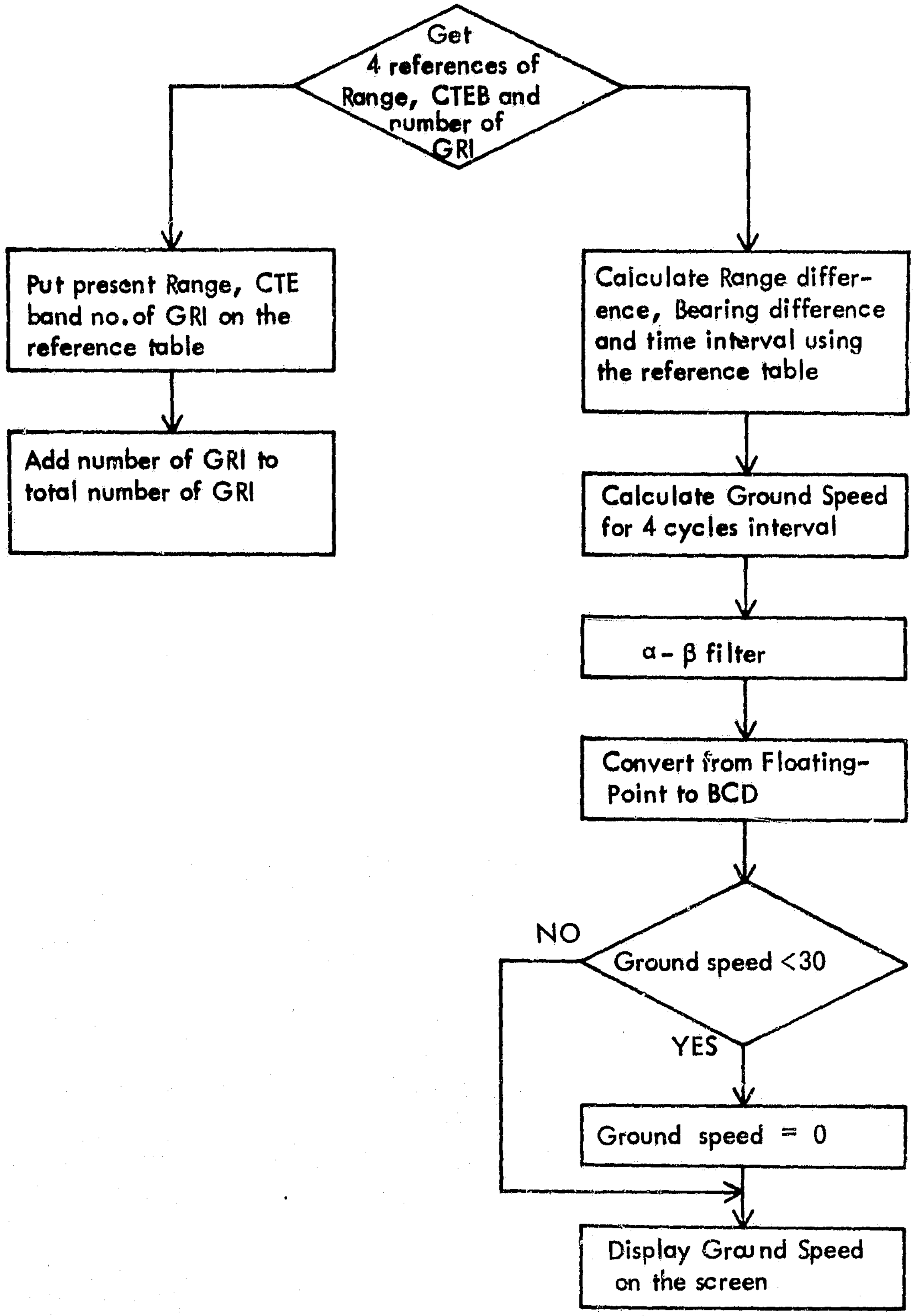


Figure 5-13. Flow Chart of Ground Speed.

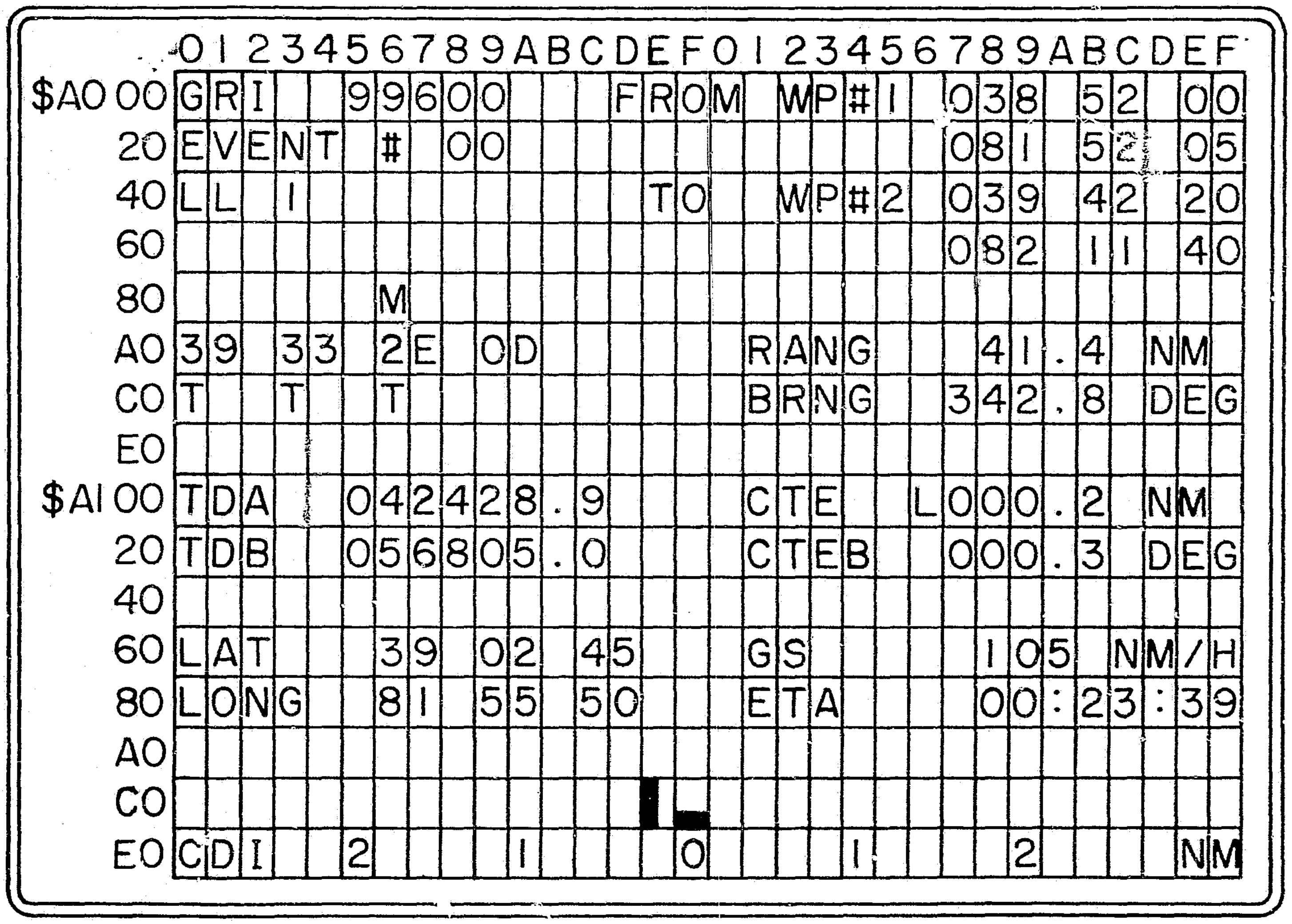


Figure 5-14. Loran-C Receiver CRT D[illegible]ay.

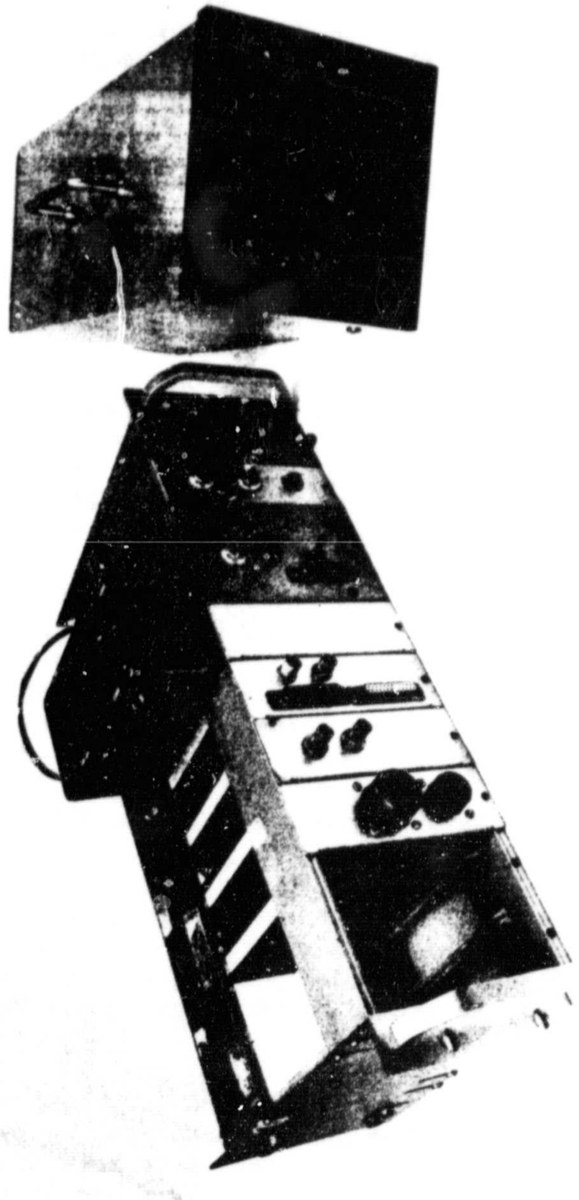


Figure 5-15. Photograph of Ohio University Loran-C Receiver.

VI. TESTS WITH MICROCOMPUTER IN AREA NAVIGATION APPLICATION

A verification of the performance capabilities of area navivation was examined by testing with simulations and flight testing. Accuracy and operational stability of the system were the focus of the testing.

A. Testing with Simulations.

Accuracy checks were accomplished by picking random locations on the earth, including some locations inside the northeast U.S. Loran-C chain because the following flight testing was performed in this chain. Coordinates of each point were stored in random access memory either as a receiver position or as a waypoint, then the range/bearing calculation routine was executed. Table 6-1 is the result, comparing the actual range to the waypoint and bearing to true north, as calculated using the elliptical model. The accuracy is consistent with the Fortran simulation using the IBM4341 computer (Chapter IV.A.4).

After testing the accuracy of the range/bearing calculation, other testing to check the CTE/CTEB calculation routine was made. As figure 6-1 shows, two desired courses were set for this test; one is from waypoint 2 to waypoint 1 and the other is from waypoint 3 to waypoint 1, and points A, B and C are observed points with respect to the desired course from wpt2 to wpt1, point D, E and F are observed points with respect to the desired course from wpt3 to wpt1. All observed points are in the northeast U.S. chain. The coordinates of each point were stored in random access memory and the RNAV program was excuted without the other programs (LORPROM5 and COORD2). Table 6-2 indicates the results of this test which are indicated with 10^{-1} resolution on the CRT screen (figure 5-14). The numbers in this table are comparisons with the results of the receiver RNAV program, and those calculated by the Fortran program using the elliptical model, since the high accuracy of the elliptical model was demonstrated in table 4-1. The correct sense of the CTE/CTEB display including the effects of ground track To or From waypoint computed by the CTE/CTEB calculation routine were verified by this test.

The ground speed calculation was tested by a simulation program which created a moving observation point; however, it was difficult to simulate these computational results without actual flights with random noise. For evaluating the ground speed computation, the flight test was necessary.

Moreover, the total program software (LORPROM5 + COORD2 + RNAV) was tested by using a Loran-C simulator manufactured by Epsco Inc. As a result of this testing, the bias errors were determined to be due to signal strengths for reception of Loran-C stations in the local area [41], and the TD-to-Geometric conversion. The simulator produced essentially no noise.

No.	Receiver Point N.LAT/W.LONG	To Waypoint N.LAT/W.LONG	Range/Bear (nm/degree)	Microcomputer RNAV Range/Bear nm/degree	Microcomputer RNAV Error of Range/Bear
1	40 30 37.8	40 00 0.0	43.448	43.448	0.0
	17 19 43.3	18 00 0.0	225.43	225.39	0.04
2	9 59 48.3	10 00 0.0	86.897	86.896	0.001
	16 31 55.9	18 00 0.0	270.26	270.25	0.01
3	69 48 5.7	70 00 0.0	173.794	173.788	0.006
	9 37 28.6	18 00 0.0	277.87	277.86	0.01
4	13 04 12.6	10 00 0.0	260.690	260.697	0.007
	14 51 13.3	18 00 0.0	225.63	225.55	0.08
5	73 35 9.2	70 00 0.0	347.588	347.576	0.012
	3 26 35.1	18 00 0.0	238.84	238.84	0.0
6	42 42 50.6	41 15 11.9	318.621	318.619	0.002
	76 49 33.9	69 58 39.1	103.57	103.68	0.09
7	42 42 50.6	46 48 27.2	452.416	452.410	0.006
	76 49 33.9	67 55 37.7	54.05	54.04	0.01
8	42 42 50.6	39 51 7.5	511.412	511.409	0.003
	76 49 33.9	87 29 12.1	253.91	253.95	0.04
9	42 42 50.6	34 03 46.0	521.045	521.052	0.007
	76 49 33.9	77 54 46.8	185.92	185.97	0.05

Table 6-1. Accuracy of Microcomputer Range/Bearing Computation with Elliptical Model.

ORIGINAL PAGE IS
OF POOR QUALITY

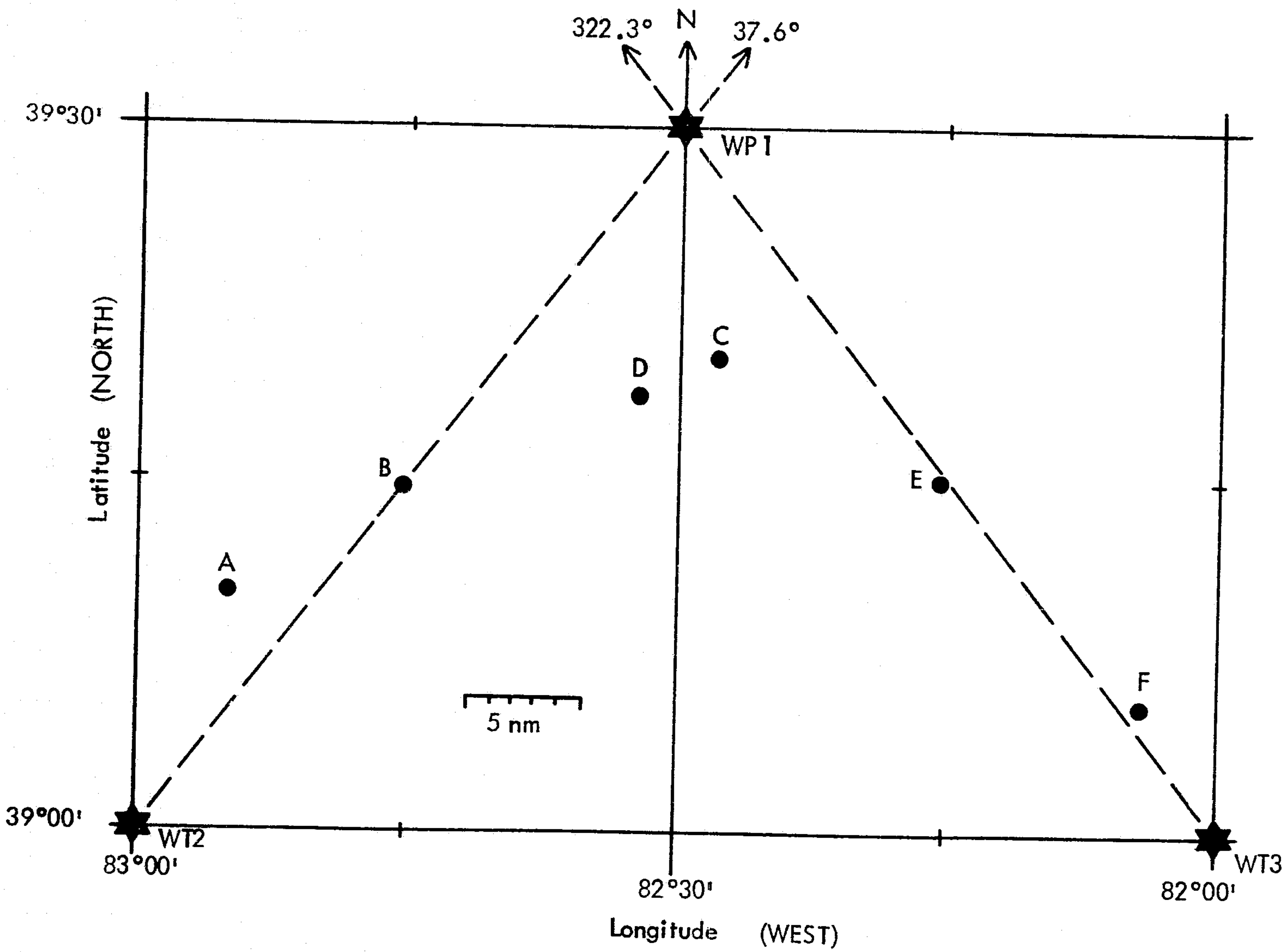


Figure 6-1. Area Navigation Computation from a Receiver's Point to a Waypoint using Fixed Time Differences.

Receiver Point	From Way Point	To Way Point	Range/ Bear (nm/°)	CTE/ CTEB (nm/°)	Receiver Display Range/ Bear (nm/°)	Receiver Display CTE/ CTEB (nm/°)
A	wp2	wp1	27.84 43.97	3.06 6.32	27.8 43.9	R 3.0 6.3
B	wp2	wp1	18.98 37.65	0.00 18.75	18.9 37.6	L 0.0 18.7
C	wp2	wp1	10.11 351.19	7.29 46.46	10.1 351.2	L 7.3 46.4
D	wp3	wp1	12.09 7.34	8.55 45.00	12.0 7.3	R 8.5 45.1
E	wp3	wp1	18.73 323.25	0.29 0.91	18.7 323.2	R 0.2 0.9
F	wp3	wp1	32.11 321.24	0.61 1.10	32.1 321.2	L 0.6 1.0

Point	Coordinate Latitude/ Longitude
wp1	39 30 00 82 30 00
wp2	39 00 00 83 00 00
wp3	39 00 00 82 00 00
A	39 10 00 82 55 00
B	39 14 59 82 45 00
C	39 20 00 82 28 00
D	39 18 00 82 32 00
E	39 15 00 82 15 30
F	39 05 00 82 04 00

Table 6-2. Test Result of Area Navigation Computation Using Fixed Time Differences

B. Flight Testing.

The purpose of the flight testing is to prove the effectiveness of the area navigation system for an actual flight. The filtering of time differences to smooth the raw data, filtering the ground speed to reduce random noise, range/bearing, CTE/CTEB and CDI indication were verified during the flight test.

Nineteen flight test paths, whose time length varied between 10 minutes to 20 minutes in a Piper Cherokee were flown with a total flight time of approximately 5 hours. Two flight tracks are shown in this paper. Figures 6-2 to 6-8 are from flight test 1 and figures 6-9 to 6-19 are from flight test 2.

1. Filtering time differences. For flight test 1 (figure 6-2), the pilot tracked from the Ohio University airport runway 25 (RWY 25) threshold, made a left 180° turn, tracking right of the desired course (from the BIAS NDB to the BIAS RWY 25 TRESH), made a left 180° turn again, passed above the NDB, making two loops around the NDB, tracking back to the RWY 25 threshold using an instrument landing system (ILS) localizer on RWY 25.

The filtered TD flight path of flight test 1 is shown in figure 6-2, and the nonfiltered TD flight path simulated by the Fortran program (appendix D) is shown in figure 6-3. As indicated by the comparison, about ±0.1nm random noise was detected with the error caused by the time delay on turns about 0.1nm. The bias error was measured about 0.7nm to the north. Figure 6-4 is the output simulated by the Fortran program (appendix E) which shows very nearly the same flight path as in figure 6-2; therefore, the Fortran simulation can be used for data analysis.

After 14 flight test paths were recorded, the 1MHz clock (which is used for the 6502 and the AM9511A), showed some instability, so 5 additional flight tests were made with a new stable clock.

For flight test 2 with the stable clock (figure 6-9), the pilot flew very similar patterns as in flight test 1, but after passing the NDB the pilot made one race track pattern and returned to the RWY 25 threshold. The coordinates of the two waypoints (Rwy 25 threshold and UNI NPB) are biased waypoints based on the previous flight tests. The flight path was smoother than the previous flight test due to the improved stability of the clock. Comparisons between figure 6-9 and figure 6-10 indicate the improved raw TD data which has less than ±0.1nm random noise with the bias reduced to about 0.5nm. Since the old effective time constant for the filtered TDs is too long for the TD's, using the same clock, the effective filter time constant can be reduced. The new time constant of 4 seconds (the previous time constant was 6 seconds) is used for the plot in figure 6-11. This reduction of the time constant improved the acceleration effect of turns, furthermore the ground speed computation process was speeded up.

2. Ground speed. The ground speed calculation is affected the most by random noise as was mentioned before. Two filters (one on TDs and another on GS) to filter the random noise were evaluated.

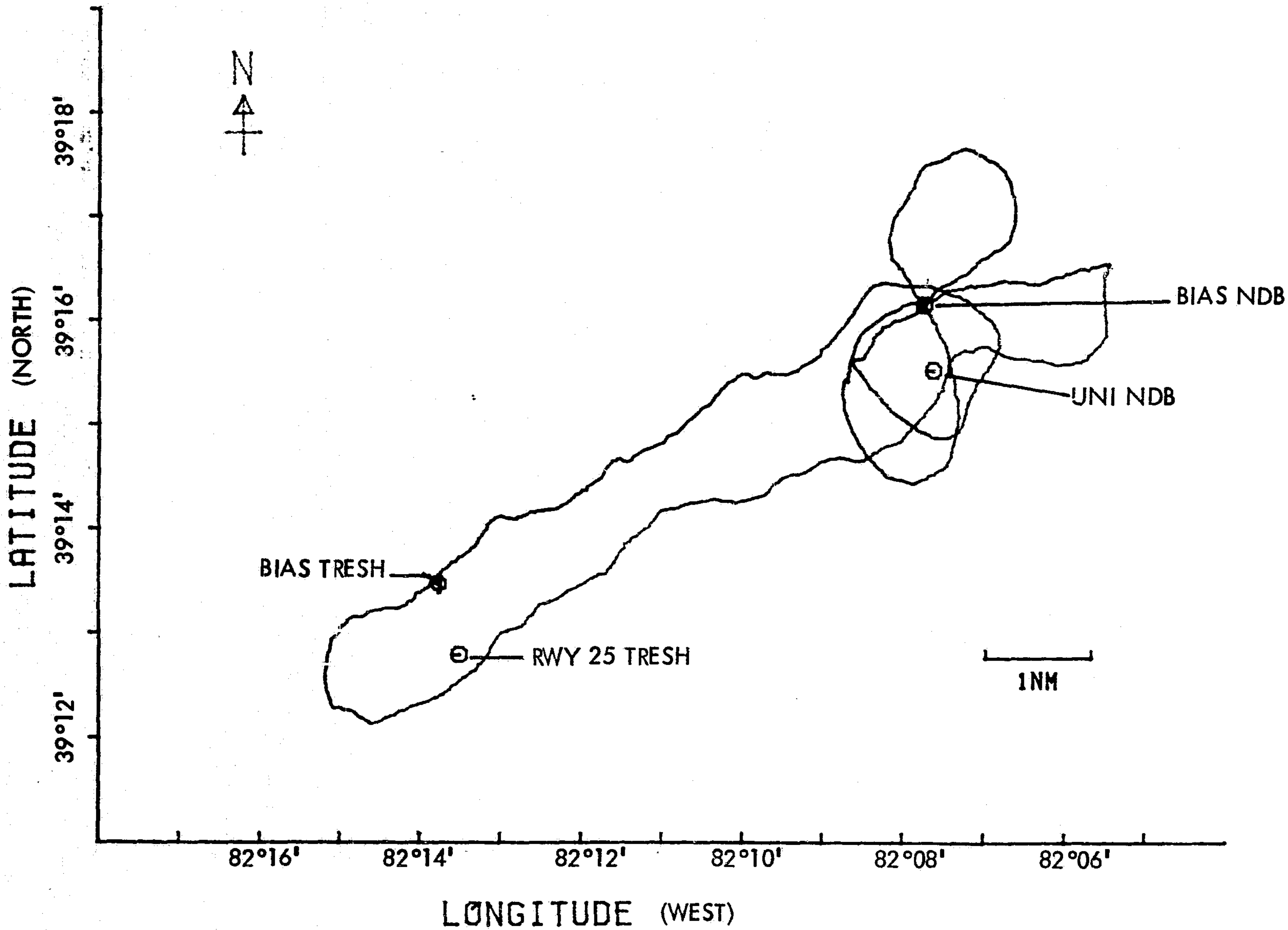


Figure 6-2. **Flight Path Plot,** Result of Flight Test 1, α-β Filter (t_f = 6 seconds, α = 0.167, β = 0.007) on TDs.

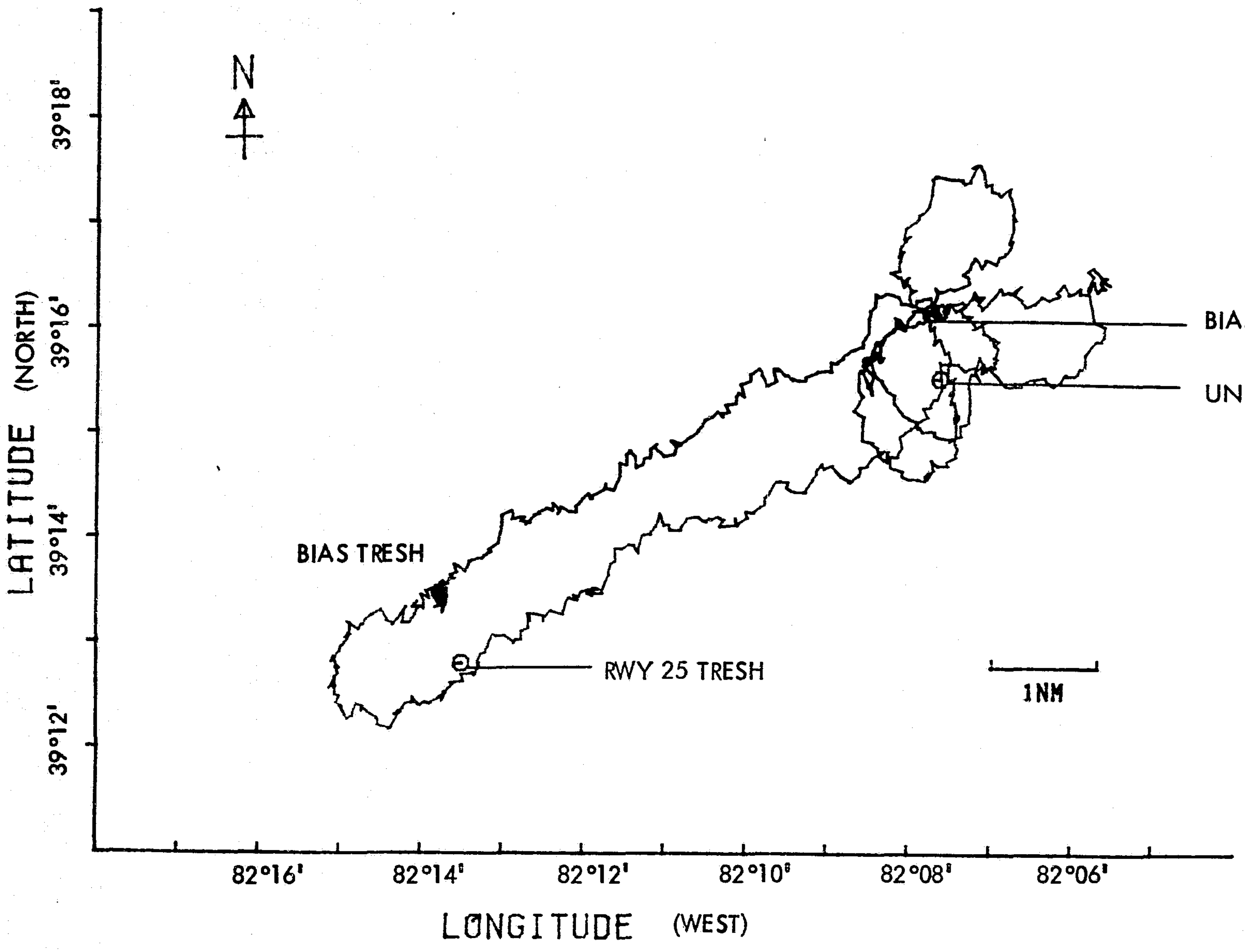


Figure 6-3. Flight Path Plot, Fortran Simulation of Flight Test 1 Using Nonfiltered TDs.

