

N O T I C E

THIS DOCUMENT HAS BEEN REPRODUCED FROM
MICROFICHE. ALTHOUGH IT IS RECOGNIZED THAT
CERTAIN PORTIONS ARE ILLEGIBLE, IT IS BEING RELEASED
IN THE INTEREST OF MAKING AVAILABLE AS MUCH
INFORMATION AS POSSIBLE

CR 160880
c. 3

THE SOFTWARE SYSTEM DEVELOPMENT FOR THE TAMU REAL-TIME FAN BEAM SCATTEROMETER DATA PROCESSORS

By

Billy V. Clark
B. Randall Jean

(NASA-CR-160880) THE SOFTWARE SYSTEM
DEVELOPMENT FOR THE TAMU REAL-TIME FAN BEAM
SCATTEROMETER DATA PROCESSORS Final Report
(Texas A&M Univ.) 262 p HC A12/MF A01

N81-11439

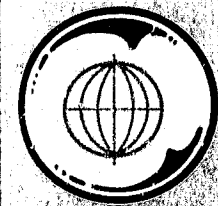
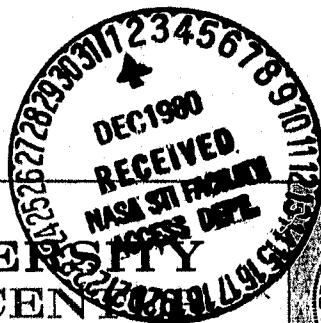
Unclas
37740

CSSL 08B G3/43

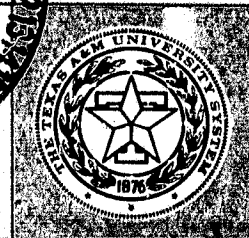
August 1980

Sponsored by:

National Aeronautics and Space Administration
Lyndon B. Johnson Space Center
Contract NAS9-15311



TEXAS A&M UNIVERSITY
REMOTE SENSING CENTER
COLLEGE STATION, TEXAS



THE SOFTWARE SYSTEM DEVELOPMENT FOR THE TAMU
REAL-TIME FAN BEAM SCATTEROMETER DATA PROCESSORS

By

Billy V. Clark
B. Randall Jean

Remote Sensing Center
Texas A&M University
College Station, Texas 77843

August 1980

Sponsored by:

National Aeronautics and Space Administration
Lyndon B. Johnson Space Center
Contract NAS9-15311

TABLE OF CONTENTS

	<u>Page</u>
LIST OF FIGURES	
LIST OF TABLES.	
1.0 SUMMARY.	1
2.0 INTRODUCTION	3
2.1 Historical Background	3
2.2 Design Objective and Overview	4
3.0 DESIGN THEORY.	9
3.1 Introduction.	9
3.2 The Scatterometer Equation and Fan Beam Systems	9
3.3 Precision in Estimating σ_{pq}^o	14
3.4 Power Spectral Estimation Using the Chirp Z-Transform	19
3.5 The Application of the CZT to Doppler Filtering	21
4.0 SYSTEM DESIGN RATIONALE.	27
4.1 Introduction.	27
4.2 Definition of System Design Parameters.	29
4.3 The Theory for Constant Precision, Constant Angular Resolution and Constant Ground Resolution Designs	35
4.3.1 A Constant Precision Design	35
4.4 A Comparison of the Design Approaches	38
5.0 SYSTEM ARCHITECTURE AND OVERVIEW	49
5.1 Target and Development System Architectures	49
5.2 An Overview of the Operation of the System	52
6.0 SOFTWARE DEVELOPMENT FOR C-BAND/L-BAND REAL-TIME RADAR DATA PROCESSOR.	59
6.1 Background.	59
6.2 Software Algorithm.	60

TABLE OF CONTENTS (continued)

	<u>Page</u>
6.2.1 General Concept	60
6.2.2 System Initialization and Set-Up	60
6.2.3 'RUN' Mode of Operation	66
6.2.3.1 Calculating σ^0	66
6.2.3.2 Calculating σ^0	72
6.2.3.3 'RUN' Termination	73
6.2.4 Testing and Evaluation	73
6.2.5 Memory Requirements	79
6.3 Interrupt Structure	79
6.4 Sub-routine Definitions	81
REFERENCES	93
APPENDIX A: System Flow Charts	94
APPENDIX B: INTCOM Listing (Main Driver)	131
APPENDIX C: Bi-Phase-L Output Frame	163
APPENDIX D: Subroutine Listings (Alphabetical)	167
APPENDIX E: AMC 95/6511 Device Routine Listings	259