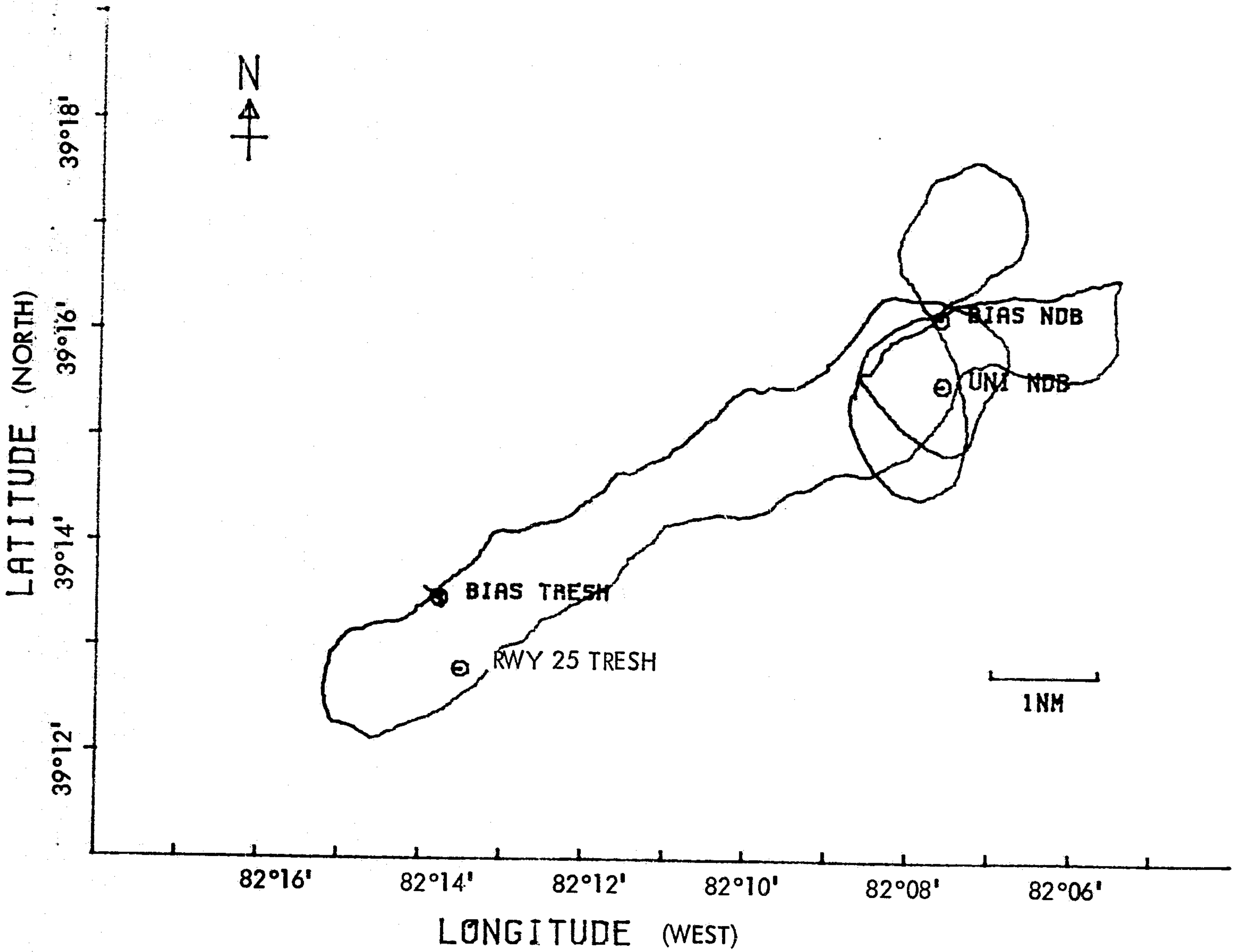


Figure 6-4. Flight Path Plot. Fortran Simulation of Flight Test 1 Using Filtered TDs $\alpha-\beta$ filter (t_f= 6 seconds, α=0.167, β=0.007) on TDs.

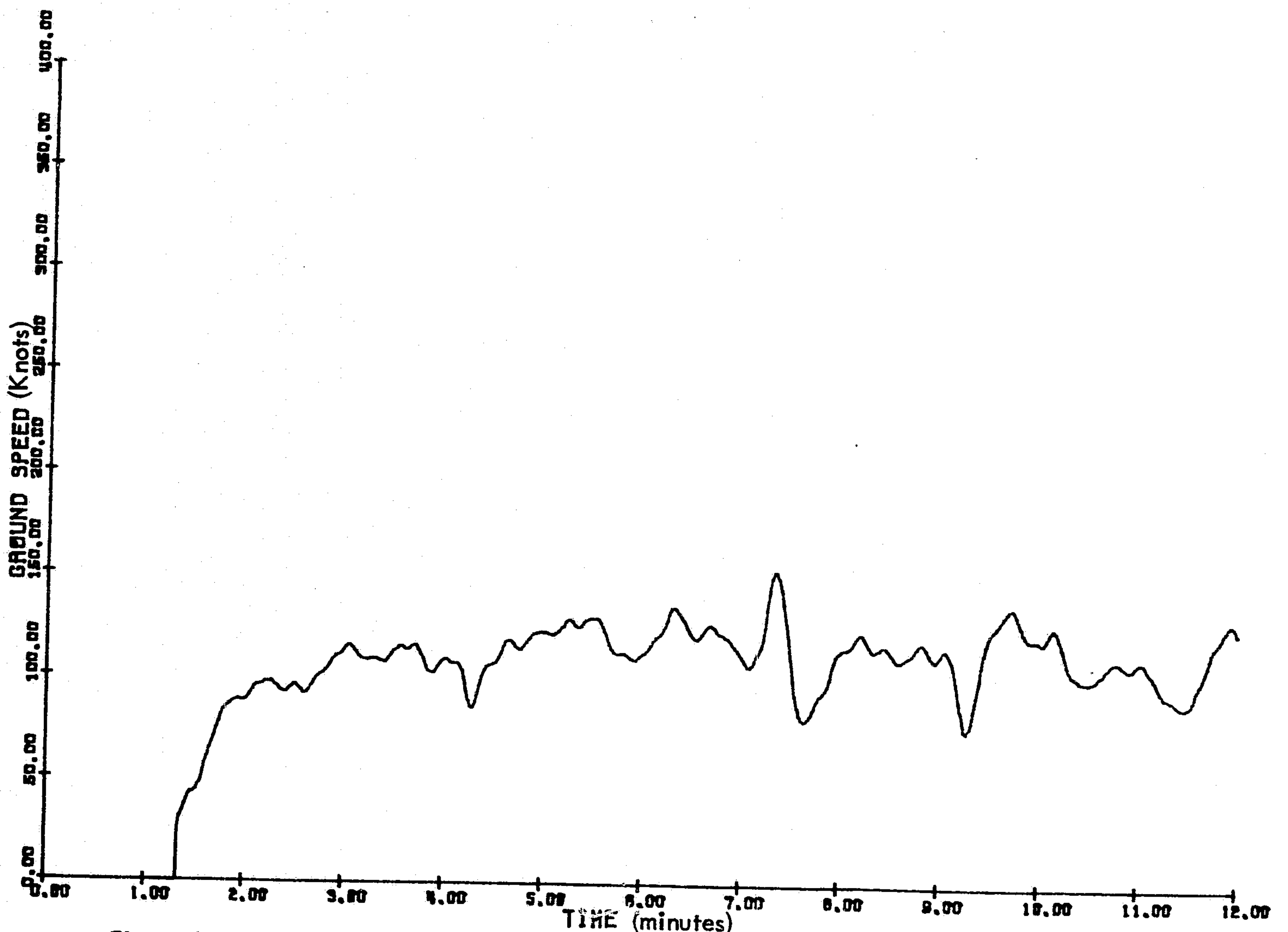


Figure 6-5. Ground Speed (Knots) – Time (minutes). Result of Flight Test 1, $\alpha - \beta$ filter (t_f = 12 seconds, α=0.084, β=0.0017) in Ground Speed Calculation.

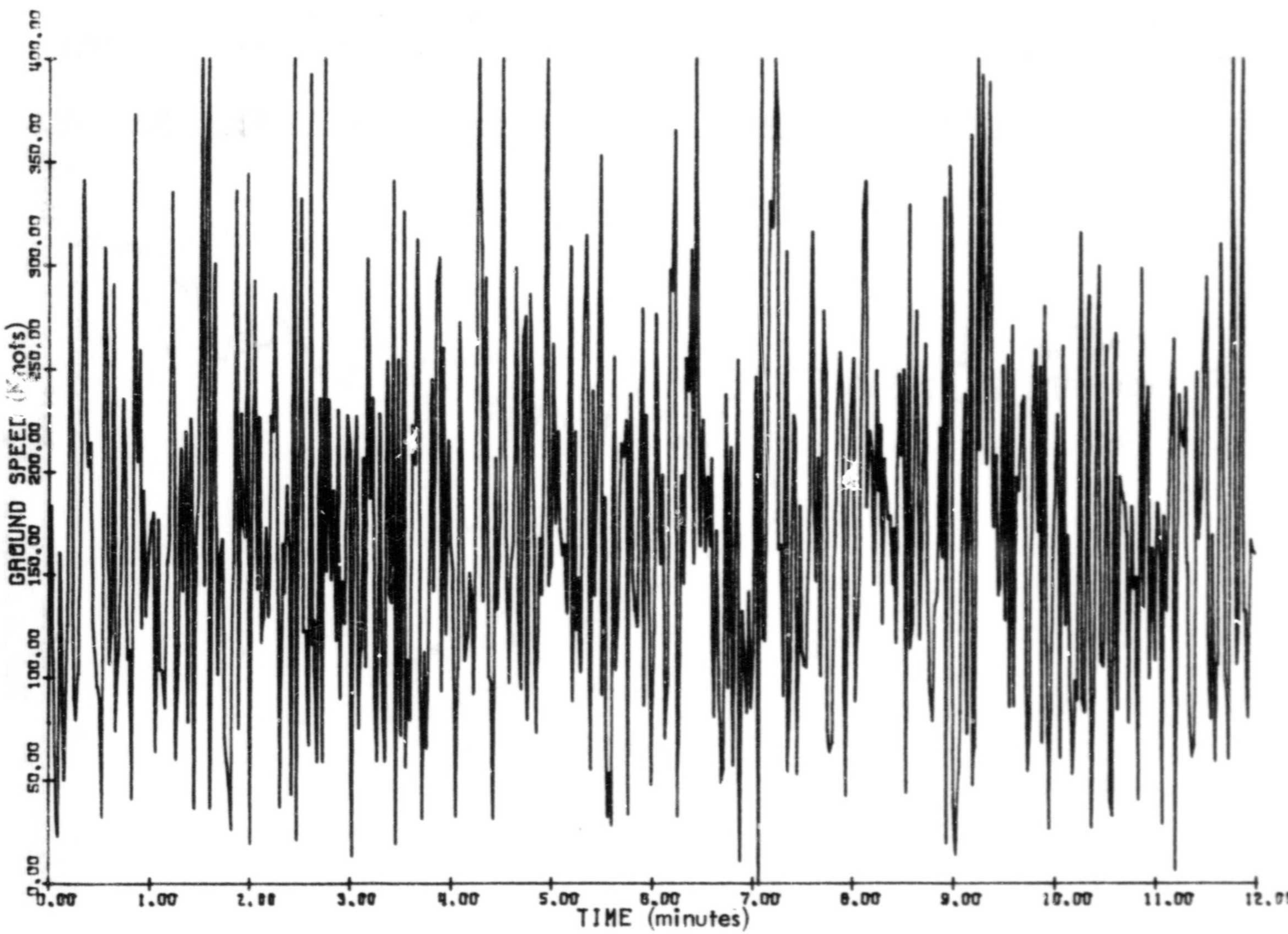


Figure 6-6. Ground Speed (Knots) - Time (minutes), Flight Test 1 Fortran Simulation of Ground Speed Using Unfiltered TDs, Nonfiltered TDs, No Filter in Ground Speed Calculation.

ORIGINAL PAGE IS
OF POOR QUALITY

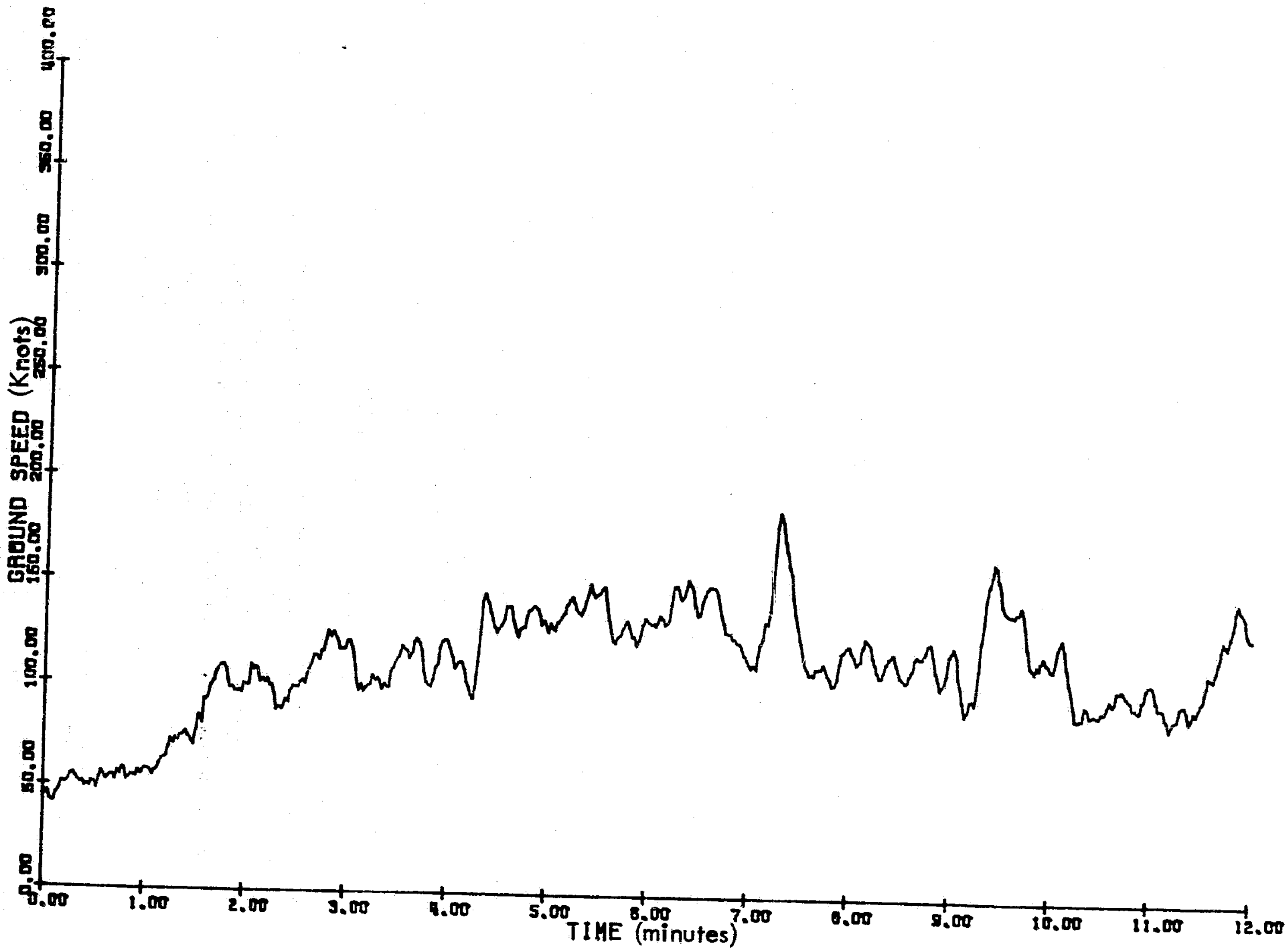


Figure 6-7. Ground Speed (Knots) - Time (minutes), Flight Test 1 Fortran Simulation of Ground Speed Using Nonfiltered TDs, $\alpha-\beta$ Filter (t_f=12 seconds, α=0.084, β=0.0017) in Ground Speed Calculation.

ORIGINAL PAGE IS
OF POOR QUALITY

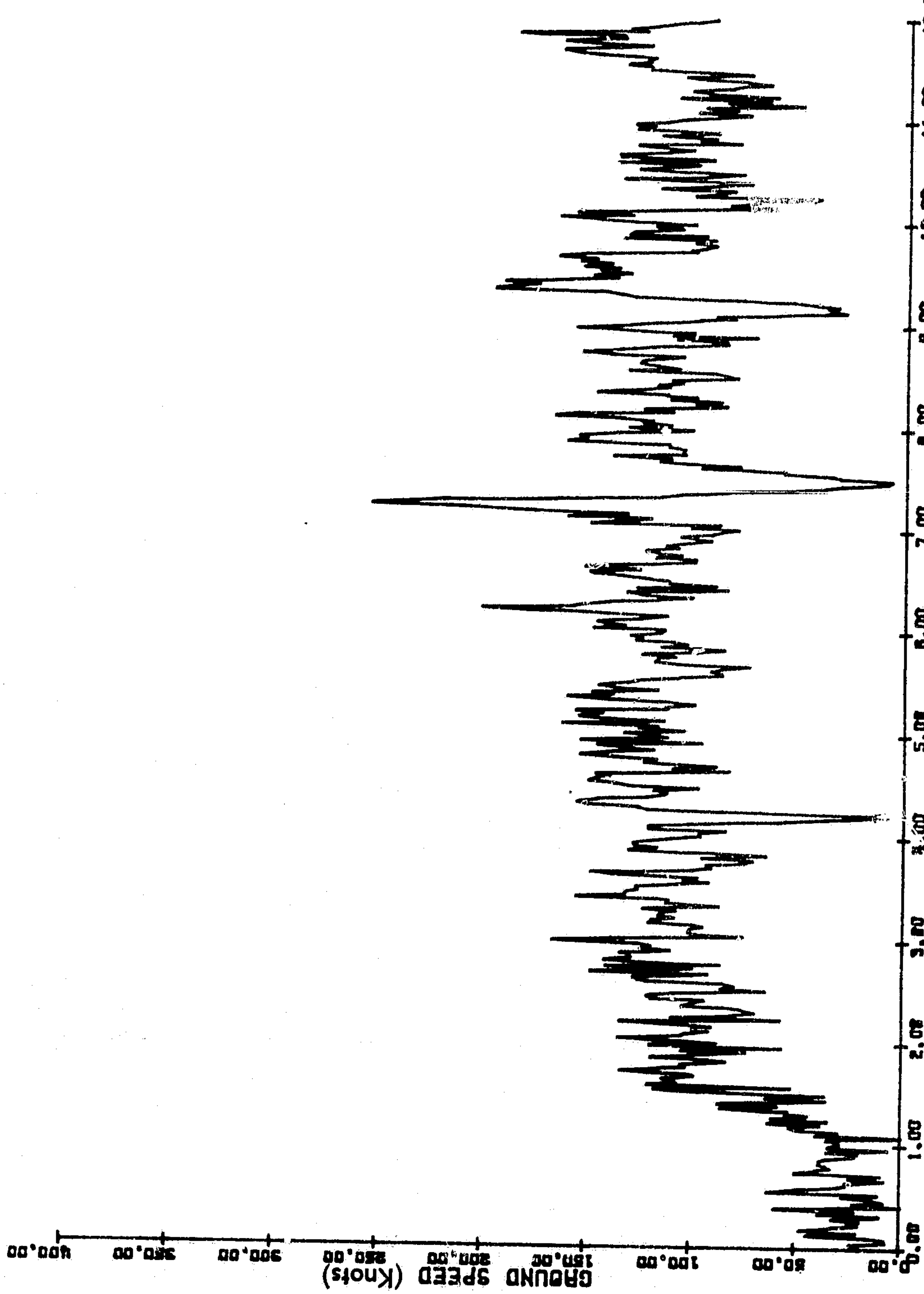


Figure 6-8. Ground Speed (Knots) - Time (minutes), Flight Test 1 Fortran Simulation of Ground Speed Using Filtered TDs, No Filter in Ground Speed Calculation.

ORIGINAL PAGE IS
OF POOR QUALITY

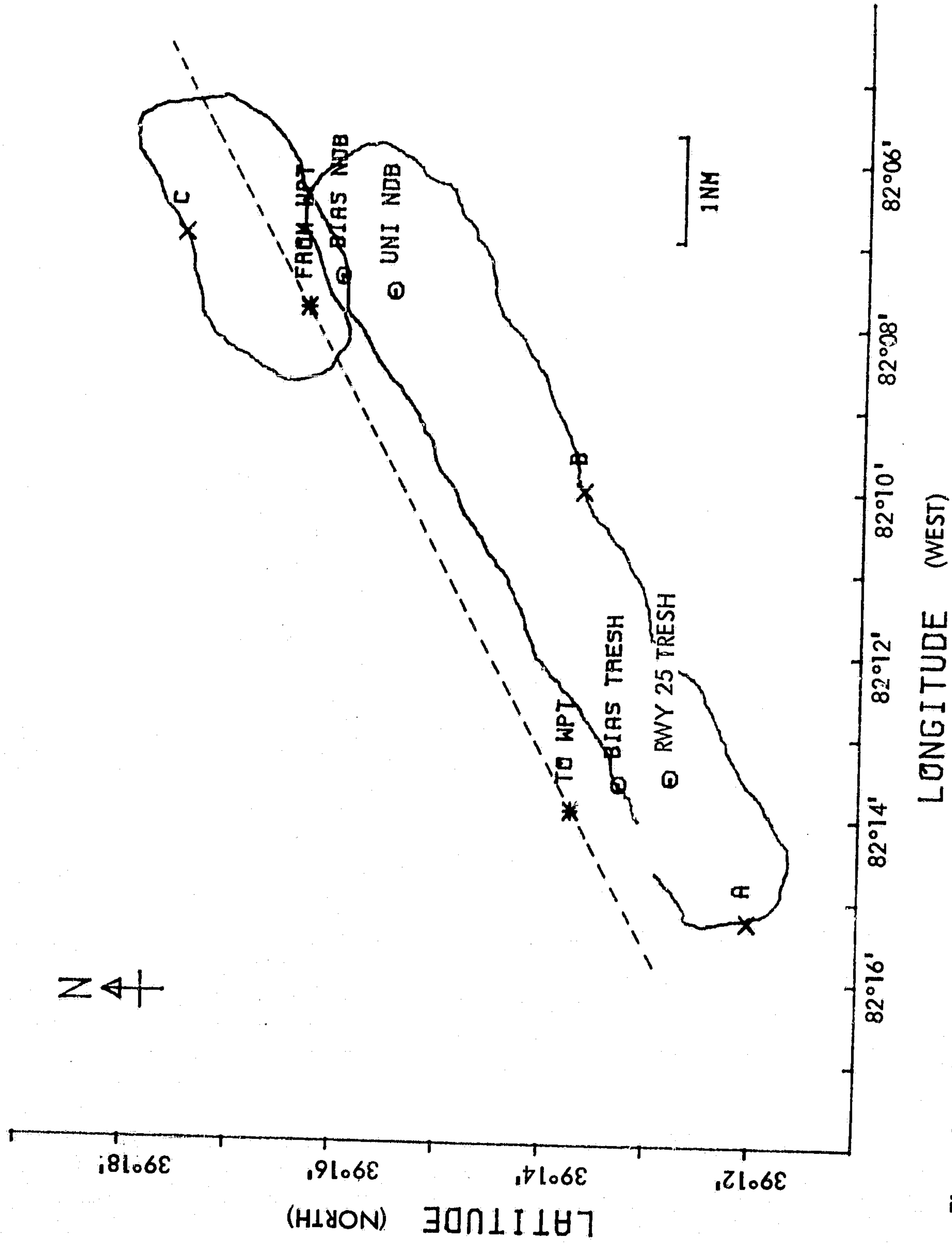


Figure 6-9. Flight Path Plot. Result of Flight Test 2, α-β Filter (t_f=6 seconds, α=0.167, β=0.007) on TDs.

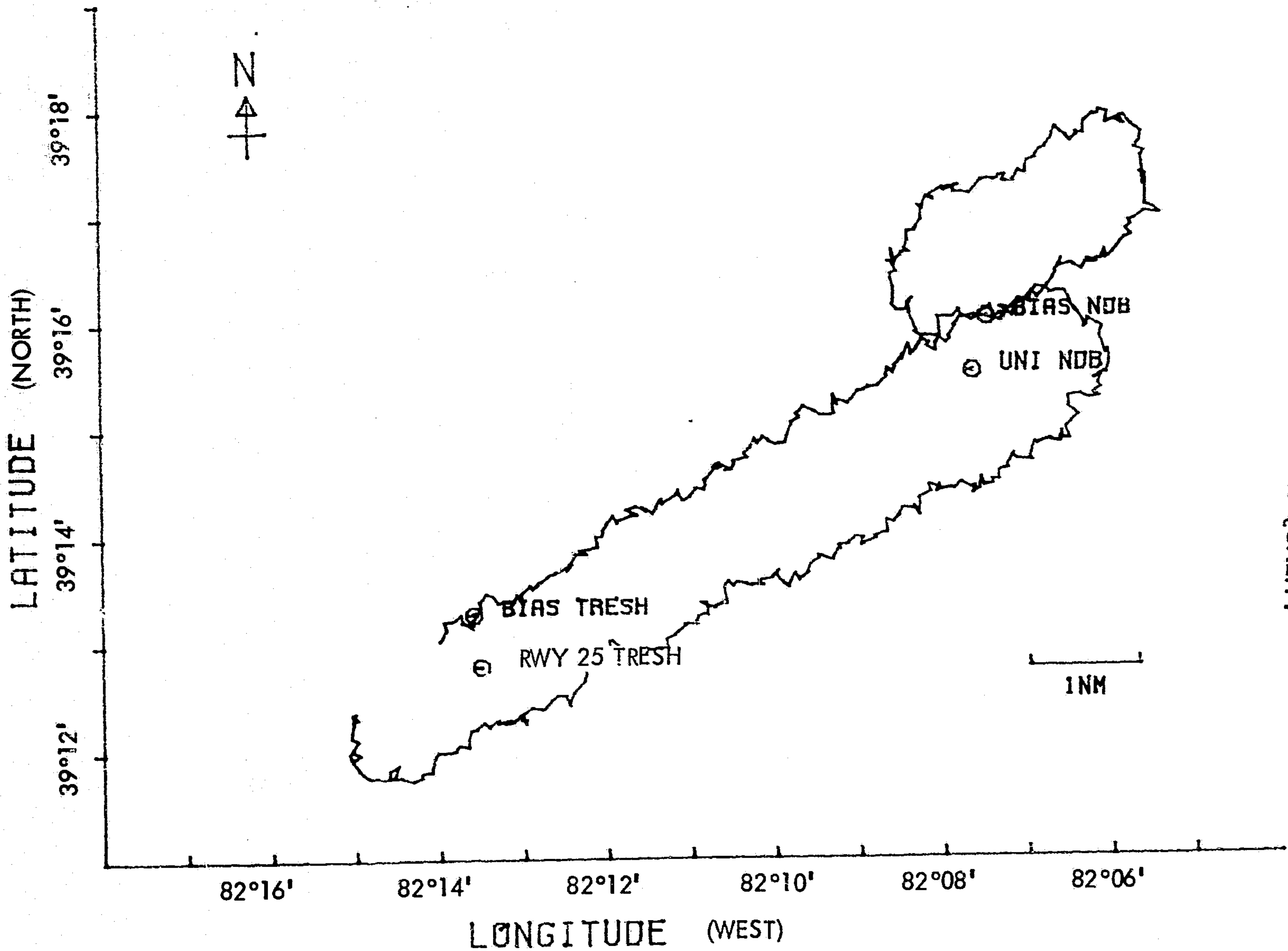


Figure 6-10. Flight Path Plot. Fortran Simulation of Flight Test 2 Using Nonfiltered TDs.

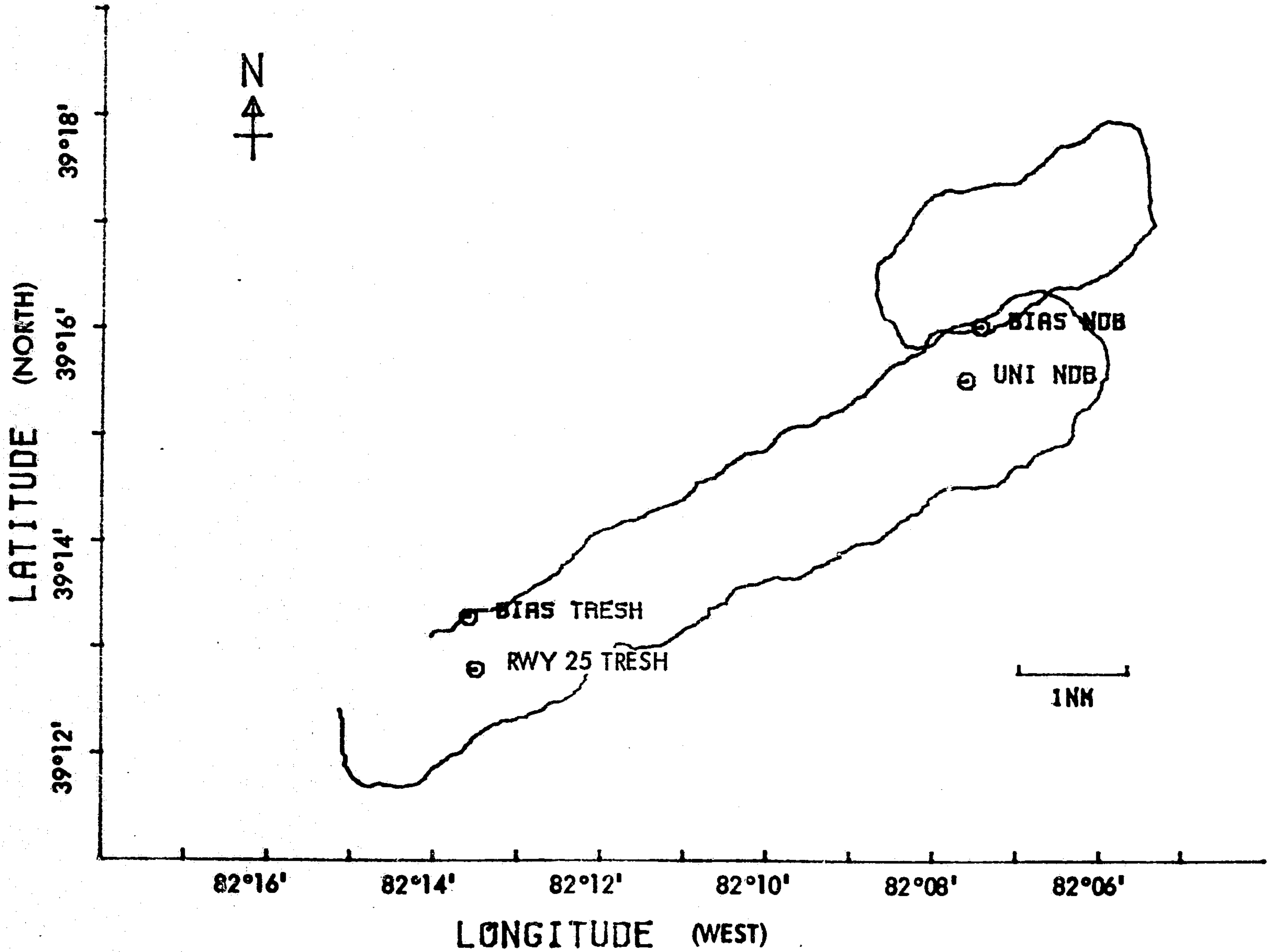


Figure 6-11. Flight Path Plot. Fortran Simulation of Flight Test 2 Using Filtered TDs $\alpha-\beta$ Filter ($t_f = 4$ seconds, $\alpha = 0.251$, $\beta = 0.016$) on TDs.

Figures 6-5 through 6-8 plot the ground speed of flight test 1. Figure 6-5 is the result of flight test 1 and figure 6-6 is the simulated data using nonfiltered TDs and nonfiltered GS. In figure 6-6, the ground speed indicated 400 knots when the actual speed did not exceed 150nm/hr. In the comparison between these two figures, the significant effect of the two filters is shown. Figure 6-7 indicates the results of filtering ground speed alone and figure 6-8 indicates filtering of TDs alone, but the best result was obtained by a combination of ground speed and TD filtering as indicated by figure 6-5.

Figures 6-12 and 6-13 plot the ground speed of flight test 2. Although the TDs have less random noise than the TDs of flight test 1, nonfiltered ground speed is not acceptable for RNAV information.

3. Range/bearing, CTE/CTEB and CDI indications. The accuracy of the range/bearing calculation and the CTE/CTEB calculation was evaluated by simulation; however, relative accuracy was tested during the flight tests.

Figures 6-14 and 6-15 are plotted using the result of flight test 2, and figures 6-16 and 6-17 are plots made by the Fortran simulation. According to these four figures, relative accuracy of the range/bearing are adequate. Figures 6-18 and 6-19 plot CTE and CTEB during the flight, and they are accurate with respect to the desired course.

However, the CDI indicator produced confusion for the pilot because the indicator showed the wrong direction to the course whenever the airplane was moving with an angle which was more than 90° to the desired course bearing angle (241.19° for flight test-2). A CDI indicator using a VOR station also causes a similar problem.

In order to indicate the desired CDI indication, it is necessary to determine the flight path vector. Since the Loran-C provides true position information, the receiver has the capability to compute a flight path vector for providing the correct CDI indication for any airplane position. A software routine for computation of a flight path vector and the angle difference between the vector angle and desired course angle is added to the CTE/CTEB calculation routine.

The corrected CTE and right/left indication of flight test 2 was plotted in figure 6-20 using the Fortran simulation. The corrected CTE/CTEB routine in the RNAV program was tested using the Loran-C simulator.

Figure 6-21 is a photograph of the Ohio University Loran-C receiver inside the Piper Cherokee for the flight test.

ORIGINAL PAGE IS
OF POOR QUALITY

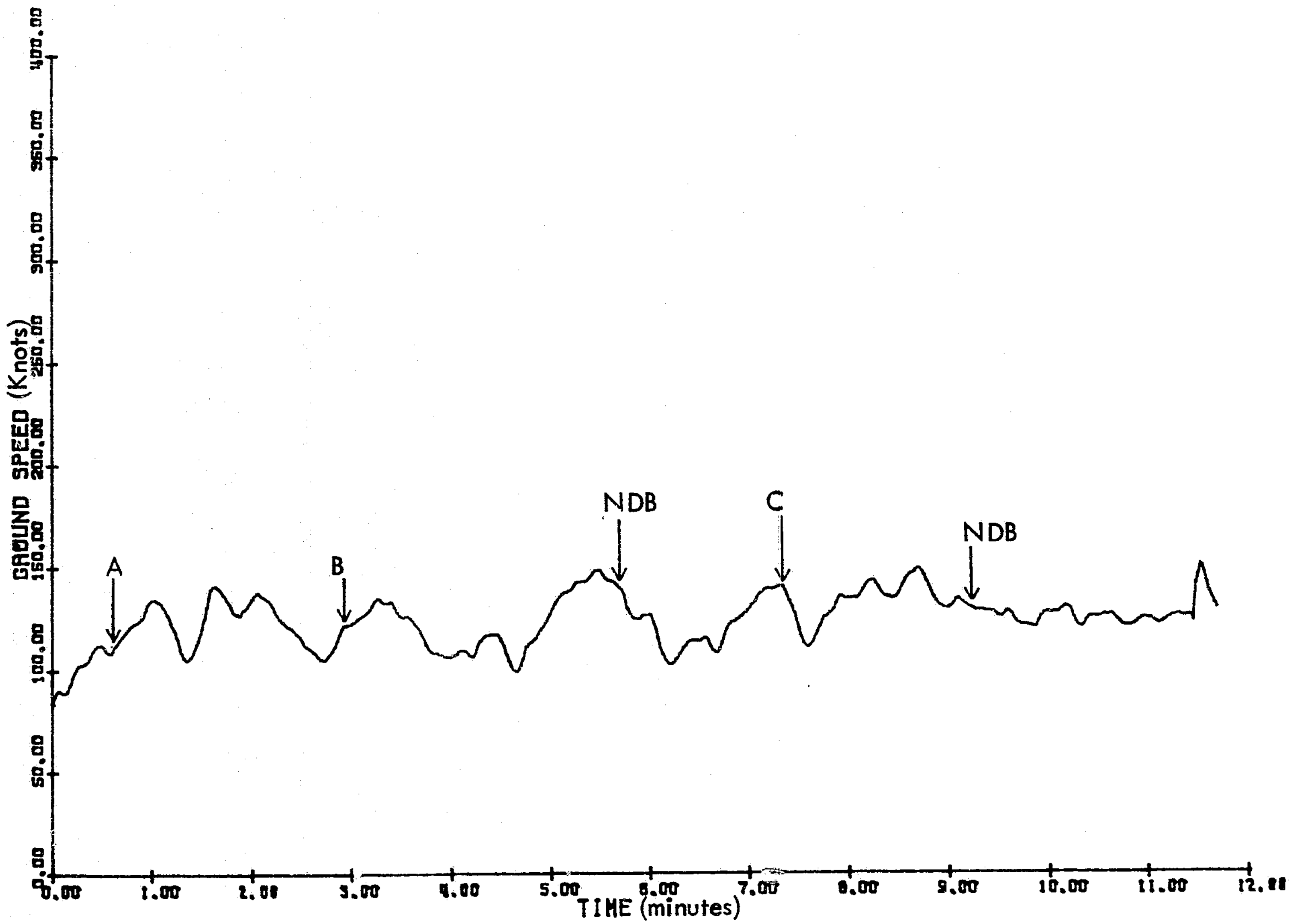


Figure 6-12. Ground Speed (Knots) - Time (minutes), Result of Flight Test 2, $\alpha - \beta$ Filter ($t_f = 12$ seconds, $\alpha = 0.084$, $\beta = 0.0017$) in Ground Speed Calculation.

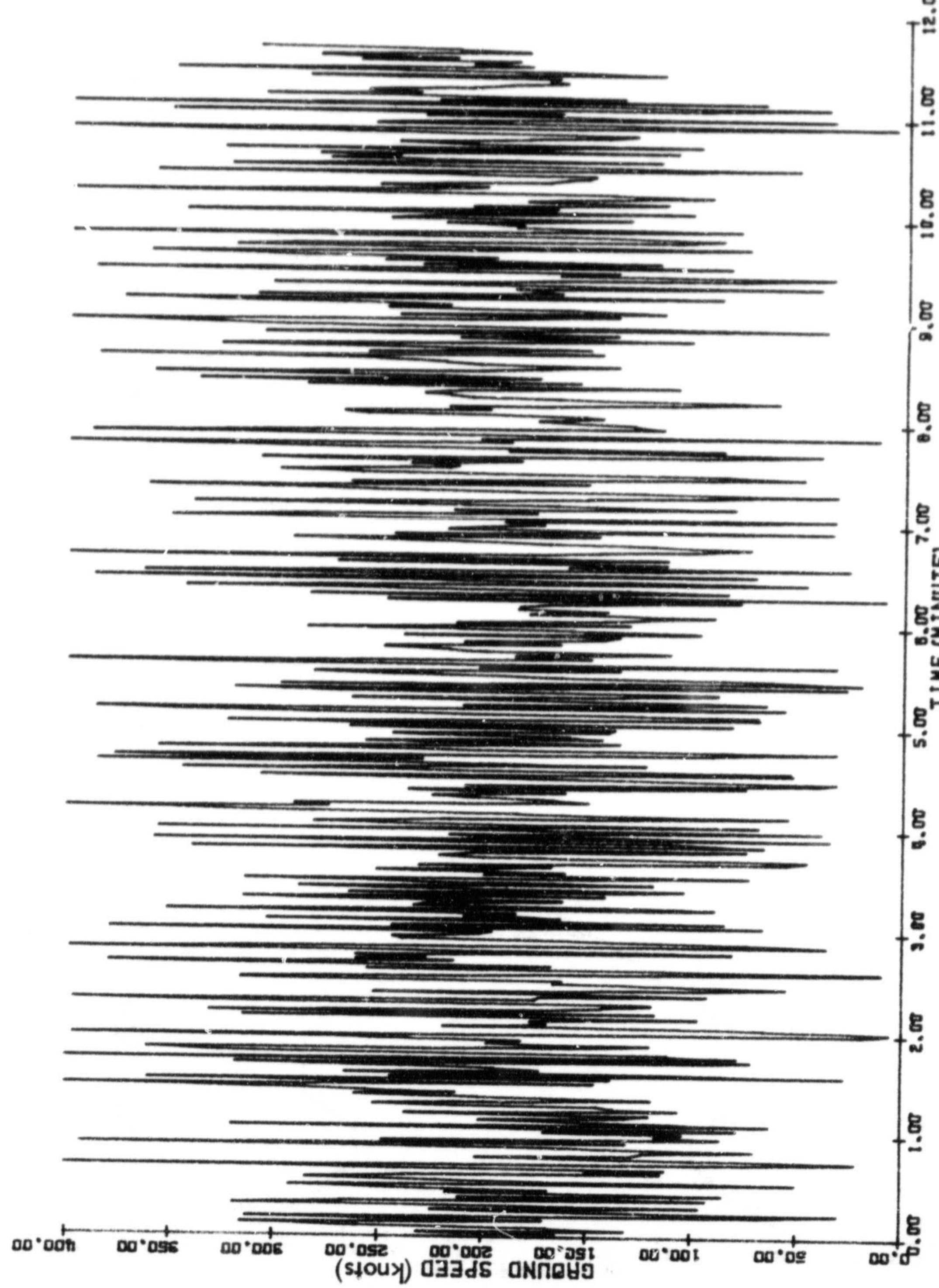


Figure 6-13. Ground Speed (Knots) - Time (minute), Flight Test 2
Fortran Simulation of Ground Speed Using Nonfiltered TDs
No Filter in Ground Speed Calculation.

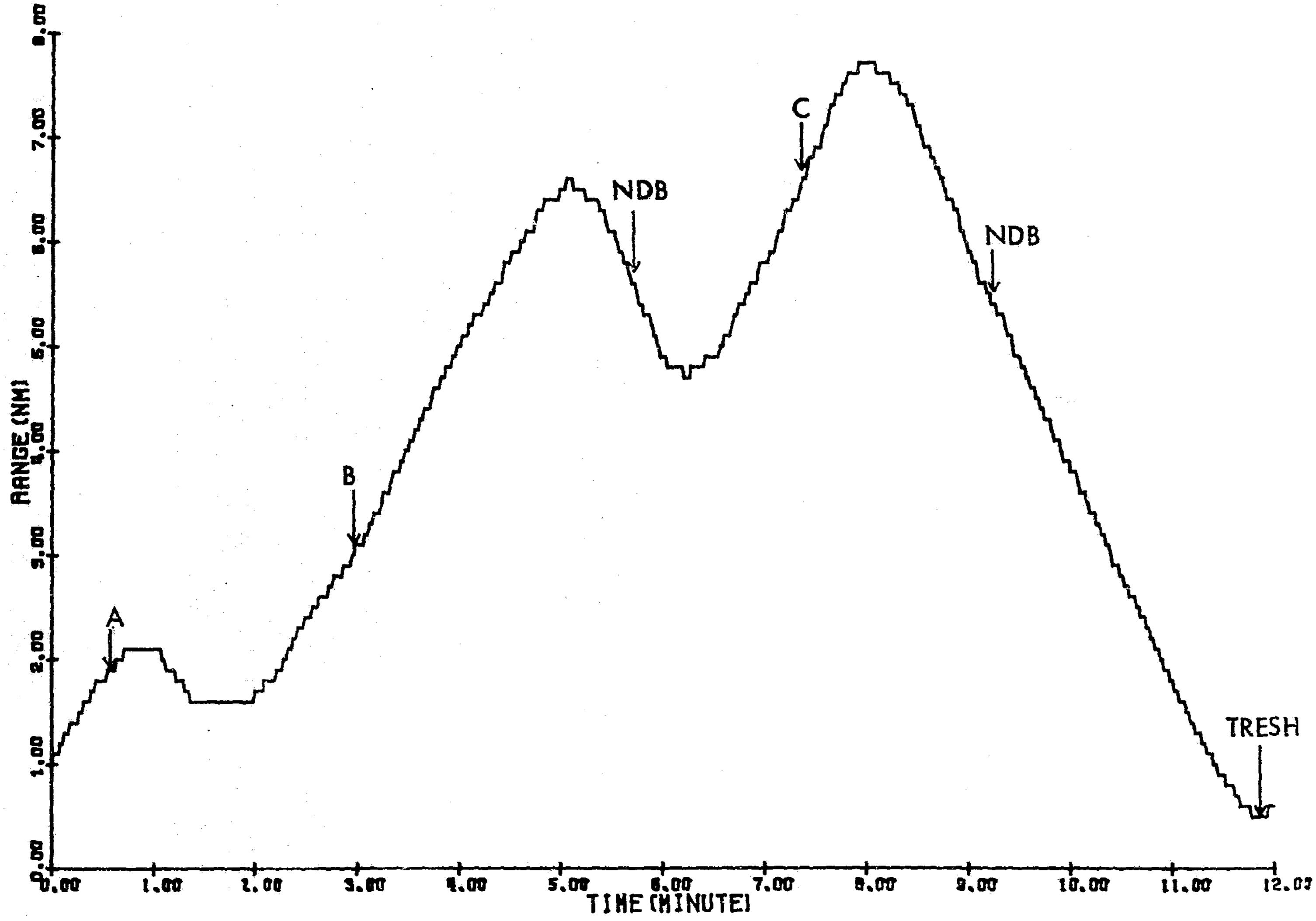


Figure 6-14. Range (NM) - Time (Minute), Result of Flight Test 2.

ORIGINAL PAGE IS
OF POOR QUALITY

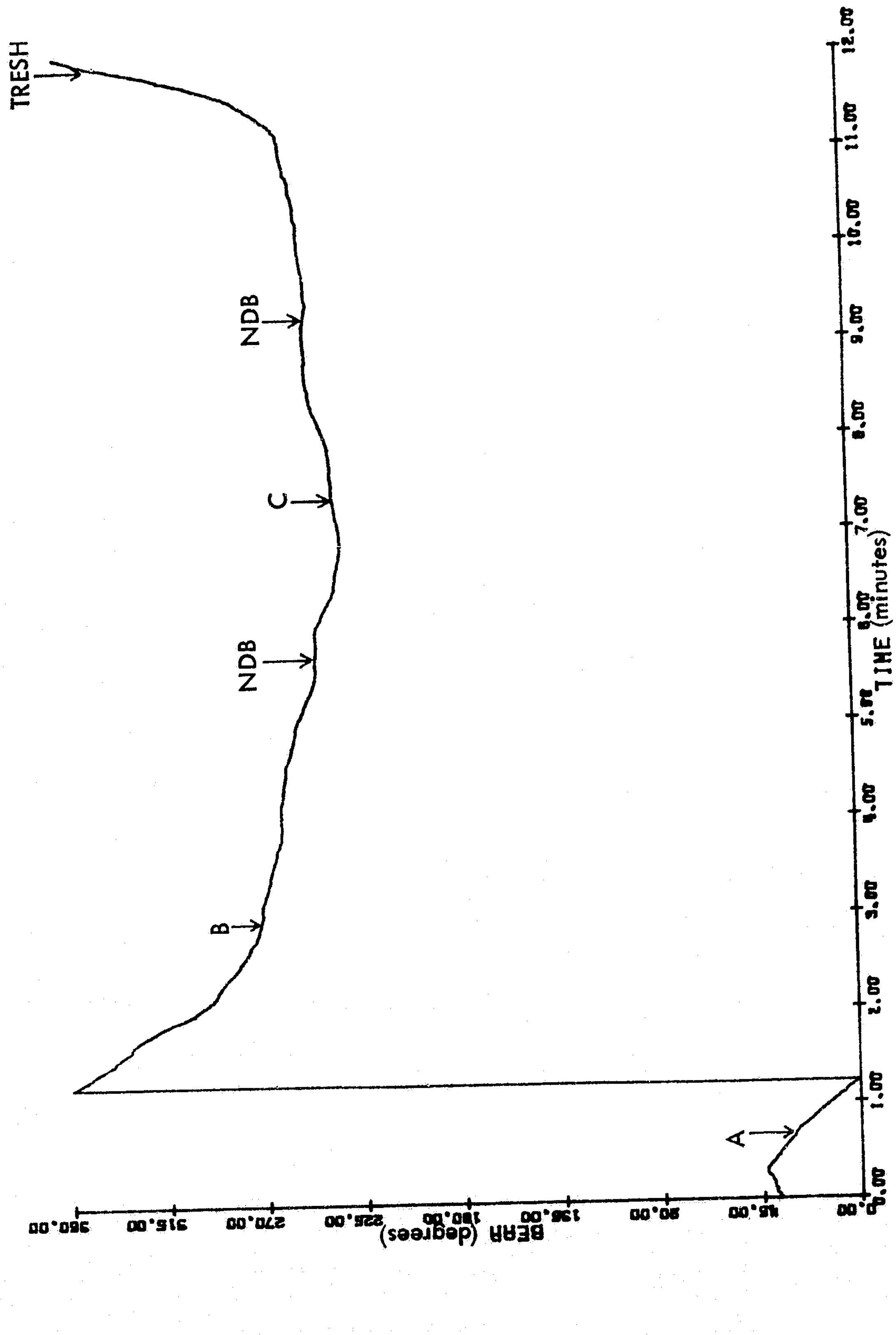


Figure 6-15. Bearing Angle (degrees) - Time (minutes), Result of Flight Test 2.

ORIGINAL PAGE IS
OF POOR QUALITY

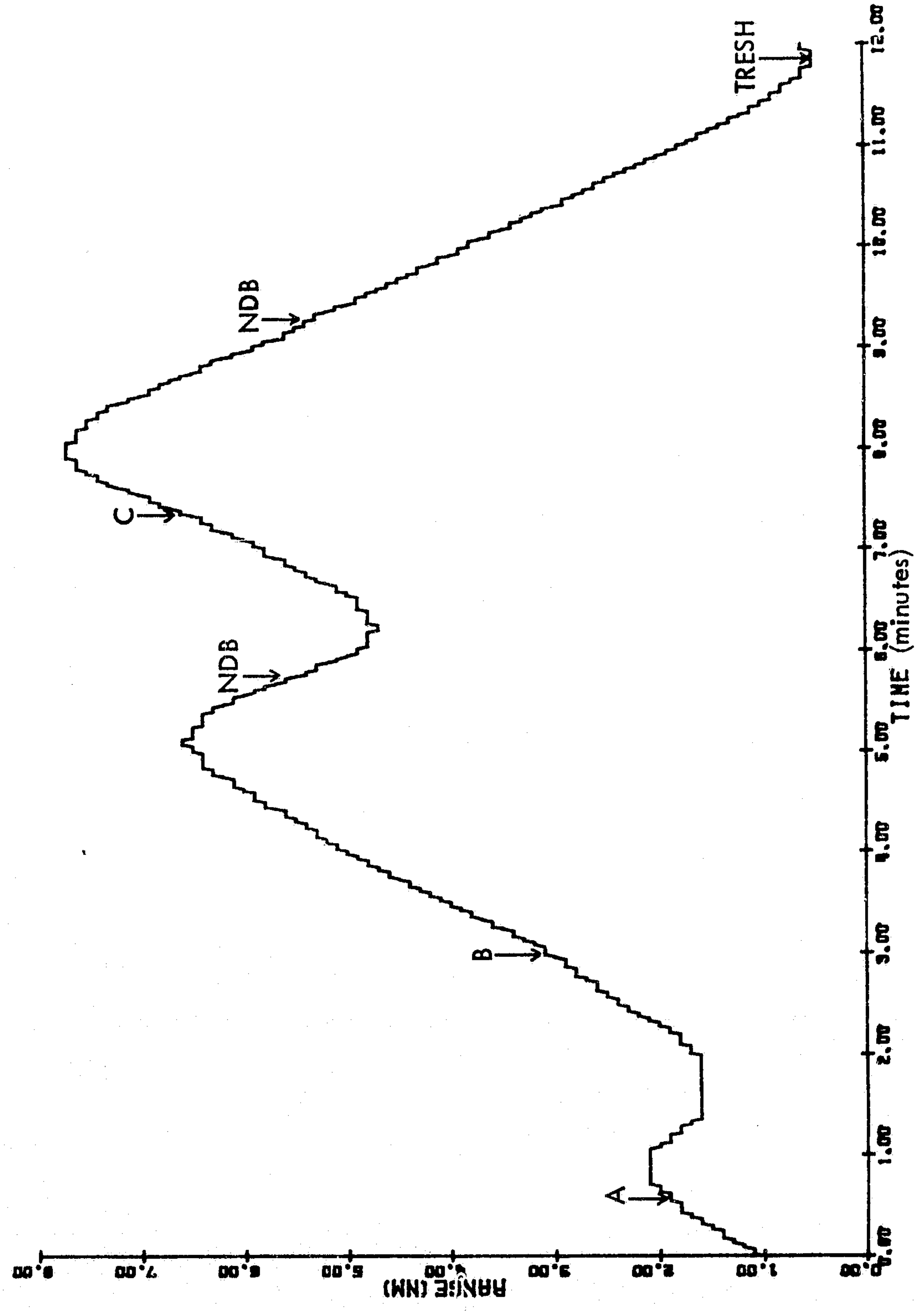


Figure 6-16. Range(NM) - Time (Minutes), Fortran Simulation of Flight Test 2.

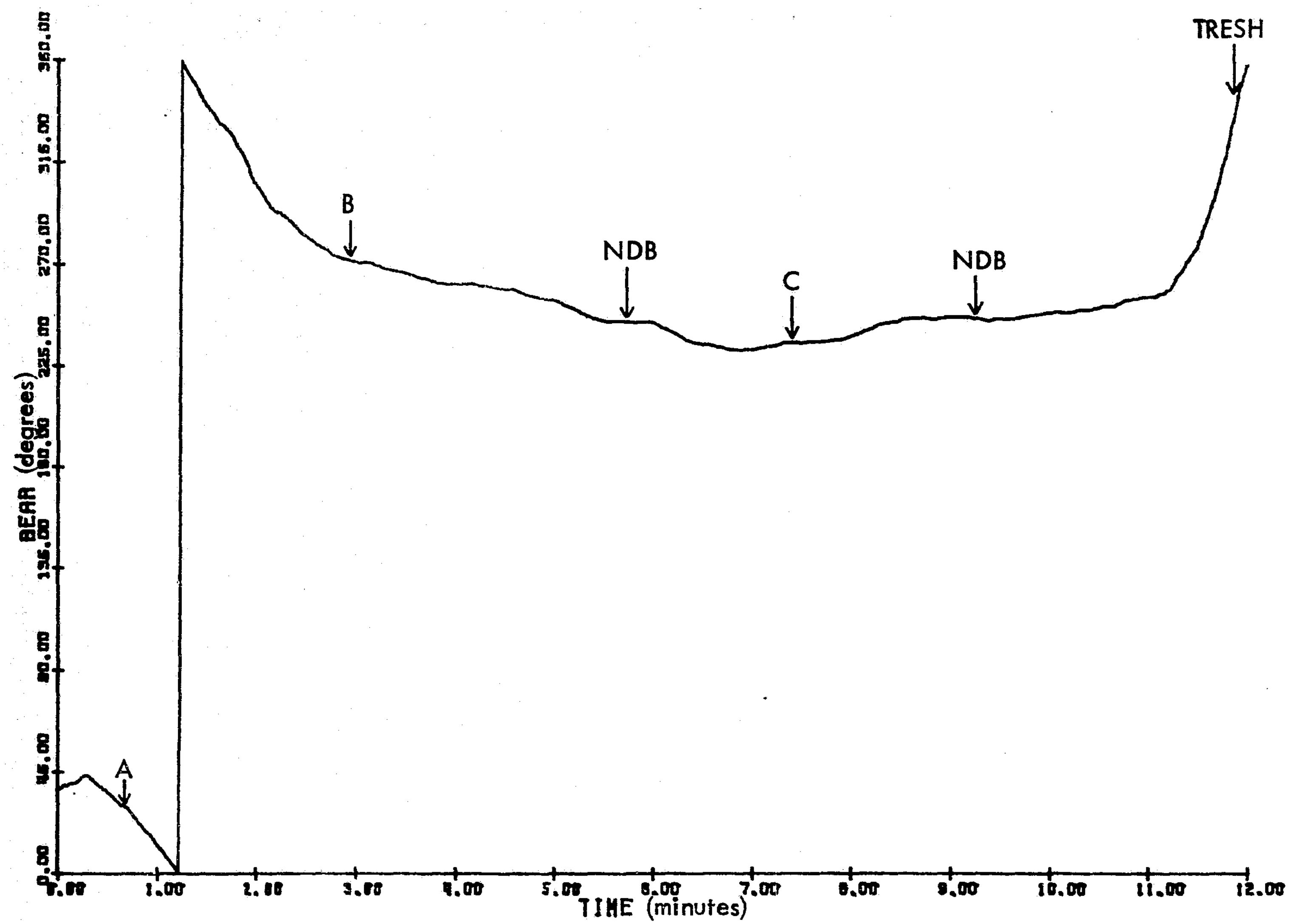


Figure 6-17. Bearing Angle (degrees) - Time (minutes), Fortran Simulation of Flight Test 2.