LIST OF FIGURES

<u>Figure</u>		<u>Page</u>
3.1	Doppler Processing and the Fan Beam Scatterometer	10
3.2	Scatterometer Geometry.	11
3.3	The Variance Ratio as a Function of Integration Time-Bandwidth Product (from Reference [2]).	16
3.4	The Improvement in Precision with the Time Bandwidth Product	17
3.5	A Simplified Block Diagram of the CW Fan Beam Scatterometer	22
3.6	Spectra of Interest	23
3.7	A Technique for Implementing the CZT Without Post Multiplication.	26
4.1	Various Resolution Parameters	31
4.2	Various Cell Parameters	32
4.3	Comparison of the Resolution Parameters when (a) $vT_1 < \rho_B$, (b) $vT_2 = \rho_B$ and (c) $vT_3 > \rho_B$	33
4.4	The Dependence of Processing Precision on View Angle for the Three Design Approaches	39
4.5	The Dependence of Angular Resolution on Viewing Angle for the Three Design Approaches	40
4.6	The Dependence of Ground Resolution on Viewing Angle for the Three Design Approaches	41
4.7	The Dependence of Beam Resolution on Viewing Angle for the Three Design Approaches	42
4.8	The Dependence of Processing Precision on View Angle for the Three Design Approaches	43
4.9	The Dependence of Angular Resolution on Viewing Angle for the Three Design Approaches	44
4.10	The Dependence of the Ground Resolution on Viewing Angle for the Three Design Approaches	45
4.11	The Dependence of Beam Resolution on Viewing Angle for the Three Design Approaches	46

LIST OF FIGURES (continued)

<u>Figure</u>		<u>Page</u>
5.1	Architecture of the Target System	50
5.2	Architecture of an Engineering Model Expandable to the Target System	53
5.3	Spectral Data Format and Related Parameters	56
5.4	Filter Power Spectral Response.	57
6.1	Overall System Structure.	61
6.2	General System Layout, Real Time Processor.	64
6.3	System Geometry	69
6.4	Module Test Set-Up.	71
6.5	System Test Set-Up.	75
6.6	Cell Overlap Geometry	77
6.7	Cell Overlap vs Cycle-Time.	78
6.8	Program Alignment within ROM and RAM.	80
6.9	CZT Interrupt Structure	82

LIST OF TABLES

<u>Table</u>		<u>Page</u>
4.1	Factors Influencing System Performance	28
4.2	Nominal Design Guidelines.	38
6.1	I/O Ports Used by NASA-CZT Processor	62

THE SOFTWARE SYSTEM DEVELOPMENT FOR THE TAMU
REAL-TIME FAN BEAM SCATTEROMETER DATA PROCESSORS

1.0 SUMMARY

The Remote Sensing Center at Texas A&M University (TAMU) has developed a real-time radar signal processor for the NASA fan beam scatterometer system. The development of the system and hardware design of the real-time processor is documented in Final Report RSC-3556, "The System and Hardware Design of Real-Time Fan Beam Scatterometer Data Processors," March 1979 [1]. This current report presents the details of the development of the software system for the signal processors.

The software package has been designed and written to process in real-time any one quadrature channel pair of radar scatterometer signals from the NASA L- or C-Band radar scatterometer systems. The software has been successfully tested in the C-Band processor breadboard hardware using recorded radar and NERDAS (NASA Earth Resources Data Annotation System) signals as the input data sources.

Contained in this report are a brief review of the processor development program, a concise yet complete description of the overall processor theory of operation and design, a detailed description and documentation of the real-time processor software system, the results of the laboratory software tests, and recommendations for the efficient application of the data processing capabilities provided by the TAMU Real-Time Scatterometer Processing System.

In the interest of efficiency and economy, some of the material presented in this report has been extracted from Final Report RSC-3556 [1] with only minor or no modification. Such material has been included to provide as nearly complete documentation in this volume of the overall processor system as is necessary to fully understand its operation, capabilities and limitations. For the detailed documentation of the breadboard hardware and the engineering model design it is still necessary to refer to the earlier report.