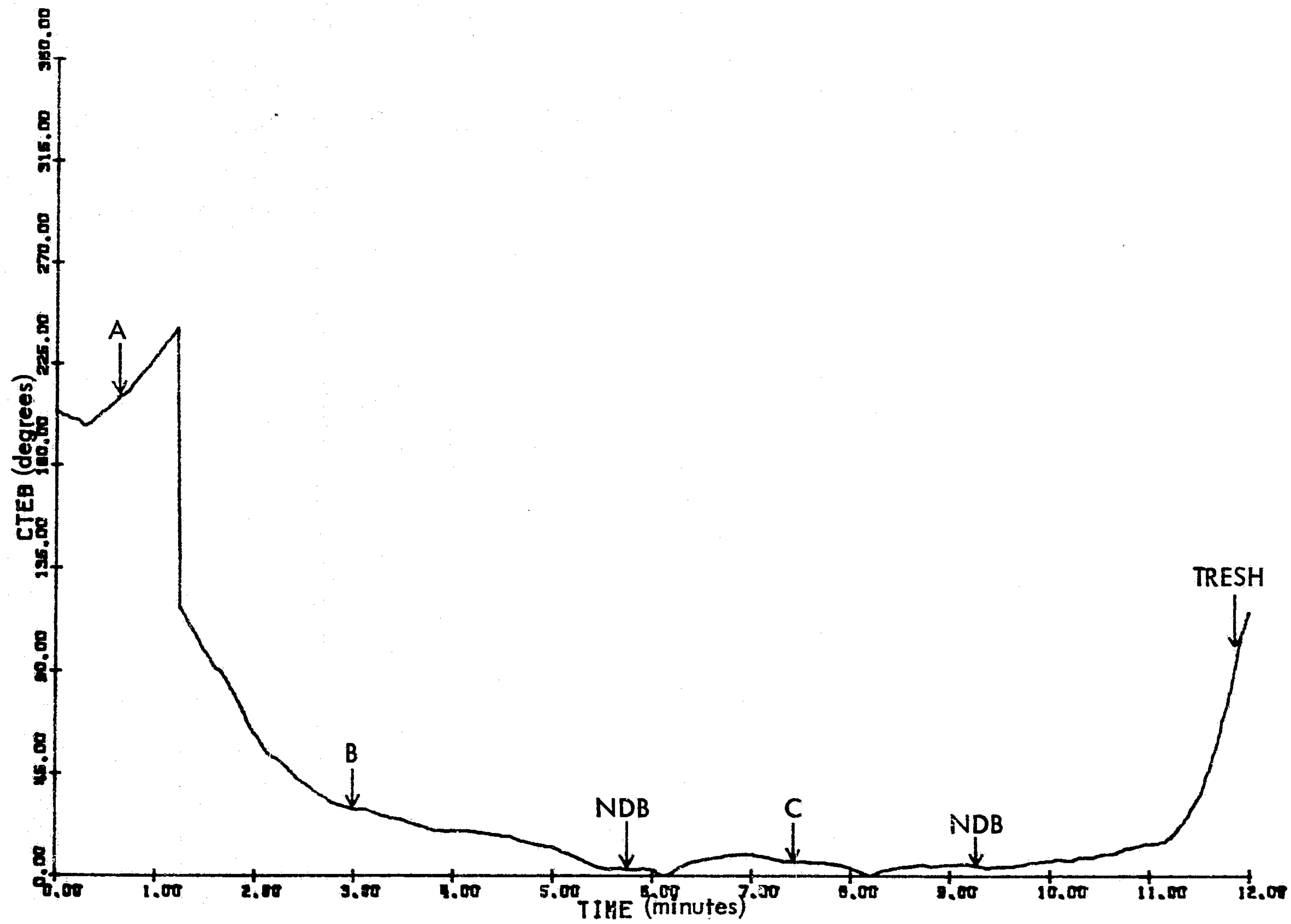


Figure 6-18. Cross-Track Error Bearing (degrees) - Time (minutes), Result of Flight Test 2.

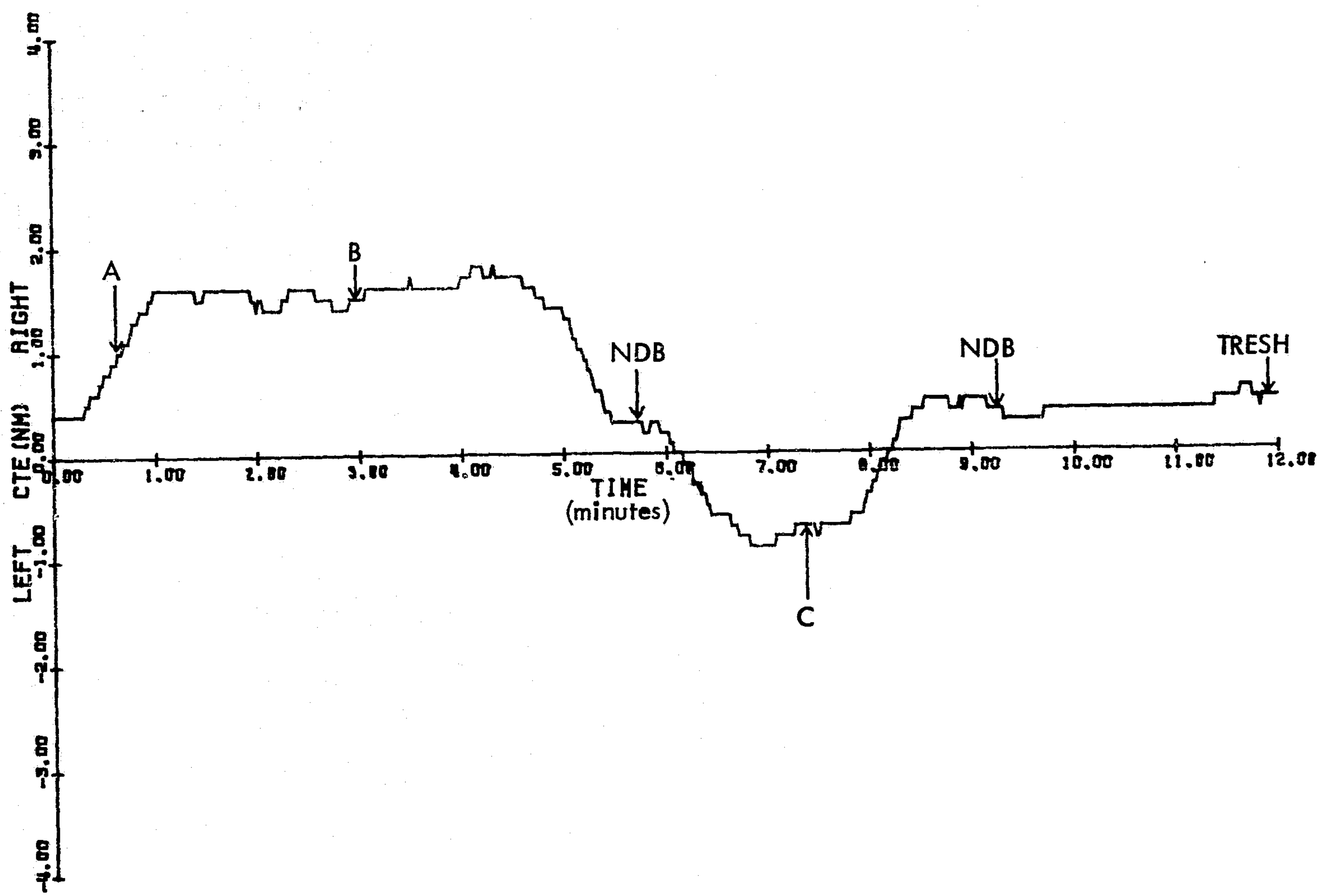


Figure 6-19. Cross-Track Error (NM) - Time(minutes), Result of Flight Test 2.

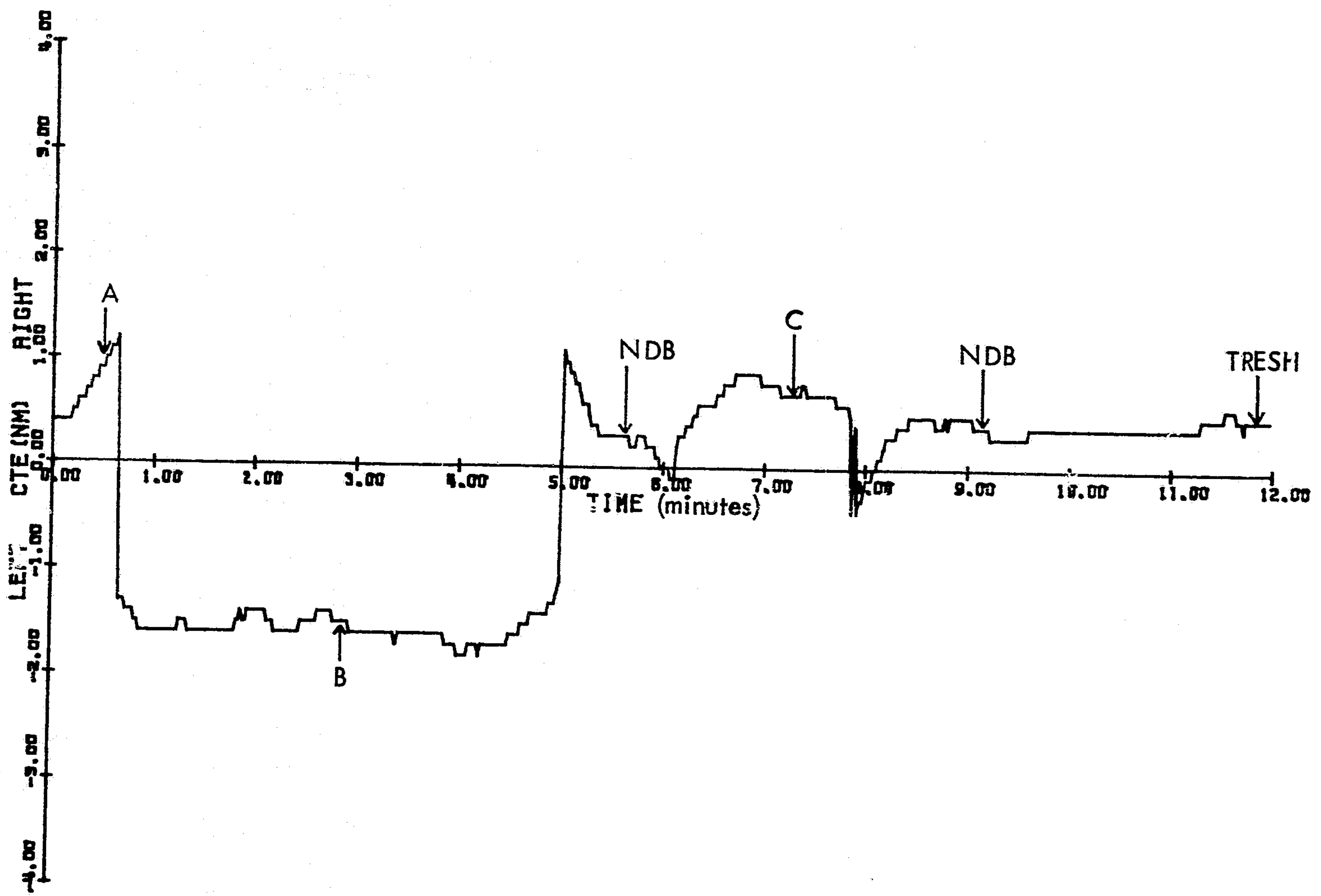


Figure 6-20. Cross-Track Error (NM) - Time (Minutes)
Fortran Simulation of Flight Test 2, Right/Left
Off-course Indication is corrected.

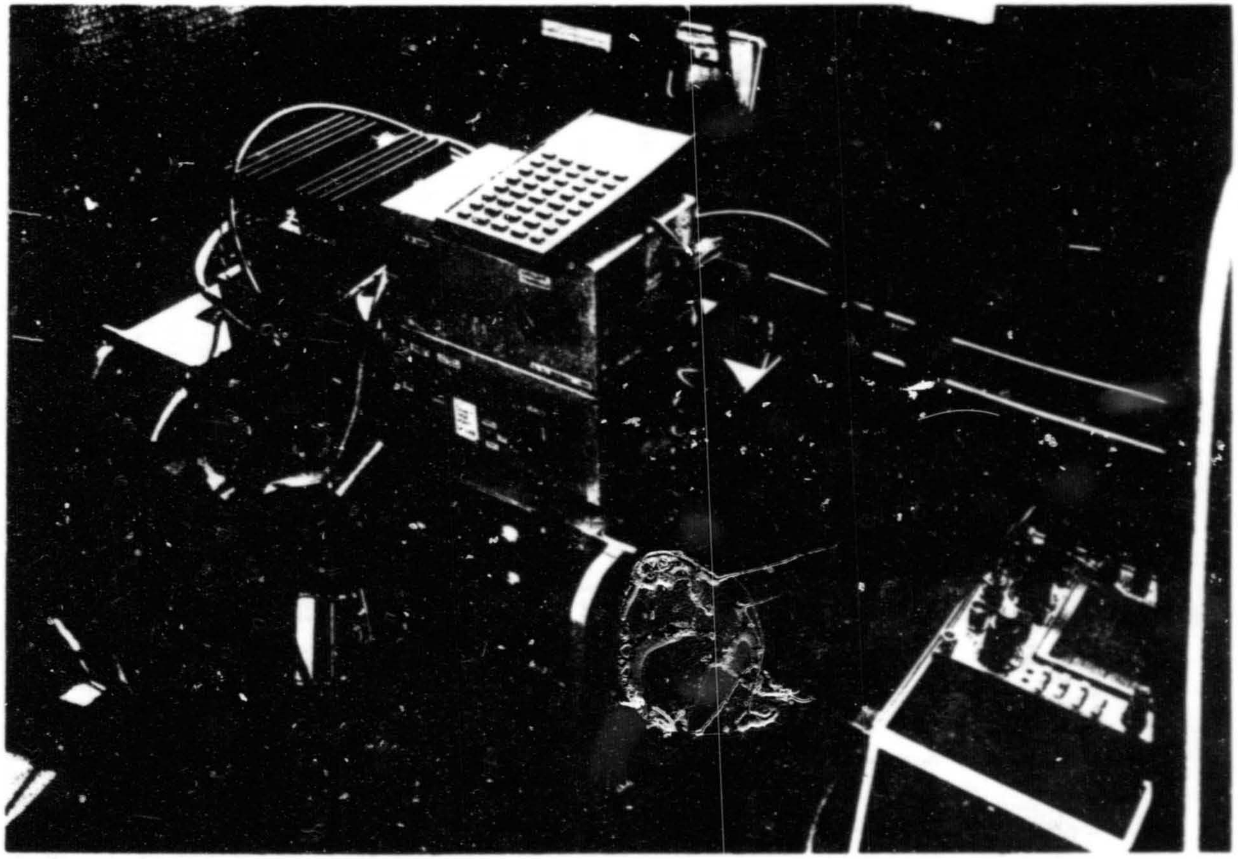


Figure 6-21. Photograph of Ohio University's Loran-C Receiver Inside the Piper Cherokee During Flight Testing.

VII. CONCLUSIONS AND RECOMMENDATIONS

Some specific conclusions can be reached as a result of the work performed in developing a microcomputer-based Loran-C receiver for general aviation application.

The objective of this area navigation software implementation is to provide high quality air navigation information by using Loran-C as a navigation system for general aviation. The following conclusions are made according to the test results in Chapter VI.

The conclusions are:

1. The high accuracy of the range/bearing calculation using the microcomputer-based Loran-C receiver was demonstrated; the error without a bias error does not exceed more than 0.012nm (range) or 0.09° (bearing) for ranges to 530nm.

2. Operational performance, as observed on a flight in a general aviation aircraft, is obtained using a α-β filter on time differences to reduce random noise. Filtering TDs with the new, stable clock, with an effective time constant is 4 seconds, effectively smooths the flight path and does not cause serious delay on the turns.

3. The ground speed calculation with 10 knots resolution has operational stability for a constant or low-acceleration flight. Since the ground speed calculation process passes through two filters, the ground speed cannot be easily updated with high acceleration. According to the effective time constants for the two filters (4 seconds for TDs and 12 seconds for the GS calculation), a step response becomes 86.3% of final value after 24 seconds. So the ground speed calculation can accept an acceleration which is less than $0.13 nm/s^2$.

4. The CTE/CTEB indication provides the relative position and proper direction respectively to any desired course inside the Loran-C coverage area. Even with an airplane very close to a To waypoint (less than 0.1nm) the CTE has sufficent sensitivity to calculate an accurate CDI indication, while the VOR navigation system at close range is too sensitive.

5. An execution time of the total navigation system routine does not exceed more than 1.5 seconds, which is short enough for adequate position update information for air navigation. In the northeast U.S. chain (GRI=99600μs), the execution time is about 1.39 seconds for RNAV position updates.

6. The Loran-C navigation system with the new stable clock recorded an average bias error of 0.5nm which meets the requirement stated in AC90-45A (enroute 2.5nm, terminal 1.5nm). Even for an approach, this system has the capability to meet the total error of 0.6nm stated in AC90-45A.

7. The Loran-C area navigation software makes it possible for the general aviation user to fly to any point inside the Loran-C coverage area in true area navigation fashion unlike VOR navigation system with its line-of-sight and range restrictions.

Some problems were identified during the testing, and these should be addressed and solved prior to implementation by general aviation.

1. The bias error to the north is due to signal-strength differences of Loran-C stations and Avionics Engineering Center's receiver implementation. The bias error can be significantly reduced with a new RF front end [42] and applying propagation corrections. The recent tests with the new RF front end indicated a bias error of 0.2nm. These data were collected in the same area as the previous flight tests. The bias error of 0.2nm could be further reduced with the application of a propagation correction.

2. An improved ground speed response for accelerated flight. The ground speed response for accelerated flight can be improved by implementing a three dimensional filter [43]; however, improvement of measuring time differences to reduce random noise should be made to provide better data for ground speed calculations.

Contemporary microprocessor technology has greatly improved the capability for quality high navigation, and allows for achieving low-cost and light weight receivers for general aviation applications. This RNAV software promises to provide the pilot with significant operational advantages through the use of a microcomputer-based Loran-C receiver.

C-2

VIII. ACKNOWLEDGEMENTS

The work reported in this paper was supported by the NASA Langley Research Center under grant NGR 36-009-017 to Ohio University. The author gratefully acknowledges the following people who aided with the research reported in this paper: Dr. Robert W. Lilley, associate director, Dr. Kent A. Chamberlin, Mr. Jim D. Nickum, research engineer, and student researchers Joseph P. Fischer, Daryl L. McCall, Steven R. Yost and Stanley Novacki, III. Special gratitude is due to Dr. Richard H. McFarland, director of the Avionics Engineering Center, who served as advisor for this paper. My appreciation is also extended to Mrs. Shirley C. Mellema for the production of this paper.

Also, the author gratefully acknowledges the Chubu Institute of Technology for giving her the opportunity of continuing her graduate study at Ohio University as an exchange student through the Kohei Miura Graduate Associateship.

Finally, the author wishes to express her sincere appreciation to her parents, for their support and encouragement.

IX. REFERENCES

[1] Frank, Robert L., "History of Loran-C," Navigation, Vol.29, No.1, Spring 1982, pp.1-5.

[2] Kayton, Myron and Walter R. Fried, "Avionics Navigation Systems," John Willey and Sons, Inc., New York, NY, 1969. pp.192-193.

[3] U. S. Department of Transportation/Federal Aviation Administration, "Summary of the FAA's Future Navigation System Mix Evaluation (Through May 1982)," Report No.DOT/FAA-EM-82-24, August, 1982, pp.4.1 - 4.9.

[4] Op cit., Kayton and Fried, pp.163-170.

[5] Ibid., Kayton and Fried, pp.181-192.

[6] "Advisory Circular," Department of Transportation/Federal Aviation Administration, AC 90-45A, February, 1975.

[7] "Private Pilot Manual," Jeppesen Sanderson, Inc., Denver, Colorado,1977, pp.8.1 - 8.18.

[8] Op cit., U. S. Department of Transportation/FAA, pp.4.18-4.20.

[9] Op cit., Kayton and Fried, pp.281-341.

[10] Milliken R. J. and C. J. Zoller,"Principle of Operation of NAVSTAR and System Characteristics," Global Position System, The Institute of Navigation, Washington D.C., 1980, pp.3-14.

[11] Kruh, P., Brady, W.F., and Schmitt, D.L., "A Strategy for Buildup to the Operational NAVSTAR GPS Constellation," proceeding of The Institute of Navigation Aerospace Meeting, Washington, D.C., March, 1983.

[12] Op cit., U. S. Department of Transportation/FAA.

[13] Natarajan, Krishnan, "Testing of Loran-C For General Aviation Aircraft," NASA Conference Publication 2176, Proceedings of Joint University Program For Air Transportation Research -1980 Conference, December 11-12, 1980.

[14] Nickum, James D., "The Effects of Precipitation Static and Lightning on the Airborne Reception of LORAN-C," U.S. Department of Transportation/Federal Aviation Administration, Report No. DOT/FAA/RD-82/45-1, April 1980.

[15] Op cit., U. S. Department of Transportation/FAA.

[16] Wong, Gene A., "Analysis of Loran-C System Reliability For Civil Aviation," Proceedings of Thirty-Seventh Annual Meeting, The Institute of Navigation, Annapolis, Maryland, June 1981.

[17] "Loran-C User Handbook," COMDINST MI6562.3, Department of Transportation, U. S. Coast Guard, May 1980, pp.1-4.

[18] Ibid.

[19] Op cit., Kayton and Fried, pp.145-146.

[20] Op cit., U. S. Department of Transportation, U. S. Coast Guard, Appendix F.

[21] Op cit., Kayton and Fried, pp.145-146.

[22] Fischer, Joseph P., "A Microcomputer-Based Position Updating System for General Aviation Utilizing Loran-C", M.S. Thesis, Ohio University, Athens, Ohio, March 1982, p.15.

[23] Samaddar, S. N., "The Theory of Loran-C Ground Wave Propagation - A Review," Navigation, Vol.26, No. 3, 1979.

[24] Op cit., Fischer, p.29.

[25] Op cit., Fischer, p.31-32.

[26] Razin, Sheldon, "Explicit (Noniterative) Loran Solution," Navigation, Vol. 14, No. 3, Fall 1967.

[27] Op cit., Fischer.

[28] Carmichal,R.D., and E.R.Smith, "Plane and Spherical Trigonometry," Gimn and Company, Boston, 1930, pp.163-189.

[29] Op cit., Fischer, p.55.

[30] Op cit., Kayton and Fried, pp.46.

[31] Aeronautical Chart and Information Center, "Geodetic Distance and Azimuth Computations for Lines over 500 Miles," ACIC Technical Report No.80, St.Louis, Mo., December 1959.

[32] Thomas, Paul D., "Mathematical Models for Navigation Systems", U. S. Naval Oceanographic Office, Washington, D.C., 1965.

[33] Op cit., Department of Transportation, U. S. Coast Guard.

[34] Shively A. Curtis, "A Real-Time Simulation for Evaluating a Low-Cost GPS Navigator," Federal Aviation Administration, Report No. FAA-EM-80-3, April, 1980, pp.B-1 - B-6.

[35] Op cit., Fischer.

[36] Lilley, R. W. and D.L. McCall, "A Loran-C Prototype Navigation Receiver For General Aviation," 4th AIAA/IEEE Digital Avionics System Conference, No. 81-2329, St. Louis, Missouri, November, 1981, pp. 614-620.

[37] Ibid.

[38] Zaks, Rodnay, "Programming the 6502," SYBEX Inc., Berkeley, California, 1980, P374-375.

[39] "Am9511A Arithmetic Processor Advanced Micro Devices Advanced MOS/LSI," Advances Micro Devices, Inc., Sunnyvale, California, 1976.

[40] Op cit., Fischer, pp.77-78.

[41] Op cit., Fischer, p.96.

[42] Yost, Stephen R., "RF Front End Interface and AGC Modification," OU/NASA TM 84, Avionics Engineering Center, Ohio University, Athens, Ohio, December 1982.

[43] Fitzgerald, Robert J., "Simple Tracking Filters Position and Velocity Measurement," IEEE Transactions on Aerospace and Electronic System Vol. AES-18, No. 5, September 1982, pp. 531-537.

X. APPENDICES

APPENDIX A. The computation for an area navigation(RNAV) equipment based on the use of VOR/DME.

$$\beta = \theta_A(t) - \theta_e$$

$$\text{Distance} = d_{tg}(t) = \sqrt{\rho^2(t) + \rho^2 e - 2\rho\rho_e \cos\beta(t)}$$

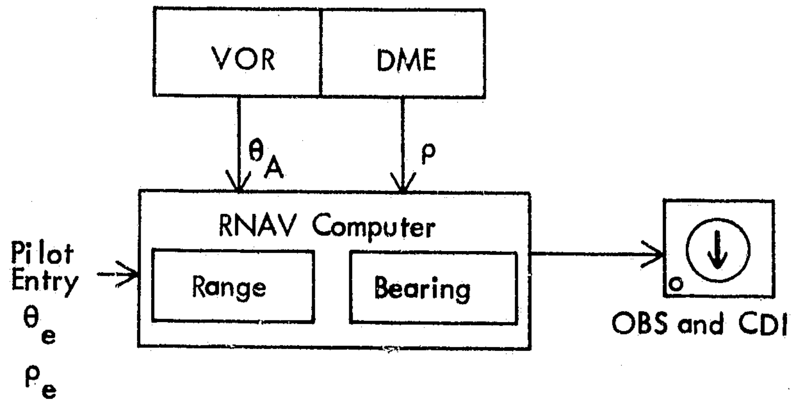
where θ_A = The angle from true north relating to the aircraft

θ_e = The angle from true north relating to the VOR/DME station

$\rho(t)$ = The distance between the VOR/DME station and the aircraft

$\rho(t)$ = The distance between the VOR/DME station and the waypoint

ORIGINAL PAGE IS
OF POOR QUALITY



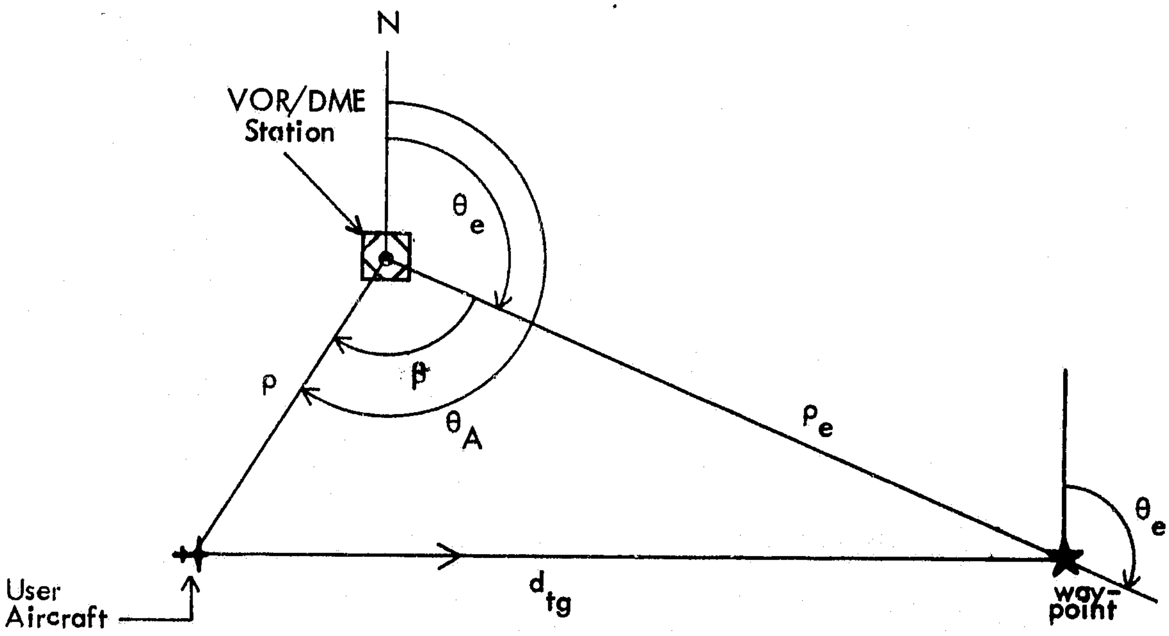


Figure A-1. Area Navigation (RNAV) Equipment.

APPENDIX B. Program listing for testing range and bearing angle computational models.

This program was written in standard Fortran IV programming language and run in the IBM4341 system at Ohio University.

```
C*********************************************************************
C
C      THIS PROGRAM CALCULATES A DISTANCE AND A BEARING BETWEEN TWO
C      WAYPOINTS .
C     COMPARISON AMONG THREE MODELS(SPHERICAL,SIMPLIFIED ELLIPTICAL
C     AND ELLIPTICAL)
C     DEVICE 5 IS A INPUT DEVICE
C     DEVICE 6 IS A OUTPUT DEVICE
C
C     FEBRUARY/1982  F.OGURI
C
C*********************************************************************
C
C      LA1= LATITUDE OF THE RECEIVER
C      LO1= LONGITUDE OF THE RECEIVER
C
C      LA2= LATITUDE OF THE SECOND POINT
C      LO2= LONGITUDE OF THE SECOND POINT
C
       REAL LA1,LA2,LO1,LO2,MU,NV,LAA,LAB,LAC,MAB
C     READ COORDINATES OF THE TWO POINTS AND CONVERT GEOCENTRIC
C     COORDINATES TO RADIAN COORDINATES .
       CALL RDLL(LA1,LO1)
       CALL RDLL(LA2,LO2)
C     READ RANGE AND BEARING ANGLE TO CALCULATE THE ERROR BETWEEN
C     MEASURED VALUES AND ACTUAL VALUES.
       CALL RABE(RANGE,BEAR)
       DATA  A/3443.917387/
       PI=3.1415926535898
C
C     ELLIPTICAL MODEL
C
      FF=1.-0.00335278
      F=0.00335278
      A=3443.917387
      B=3432.370680
       DLO=LO1-LO2
       TB=FF*TAN(LA1)
       TBI=FF*TAN(LA2)
       CB=SQRT(1.+TB**2)
       CBI=SQRT(1.+TBI**2)
       SB=TB/CB
       SBI=TBI/CBI
       CD=COS(DLO)
       C1=SIN(DLO)
       C2=(TBI-TB*CD)/CB
       C3=(TBI*TB+CD)/CB
       TBA=C1/C2
       BA=ATAN(TBA)
       TAA=SQRT(C1**2+C2**2)/C3
       AA=ATAN(TAA)
       CAA=1./SQRT(1.+TAA**2)
       SAA=TAA*CAA
       MU=(AA-SAA)*(SB+SBI)**2/(1.+CAA)
       NV=(AA+SAA)*(SB-SBI)**2/(1.-CAA)
       DIS1=ABS(A*(AA-F*(MU+NV)/4.))
       BA1D=BA*180./PI
       IF(C2.LE.0.) BA1D=BA1D+180.
       IF(C1.LE.0..AND.C2.GE.0.) BA1D=360.+BA1D
C
C     SPHERICAL MODEL
C
```

```
      MAB=(LA2-LA1)/2.
      PAB=(LA1+LA2)/2.
      TA=TAN(DLO/2.)
      T1=ATAN(SIN(MAB)/(COS(PAB)*TA))
      T2=ATAN(COS(MAB)/(SIN(PAB)*TA))
      BET=T2-T1
      BA2D=BET*180./PI
      IF(C2.LE.0..AND.C1.LE.0.) BA2D=BA2D+360.
      IF(C1.LE.0..AND.C2.GE.0.) BA2D=360.+BA2D
      LAC=2.*ATAN(TAN(MAB)*SIN(T2)/SIN(T1))
      LAC=LAC*180./PI
      DIS2=60.0*LAC
C
C     SIMPLIFIED ELLIPTICAL MODEL
C
      X=DLO*COS(PAB)
      Y=LA2-LA1
      RR=((A*SIN(PAB))**2+(B*COS(PAB))**2)/B
      DIS3=RR*SQRT(X**2+Y**2)
      BA3=ATAN(X/Y)
      BA3D=BA3*180./PI
      IF(X.GE.0..AND.Y.GE.0.) CONTINUE
      IF(X.LE.0..AND.Y.GE.0.) BA3D=360.+BA3D
      IF(Y.LT.0.) BA3D=BA3D+180.
C
C     ERROR CALCULATIONS FOR THREE MODELS
C
      ERR1=DIS1-RANGE
      ERR2=DIS2-RANGE
      ERR3=DIS3-RANGE
      ERRB1=BA1D-BEAR
      ERRB2=BA2D-BEAR
      ERRB3=BA3D-BEAR
      ERROR1=((DIS1-RANGE)/RANGE)*100.
      ERROR2=((DIS2-RANGE)/RANGE)*100.
      ERROR3=((DIS3-RANGE)/RANGE)*100.
      ERRBA1=((BA1D-BEAR)/BEAR)*100.
      ERRBA2=((BA2D-BEAR)/BEAR)*100.
      ERRBA3=((BA3D-BEAR)/BEAR)*100.
      WRITE(6,200) DIS1,ERR1,BA1D,ERRB1,DIS2,ERR2,BA2D,ERRB2,
     1             DIS3,ERR3,BA3D,ERRB3
  200 FORMAT(1X,' DIS1=',F11.5,5X,' ERR1=',F9.5,5X,' BA1D=',F7.3,
     1       4X,' ERRB1=',F6.3/
     2       1H ,' DIS2=',F11.5,5X,' ERR2=',F9.5,5X,' BA2D=',F7.3,
     3       4X,' ERRB2=',F6.3/
     4       1H ,' DIS3=',F11.5,5X,' ERR3=',F9.5,5X,' BA3D=',F7.3,
     5       4X,' ERRB3=',F6.3)
      WRITE(6,300) ERROR1,ERROR2,ERROR3,ERRBA1,ERRBA2,ERRBA3
  300 FORMAT(1X,' ERROR1=',F9.6,5X,' ERROR2=',F9.6,3X,' ERROR3=',F9.6/
     3       1H ,' ERRBA1=',F9.6,5X,' ERRBA2=',F9.6,3X,' ERRBA3=',F9.6)
      STOP
      END
      SUBROUTINE RDLL(PHI,THE)
C
C*******************************************************************
C
C       THIS SUBROUTINE CONVERTS GEOCENTRIC COORDINATES ENTERED BY THE
C       USER TO RADIAN COORDINATES.  INPUT FORM IS:  DDDD MM SS.SS
C       WHERE  'DDDD' IS THE DEGREES PORTION OF THE LAT. OR LONG.,
C       INCLUDING SING, 'MM' IS THE MINUTES PORTION, AND 'SS.SS' IS THE
C       SECONDS PORTION.  READ FORMAT I:  I4,1X,I2,1X,F5.0.
C
C*******************************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*4 PHI,THE
      DATA  PI/3.1415926535898/
      DATA MSG1/'LATI'/,MSG2/'TUDE'/,MSG3/':    '/,MSG4/'LONG'/,MSG5/
     1'ITUD'/,MSG6/'E:  '/
      PI1=PI/180.
C
C          PROMPT USER.
```

```
C
      WRITE(6,1) MSG1,MSG2,MSG3
      READ(5,10) ID1,IM1,SS1
      PHI=SNGL(PI1*(DFLOAT(ID1)+(DFLOAT(IM1)+SS1/60.)/60.))
C
C         PROMPT USER FOR LONGITUDE ENTRY.
C
      WRITE(6,1) MSG4,MSG5,MSG6
      READ(5,10) ID1,IM1,SS1
      THE=SNGL(PI1*(DFLOAT(ID1)+(DFLOAT(IM1)+SS1/60.)/60.))
C
      RETURN
    1 FORMAT(' ENTER ',3A4/'  DDDD MM SS.SS')
   15 FORMAT(2X,I4,1X,I2,1X,F5.2)
   10 FORMAT(I4,1X,I2,1X,F5.2)
    3 FORMAT(' LATITUDE =  ',I4,1X,I2,1X,F5.2/' LONGITUDE = ',I4,1X,I2,
     *        1X,F5.2)
      END
      SUBROUTINE RABE(DIST,DEGR)
C
C***********************************************************************
C
C      THIS SUBROUTINE READS RANDG AND BEARING WHICH ARE PUBLISHED.
C
C***********************************************************************
C
      DATA MSG7/'DIST'/,MSG8/'ANCE'/,MSG9/'BEAR'/,MSG10/'ING '/
C
      WRITE(6,2) MSG7,MSG8
      READ(5,20) DIST
C
      WRITE(6,4) MSG9,MSG10
      READ(5,40) DEGR
C
      RETURN
    2 FORMAT(' ENTER ',2A4/'  DDDD.DDDDD')
    4 FORMAT(' ENTER ',2A4/'  BBB.BBB')
   20 FORMAT(F10.5)
   40 FORMAT(F7.3)
   25 FORMAT(2X,F10.5)
   45 FORMAT(2X,F7.3)
      END
```