2.0 INTRODUCTION

2.1 Historical Background

The NASA Johnson Space Center operates a set of airborne fan beam scatterometers for support of various earth and space related programs. Data from an early system proved helpful in identifying an ocean wind measurement technique. This work eventually led to the scatterometer system aboard SEASAT A. Raw scatterometer signals from these early experiments were post processed into quantitative engineering unit data using a general purpose computer. The time and expense required to process data caused the delay between data acquisition and data product delivery to be excessive. As a result, utilization of the scatterometers was limited and they were eventually removed from service.

More recently, radar data requests by the NASA soil moisture program provided enough justification to warrant reinstating the 0.4 GHz, 1.6 GHz and 13.3 GHz scatterometers into service, and constructing a new 4.75 GHz scatterometer. The earlier experiences with scatterometer data processing led to a program to develop faster, cheaper methods of data handling and processing. The initial thrust of the program produced a demonstration processor for use with the 13.3 GHz scatterometer. The philosophy behind this processor was to provide a real-time quick capability for verifying data characteristics (i.e., system operation) and 2) provide a method of identifying those data to be post processed on a large computer to the accuracies needed for analysis. This processor was developed and constructed at TAMU under NASA contract [2] to process

polarization channel over a limited set of viewing angles. In addition to this hardware processor, TAMU developed software routines for general purpose computers to reduce raw data to calibrated engineering units.

Recent advances in signal processing technology have suggested that by combining analog and digital processing methods into a single processor, real-time on board processing and real-time rate post time processing of scatterometer data to calibrated engineering units could be accomplished. Such a system could provide all of the capability in terms of viewing angles, resolution and adaptability that the post time software systems previously developed could provide, with a potential for more accurate results as a result of eliminating the analog recorder when operating in a real-time mode. Such a system would provide experimenters with calibrated data on a timely basis with fewer manhours from data flight to delivery. This realization provided the basis for the current efforts reported in this document.

2.2 Design Objective and Overview

Designs for two airborne radar scatterometer processors for use with the NASA 1.6 GHz and 4.75 GHz scatterometers were identified and analyzed. A portion of the processor was implemented to evaluate a "state-of-the-art" component proposed for use in the processor. This component permitted a standardized design approach which is extendable to other NASA fan beam scatterometers. The current effort exploited design experiences from previous hardware and software processors to minimize

significant error contributions and to assure repeatability in performance. However, innovations were also introduced as a result of the hybrid sampled analog and digital approach to provide a flexible operator/experimenter oriented system. As a result of these new insights, major improvements were also identified for use in the purely software approaches to processing scatterometer data. This benefitted another NASA sponsored program to develop a more efficient software routine to process scatterometer data on an interim basis while the hardware processors undergo development. This latter effort ran concurrently with the processor development program and afforded an opportunity to also test, anticipate and prove the characteristics of the hardware design.

The hardware design features a chirp Z-transform (CZT) approach to filtering the Doppler spread radar return. The CZT is implemented with multiplying digital to analog converters and a charge coupled transversal filter. The filtering operation reduces to that of performing a discrete Fourier transform (DFT) of the radar return when represented in complex valued form. As a consequence, no Hilbert transform operation (sign sensing) is required to separate fore and aft returns. Both are provided simultaneously with considerable reduction in circuit complexity. The subsequent processing, is actually limited to the aft data; however, the fore data is available within the processor should future efforts require it.

There are many advantages in the CZT approach. It permits high frequency resolution of Doppler return. As a consequence, the return

may be measured with good angular resolution. This also permits the processor to adapt with changes in aircraft velocity to track the desired viewing angles by simply using a different set of spectral outputs. It will also permit arbitrary choices in viewing angles desired. The CZT approach can be readily applied to scatterometers operating at other wavelengths by altering the sampling frequency.

The power spectral density (PSD) of the total return is formed from the chirp Z-transformed data. The formation of the PSD requires that the spectral data be detected (squared) and accumulated (averaged) over a period of time. To achieve the accuracy and the dynamic range required in scatterometry, the detection and accumulation are accomplished digitally.

The detected and averaged data are converted to estimates of the scattering coefficients σ^0 at eight viewing angles over the aft sector. The conversion is implemented in software and requires the application of radar range, pattern data, viewing angle, and transmitted power to yield a calibrated result. In addition, the software permits interactive control of the processor. Since the computations and control are provided by software, any portion of the operating system can be altered should the need arise.