APPENDIX C. Program listing for microprocessor version of area navigation (RNAV) program. This program is written in standard MOS6502 assembly language and assembled by a cross assembler on the IBM4341.

```
************************************************************************ C0000010
*                                                                      * C0000020
*      THIS PROGRAM PROVIDES NAVIGATIONAL INFORMATION USING THE        *
*      MICROCOMPUTER 6502 AND THE AM9511A MATH CHIP. THERE ARE TWO     *
*      PARTS.  THE FIRST PART IS DESIGNED TO CONVERT LORAN-C TIME-     *
*      DIFFERENCES TO LATITUDE/LONGITUDE BY J.P.FISCHER(TRACKING       *
*      FILTER ON TIME DIFFERENCES WERE ADDED LATER BY F.OGURI), AND THE
*      SECOND PART IS DESIGNED TO CALCULATE RANGE/BEARING TO A WAY-    *
*      POINT, CROSS TRACK ERROR FROM DESIRED COURSE, GROUND SPEED      *
*      AND ESTIMATED TIME OF ARRIVAL TO THE WAYPOINT BY F. OGURI.      *
*      BDC-TO-HEX CONVERSIONS ARE MADE FOR TIME-DIFFERENCES,           *
*      WAYPOINTS AND GRI, AND HEX-TO-BCD CONVERSIONS ARE MADE FOR      *
*      ALL CALCULATED NAVIGATIONAL INFORMATION;  ALL INTERNAL          *
*      CALCULATIONS ARE MADE USING BINARY FLOATING-POINT.  ALL SUB-    *
*      ROUTINE ARE AT THE END OF THE MAIN PROGRAM.  THE NUMBER TABLE   *
*      AREA IS DESIGNED TO BE PLACED AFTER THE SUBROUTINES;            *
*      CONSTANTS ARE FIRST, CALCULATED VARIABLES LAST.                 *
*      FIRST PART:  ALGORITHM IS BASED ON FORTRAN PROGRAM 'DEXLRN.'    *
*                                     7/1981, J. P. FISCHER            *
*      SECOND PART:  ALGORITHM IS BASED ON FORTRAN PROGRAM 'DIST3.'    *
*      ELLIPTICAL MODEL,  1/1982,  F. OGURI                            *
*      CHANGE FOR THE VIDEO BOARD,   10/1982,  F. OGURI                *
*                                                                      * C0000150
************************************************************************ C0000160
*                                                                        C0000180
PIAA   EQU $9000        PERIPHERAL AND DDR SIDE A
PIAB   EQU $9002        PERIPHERAL AND DDR SIDE B
*                                                                        C0000210
*        AM9511A COMMANDS.                                               C0000220
*                                                                        C0000230
FADD   EQU $10
FSUB   EQU $11
FMUL   EQU $12
FDIV   EQU $13
SQRT   EQU 1
SIN    EQU 2
COS    EQU 3
ATAN   EQU 7
PTOF   EQU $17
PUPI   EQU $1A
FLTD   EQU $1C
FIXD   EQU $1E
CHSF   EQU $15
TAN    EQU $04
XCHF   EQU $19
ASIN   EQU $05
DMUL   EQU $2E
*                                                                        C0000370
*        VARIABLE NAME TABLE(FOR TD-TO-POSITION CONVERSION)
*                                                                        C0000390
*        THE FOLLOWING ARE CONSTANTS.                                    C0000410
*                                                                        C0000420
TCY    EQU 0
TCZ    EQU TCY+4
THMY   EQU TCZ+4
THMZ   EQU THMY+4
XNR    EQU THMZ+4
CTMY   EQU XNR+4
```

```
STMY    EQU CTMY+4
CTMZ    EQU STMY+4
STMZ    EQU CTMZ+4
CXK     EQU STMZ+4
SXK     EQU CXK+4
C1      EQU SXK+4
C2      EQU C1+4
C3      EQU C2+4
C4      EQU C3+4
C5      EQU C4+4
C6      EQU C5+4
C7      EQU C6+4
C8      EQU C7+4
C9      EQU C8+4
C10     EQU C9+4
C11     EQU C10+4
C12     EQU C11+4
C14     EQU C12+4
EM6     EQU C14+4
C256    EQU EM6+4
P180    EQU C256+4
C60     EQU P180+4
ALP     EQU C60+4        ALPHA FOR FILTER ON TDS
BET     EQU ALP+4        BETA FOR FILTER ON TDS
TM      EQU BET+4        TIME INTERVAL FOR FILTER ON TDS
*
*          THE FOLLOWING ARE CALCULATED VARIABLES.
*          (FOR TD-TO-POSITION CONVERSION)
*
TY      EQU TM+4
TZ      EQU TY+4
PY      EQU TZ+4
PZ      EQU PY+4
CPY     EQU PZ+4
SPY     EQU CPY+4
CPZ     EQU SPY+4
SPZ     EQU CPZ+4
AY      EQU SPZ+4
AZ      EQU AY+4
BY      EQU AZ+4
BZ      EQU BY+4
U1      EQU BZ+4
U2      EQU U1+4
U3      EQU U2+4
UU      EQU U3+4
CDBY    EQU UU+4
THMS    EQU CDBY+4
CB      EQU THMS+4
CA      EQU CB+4
CC      EQU CA+4
F       EQU CC+4
G       EQU F+4
H       EQU G+4
THGS    EQU H+4          LONGITUDE OF THE RECEIVER
PHGS    EQU THGS+4       LATITUDE OF THE RECEIVER
TEMP    EQU PHGS+4
TYP     EQU TEMP+4
TZP     EQU TYP+4
TYS     EQU TZP+4
TZS     EQU TYS+4
VTYP    EQU TZS+4
VTZP    EQU VTYP+4
*
*          PAGE-ZERO ASSIGNMENTS
*
        ORG 0
TDA     BSS 4            PACKED BCD /W TENTH DIGIT
        BSS 4            PACKED BCD /W TENTH DIGIT
        ORG $68
AGCF    BSS 1
AGCB    BSS 1
BASE1   BSS 2            BASE ADDRESS FOR NUMBER MOVE
XLIM    BSS 1            INDEX LIMIT USED IN NUMBER CONVERSION ROUTINES
```

```
XTEMP  BSS 1            SAVE AREA FOR X-REGESTER
YTEMP  BSS 1            SAVE AREA FOR Y-REGESTER
COUNT  BSS 1            COUNT-DOWN REGGSTER
VY     BSS 1
FLAG   BSS 1
HEX    BSS 3            TABLE FOR BUILDING UP HEX NUMBER FROM BCD
RES    BSS 3            RESIDUE TABLE FOR BINARY MULTIPLICATION
DVDN   BSS 2            HEX NUMBER TO BE CONVERTED TO BCD
CNT2   BSS 1
CNT3   BSS 1
CNT4   BSS 1
CNT5   BSS 1
PREWP  BSS 1            PREVIOUS WAYPOINT NUMBERS
FLG3   BSS 1
TMP    BSS 2            TEMPORARY REGISTER FOR GRI CONVERSION
GRI1   BSS 1            ADDRESS OF GS REFERENCE TABLE(GRI LOOP COUNT)
ARCN   BSS 1            ADDRESS OF GS REFERENCE TABLE(RANGE)
CTEBN  BSS 1            ADDRESS OF GS REFERENCE TABLE(CTEB)
LOLA   BSS 1
LOPG   BSS 1            FLAG FOR LOWPAGE OF VIDEO SCREEN
FLT    BSS 1
NOLP   BSS 1
TDTMP  BSS 1
PRTMP  BSS 1
SMTMP  BSS 1
VLTMP  BSS 1
*
*       THE FOLLOWING ARE THE
*       COMPUTED LATITUDE/LONGITUDE AND RANGE/BEARING.  ALL
*       ARE IN PACKED BCD FORMAT.  THESE ARE DESIGNED TO
*       INTERFACE TO SENSOR SOFTWARE.
*
LAT    BSS 3            FORMAT: (EX.) 39 19 20 - DEGREES, MINUTES, SECON
       BSS 3            FORMAT: (EX.) 82 05 56 - DEGREES, MINUTES, SECON
       BSS 3            FORMAT: (EX.) 01 34 52 - HOURS, MINUTES, SECOND
GSP    BSS 10           FORMAT: (EX.) 20 56 -  205.6 NM
BASE   BSS 2            ADDRESS OF FLOATING-POINT TABLE
VIDEO  BSS 2            ADDRESS OF VIDEO SCREEN MAP
WP     BSS 1            MN : WAYPOINTN M=1,5  N=1,5
WPG    BSS 1            TO WAYPOINT
WPS    BSS 1            FROM WAYPOINT
LLLP   BSS 1
WPTB   BSS 48           ND DD MM SS OO DD MM SS : LA2 AND LO2 IN DEGREE
*
*
*     USER'S WAYPOINT TABLE
*
       ORG $AC
       HEX 12           -WP
       HEX 00           -WPG
       HEX 00           -WPS
*
*
*     ADDRESS OF GRI AND BIF1
*
GRI    EQU $0012        GRI DATA BUFFER
BIF1   EQU $0041        LOOP COUNTER
GRIN   EQU $03E0        REFERENCE TABLE FOR GRI LOOP COUNT
*
*     THE FOLLOWING ARE CONSTANTS.(FOR RNAV)
*
RCR2   EQU 0            3443.9174NM
F1     EQU RCR2+4       0.00335278
F2     EQU F1+4         1-F1
ONE    EQU F2+4         1
FOUR   EQU ONE+4        4
D3618  EQU FOUR+4       3600*180/PI
D36E6  EQU D3618+4      36.0E6
F20    EQU D36E6+4      20
P18    EQU F20+4        180/PI
PAI2   EQU P18+4        2*PI RADIANS
PI2D   EQU PAI2+4       360 DEGREE
D60    EQU PI2D+4       60
```

```
ALPG    EQU D60+4        ALPHA FOR FILTER IN GS CALCULATION
BETG    EQU ALPG+4       BETA FOR FILTER IN GS CALCULATION
LO1     EQU BETG+4       LONGITUDE OF FROM WAYPOINT
LA1     EQU LO1+4        LATITUDE OF FROM WAYPOINT
L02     EQU LA1+4        LONGITUDE OF TO WAYPOINT
LA2     EQU L02+4        LATITUDE OF TO WAYPOINT
*
*      CALCULATED VARIABLES.(FOR RNAV)
*
COB     EQU LA2+4
SB      EQU COB+4
CBI     EQU SB+4
SBI     EQU CBI+4
CO1     EQU SBI+4
CO2     EQU CO1+4
CO3     EQU CO2+4
BA      EQU CO3+4
SAA     EQU BA+4
CAA     EQU SAA+4
AA      EQU CAA+4
MU      EQU AA+4
NV      EQU MU+4
ARC1    EQU NV+4
ARC2    EQU ARC1+4
PSI1    EQU ARC2+4
PSI2    EQU PSI1+4
BNYY    EQU PSI2+4
HELPP   EQU BNYY+4
TIME    EQU HELPP+4
CTEB    EQU TIME+4
GS      EQU CTEB+4
CTEB1   EQU GS+4
WW      EQU CTEB1+4
*
*       FOR GROUND SPEED CALCULATION.
*
CTEBO   EQU WW+4         PRIVIOUS CTEB FOR GS CALCULATION
ARCO    EQU CTEBO+4      PRIVIOUS RANGE FOR GS CALCULATION
GRITT   EQU ARCO+4       TOTAL GRI LOOP COUNT
GSPRD   EQU GRITT+4      GS PREDICTED VALUE
ACPRD   EQU GSPRD+4
GSSM    EQU ACPRD+4
*
*      CALCULATED VARIABLES FOR BCD TO FLOATING POINT FORMATCONVERSION.
*
        ORG $038C
BNY     BSS 4            WAYPOINT REGISTER
HELP    BSS 4            GRI CONVERSION REGISTER
*
        ORG $03B0
GRIT    BSS 4            TOTAL REFERENCE OF GRI LOOP COUNT
*
        ORG $3A4
W       BSS 4            TEMPORARY REGISTER FOR WAYPOINT CONVERSION
*
*
*
        ORG $1800
*
*           CONVERT TD'S FROM BCD TO FLOATING-POINT
*
        LDX =0
        LDY =TY          CONVERT TDA FIRST
        STY YTEMP
        LDA =2
        STA COUNT        CONVERT TWO TDS
        STA BASE+1
        CLD              SET DECIMAL MODE OFF
*
*           BCD-TO-HEX CONVERSION
*
TOHEX   LDY =6           SIX DIGITS
        LDA =0
```

```
        STA HEX
        STA HEX+1
        STA HEX+2
        STA FLAG
        JMP LFOUR       DO LOWER PART FIRST
UFOUR   DEC FLAG
        LDA TDA,X       GET A BYTE
        LSR A
        LSR A
        LSR A
        LSR A
        JMP TO3         DO THE MAIN CONVERSION
LFOUR   INC FLAG
        LDA TDA,X       GET A BYTE
        AND =$F         REMOVE UPPER FOUR BITS
        INX
TO3     CLC             FOR ADDITION
        ADC HEX+2
        STA HEX+2       ADD DIGIT TO PARTIAL SUM
        BCC NOC         GO IF NO CARRY OUT
        INC HEX+1
NOC     DEY             NEXT DIGIT
        BEQ TOD         IF DONE, LEAVE
        STY XTEMP       SAVE COUNTER
*
*           MULTIPLY PARTIAL SUM BY TEN
*
        LDA =0
        STA RES         CLEAR MULT. TABLE
        STA RES+1
        STA RES+2
        LDA =10         DIVISOR
        LDY =8
TO2     CLC
        ROL RES+2
        ROL RES+1
        ROL RES
        ASL A
        BCC NOC2
        PHA
        CLC
        LDA RES+2
        ADC HEX+2
        STA RES+2
        LDA RES+1
        ADC HEX+1
        STA RES+1
        LDA RES
        ADC HEX
        STA RES
        PLA
NOC2    DEY
        BNE TO2
        LDY XTEMP
        LDA RES
        STA HEX
        LDA RES+1
        STA HEX+1
        LDA RES+2
        STA HEX+2
        LDA FLAG
        BNE UFOUR
        JMP LFOUR
*
*           NOW CHANGE INTEGER PART TO FLOATING-POINT
*           THEN ADD IN THE FRACTIONAL PART.
*
TOD     LDY YTEMP
        STX XTEMP       SAVE THE CURRENT DIGIT LOCATION
        LDA =0
        STA (BASE),Y    CLEAR THE UPPER TWO BYTES
        INY
        LDA HEX         MSB OF HEX INTEGER
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
        STA (BASE),Y    PUT IN TABLE FOR 9511
        INY
        LDA HEX+1       LSB OF HEX INTEGER
        STA (BASE),Y
        INY
        LDA HEX+2
        STA (BASE),Y
        LDY YTEMP       POINT TO TD NUMBER
        JSR PUSH        GIVE IT TO 9511
        LDA =FLTD
        JSR CMND        CONVERT INTEGER TO FLOATING-POINT
        LDY =EM6        CONSTANT 1E-7
        JSR PUSH
        LDA =FMUL       CONVERT FROM MICROSECONDS...
        JSR CMND        TO SECONDS
        LDY YTEMP       GET THE TD LOCATION AGAIN
        JSR POP         AND STORE THE TD
        DEC COUNT       SEE IF BOTH TDS CONVERTED
        BEQ ABFLT       IF SO, START TD-TO-POSITION CONVERSION
TOD1    LDX XTEMP       SET UP FOR NEXT TD
        INX
        LDY =TZ         ADDRESS OF TDB
        STY YTEMP       STORE IT
        JMP TOHEX       REPEAT
*
*    ALPHA BETA FILTERING
*
ABFLT   LDA NOLP
        CMP =1
        BNE ABFLT2
        LDX =0
        LDA =0
ABFLT1  STA $02F8,X
        INX
        CPX =8
        BNE ABFLT1
        LDY =TY
        JSR PUSH
        LDY =TZ
        JSR PUSH
        LDY =TZP
        JSR POP
        LDY =TYP
        JSR POP
        JMP SUTD
*
ABFLT2  LDA =$7C
        STA TDTMP       TY
        LDA =$E8
        STA PRTMP       TYP
        LDA =$F0
        STA SMTMP       TYS
        LDA =$F8
        STA VLTMP       VTYP
ABFLT3  LDY TDTMP
        JSR PUSH
        LDY PRTMP
        JSR PUSH
        LDA =FSUB
        JSR CMND        TY-TYP
        LDA =PTOF
        JSR CMND
        LDY =ALP
        JSR PUSH
        LDA =FMUL
        JSR CMND        ALP*(TY-TYP)
        LDY PRTMP
        JSR PUSH
        LDA =FADD
        JSR CMND        TYP+ALP*(TY-TYP)
        LDY SMTMP
        JSR POP         TYS=TYP+ALP*(TY-TYP)
*
```

```
        LDY =BET
        JSR PUSH
        LDA =FMUL
        JSR CMND         BET*(TY-TYP)
        LDY =TM
        JSR PUSH
        LDA =FDIV
        JSR CMND
        LDY VLTMP
        JSR PUSH
        LDA =FADD
        JSR CMND
        LDA =PTOF
        JSR CMND
        LDY VLTMP
        JSR POP          VTYP=VTYP+BET*(TY-TYP)/TM
        LDY =TM
        JSR PUSH
        LDA =FMUL
        JSR CMND         TM*VTYS
        LDY SMTMP
        JSR PUSH
        LDA =FADD
        JSR CMND         TYS+TM*VTYS
        LDA =PTOF
        JSR CMND
        LDY PRTMP
        JSR POP          TYP=TYS+TM*VTYS
        LDY TDTMP
        JSR POP          TY=TYS+TM*VTYS
        LDA TDTMP
        CMP =$80
        BEQ SUTD
ABFLT4  INC TDTMP
        INC PRTMP
        INC SMTMP
        INC VLTMP
        LDA TDTMP
        CMP =$80
        BNE ABFLT4
        JMP ABFLT3
*
*       CALCULATE 'PY'
*
SUTD    LDY =XNR         'XNR'
        JSR PUSH         PUSH ON STACK
        LDY =TY          'TY'
        JSR PUSH
        LDY =TCY         'TCY'
        JSR PUSH
        LDA =FSUB
        JSR CMND         SUBTRACT TY-TYC
        LDA =FMUL
        JSR CMND         XNR*(TY-TYC)
        LDY =THMY        'THMY'
        JSR PUSH
        LDA =FSUB
        JSR CMND         XNR*(TY-TZ)-THMY
        LDY =PY          'PY'
        JSR POP          PUT 'PY' INTO TABLE
*
*       CALCULATE 'PZ'
*
        LDY =XNR         'XNR'
        JSR PUSH         PUSH ONTO STACK
        LDY =TZ          'TZ'
        JSR PUSH         PUSH ONTO STACK
        LDY =TCZ         'TCZ'
        JSR PUSH         PUSH ONTO STACK
        LDA =FSUB
        JSR CMND         TZ-TCZ
        LDA =FMUL
        JSR CMND         XNR*(TZ-TCZ)
```

```
         LDY =THMZ        'THMZ'
         JSR PUSH
         LDA =FSUB
         JSR CMND         XNR*(TZ-TCZ)-THMZ
         LDY =PZ          LOCATION FOR 'PZ'
         JSR POP          STORE RESULT IN 'PZ'
*
*            CALCULATE CPY,SPY,CPZ,SPZ
*
         LDY =PY          'PY'
         JSR PUSH         PUSH IT
         LDY =PY          'PY'
         JSR PUSH         DUPLICATE STACK
         LDA =COS
         JSR CMND         COS(PY)
         LDY =CPY         FOR CPY
         JSR POP          GET IT
         LDA =SIN
         JSR CMND         SIN(PY)
         LDY =SPY
         JSR POP          GET SPY
         LDY =PZ          'PZ'
         JSR PUSH
         LDY =PZ          'PZ'
         JSR PUSH
         LDA =COS
         JSR CMND         COS(PZ)
         LDY =CPZ         LOCATION FOR CPZ
         JSR POP
         LDA =SIN
         JSR CMND         SIN(PZ)
         LDY =SPZ         LOCATION FOR 'SPZ'
         JSR POP
*
*            CACULATE 'AY'
*
         LDY =CPY         'CPY'
         JSR PUSH
         LDY =CTMY        'CTMY'
         JSR PUSH
         LDA =FSUB
         JSR CMND         CPY-CTMY
         LDY =STMY        'STMY'
         JSR PUSH
         LDA =FDIV
         JSR CMND         (CPY-CTMY)/STMY
         LDY =AY          'AY'
         JSR POP
*
*            CALCULATE 'AZ'
*
         LDY =CPZ         'CPZ'
         JSR PUSH
         LDY =CTMZ        'CTMZ'
         JSR PUSH
         LDA =FSUB
         JSR CMND         CPZ-CTMZ
         LDY =STMZ        'STMZ'
         JSR PUSH
         LDA =FDIV
         JSR CMND         (CPZ-CTMZ)/STMZ
         LDY =AZ          LOCATION FOR 'AZ'
         JSR POP
*
*           CALCULATE 'BY'
*
         LDY =SPY         'SPY'
         JSR PUSH
         LDY =STMY        'STMY'
         JSR PUSH
         LDA =FDIV
         JSR CMND         SPY/STMY
         LDY =BY          LOCATION FOR 'BY'
```

```
        JSR POP         GET IT
*
*          CALCULATE 'BZ'
*
        LDY =SPZ        'SPZ'
        JSR PUSH
        LDY =STMZ       'STMZ'
        JSR PUSH
        LDA =FDIV
        JSR CMND        SPZ/STMZ
        LDY =BZ         LOCATION FOR 'BZ'
        JSR POP         GET IT
*
*          CALCULATE 'U1'
*
        LDY =AY         'AY'
        JSR PUSH
        LDY =CXK        'CXK'
        JSR PUSH
        LDA =FMUL
        JSR CMND        AY*CXK
        LDY =AZ         'AZ'
        JSR PUSH
        LDA =FSUB
        JSR CMND        AY*CXK-AZ
        LDY =U1         LOCATION FOR 'U1'
        JSR POP
*
*          CALCULATE 'U2'
*
        LDY =AY         'AY'
        JSR PUSH
        LDY =SXK        'SXK'
        JSR PUSH
        LDA =FMUL
        JSR CMND        AY*SXK
        LDY =U2
        JSR POP         GET 'U2'
*
*          CALCULATE 'U3'
*
        LDY =AZ         'AZ'
        JSR PUSH
        LDY =BY         'BY'
        JSR PUSH
        LDA =FMUL
        JSR CMND        AZ*BY
        LDY =AY         'AY'
        JSR PUSH
        LDY =BZ         'BZ'
        JSR PUSH
        LDA =FMUL
        JSR CMND        AY*BZ
        LDA =FSUB
        JSR CMND        AZ*BY-AY*BZ
        LDY =U3         LOCATION FOR 'U3'
        JSR POP
*
*          CALCULATE 'UU'
*
        LDY =U1         'U1'
        JSR PUSH
        LDY =U1         'U1'
        JSR PUSH
        LDA =FMUL
        JSR CMND        U1*U1
        LDY =U2         'U2'
        JSR PUSH
        LDY =U2
        JSR PUSH
        LDA =FMUL
        JSR CMND        U2*U2
        LDA =FADD
        JSR CMND        U1*U1+U2*U2
```

```
        LDY =UU         LOCATION FOR 'UU'
        JSR POP
*
*           CALCULATE 'CDBY'
*
        LDY =UU         'UU'
        JSR PUSH
        LDY =U3         'U3'
        JSR PUSH
        LDY =U3         'U3'
        JSR PUSH
        LDA =FMUL
        JSR CMND        U3*U3
        LDA =FSUB
        JSR CMND        UU-U3*U3
        LDA =SQRT
        JSR CMND        SQRT(UU-U3*U3)
        LDY =U2         'U2'
        JSR PUSH
        LDA =FMUL
        JSR CMND        U2*SQRT(UU-U3*U3)
        LDY =U3         'U3'
        JSR PUSH
        LDY =U1         'U1'
        JSR PUSH
        LDA =FMUL
        JSR CMND        U3*U1
        LDA =FADD
        JSR CMND        U3*U1+U2*SQRT(UU-U3*U3)
        LDY =UU         'UU'
        JSR PUSH
        LDA =FDIV
        JSR CMND        (U3*U1+U2*SQRT(UU-U3*U3))/UU
        LDY =CDBY       LOCATION FOR 'CDBY'
        JSR POP
*
*           CALCULATE 'THMS'
*
        LDY =AY         'AY'
        JSR PUSH
        LDY =BY         'BY'
        JSR PUSH
        LDY =CDBY       'CDBY'
        JSR PUSH
        LDA =FADD
        JSR CMND        BY+CDBY
        LDA =FDIV
        JSR CMND        AY/(BY+CDBY)
        LDA =ATAN
        JSR CMND        ATAN(AY/(BY+CDBY))
        LDY =THMS       LOCATION FOR 'THMS'
        JSR POP
*
*           CALCULATE 'CB'
*
        LDY =THMS       'THMS'
        JSR PUSH
        LDA =COS
        JSR CMND        COS(THMS)
        LDY =CB         LOCATION FOR 'CB'
        JSR POP
*
*           CALCULATE 'CA'
*
        LDY =THMS       'THMS'
        JSR PUSH
        LDY =PY         'PY'
        JSR PUSH
        LDA =FADD
        JSR CMND        THMS+PY
        LDA =COS
        JSR CMND        COS(THMS+PY)
```

```
        LDY =CA         LOCATION FOR 'CA'
        JSR POP
*
*           CALCULATE 'CC'
*
        LDY =THMS       'THMS'
        JSR PUSH
        LDY =PZ         'PZ'
        JSR PUSH
        LDA =FADD
.t70

        LDA =FADD
        JSR CMND        THMS+PZ
        LDA =COS
        JSR CMND        COS(THMS+PZ)
        LDY =CC         LOCATION FOR 'CC'
        JSR POP
*
*           CALCULATE 'F'
*
        LDY =C1         'C1'
        JSR PUSH
        LDY =CA         'CA'
        JSR PUSH
        LDA =FMUL
        JSR CMND        C1*CA
        LDY =C2         'C2'
        JSR PUSH
        LDY =CB         'CB'
        JSR PUSH
        LDA =FMUL
        JSR CMND        C2*CB
        LDA =FADD
        JSR CMND        C1*CA+C2*CB
        LDY =C3         'C3'
        JSR PUSH
        LDY =CC         'CC'
        JSR PUSH
        LDA =FMUL
        JSR CMND        C3*CC
        LDA =FADD
        JSR CMND        C1*CA+C2*CB+C3*CC
        LDY =F          LOCATION FOR 'F'
        JSR POP
*
*           CALCULATE 'G'
*
        LDY =C4         'C4'
        JSR PUSH
        LDY =CA         'CA'
        JSR PUSH
        LDA =FMUL
        JSR CMND        C4*CA
        LDY =C5         'C5'
        JSR PUSH
        LDY =CB         'CB'
        JSR PUSH
        LDA =FMUL
        JSR CMND        C5*CB
        LDA =FADD
        JSR CMND        C4*CA+C5*CB
        LDY =C6         'C6'
        JSR PUSH
        LDY =CC         'CC'
        JSR PUSH
        LDA =FMUL
        JSR CMND        C6*CC
        LDA =FADD
        JSR CMND        C4*CA+C5*CB+C6*CC
        LDY =G          'G'
        JSR POP         GET 'G'
```

```
*
*         CALCULATE 'H'
*
      LDY =C7         'C7'
      JSR PUSH
      LDY =CA         'CA'
      JSR PUSH
      LDA =FMUL
      JSR CMND        C7*CA
      LDY =C8         'C8'
      JSR PUSH
      LDY =CB         'CB'
      JSR PUSH
      LDA =FMUL
      JSR CMND        C8*CB
      LDA =FADD
      JSR CMND        C7*CA+C8*CB
      LDY =C9         'C9'
      JSR PUSH
      LDY =CC         'CC'
      JSR PUSH
      LDA =FMUL
      JSR CMND        C9*CC
      LDA =FADD
      JSR CMND        C7*CA+C8*CB+C9*CC
      LDY =H          LOCATION 'H'
      JSR POP
*
*         CALCULATE 'THGS'
*
      LDY =G          'G'
      JSR PUSH
      LDY =C10        'C10'
      JSR PUSH
      LDA =FADD
      JSR CMND        G+C10
      LDY =F          'F'
      JSR PUSH
      LDY =C11        'C11'
      JSR PUSH
      LDA =FADD
      JSR CMND        F+C11
      LDA =FDIV
      JSR CMND        (G+C10)/(F+C11)
      LDA =ATAN
      JSR CMND        ATAN((G+C10)/(F+C11))
      LDA =PTOF
      JSR CMND        DUPLICATE STACK
      LDY =THGS
      JSR POP         GET THGS
      LDY =P180       180/PI
      JSR PUSH
      LDA =FMUL
      JSR CMND        CONVERT FROM RADIANS TO DEGREES
      LDX =3          POINT TO LONGITUDE FIELD
      LDA =6          INDEX LIMIT
      STA XLIM
      LDA =$86        VIDEO LOCATION FOR LONG.
      STA VY
      JSR TOBCD2      CONVERT TO DEGREES, MINUTES, SECONDS
*
*         CALCULATE 'PHGS'
*
      LDY =THGS       'THGS'
      JSR PUSH
      LDA =SIN
      JSR CMND        SIN(THGS)
      LDY =C14
      JSR PUSH
      LDY =C14
      JSR PUSH
      LDA =FMUL
```

```
        JSR CMND        C14*C14
        LDA =FMUL
        JSR CMND        C14*C14*SIN(THGS)
        LDY =H          'H'
        JSR PUSH
        LDY =C12        'C12'
        JSR PUSH
        LDA =FADD
        JSR CMND        H+C12
        LDA =FMUL
        JSR CMND        C14*C14*SIN(THGS)*(H+12)
        LDY =G          'G'
        JSR PUSH
        LDY =C10        'C10'
        JSR PUSH
        LDA =FADD
        JSR CMND        G+C13
        LDA =FDIV
        JSR CMND        C14*C14*SIN(THGS)*(H+C12)/(G+C10)
        LDA =ATAN
        JSR CMND        ARCTAN(C14*C14*SIN(THGS)*(H+C12)/(G+C13))
        LDA =PTOF
        JSR CMND        DUPLICATE STACK LOCATIONS
        LDY =PHGS
        JSR POP         GET PHGS
        LDY =P180       180/PI
        JSR PUSH
        LDA =FMUL
        JSR CMND        CONVERT PHGS FROM RADIANS TO DEGREES
        LDX =0          LOCATION FOR LATITUDE FIELD
        LDA =3          INDEX LIMIT
        STA XLIM
        LDA =$66        VIDEO LOCATION FOR LAT.
        STA VY
        JSR TOBCD2      CONVERT TO DEGREES, MINUTES, SECONDS
*                                                               C0006700
*    AREA NAVIGATION CALCULATION
*
RNAV    LDA =3
        STA BASE+1      BASE=$300
        CLD
*
        LDX =1
        STX FLG3        FLG3=1
        LDA WP
        CMP PREWP       IS A WAYPOINT CHANGED?
        BNE WPCV2       IF YES, GET NEW WAYPOINT COORDINATE.
        DEC FLG3        FLG3=0
        JMP RABAO
*
*    CONVERT WAYPOINT(LO2,LA2) FROM BCD TO FLOATING POINT.
*
WPCV1   LDA WP          READ WAYPOINT NUMBERS
WPCV2   STA PREWP       STORE NEW WAYPOINT NUMBERS
        LDX FLG3        WHEN FLG3=1, FROM WAYPOINT CASE.
        BEQ WPCV3       WHEN FLG=0, TO WAYPOINT CASE.
        AND =$F0        READ FROM WAYPOINT NUMBER
        LSR A
        LSR A
        LSR A
        LSR A
        STA WPS         STORE FROM WAYPOINT NUMBER
        ORA =$30
        STA $A015       DISPLAY FROM WAYPOINT NO.
        JMP WPCV4
WPCV3   AND =$0F        READ TO WAYPOINT NUMBER
        STA WPG         STORE TO WAYPOINT NO.
        ORA =$30
        STA $A055       DISPLAY TO WAYPOINT NO.
WPCV4   LDX =0
        STX CNT2
LO      LDA WPTB,X
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
        AND =$F0        READ WAYPOINT NO. ON WAYPOINT TABLE
        LSR A
        LSR A
        LSR A
        LSR A
        LDY FLG3
        CMP WPG,Y       FIND WAYPOINT NUMBER
        BEQ L00         IF FIND NO., BRANCH TO L00
        LDY =8
WPCV5   INX             INCREMENT 8 TIMES
        DEY
        BNE WPCV5
        JMP L0          TRY TO FIND AGAIN
L00     LDA =0
        STA CNT1        CLEAR COUNTER1
        STA W+1         CLEAR TEMPORARY REGISTER
        STA W+2
        STA W+3
        STA BNY         CLEAR MSB OF WAYPOINT REGISTER
L10     LDA WPTB,X
        AND =1
        BEQ L7          IF DEGREE IS LESS THAN 100, BRANCH TO L7
        LDA =$64
        STA W+3
        LDA =1
L7      LDY FLG3
        BEQ L2
        LDY CNT2
        BNE L77
        LDY =0
        JMP L3
L77     LDY =$20
        JMP L3
L2      LDY CNT2
        BNE L22
        LDY =$40
        JMP L3
L22     LDY =$60
L3      ORA =$30
        STA $A017,Y
        INY
L4      INX
        LDA WPTB,X
        LSR A
        LSR A
        LSR A
        LSR A
        ORA =$30
        STA $A017,Y
        INY
        LDA WPTB,X
        AND =$0F
        ORA =$30
        STA $A017,Y
        INY
        INY
        CPY =$0A
        BEQ L6
        CPY =$2A
        BEQ L6
        CPY =$4A
        BEQ L6
        CPY =$6A
        BEQ L6
        JMP L4
L6      DEX
        DEX
*
*      UPDIGIT*10+LOWDIGIT
*
L5      LDA =0
        STA BNY+3       CLEAR LSB OF WAYPOINT REGISTER
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
        LDY =4
L8      LDA WPTB,X
        STA LOLA
L88     ASL BNY+3
        ASL LOLA
        BCC L9
        LDA =10
        CLC
        ADC BNY+3
        STA BNY+3
L9      DEY
        BNE L88         WHEN IT'S DONE, (BNY+3)=UPDIGIT*10.
        LDA WPTB,X
        AND =$0F        READ LOWDIGIT
        CLC
        ADC BNY+3
        STA BNY+3       (BNY+3)=UPDIGIT*10+LOWDIGIT
L25     INC CNT1
        INX             NEXT BYTE
*   WPT. REGISTER = TEMP.REGISTER+WPT.REGISTER
        CLC
        LDA W+3
        ADC BNY+3
        STA BNY+3
        BCC L28
        INC W+2
L28     LDA W+2
        STA BNY+2
        LDA W+1
        STA BNY+1
        LDY CNT1
        CPY =3
        BNE L35         IF IT'S NOT DONE, BRANCH TO L35
        JMP L50         IF THE CONVERTION IS DONE, JMP TO L50
*
*    MULTIPLY BY 60
*
L35     STX CNT3
        LDY =BNYY
        JSR PUSH
        LDY =D60
        JSR PUSH
        LDA =DMUL
        JSR CMND
        LDY =WW
        JSR POP         TEMP.REGISTER = WPT.REGISTER*60.
        LDX CNT3
        JMP L5          IF THE MULTIPLICATION IS DONE, JUMP TO L5
*
L50     STX CNT3
        LDY =BNYY
        JSR PUSH
        LDA =FLTD       CHANGE FIXD TO FLOATING POINT
        JSR CMND
        LDY =D3618
        JSR PUSH
        LDA =FDIV       DIVIDE BY 3600 AND CONVERT IT INTO RADIAN UNIT
        JSR CMND
        LDA FLG3
        BEQ L70
        LDA CNT2
        BNE L60
        LDY =LA1
        JSR POP         CONVERSION FOR LATITUDE OF FROM WPT. IS DONE
        INC CNT2
        LDX CNT3
        JMP L00
L60     LDY =L01
        JSR POP         CONVERSION FOR LONGITUDE OF FROM WPT. IS DONE
        DEC FLG3
        JMP WPCV1
L70     LDA CNT2
```

```
        BNE L80
        LDY =LA2
        JSR POP         CONVERSION FOR LATITUDE OF TO WPT. IS DONE.
        INC CNT2
        LDX CNT3
        JMP L00
L80     LDY =LO2
        JSR POP         CONVERSION FOR LONGITUDE OF TO WPT. IS DONE.
        INC FLG3        FLG3=1
*
*     TRANSFER CONTENTS OF THGS, PHGS TO LO1, LA1 IN THIS PROGRAM
*
        LDY =LA1        WHEN WAYPOINT ARE NOT CHANGED.
        JSR PUSH        DESIRED COURSE IS CALCULATED.
        JMP TRAN1
*
*      CALCULATE 'COB' AND 'SB'
*
RABA0   DEC BASE+1      BASE=$200
        LDY =PHGS       PHGS IS LATITUDE OF THE RECEIVER.
        JSR PUSH
        INC BASE+1      BASE=$300
TRAN1   LDA =TAN
        JSR CMND        TAN(LA1)
        LDY =F2
        JSR PUSH
        LDA =FMUL
        JSR CMND        F2*TAN(LA1)
        LDA =ATAN
        JSR CMND        B=ATAN(F2*TAN(LA1))
        LDA =PTOF
        JSR CMND
        LDA =COS
        JSR CMND        COS(B)
        LDY =COB
        JSR POP         COB=COS(B)
        LDA =SIN
        JSR CMND        SIN(B)
        LDY =SB
        JSR POP         SB=SIN(B)
*
*      CALCULATE 'CBI' AND 'SBI'
*
        LDY =LA2
        JSR PUSH
        LDA =TAN
        JSR CMND        TAN(LA2)
        LDY =F2
        JSR PUSH
        LDA =FMUL
        JSR CMND        F2*TAN(LA2)
        LDA =ATAN
        JSR CMND        BI=ATAN(F2*TAN(LA2))
        LDA =PTOF
        JSR CMND
        LDA =SIN
        JSR CMND        SIN(BI)
        LDY =SBI
        JSR POP         SBI=SIN(BI)
        LDA =COS
        JSR CMND        COS(BI)
        LDY =CBI
        JSR POP         CBI=COS(BI)
*
*     CALCULATE 'CO1'
*
        LDA FLG3
        BEQ RABA1       IF WAYPOINTS ARE NOT CHANGED, BRANCH TO RABA1
        LDY =LO1        WHEN WAYPOINTS ARE CHANGED,
        JSR PUSH        DESIRED COURSE IS CALCULATED.
        JMP RABA2
RABA1   DEC BASE+1
```

```
        LDY =THGS
        JSR PUSH
        INC BASE+1
RABA2   LDY =LO2
        JSR PUSH
        LDA =FSUB
        JSR CMND        LO1-LO2
        LDA =PTOF
        JSR CMND
        LDA =SIN
        JSR CMND        SIN(DLO)
        LDY =CBI
        JSR PUSH
        LDA =FMUL
        JSR CMND        COS(BI)*SIN(DLO)
        LDY =CO1
        JSR POP         CO1=COS(BI)*SIN(DLO)
*
*     CALCULATE 'CO2'
*
        LDA =COS
        JSR CMND        COS(DLO)
        LDY =CBI
        JSR PUSH
        LDA =FMUL
        JSR CMND        COS(BI)*COS(DLO)
        LDA =PTOF
        JSR CMND
        LDY =SB
        JSR PUSH
        LDA =FMUL
        JSR CMND        SIN(B)*COS(BI)*COS(DLO)
        LDY =COB
        JSR PUSH
        LDY =SBI
        JSR PUSH
        LDA =FMUL
        JSR CMND        COS(B)*SIN(BI)
        LDA =XCHF
        JSR CMND
        LDA =FSUB
        JSR CMND        COS(B)*SIN(BI)-SIN(B)*COS(BI)*COS(DLO)
        LDY =CO2
        JSR POP         C2=COS(B)*SIN(BI)-SIN(B)*COS(BI)*COS(DLO)
*
*     CALCULATE 'CO3'
*
        LDY =COB
        JSR PUSH
        LDA =FMUL
        JSR CMND        COS(B)*COS(BI)*COS(DLO)
        LDY =SB
        JSR PUSH
        LDY =SBI
        JSR PUSH
        LDA =FMUL
        JSR CMND        SIN(B)*SIN(BI)
        LDA =FADD
        JSR CMND        SIN(B)*SIN(BI)+COS(B)*COS(BI)*COS(DLO)
        LDY =CO3
        JSR POP         CO3=SIN(B)*SIN(BI)+COS(B)*COS(BI)*COS(DLO)
*
*     CALCULATE 'BA'
*
        LDY =CO1
        JSR PUSH
        LDY =CO2
        JSR PUSH
        LDA =FDIV
        JSR CMND        CO1/CO2
        LDA =ATAN
        JSR CMND        ATAN(CO1/CO2)
```

ORIGINAL PAGE IS OF POOR QUALITY

```
        LDA =PTOF
        JSR CMND
        LDY =BA
        JSR POP          BA=ATAN(CO1/CO2)
*
*   CALCULATE 'AA'
*
        LDA =PTOF
        JSR CMND
        LDA =COS
        JSR CMND         COS(BA)
        LDY =CO2
        JSR PUSH
        LDA =FMUL
        JSR CMND         CO2*COS(BA)
        LDA =XCHF
        JSR CMND
        LDA =SIN
        JSR CMND         SIN(BA)
        LDY =CO1
        JSR PUSH
        LDA =FMUL
        JSR CMND         CO1*SIN(BA)
        LDA =FADD
        JSR CMND         CO2*COS(BA)+CO1*SIN(BA)
        LDY =CO3
        JSR PUSH
        LDA =FDIV
        JSR CMND         (CO2*COS(BA)+CO1*SIN(BA))/CO3
        LDA =ATAN
        JSR CMND         ATAN((CO2*COS(BA)+CO1*SIN(BA))/CO3)
        LDA =PTOF
        JSR CMND
        LDY =AA
        JSR POP          AA=ATAN((CO2*COS(BA)+CO1*SIN(BA))/CO3)
*
*    CALCULATE 'SAA' AND 'CAA'
*
*
        LDA =PTOF
        JSR CMND
        LDA =SIN
        JSR CMND         SIN(AA)
        LDY =SAA
        JSR POP          SAA=SIN(AA)
        LDA =COS
        JSR CMND         COS(AA)
        LDY =CAA
        JSR POP          CAA=COS(AA)
*
*    CALCULATE 'MU'
*
        LDY =SB
        JSR PUSH
        LDY =SBI
        JSR PUSH
        LDA =FADD
        JSR CMND         SIN(B)+SIN(BI)
        LDA =PTOF
        JSR CMND
        LDA =FMUL
        JSR CMND         M=(SIN(B)+SIN(BI))**2
*
        LDY =ONE
        JSR PUSH
        LDY =CAA
        JSR PUSH
        LDA =FSUB
        JSR CMND         1.-COS(AA)
        LDY =SAA
        JSR PUSH
        LDA =FDIV
```

```
        JSR CMND        (1.-COS(AA))/SIN(AA)
        LDY =AA
        JSR PUSH
        LDY =SAA
        JSR PUSH
        LDA =FSUB
        JSR CMND        AA-SIN(AA)
        LDY =SAA
        JSR PUSH
        LDA =FDIV
        JSR CMND        (AA-SIN(AA))/SIN(AA)
        LDA =FMUL
        JSR CMND        U=(1.-COS(AA))/SIN(AA) * (AA-SIN(AA))/SIN(AA)
        LDA =FMUL
        JSR CMND        M*U
        LDY =MU
        JSR POP         MU=M*U
*
*     CALCULATE 'NV'
*
        LDY =SB
        JSR PUSH
        LDY =SBI
        JSR PUSH
        LDA =FSUB
        JSR CMND        SIN(B)-SIN(BI)
        LDY =SAA
        JSR PUSH
        LDA =FDIV
        JSR CMND        (SIN(B)-SIN(BI))/SIN(AA)
        LDA =PTOF
        JSR CMND
        LDA =FMUL
        JSR CMND        N=( (SIN(B)-SIN(BI))/SIN(AA) )**2
*
        LDY =ONE
        JSR PUSH
        LDY =CAA
        JSR PUSH
        LDA =FADD
        JSR CMND        1.+COS(AA)
        LDY =AA
        JSR PUSH
        LDY =SAA
        JSR PUSH
        LDA =FADD
        JSR CMND        AA+SIN(AA)
        LDA =FMUL
        JSR CMND        V=(1.+COS(AA))*(AA+SIN(AA))
        LDA =FMUL
        JSR CMND        N*V
        LDY =NV
        JSR POP         NV=N*V
*
*     CALCULATE 'RANGE'
*
        LDY =AA
        JSR PUSH
        LDY =MU
        JSR PUSH
        LDY =NV
        JSR PUSH
        LDA =FADD
        JSR CMND        MU+NV
        LDY =F1
        JSR PUSH
        LDA =FMUL
        JSR CMND        F1*(MU+NV)
        LDY =FOUR
        JSR PUSH
        LDA =FDIV
        JSR CMND        F1*(MU+NV)/4.
```

```
        LDA =FSUB
        JSR CMND          AA - F1*(MU+NV)/4.
        LDY =RCR2
        JSR PUSH
        LDA =FMUL
        JSR CMND          RCR2*( AA - F1*(MU+NV)/4. )
        LDY =ARC1
        JSR POP           ARC1=RCR2*(AA-F1*(MU+NV)/4.)
*
        LDY =ARC1
        LDA (BASE),Y      IF ARC1 IS NEGATIVE, CHANGE THE SIGN TO POSITIVE
        BPL L44
        JSR PUSH
        LDA =CHSF
        JSR CMND          ARC1=-ARC1
        LDY =ARC1
        JSR POP
*
*   CALCULATE 'BEARING'
*
*
L44     LDY =BA
        JSR PUSH
        LDY =CO2
        LDA (BASE),Y
        BPL L55           IF CO2>0, JUMP TO L55
        LDA =PUPI
        JSR CMND
        LDA =FADD
        JSR CMND          BA+180
        JMP L66
L55     LDY =CO1
        LDA (BASE),Y
        BPL L66           IF CO1>0, JUMP TO L66
        LDY =PAI2
        JSR PUSH
        LDA =FADD
        JSR CMND          BA+360
L66     LDA =PTOF
        JSR CMND
        LDY =PSI1
        JSR POP
        LDY =P18
        JSR PUSH
        LDA =FMUL
        JSR CMND
*
*   VIDEO DISPLAY FOR ARC1(RANGE) AND PSI1(BEARING)
*
        DEC VIDEO+1
        DEC BASE+1        BASE=$200
        LDX =2            LOCATION FOR BEARING
        LDA =$D7          VIDEO LOCATION FOR BEARING
        STA VY
        JSR RNGB
*
        INC BASE+1        BASE=$300
        LDY =ARC1
        JSR PUSH
        DEC BASE+1        BASE=$200
        LDX =0            LOCATION FOR RANGE
        LDA =$B7          VIDEO LOCATION FOR RANGE
        STA VY
        JSR RNGB
        INC VIDEO+1
        INC BASE+1        BASE=$300
*
*   CALCULATE GRI
*
        LDA FLG3          IS A WAYPOINT CHANGED?
        BEQ N55           IF FLAG3=0, GO TO N55
        LDX =0
```

```
        STX TMP         (TMP)=0
        STX TMP+1       (TMP+1)=0
        STX HELP        (HELP)=0
        STX HELP+1      (HELP+1)=0
        LDY =0
N1      LDA GRI,X       A=(GRI+X)
        CPY =0
        BEQ N2          IF FLG2=0, GO TO N2
        AND =$F0        UPPER BYTE
        LSR A
        LSR A
        LSR A
        LSR A
        DEY
        JMP N3          JUMP TO N3
N2      AND =$0F        LOWER BYTE
        INX             X=X+1
        INY
N3      CLC             CLEAR CARRY
        ADC TMP+1       A+(TMP+1)
        STA TMP+1       (TMP+1)=(TMP+1)+A
        BCC N4          IF CARRY=0, GO TO N4
        INC TMP         (TMP)=1
N4      CPX =2
        BEQ N5          IF X=2,  GO TO N5
        ASL TMP+1       (TMP+1)*2
        ROL TMP         (TMP)*2
        LDA TMP+1
        STA HELP+3      (HELP+3)=(TMP+1)
        LDA TMP
        STA HELP+2      (HELP+2)=(TMP)
        ASL TMP+1       (TMP+1)*2
        ROL TMP         (TMP)*2
        ASL TMP+1       (TMP+1)*2
        ROL TMP         (TMP)*2
        CLC             CLEAR CARRY
        LDA TMP+1
        ADC HELP+3
        STA TMP+1       (TMP+1)=(TMP+1)+(HELP+3)
        LDA TMP
        ADC HELP+2
        STA TMP         (TMP)=(TMP)+(HELP+2)
        JMP N1          JUMP BACK TO N1
N5      LDA TMP
        STA HELP+2
        LDA TMP+1
        STA HELP+3
        JMP BLNK1       JUMP TO BLNK1
*
*     CALCULATE CROSS TRACK ERROR IN DEGREE AND DISTANCE.
*
N55     LDY =PSI2
        JSR PUSH
        LDY =PSI1
        JSR PUSH
        LDA =FSUB
        JSR CMND        PSI2-PSI1
        LDA =PTOF
        JSR CMND
        LDY =CTEB
        JSR POP         CTEB=PSI2-PSI1
*
        LDA =1
        STA FLT         FLT=1
        LDY =CTEB
        LDA (BASE),Y
        BPL Z10         IF CTEB IS POSITIVE, BRANCH TO Z10
        LDA =0
        STA FLT         IF CTEB IS NEGATIVE, FLT=0
        LDA =CHSF       IF CTEB IS NEGATIVE, CTEB=-CTEB
        JSR CMND
*
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
Z10     LDA =PUPI
        JSR CMND
        LDA =FSUB
        JSR CMND        CTEB-PI
        LDA =PTOF
        JSR CMND
        LDY =CTEB1
        JSR POP
        LDA (BASE),Y
        BMI Z40         IF CTEB .LT. PI, BRANCH TO Z40
*
Z20     LDA =PUPI
        JSR CMND
        LDA =FSUB
        JSR CMND        CTEB-PI
        LDA =CHSF
        JSR CMND        CTEB=-CTEB
        LDA =PTOF
        JSR CMND
        LDY =CTEB
        JSR POP
*
Z30     LDA FLT
        BEQ Z40         IF CTEB IS NEGATIVE, GO TO Z40
        LDY =P18
        JSR PUSH
        LDA =FMUL       CONVERT TO DEGREE
        JSR CMND
        LDA =$0C
        STA $A116       INDICATE 'L' ON CRT SCREEN
        LDA =0
        STA CNT2        WHEN CNT2=0,'LEFT' INDICATION
        JMP Z50
*
Z40     LDA FLT
        BEQ Z30         IF CTEB IS NEGATIVE, GO TO Z30
        LDA =PTOF
        JSR CMND
        LDA =CHSF
        JSR CMND
        LDY =CTEB
        JSR POP
        LDY =P18
        JSR PUSH
        LDA =FMUL
        JSR CMND        CONVERT TO DEGREE
        LDA =$12
        STA $A116       INDICATE 'R' ON CRT SCREEN
        LDA =1
        STA CNT2        WHEN CNT2=1, 'RIGHT' INDICATION
*
Z50     DEC BASE+1      BASE=$200
        LDX =6          LOCATION FOR CROSS TRACK ERROR BEARING
        LDA =$37        VIDEO LOCATION FOR CTEB
        STA VY
        JSR RNGB
        INC BASE+1      BASE=$300
        LDY =CTEB
        JSR PUSH
        LDA =SIN
        JSR CMND        SIN(CTEB)
        LDY =ARC1
        JSR PUSH
        LDY =RCR2
        JSR PUSH
        LDA =FDIV
        JSR CMND        ARC1/RCR2
        LDA =SIN
        JSR CMND        SIN(ARC1/RCR2)
        LDA =FMUL
        JSR CMND        SIN(CTEB)*SIN(ARC1/RCR2)
        LDA =ASIN
```

```
        JSR CMND        ASIN(SIN(CTEB)*SIN(ARC1/RCR2))
        LDY =RCR2
        JSR PUSH
        LDA =FMUL
        JSR CMND        RCR2*ASIN(SIN(CTEB)*SIN(ARC1/RCR2))
        LDA =PTOF
        JSR CMND
        LDY =SB
        JSR POP
        LDY =SB
        LDA (BASE),Y
        BPL CTE3
        JSR PUSH
        LDA =CHSF
        JSR CMND
CTE3    DEC BASE+1      BASE=$200
        DEC FLAG
        LDX =8          LOCATION FOR CROSS TRACK ERROR
        LDA =$17        VIDEO LOCATION FOR CTE.
        STA VY
        JSR RNGB
        INC BASE+1      BASE=$300
*
*    CDI DISPLAY
*
        LDX =$C0
        LDA =00
CDI0    STA $A100,X     ERASE PREVIOUS CDI NEEDLE INDICATION
        INX
        CPX =$E0
        BCC CDI0
        LDA =3
        STA $A1CF       INDICATE CENTER POINT OF CDI
        LDA =01
        STA CNT1        CDI=0.1
        LDX =$CF
        LDY CNT2
        BNE CDI4        IF CNT2=1, CDI INDICATION ON RIGHT SIDE.
        LDA =$6A
        LDY GSP+8
        BEQ CDI1        IF CTE .LT. 10.0 NM, BRANCH TO CDI1
CDI00   LDX =$C4        INDICATE CDI=2.1 NM LEFT, WHEN CDI .GE. 10.0 NM.
        LDA =$55
        JMP CDI8
CDI1    LDY GSP+9
        CPY =$21
        BCS CDI00       IF CTE .GT. 2.1 NM, BRANCH BACK TO CDI00
        CPY CNT1
        BCC CDI8        IF CTE .LT. CNT1, BRANCH TO CDI8
        INC CNT1
        LDY CNT1
        CPY =$0A
        BNE CDI2        IF CDI .LT. $0A, BRANCH TO CDI2
        LDY =$10
        STY CNT1        CNT1=1.0
CDI2    CPY =$1A
        BNE CDI3        IF CDI .LT. $1A, BRANCH TO CDI3
        LDY =$20
        STY CNT1        CNT1=2.0
CDI3    EOR =$3F
        CMP =$55
        BNE CDI1
        DEX             DECREASE POSITION REGISTER
        CPX =$C5
        BCS CDI1
        JMP CDI8
CDI4    LDA =$55
        LDY GSP+8
        BEQ CDI5        IF CTE .LT. 10.0, BRANCH TO CDI5
CDI44   LDX =$DA        INDICATE CDI=2.1 NM RIGHT, WHEN CDI .GE. 10.0 NM
        LDA =$6A
        JMP CDI8
```

```
CD15    LDY GSP+9
        CPY =$21
        BCS CD144        IF CTE .GE. 2.1 NM, BRANCH TO CD144
        CPY CNT1
        BCC CD18         IF CTE .LT.CNT1, BRANCH TO CD18
        INC CNT1
        LDY CNT1
        CPY =$0A
        BNE CD16         IF CNT1 .LT. $0A, BRANCH TO CD16
        LDY =$10         CNT1=1.0
        STY CNT1
CD16    CPY =$1A
        BNE CD17         IF CNT1 .LT. $1A, BRANCH TO CD17
        LDY =$20         CNT1=2.0
        STY CNT1
CD17    EOR =$3F
        CMP =$6A
        BNE CD15
        INX              INCREASE POSITION REGISTER
        CPX =$0A
        BCC CD15
CD18    STA $A100,X      INDICATE CDI ON CRT SCREEN
*
*   DISPLAY GROUND SPEED AFTER 56 LOOPS
*
        LDA NOLP
        CMP =$70
        BEQ AV1
        INC NOLP
        INC NOLP
        JMP BOTM3
*
*   CALCULATE GROUND SPEED
*
AV1     LDA CNT4
        CMP =4           COLLECT 4 REFERENCES?
        BEQ AV6          IF YES, BRANCH TO AV6
        LDY =ARC1
        JSR PUSH
        LDY =CTEB
        JSR PUSH
        LDY CTEBN
        JSR POP
        LDY ARCN
        JSR POP
*
        LDX CNT4         X=CNT4
        LDA GRI1         READ GRI LOOP COUNT
        STA GRIN,X       (GRIN+X)=GRI1
        CLC              CLEAR CARRY
        ADC GRIT+3       (GRIT+3)+A
        BCC AV4          IF CARRY=0,  GO TO AV4
        INC GRIT+2       (GRIT+2)=1
AV4     STA GRIT+3       (GRIT+3)=(GRIT+3)+(GRIN+X)
        JMP BOTM2
*
AV6     LDA ARCN
        CMP =$D0         ADDDRESS POINTER REACH TO $D0
        BNE AV7          IF ARCN=$D0, NO BRANCH
        LDA =$C0
        STA ARCN         ARCN=$C0
        LDA =$D0
        STA CTEBN        CTEBN=$D0
        LDA =0
        STA CNT5         CNT5=0
AV7     LDY =ARC1
        JSR PUSH
        LDY ARCN
        JSR PUSH         PUSH ARCN
        LDY =ARCO
        JSR POP          ARCO=ARCN
        LDY ARCN
```

ORIGINAL PAGE IS OF POOR QUALITY

```
        JSR POP         ARCN=ARC1
*
        LDX CNT5        X=CNT5
        LDA GRIT+3
        SEC             CLEAR CARRY
        SBC GRIN,X      (GRIT+3)-(GRIN+X)
        BCS AV8         IF CARRY=0,  GO TO AV8
        DEC GRIT+2      (GRIT+2)=0
AV8     CLC             CLEAR CARRY
        ADC GRI1
        BCC AV9         IF CARRY=0,  GO TO AV9
        INC GRIT+2      (GRIT+2)=1
AV9     STA GRIT+3      (GRIT+3)=(GRIT+3)-(GRIN+X)+(GRI1)
        LDA GRI1
        STA GRIN,X
*
        LDY =CTEB
        JSR PUSH
        LDY CTEBN
        JSR PUSH        PUSH CTEBN
        LDY =CTEBO
        JSR POP         CTEBO=CTEBN
        LDY CTEBN
        JSR POP         CTEBN=CTEB
*   CALCULATE TIME=GRI*(NO.OF GRI)
CALTM   LDY =HELPP
        JSR PUSH
        LDY =GRITT
        JSR PUSH        READ TOTAL GRI LOOP COUNT
        LDA =DMUL
        JSR CMND        GRI*GRITT
        LDA =FLTD
        JSR CMND
        LDY =D36E6
        JSR PUSH
        LDA =FDIV
        JSR CMND        CONVERT IN HOUR
        LDY =TIME
        JSR POP
*
*   CALCULATE AVERAGE GROUND SPEED
*
GS1     LDY =ARCO
        JSR PUSH
        LDY =ARC1
        JSR PUSH
        LDA =FSUB
        JSR CMND        ARCO-ARC1
        LDA =PTOF
        JSR CMND
        LDA =FMUL
        JSR CMND        (ARCO-ARC1)**2
        LDY =CTEBO
        JSR PUSH
        LDY =CTEB
        JSR PUSH
        LDA =FSUB
        JSR CMND        CTEBO-CTEB
        LDY =ARC1
        JSR PUSH
        LDA =FMUL
        JSR CMND        ARC1*DCTEB
        LDA =PTOF
        JSR CMND
        LDA =FMUL
        JSR CMND        (ARC1*DCTEB)**2
        LDA =FADD
        JSR CMND        (ARCO-ARC1)**2+(ARC1*DCTEB)**2
        LDA =SQRT
        JSR CMND        SQRT((ARCO-ARC1)**2+(ARC1*DCTEB)**2)
        LDY =TIME
        JSR PUSH
```

```
        LDA =FDIV
        JSR CMND        SQRT((ARCO-ARC1)**2+(ARC1*DCTEB)**2)/TIME
*
        LDA =PTOF
        JSR CMND
        LDY =F20
        JSR PUSH
        LDA =FSUB
        JSR CMND
        LDY =C01
        JSR POP
        LDY =C01
        LDA (BASE),Y
        BPL GS2         IF GS .GE. 0.0, BRANCH TO GS2
        LDY =GSPRD
        JSR PUSH
        LDA =CHSF
        JSR CMND        -GSPRD
        JMP ABGS
*
GS2     LDY =GSPRD
        JSR PUSH
        LDA =FSUB
        JSR CMND        GSOBS-GSPRD
*
*   ALPHA BETA FILTER FOR GS
*
ABGS    LDA =PTOF
        JSR CMND
        LDY =ALPG
        JSR PUSH
        LDA =FMUL
        JSR CMND        ALPG*(GSOGS-GSPRD)
        LDY =GSPRD
        JSR PUSH
        LDA =FADD
        JSR CMND        GSPRD+ALPG*(GSOBS-GSPRD)
        LDY =GSSM
        JSR POP         GSSM=GSPRD+ALPG*(GSOBS-GSPRD)
*
        LDY =BETG
        JSR PUSH
        LDA =FMUL
        JSR CMND        BETG*(GSOBS-GSPRD)
        LDY =TIME
        JSR PUSH
        LDY =FOUR
        JSR PUSH
        LDA =FDIV
        JSR CMND
        LDA =FDIV
        JSR CMND        BETG*(GSOBS-GSPRD)/TIME
        LDY =ACPRD
        JSR PUSH
        LDA =FADD
        JSR CMND        ACPRD+BETG*(GSOBS-GSPRD)/TIME
        LDA =PTOF
        JSR CMND
        LDY =ACPRD
        JSR POP
*
        LDY =TIME
        JSR PUSH
        LDY =FOUR
        JSR PUSH
        LDA =FDIV
        JSR CMND
        LDA =FMUL
        JSR CMND        TIME*ACSM
        LDY =GSSM
        JSR PUSH
        LDA =FADD
```

```
        JSR CMND        GSSM+TIME*ACSM
*
GS3     LDA =PTOF
        JSR CMND
        LDY =GS
        JSR POP         GS=GSPRD+GAIN*(GSOBS-GSPRD)
        LDA =PTOF
        JSR CMND
        LDY =GSPRD
        JSR POP         GSPRD=GS
        LDY =GS
        LDA (BASE),Y
        BPL GS5         IF GS IS NEGATIVE,
        LDA =CHSF
        JSR CMND        GS=-GS
GS4     LDA =PTOF
        JSR CMND
        LDY =GS
        JSR POP
GS5     DEC BASE+1      BASE=$200
        LDA =1
        STA LOPG
        LDX =4          LOCATION FOR GROUND SPEED.
        LDA =$78        VIDEO LOCATION FOR GS.
        STA VY
        JSR RNGB        DISPLAY GS ON CRT SCREEN
        INC BASE+1
*
*
*   CALCULATE ESTIMATE TIME OF ARRIVAL
*
ETOA00  LDA GSP+4
        BNE ETOA2
        LDA GSP+5
        BNE ETOA2
        LDX =0
        LDA =$20
ETOA0   LDY BLANK3,X    BLANK DISPLAY FOR ETA, WHEN GS=0.
        STA $A100,Y
        INX
        CPX =6
        BCC ETOA0
        JMP BOTM
ETOA2   LDY =ARC1
        JSR PUSH
        LDY =GS
        JSR PUSH
        LDA =FDIV
        JSR CMND        ETA=ARC1/GS
        DEC BASE+1
        DEC FLAG
        LDX =6          LOCATION FOR ESTIMATE TIME OF ARRIVAL.
        LDA =9          INDEX LIMIT
        STA XLIM
        LDA =$98        VIDEO LOCATION FOR ETA.
        STA VY
        JSR TOBCD2      DISPLAY ETA ON CRT SCREEN.
        INC BASE+1
        JMP BOTM
BOTM2   INC CNT4        CNT4=CNT4+1
BOTM    INC CNT5        CNT5=CNT5+1
        LDY =4
AV10    INC ARCN        ARCN=ARCN+4
        INC CTEBN       CTEBN=CTEBN+4
        DEY
        BNE AV10        IF Y=0,  NO BRANCH
BOTM3   SED             SET DECIMAL MODE FOR SENSOR
        RTS             RETURN TO MAIN PROGRAM
*
BLNK1   LDX =0
BLNK2   LDA =$20        BLANK DISPLAY FOR GS. , ETA. , CTEB. AND CTE.,
        LDY BLANK1,X    WHEN WAYPOINT IS CHANGED.
```

```
        STA $A100,Y
        INX
        CPX =18
        BCC BLNK2
        LDY =PSI1
        JSR PUSH
        LDY =PSI2
        JSR POP         STORE THE DESIRED COURSE BEARING
        LDY =ARC1
        JSR PUSH
        LDY =ARC2
        JSR POP         STORE THE DESIRED COURSE RANGE
        LDA =0
        STA CNT5
        STA CNT4
        STA LOPG
        STA NOLP
        LDY =$B0
REF     STA $0300,Y
        INY
        CPY =$BC
        BNE REF
        LDA =$C0
        STA ARCN        ARCN=$C0
        LDA =$D0
        STA CTEBN       CTEBN=$D0
        SED             SET DECIMAL MODE FOR SENSOR
        RTS             RETURN TO THE MAIN PROGRAM.
*
*********************************************************************
*                                                                   *
*       THIS SUBROUTINE INITIALIZES THE PIA FOR USE WITH            *
*       THE MATH CHIP AND THEN SETS THE CONTROL INPUTS OF THE       *
*       MATH CHIP TO INACTIVE STATES.                               *
*                                                                   *
*********************************************************************
*
PINT    LDA PIAA        CLEAR INTERRUPTS
        LDA PIAB
        LDA =$14
        STA PIAA+1      SET INTERRUPT CONTROL AND DDR
        LDA =0
        STA PIAB+1      SET DDR LOW
        LDA =$CF
        STA PIAB        SET INPUTS AND OUTPUTS FOR 9511
        LDA =4
        STA PIAB+1      SET DDR BIT HIGH
        LDA =7
        STA AGCB        SET BACKGROUND COORDINATION BYTE
        STA PIAB        SET CD, RD AND WR HIGH
        LDA =$F
        STA PIAB        SET SVACK HIGH
        LDA PIAA        CLEAR ANY INTERRUPTS
        LDA =0          GET LSB OF NUMBER TABLE
        STA BASE        SAVE FOR INDEXING
        LDA =2          GET MSB
        STA BASE+1
        LDA =$A
        STA DVSR        STORE 10 FOR BINARY DIVISION
        LDA =0
        STA VIDEO
        LDA =$A1
        STA VIDEO+1
*
*       MOVE NUMBER TABLE TO READ/WRITE SPACE
*
        LDY =0          CLEAR INDEX ONE
        LDX =0          CLEAR INDEX TWO
        LDA TABLE
        STA BASE1
        LDA TABLE+1
        STA BASE1+1
```

```
MVE1    LDA (BASE1),Y    GET A DIGIT
        STA $200,X       STORE IT IN NEW LOCATION
        INX
        INY
        CPY =124         MOVE 120 YET?
        BCC MVE1         IF NOT, CONTINUE
*
        LDY =0
        LDX =0
        LDA TABLE2
        STA BASE1
        LDA TABLE2+1
        STA BASE1+1
MVE2    LDA (BASE1),Y
        STA $300,X
        INX
        INY
        CPY =56
        BCC MVE2
*
        LDY =0
        LDX =0
        LDA TABLE3
        STA BASE1
        LDA TABLE3+1
        STA BASE1+1
MVE3    LDA (BASE1),Y
        STA $B0,X
        INX
        INY
        CPY =16
        BCC MVE3
*
*           INITIALIZE VIDEO DISPLAY OUTPUT
*
        LDX =0
WRSCN   LDY SCLC,X
        LDA LSCRN,X
        AND =$3F
        STA $A000,Y
        INX
        CPX =27
        BCC WRSCN
        LDX =0
WRSN2   LDY SCLC2,X
        LDA LSCN2,X
        AND =$3F
        STA $A100,Y
        INX
        CPX =36
        BCC WRSN2
*
        RTS
*
*
*************************************************************************
*                                                                       *
*       SUBROUTINE TO SEND DATA TO 9511.                                *
*       ITEM NUMBER IS IN REG-Y.  BASE ADDRESS                          *
*       IS IN 'BASE.'                                                   *
*                                                                       *
*************************************************************************
*
PUSH    JSR SAO          SET PIAA TO OUTPUTS
        INY              ADJUST REG-Y SO IT
        INY              POINTS TO LSB OF NUMBER
        INY
        LDX =4           LOAD COUNT OF 4
PSH1    LDA (BASE),Y     GET A BYTE OF THE NUMBER
        STA PIAA         GIVE IT TO 9511
        LDA =$A
        JSR SV10
```

```
        LDA PIAA        CLEAR ANY INTERRUPTS
        DEY             NEXT BYTE
        DEX
        BNE PSH1        LOOP UNTIL ENTIRE WORD WRITTEN
        RTS             IF DONE, RETURN
*
*
*********************************************************************
*                                                                   *
*       SUBROUTINE SETS UP PIA SIDE A AS OUTPUTS.                   *
*                                                                   *
*********************************************************************
*
SAO     LDA PIAA+1      GET THE CONTROL REGESTER
        AND =$FB        SET ACCESS THE DDR
        STA PIAA+1      RETURN IT
        LDA =$FF
        STA PIAA        SET ALL TO OUTPUTS
        LDA PIAA+1
        ORA =4          SET DDR BIT HIGH
        STA PIAA+1      RETURN IT
        RTS
*
*********************************************************************
*                                                                   *
*       THIS SUBROUTINE POPS A NUMBER OFF OF THE                    *
*       9511 STACK.  NUMBER IS RETURNED TO LOCATION                 *
*       WITH 'BASE' AS BASE ADDRESS; ITEM NUMBER IN Y.              *
*                                                                   *
*********************************************************************
*
POP     JSR SAI         SET PIA AS INPUTS
        LDX =4          LOAD COUNT OF 4
POP1    LDA =9
        JSR SV9
        STA (BASE),Y    STORE IN TABLE
        LDA =$B
        STA AGCB
        ORA AGCF
        STA PIAB        SET RD HIGH TO INCR. STACK POINTER
        INY
        DEX
        BNE POP1        DO 4 BYTES
        RTS
*
*
*********************************************************************
*                                                                   *
*       SUBROUTINE SETS UP PIAA AS INPUTS.                          *
*                                                                   *
*********************************************************************
*
SAI     LDA PIAA+1
        AND =$FB        SET DDR BIT LOW
        STA PIAA+1
        LDA =0
        STA PIAA        SET SIDE A TO ALL INPUTS
        LDA PIAA+1
        ORA =4          SET DDR BIT HIGH
        STA PIAA+1
        RTS
*
*
*********************************************************************
*                                                                   *
*       SUBROUTINE SENDS THE COMMAND BYTE IN ACCUM.                 *
*       TO THE 9511 FOR EXECUTION.  ROUTINE RETURNS TO              *
*       CALLER REGARDLESS OF WHETHER EXECUTION IS COMPLETED         *
*       OR NOT.                                                     *
*                                                                   *
*********************************************************************
*
```

```
CMND    PHA
        JSR SAO         SET PIA SIDE A AS OUTPUTS
        PLA             GET THE COMMAND
        STA PIAA        SEND TO 9511
        LDA =$E
        JSR SV10
        BIT PIAA+1      TEST IF COMMAND DONE
        BPL *-3         KEEP TESTING UNTIL DONE
        LDA PIAA        CLEAR THE INTERRUPT BIT
*
*           READ THE STATUS REGESTER; RETURN IF INVALID CODE
*           WAS PRODUCED.
*
        JSR SAI         SET PIAA SIDE A AS INPUTS
        LDA =$D         SET C/D HIGH, RD LOW
        JSR SV9
        PHA             SAVE IT
        LDA =$F         RETURN 9511 TO INACTIVE STATE
        STA AGCB
        ORA AGCF
        STA PIAB
        PLA
        AND =%10011110  ZERO OUT UNIMPORTANT BITS
        BEQ OK          IF ZERO, CONTINUE PROCESSING
        PLA             POP THE STACK
        PLA
        SED             SET DECIMAL FOR SENSOR ROUTINE
OK      RTS             RETURN
*
*
SV9     STA AGCB
        ORA AGCF
        STA PIAB
        BIT PIAA+1
        BVC *-3
        LDA PIAA
        RTS
*
*
SV10    PHA
        STA AGCB
        ORA AGCF
        STA PIAB
        PLA
        CLC
        ADC =1
        STA AGCB
        ORA AGCF
        STA PIAB
        RTS
*
*
************************************************************************
*                                                                      *
*       THIS ROUTINE CONVERTS THE BINARY FLOATING-POINT LATITUDE       *
*       AND LONGITUDE SEPARATELY INTO THE STANDARD DEGREE, MINUTES,    *
*       AND SECOND FORMAT.  THE RESULT IS STORED IN BCD.               *
*                                                                      *
************************************************************************
*
INTG    CLD             SET DECIMAL MODE OFF
        LDA =PTOF
        JSR CMND        DUPLICATE STACK LOCATIONS
        LDA =FIXD
        JSR CMND        CONVERT POSITION TO AN INTEGER
        LDA =PTOF
        JSR CMND        DUPLICATE IT
        LDY =TEMP       WORK AREA
        STX XTEMP
        JSR POP
        DEY
        LDA (BASE),Y    GET THE HEX RESULT
```

```
        STA DVDN+1       AND PLACE FOR HEX-TO-BCD CONVERSION
        DEY
        LDA (BASE),Y
        STA DVDN
*
*           ROUTINE TO CONVERT A TWO-BYTE HEX NUMBER TO A FIVE-BYTE BCD
*           NUMBER.
*
        LDX =4           COUNT OF FOUR
*
*           DIVIDE BY TEN
*
UNSPD   LDA =0
        STA RMNDR        CLEAR REMAINDER
        LDY =17          SET UP COUNT
        JMP D01
D02     LDA RMNDR
        SEC              SET CARRY FOR  SUBTRACT
        SBC DVSR
        BPL NREST        GO IF NO RESTORE
D01     CLC
        JMP MERGQ        GO TO SET Q
NREST   STA RMNDR        NEW RESIDUE
        SEC              Q=1
MERGQ   ROL DVDN+1
        ROL DVDN
        DEY              DECREMENT COUNT
        BEQ RTN
        ROL RMNDR        SHIFT LEFT
        JMP D02          CONTINUE
*
RTN     LDA RMNDR        GET REMAINDER
        STA DRES,X       STORE IT
        DEX
        BPL UNSPD        DO UNTIL DONE
*
        LDX XTEMP
        RTS
*
*
TOBCD2  JSR INTG         GET THE INTEGER PART
*
*           STORE THE POSITION COORDINATES ON VIDEO SCREEN
*
        LDY VY           GET VIDEO LOCATION
        LDA DRES+3       LOAD MSB
        ORA =$30         CHANGE TO ASCII
        STA (VIDEO),Y
        INY
        LDA DRES+4       GET LSB
        ORA =$30
        STA (VIDEO),Y
TOB22   INY
        INY
        STY VY
*
        LDA DRES+3
        ASL A            MOVE LOWER FOUR BITS TO UPPER FOUR BITS
        ASL A
        ASL A
        ASL A
        STA LAT,X        STORE THE DIGIT IN POSITION FIELD
        LDA DRES+4       GET NEXT DIGIT
        ORA LAT,X        MERGE WITH UPPER DIGIT
        STA LAT,X        REPLACE
TOB33   INX
        CPX XLIM         SEE IF DONE
        BEQ OUT2         RETURN IF DONE
        LDA =FLTD
        JSR CMND         CHANGE INTEGER TO FLOATING-POINT
        LDA =FSUB
        JSR CMND         SUBTRACT OFF INTEGER PART
```

```
        LDY =C60            GET CONSTANT "60"
        STX XTEMP
        JSR PUSH
        LDX XTEMP
        LDA =FMUL
        JSR CMND            MULTIPLY RESIDUE BY 60
        JMP TOBCD2
OUT2    RTS
*
*
**********************************************************************
*                                                                    *
*       ROUTINE CONVERTS RANGE AND BEARING FROM BINARY TO BCD        *
*       AND STORES THEM.                                             *
*                                                                    *
**********************************************************************
*
RNGB    JSR INTG            GET THE INTEGER PART
*
*           VIDEO OUTPUT
*
        LDY VY
        LDA DRES+2
        ORA =$30
        STA (VIDEO),Y
        INY
        LDA LOPG
        BEQ RTNOO           IF THE DISPLAY NOT FOR GS, BRANCH TO RTNOO
        LDA DRES+2
        BNE RTNOO
        LDA [illegible]S+3
        CMP =[illegible]
        BCS RTNOO           IF GS .LE. 30 NM, BRANCH TO RTNOO
        LDA =0
        STA DRES+3
        STA DRES+4
RTNOO   LDA DRES+3
        ORA =$30
        STA (VIDEO),Y
        INY
RTNO    LDA DRES+4
        ORA =$30
        STA (VIDEO),Y
RTN1    INY
        INY
        STY VY
*
        LDA DRES+2          3RD ORDER DIGIT
        ASL A               MOVE LOWER DIGIT TO UPPER
        ASL A
        ASL A
        ASL A
        STA GSP,X           PUT IT IN RESULT
        LDA DRES+3          2ND ORDER DIGIT
        ORA GSP,X           MERGE WITH LAST DIGIT
        STA GSP,X
        INX
        LDA DRES+4          FIRST DIGIT
        ASL A
        ASL A
        ASL A
        ASL A
        STA GSP,X           STORE IT
        LSR LOPG
        BCS RTN2
        LDA =FLTD
        JSR CMND            CHANGE TO FLOATING-POINT
        LDA =FSUB
        JSR CMND            SUBTRACT OFF INTEGER
        LDY =C256           "256"
        STX XTEMP
        JSR PUSH
```

```
        LDA =FMUL
        JSR CMND            GET FRACTION AS INTEGER
        LDA =FIXD
        JSR CMND
        LDY =TEMP           WORK AREA
        JSR POP
        DEY
        LDA (BASE),Y        GET THE FRACTION
        LSR A               SEARCH-TABLE METHOD OF
        LSR A               OBTAINING CORRESPONDING
        LSR A               DECIMAL VALUE
        LSR A
        TAX                 USE AS INDEX
        LDA FRTBL1,X        GET THE DECIMAL EQUIVALENT
        LDX XTEMP
        LDY VY
        ORA =$30
        STA (VIDEO),Y
        AND =$F
        ORA GSP,X           STORE THE FRACTION
        STA GSP,X
RTN2    RTS
*
*
***********************************************************************
*                                                                     *
*       CONSTANT TABLE OF NUMBERS USED IN CALCULATIONS.               *
*                                                                     *
***********************************************************************
*
TABLE   ADR (CNTS)
CNTS    EQU *
        HEX 7C,9F,BE,77   00 - TCY
        HEX 7C,DD,2F,1B   04 - TCZ
        HEX 7E,9A,EC,71   08 - THMY
        HEX 7E,98,0F,39   0C - THMZ
        HEX 06,BB,F2,6E   10 - XNR
        HEX 00,FD,13,63   14 - CTMY
        HEX 7E,9A,55,50   18 - STMY
        HEX 00,FD,2E,C3   1C - CTMZ
        HEX 7E,97,80,51   20 - STMZ
        HEX 7F,BF,D3,EB   24 - CXK
        HEX 00,ED,5A,69   28 - SXK
        HEX 02,BE,EC,DD   2C - C1
        HEX 03,89,F5,17   30 - C2
        HEX 83,E5,7E,A9   34 - C3
        HEX 03,82,DE,4C   38 - C4
        HEX 83,8E,DC,13   3C - C5
        HEX 01,93,27,AA   40 - C6
        HEX 83,A1,A2,81   44 - C7
        HEX 03,A3,CA,CE   48 - C8
        HEX 00,8F,28,B3   4C - C9
        HEX 77,CF,C0,08 50 - C10
        HEX 75,87,75,21 54 - C11
        HEX F8,C2,1E,A6 58 - C12
        HEX 01,80,6F,75 5C - C14
        HEX 69,D6,BF,95 60 - 1E-7
        HEX 09,80,00,00   6C - C256 ("256")
        HEX 06,E5,2E,E1   70 - P180 (180/PI)
        HEX 06,F0,00,00   74 - C60 (60)
        HEX 7E,AB,58,08 -ALP
        HEX 79,E5,5D,67 -BET
        HEX 01,B2,7B,B3 -TM
*
*       END OF NUMBER TABLE; START SCRATCH SPACE
*
FRTBL1 HEX 00,01,01,02,03,03,04,04
        HEX 05,06,06,07,08,08,09,09
LSCRN   ASC 'FROMWP#TOWP#RANG.NMBRNG.DEG'
SCLC    HEX 0D,0E,0F,10,12,13,14
        HEX 4E,4F,52,53,54
        HEX B1,B2,B3,B4,BA,BD,BE
```

```
        HEX D1,D2,D3,D4,DA,DD,DE,DF
LSCN2   ASC 'CTE.NMCTEB.DEG'
        ASC 'GSNM/HETA::CDI21012NM'
        HEX 03
SCLC2   HEX 11,12,13,1A,1D,1E
        HEX 31,32,33,34,3A,3D,3E,3F
        HEX 71,72,7C,7D,7E,7F
        HEX 91,92,93,9A,9D
        HEX E0,E1,E2,E5,EA,EF,F4,F9,FE,FF
        HEX CF
BLANK   HEX B7,B8,B9,BB
        HEX D7,D8,D9,DB
        HEX 66,67,69,6A,6C,6D
        HEX 86,87,89,8A,8C,8D
BLANK1  HEX 16,17,18,19,1B
        HEX 37,38,39,3B
BLANK2  HEX 78,79,7A
BLANK3  HEX 98,99,9B,9C,9E,9F
*
*    CONSTANT TABLE OF NUMBERS USED IN CALCULATIONS
*
TABLE2  ADR (CST)
CST     EQU *
        HEX 0C,D7,3E,AE -RCR2
        HEX 78,DB,BA,50 -F1
        HEX 00,FF,24,46 -F2
        HEX 01,80,00,00 -ONE
        HEX 03,80,00,00 -FOUR
        HEX 12,C9,6E,34 -D3618
        HEX 1A,89,54,40 -D36E6
        HEX 05,A0,00,00 -F20
        HEX 06,E5,2E,E1 -P18
        HEX 03,C9,0F,DA -PA12
        HEX 09,B4,00,00 -P12D
        HEX 00,00,00,3C -D60
        HEX 7D,AB,58,08 -ALFG
        HEX 77,E5,5D,92 -BETG
TABLE3  ADR (CST3)
CST3    EQU *
        HEX 10,39,16,20 -WP1(BIASED UNI NDB)
        HEX 00,82,07,50
        HEX 20,39,13,44 -WPT2(BIASED THRSH - O.U.AIRPORT)
        HEX 00,82,13,55
*
        END
```