The design approach was partially evaluated by actually implementing a subsystem of the scatterometer processor. An evaluation to this detail was required to validate the performance and dynamic range of the charge coupled devices and associated circuitry since this is a "state-of-the-art" item.

An earlier report [1] described the system design theory, the system operating rationale and architecture, the hardware and software designs and an evaluation of the CZT approach for the scatterometer processors profiled above. This report reviews the relevant system and hardware design considerations and provides detailed documentation of the newly developed processor software. In particular, Section 3.0 reviews the system design theory background. The characteristics of CW fan beam scatterometers are related to the scatterometer equation to identify the measurement theory. It is shown that the angular scattering characteristic can be resolved by estimating the PSD of the radar return. The precision by which the PSD estimated is related to the time bandwidth product by analogy with classical fading theory. The technique by which the fore and aft spectra are separated using a DFT method is then identified. The DFT is related to the CZT and the means by which the CZT may be implemented is then established.

Section 4.0 is dedicated to establishing a suitable operating rationale for the processor. Trade-offs between angular resolution, ground resolution, precision and beam resolution are established and evaluated to identify a suitable operating mode to satisfy user requirements and system constraints.

Section 5.0 describes in detail the system architecture and carefully distinguishes between the target system and the engineering development model. An overview of the internal operation of the system is also presented.

Section 6.0 provides a complete description of the software system that has been developed for the real-time data processors. The software system provides fully calibrated normalized radar cross-section data for eight angles of incidence. The output data are fully annotated with all relevant aircraft, sensor, and processor ancillary data in a single serial bi- ϕ L channel.

3.0 DESIGN THEORY

3.1 Introduction

Airborne fan beam scatterometers permit simultaneous backscatter observations over a range of incident angles. By confining the antenna beam width in the crosstrack dimension and spreading the beam in the along track dimension, Doppler filtering may be employed in a CW system to resolve the average return power at various incident angles, each of which is spanned by a small angular window as illustrated by Figure 3.1. Combinations of transmit and receive polarizations permit like and cross polarized scattering properties of distributed targets to be measured. When the aircraft is flown over the same distributed target at different headings about the compass, the azimuthal as well as the incident angular behaviors may be documented.

3.2 The Scatterometer Equation and Fan Beam Systems

For a large class of distributed targets the returns from slightly different angular directions are essentially uncorrelated. Where a particular direction is denoted by (θ, ϕ) within the coordinate system of Figure 3.2, the total return power may be described by summing returns from patches of the target located in various angular directions (θ_i, ϕ_j) . If the radar cross section in direction (θ_i, ϕ_j) is denoted by $\sigma_{pq}(\theta_i, \phi_j)$, the total return power may be expressed as

$$W_r' = \frac{\lambda^2}{(4\pi)^3} W_t \sum_{i=1}^N \sum_{j=1}^M G_{tp}(\theta_i, \phi_j) G_{rq}(\theta_i, \phi_j) \sigma_{pq}(\theta_i, \phi_j) / R^4 \quad (3.1)$$