APPENDIX D. Program Listing for Testing Flight Test Data.

This program converts time differences to coordinates of position and calculates area navigation information (range, bearing, nonfiltered ground speed and filtered ground speed). It is written in standard Fortran IV programming language and run in the IBM4341.

```
C************************************************************************
C    THIS PROGRAM CONVERT TDS TO COORDINATES OF POSITION.
C    AFTER THE CONVERSION, AREA NAVIGATIONAL INFORMATIONS ARE COMPUTED.
C    (RANGE,BEARING AND GROUND SPEED)
C
C                                  MARCH/1983   FUJIKO OGURI
C************************************************************************
C
      DIMENSION TH(2),POS(2),IPOS(2),RANGE(1000),BEAR(1000)
      DIMENSION VLS(2),VLP(2)
      INTEGER EVENT
      PI=3.14159265
      J=1
      T=0.996*1.4
      VLP(1)=0.
      VLP(2)=0.
      TF=12.
      ALP=0.72*T/TF
      BET=ALP**2/4.
      TIME=T/3600.
   10 READ (1,100,END = 300) TH,EVENT,GSD
  100 FORMAT (F9.2,1X,F9.2,1X,I2,34X,F5.1)
C
C    CONVERT TIME DIFFERENCES TO COORDINATES OF POSITION
C
      CALL DEXLRN(TH,POS)
      DO 20 I=1,2
      IPOS(I) = IFIX(POS(I)) * 10000
      TMP = (POS(I) - IFIX(POS(I))) * 60.
      IPOS(I) = IPOS(I) + (IFIX(TMP) * 100)
      TMP = (TMP - IFIX(TMP)) * 60.
   20 IPOS(I) = IPOS(I) + IFIX(TMP)
C
      IF(J.EQ.1001) GO TO 300
C
C    RANGE AND BEARING ANGLE CALCULATION
C
      CALL RABE(POS, RANGE,BEAR,J)
      IF(J.EQ.1) GO TO 9
C
C   NONFILTERED GROUND SPEED CALCULATION
C
      DRANGE=RANGE(J)-RANGE(J-1)
      DBEAR=(BEAR(J)-BEAR(J-1))*PI/180.
      IF(DBEAR.GE.4.46804) DBEAR=2.*PI-DBEAR
      IF(DBEAR.LE.-4.46804) DBEAR=2.*PI+DBEAR
      GSO=SQRT(DRANGE**2+(RANGE(J)*DBEAR)**2)/TIME
      IF(J.NE.2) GO TO 199
      GSP=GSO
      ACP=0.0
      GO TO 9
C
C    FILTERED GROUND SPEED CALCULATION, INTERVAL=1 CYCLE
C
  199 GSS=GSP+ALP*(GSO-GSP)
      ACS=ACP+BET*(GSO-GSP)/TIME
      GSF=GSS+TIME*ACS
      GSP=GSF
      ACP=ACS
      GSF=ABS(GSF)
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      IF(J.LE.4) GO TO 9
C
C     FILTERED GROUND SPEED CALCULATION, INTERVAL=4 CYCLES
C
      DRANGE=RANGE(J)-RANGE(J-4)
      DBEAR=(BEAR(J)-BEAR(J-4))*PI/180.
      IF(DBEAR.GE.4.46804) DBEAR=2.*PI-DBEAR
      IF(DBEAR.LE.-4.46804) DBEAR=2.*PI+DBEAR
      GS40=SQRT(DRANGE**2+(RANGE(J)*DBEAR)**2)/(TIME*4.)
      IF(J.EQ.5) GS4P=GS40
      IF(J.EQ.5) AC4P=0.0
      GS4S=GS4P+ALP*(GS40-GS4P)
      AC4S=AC4P+BET*(GS40-GS4P)/TIME
      GS4=GS4S+AC4S*TIME
      GS4P=GS4
      IF(J.LE.16) GO TO 9
C
C     FILTERED GROUND SPEED CALCULATION, INTERVAL=16 CYCLES
C
      DRANGE=RANGE(J)-RANGE(J-16)
      DBEAR=(BEAR(J)-BEAR(J-16))*PI/180.
      IF(DBEAR.GE.4.46804) DBEAR=2.*PI-DBEAR
      IF(DBEAR.LE.-4.46804) DBEAR=2.*PI+DBEAR
      GSAO=SQRT(DRANGE**2+(RANGE(J)*DBEAR)**2)/(TIME*16.)
      IF(J.EQ.17) GSAP=GSAO
      IF(J.EQ.17) ACAP=0.0
      GSAS=GSAP+ALP*(GSAO-GSAP)
      ACAS=ACAP+BET*(GSAO-GS4P)/TIME
      GSA=GSAS+ACAS*TIME
      GSAP=GSA
      WRITE(2,200) TH,EVENT,IPOS,RANGE(J),BEAR(J),GSD,GSO,GSF,GSA,GS4
  200 FORMAT (F9.2,1X,F9.2,1X,I2,1X,I6,1X,I6,8X,F5.1,1X,F5.1,1X,5F8.1)
    9 J=J+1
      GO TO 10
  300 STOP
      END
C
C     THIS SUBROUTINE CONVERTS TIME DIFFERENCES TO COORDINATES OF
C     POSITION.
C
      SUBROUTINE DEXLRN(TH,POS)
      DIMENSION TH(2),POS(2)
      DATA TCY,TCZ/3.9E-2,5.4E-2/
      DATA THMY/0.15129258/,THMZ/0.14849557/,XNR/46.986746/
      DATA CTMY/0.98857709/,STMY/0.15071607/,CTMZ/0.98899478/
      DATA STMZ/0.14795043/,CXK/0.37466368/,SXK/0.92716079/
      DATA C1/2.9832071/,C2/4.3111683/,C3/-7.1717116/
      DATA C4/4.0896360/,C5/-4.4643647/,C6/1.1496479/
      DATA C7/-5.0510869/,C8/5.1185063/,C9/0.55921480/
      DATA C10/1.5850077E-3/,C11/2.5836473E-4/,C12/-2.9620318E-3/
      DATA C13/1.5850077E-3/,C14/1.0034014/
      DATA PI/3.141592/
C
      TY=TH(1)*1.E-6
      TZ=TH(2)*1.E-6
C
      PY=XNR*(TY-TCY)-THMY
      PZ=XNR*(TZ-TCZ)-THMZ
      CPY=COS(PY)
      SPY=SIN(PY)
      CPZ=COS(PZ)
      SPZ=SIN(PZ)
      AY=(CPY-CTMY)/STMY
      AZ=(CPZ-CTMZ)/STMZ
      BY=SPY/STMY
      BZ=SPZ/STMZ
      U1=AY*CXK-AZ
      U2=AY*SXK
      U3=AZ*BY-AY*BZ
      UU=U1*U1+U2*U2
      CDBY=(U3*U1+U2*SQRT(UU-U3*U3))/UU
      THMS=ATAN(AY/(BY+CDBY))
```

```
      CB=COS(THMS)
      CA=COS(THMS+PY)
      CC=COS(THMS+PZ)
C
      F=C1*CA+C2*CB+C3*CC
      G=C4*CA+C5*CB+C6*CC
      H=C7*CA+C8*CB+C9*CC
C
      THGS=ATAN((G+C10)/(F+C11))
      PHGS=ATAN(C14*C14*SIN(THGS)*(H+C12)/(G+C13))
      POS(2)=THGS*180./PI
      POS(1)=PHGS*180./PI
      RETURN
      END
C
C     THIS SUBROUTINE CALCULATES RANGE TO THE WAYPOINT AND BEARING
C     TO TRUE NORTH.
C
      SUBROUTINE RABE(POS, RANGE, BEAR,J)
      DIMENSION POS(2),RANGE(1000),BEAR(1000)
      REAL LA1,LO1,LA2,LO2,M,N
      DATA PI/3.1415926535898/,A/3443.9174/
      LA2=0.68467327
      LO2=1.43521818
C
      LA1=POS(1)*PI/180.
      LO1=POS(2)*PI/180.
      FF=1. - 1./298.2
      F=1./298.2
      DLO=LO1-LO2
      TB=FF*TAN(LA1)
      TBI=FF*TAN(LA2)
      B=ATAN(TB)
      BI=ATAN(TBI)
      C1=COS(BI)*SIN(DLO)
      C2=COS(B)*SIN(BI)-SIN(B)*COS(BI)*COS(DLO)
      C3=SIN(B)*SIN(BI)+COS(B)*COS(BI)*COS(DLO)
      BA=ATAN(C1/C2)
      TH=ATAN((C2*COS(BA)+C1*SIN(BA))/C3)
      M=(SIN(B)+SIN(BI))**2
      N=((SIN(B)-SIN(BI))/SIN(TH))**2
      U=((1.-COS(TH))/SIN(TH))**2
      V=(1.+COS(TH))*(TH+SIN(TH))
      RANGE(J)=ABS(A*(TH-F*(M*U+N*V)/4.))
      BEAR(J)=BA*180./PI
      IF(C2.LE.0.) BEAR(J)=BEAR(J)+180.
      IF(C1.LE.0..AND.C2.GE.0.) BEAR(J)=360.+BEAR(J)
      RETURN
      END
```

APPENDIX E. Program Listing for Testing Flight Test Data.

This program filters time differences (TDs), converts filtered TDs to coordinates of position and calculates area navigation information (range, bearing, CTE/CTEB, nonfiltered ground speed, filtered ground speed and estimated time of arrival). It is written in Standard Fortran IV programming language and run in the IBM4341.

```
C***********************************************************************
C    THIS PROGRAM CONVERT TDS TO COORDINATES OF POSITION.
C    ALPHA-BETA FILTER IS USED TO SMOOTH TD DATA.
C    AFTER THE CONVERSION, AREA NAVIGATIONAL INFORMATIONS ARE COMPUTED.
C    (RANGE,BEARING,CROSS TRACK ERROR/BEARING, GROUND SPEED AND
C     ESTIMATED TIME OF ARRIVAL)
C
C                                    MARCH/1983  FUJIKO OGURI
C***********************************************************************
C
      DIMENSION TH(2),POS(2),IPOS(2),RANGE(1000),BEAR(1000)
      DIMENSION TDS(2),TDP(2),TDO(2),VLS(2),VLP(2),PPOS(2)
      INTEGER EVENT,SIDE
      PI=3.14159265
      R=3443.9174
C    ANGLE IS A DESIRED COURSE BEARING TO TRUE NORTH
      ANGLE=241.19
      J=1
      T=0.996*1.4
      TF=6.
      ALPHA=0.72*T/TF
      BETA=ALPHA**2/4.
      VLP(1)=0.
      VLP(2)=0.
      TF=12.
      ALP=0.72*T/TF
      BET=ALP**2/4.
      TIME=T/3600.
C
      READ(1,100,END=300) TDP,EVENT,GSD
C
   10 READ (1,100,END = 300) TDO,EVENT,GSD
  100 FORMAT (F9.2,1X,F9.2,1X,I2,34X,F5.1)
C    FILTER TIME DIFFERENCE
      DO 15 I=1,2
      TDS(I)=TDP(I)+ALPHA*(TDO(I)-TDP(I))
      VLS(I)=VLP(I)+BETA*(TDO(I)-TDP(I))/T
      TDP(I)=TDS(I)+VLS(I)*T
      VLP(I)=VLS(I)
   15 TH(I)=TDP(I)
C    CONVERT TIME DIFFERENCES TO COORDINATES OF POSITION
      CALL DEXLRN(TH,POS)
      DO 20 I=1,2
      IPOS(I) = IFIX(POS(I)) * 10000
      TMP = (POS(I) - IFIX(POS(I))) * 60.
      IPOS(I) = IPOS(I) + (IFIX(TMP) * 100)
      TMP = (TMP - IFIX(TMP)) * 60.
   20 IPOS(I) = IPOS(I) + IFIX(TMP)
      IF(J.EQ.1001) GO TO 300
      CALL RABE(POS, RANGE,BEAR,J)
      IF(J.EQ.1) GO TO 9
C
      DLONG=PPOS(2)-POS(2)
      DLAT=POS(1)-PPOS(1)
      PLAT=(POS(1)+PPOS(1))*PI/360.
```

```
      X=DLONG*COS(PLAT)
      Y=DLAT
      DRECT=ATAN(X/Y)*180./PI
      IF(X.LE.0..AND.Y.GE.0.) DRECT=360.+DRECT
      IF(Y.LT.0.) DRECT=DRECT+180.
      Z=ANGLE-DRECT
      IF(Z.GE.270.) Z=360.-Z
      IF(Z.LE.-270.) Z=360.+Z
C
C     NONFILTERED GROUND SPEED CALCULATION
C
      DRANGE=RANGE(J)-RANGE(J-1)
      DBEAR=(BEAR(J)-BEAR(J-1))*PI/180.
      IF(DBEAR.GE.4.46804) DBEAR=2.*PI-DBEAR
      IF(DBEAR.LE.-4.46804) DBEAR=2.*PI+DBEAR
      GSO=SQRT(DRANGE**2+(RANGE(J)*DBEAR)**2)/TIME
      IF(J.NE.2) GO TO 199
      IF(J.NE.2) GO TO 199
      GSP=GSO
      ACP=0.0
      GO TO 9
C
C     FILTERED GROUND SPEED CALCULATION, INTERVAL=1 CYCLE
C
  199 GSS=GSP+ALP*(GSO-GSP)
      ACS=ACP+BET*(GSO-GSP)/TIME
      GSF=GSS+TIME*ACS
      GSP=GSF
      ACP=ACS
      GSF=ABS(GSF)
      IF(J.LE.4) GO TO 9
C
C     FILTERED GROUND SPEED CALCULATION, INTERVAL=4 CYCLES
C
      DRANGE=RANGE(J)-RANGE(J-4)
      DBEAR=(BEAR(J)-BEAR(J-4))*PI/180.
      IF(DBEAR.GE.4.46804) DBEAR=2.*PI-DBEAR
      IF(DBEAR.LE.-4.46804) DBEAR=2.*PI+DBEAR
      GS40=SQRT(DRANGE**2+(RANGE(J)*DBEAR)**2)/(TIME*4.)
      IF(GS40.LE.20.0) GS40=0.0
      IF(J.EQ.5) GS4P=GS40
      IF(J.EQ.5) AC4P=0.0
      GS4S=GS4P+ALP*(GS40-GS4P)
      AC4S=AC4P+BET*(GS40-GS4P)/TIME
      GS4=GS4S+AC4S*TIME
      GS4P=GS4
      IF(GS4.LE.30.0) GS4=0.0
      IF(GS4.EQ.0.) GO TO 444
C
C     ESTIMATED TIME OF ARRIVAL CALCULATION
C
      ETA=RANGE(J)/GS4
      IETA=IFIX(ETA)*10000
      TMP2=(ETA-IFIX(ETA))*60.
      IETA=IETA+(IFIX(TMP2)*100)
      TMP2=(TMP2-IFIX(TMP2))*60.
      IETA=IETA+IFIX(TMP2)
C
C     CTE/CTEB CALCULATION
C
  444 CTEB=BEAR(J)-ANGLE
      IF(CTEB.LT.0.) GO TO 111
C
      SIDE=1
      IF(CTEB.GE.270.) CTEB=360.-CTEB
      IF(CTEB.GT.180.) SIDE=0
      GO TO 222
C
  111 SIDE=1
      IF(CTEB.GE.-180.) SIDE=0
      IF(CTEB.LE.-270.) CTEB=CTEB+360.
C
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
 222  CTEB=ABS(CTEB)
      CC=RANGE(J)/R
      BB=CTEB*PI/180.
      AA=SIN(BB)*SIN(CC)
      CTE=ARSIN(AA)*R
      CTE=ABS(CTE)
C    CTE=RIGHT, -CTE=LEFT
      IF(SIDE.EQ.0) CTE=-CTE
      IF(Z.GE.-90..AND.Z.LT.90.) GO TO 333
      CTE=-CTE
C
 333  WRITE(2,200) TH,EVENT,IPOS,IETA,RANGE(J),BEAR(J),GSD,CTEB,CTE,GSO,
     1    GSF,GS4
 200  FORMAT(2(F9.2,1X),I2,3(1X,I6),8(F6.1))
   9  J=J+1
      PPOS(1)=POS(1)
      PPOS(2)=POS(2)
      IETA=999999
      GO TO 10
 300  STOP
      END
C
C    THIS SUBROUTINE CONVERTS TIME DIFFERENCES TO COORDINATES OF
C    POSITION.
C
      SUBROUTINE DEXLRN(TH,POS)
C
      DIMENSION TH(2),POS(2)
      DATA TCY,TCZ/3.9E-2,5.4E-2/
      DATA THMY/0.15129258/,THMZ/0.14849557/,XNR/46.986746/
      DATA CTMY/0.98857709/,STMY/0.15071607/,CTMZ/0.98899478/
      DATA STMZ/0.14795043/,CXK/0.37466368/,SXK/0.92716079/
      DATA C1/2.9832071/,C2/4.3111683/,C3/-7.1717116/
      DATA C4/4.0896360/,C5/-4.4643647/,C6/1.1496479/
      DATA C7/-5.0510869/,C8/5.1185063/,C9/0.55921480/
      DATA C10/1.5850077E-3/,C11/2.5836473E-4/,C12/-2.9620318E-3/
      DATA C13/1.5850077E-3/,C14/1.0034014/
      DATA PI/3.141592/
C
      TY=TH(1)*1.E-6
      TZ=TH(2)*1.E-6
C
      PY=XNR*(TY-TCY)-THMY
      PZ=XNR*(TZ-TCZ)-THMZ
      CPY=COS(PY)
      SPY=SIN(PY)
      CPZ=COS(PZ)
      SPZ=SIN(PZ)
      AY=(CPY-CTMY)/STMY
      AZ=(CPZ-CTMZ)/STMZ
      BY=SPY/STMY
      BZ=SPZ/STMZ
      U1=AY*CXK-AZ
      U2=AY*SXK
      U3=AZ*BY-AY*BZ
      UU=U1*U1+U2*U2
      CDBY=(U3*U1+U2*SQRT(UU-U3*U3))/UU
      THMS=ATAN(AY/(BY+CDBY))
      CB=COS(THMS)
      CA=COS(THMS+PY)
      CC=COS(THMS+PZ)
C
      F=C1*CA+C2*CB+C3*CC
      G=C4*CA+C5*CB+C6*CC
      H=C7*CA+C8*CB+C9*CC
C
      THGS=ATAN((G+C10)/(F+C11))
      PHGS=ATAN(C14*C14*SIN(THGS)*(H+C12)/(G+C13))
      POS(2)=THGS*180./PI
      POS(1)=PHGS*180./PI
      RETURN
      END
```

```
C
C     THIS SUBROUTINE CALCULATES RANGE TO THE WAYPOINT AND BEARING
C     TO TRUE NORTH.
C
      SUBROUTINE RABE(POS, RANGE, BEAR,J)
C
      DIMENSION POS(2),RANGE(1000),BEAR(1000)
      REAL LA1,LO1,LA2,LO2,M,N
      DATA PI/3.1415926535898/,A/3443.9174/
C     COORDINATES OF THE TO WAYPOINT
      LA2=0.68467327
      LO2=1.43521818
C     COORDINATES OF THE RECEIVER POINT
      LA1=POS(1)*PI/180.
      LO1=POS(2)*PI/180.
      FF=1. - 1./298.2
      F=1./298.2
      DLO=LO1-LO2
      TB=FF*TAN(LA1)
      TBI=FF*TAN(LA2)
      B=ATAN(TB)
      BI=ATAN(TBI)
      C1=COS(BI)*SIN(DLO)
      C2=COS(B)*SIN(BI)-SIN(B)*COS(BI)*COS(DLO)
      C3=SIN(B)*SIN(BI)+COS(B)*COS(BI)*COS(DLO)
      BA=ATAN(C1/C2)
      TH=ATAN((C2*COS(BA)+C1*SIN(BA))/C3)
      M=(SIN(B)+SIN(BI))**2
      N=((SIN(B)-SIN(BI))/SIN(TH))**2
      U=((1.-COS(TH))/SIN(TH))**2
      V=(1.+COS(TH))*(TH+SIN(TH))
      RANGE(J)=ABS(A*(TH-F*(M*U+N*V)/4.))
      BEAR(J)=BA*180./PI
      IF(C2.LE.0.) BEAR(J)=BEAR(J)+180.
      IF(C1.LE.0..AND.C2.GE.0.) BEAR(J)=360.+BEAR(J)
      RETURN
      END
```