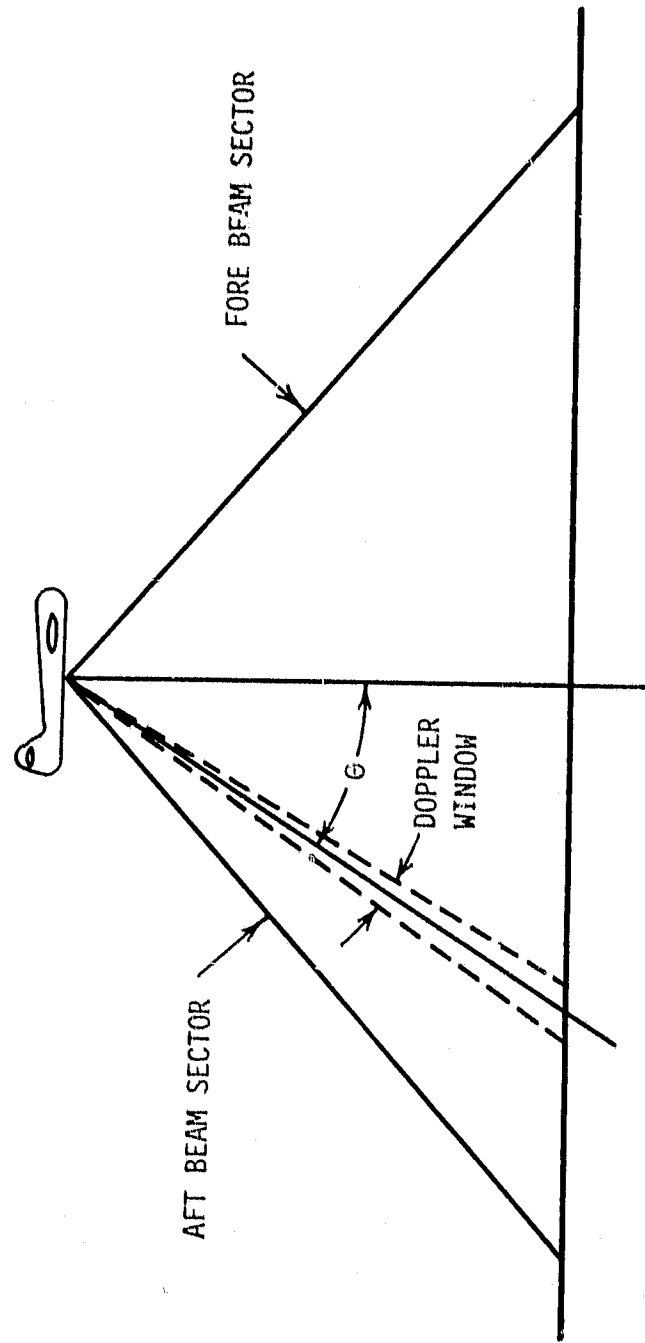


FIGURE 3.1 DOPPLER PROCESSING AND THE FAN BEAM SCATTEROMETER

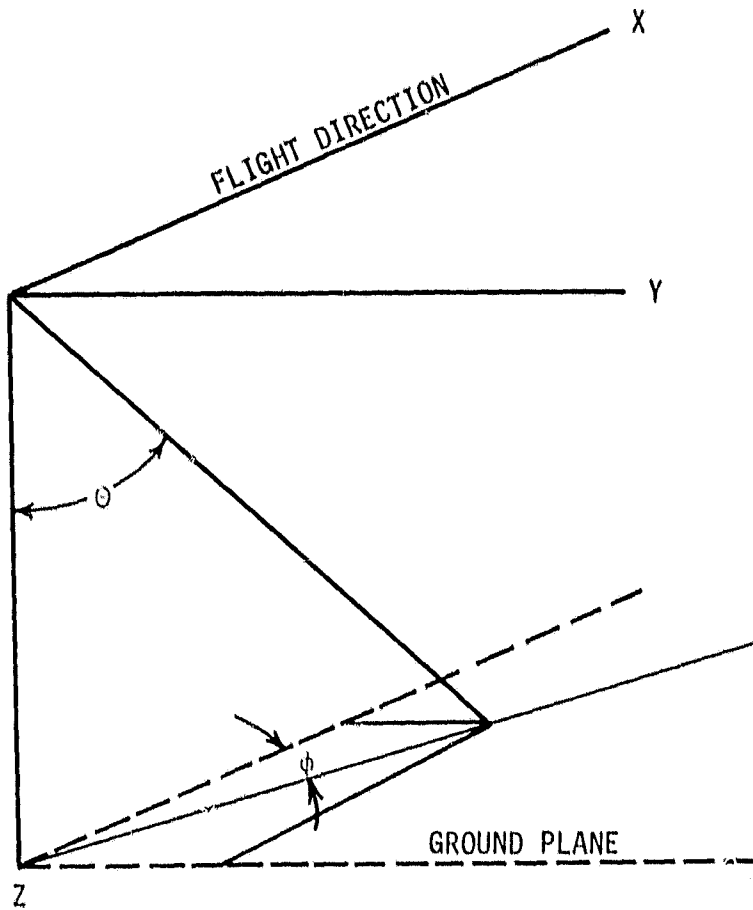


FIGURE 3.2 SCATTEROMETER GEOMETRY

where λ = radar wavelength

G_{tp} = transmit pattern directivity for polarization state p

G_{rq} = receive pattern directivity for polarization state q

W_t = total transmitted power

R_i = $h/\cos\theta_i$

h = aircraft altitude

p, q = indices denoting the transmit and receive polarization states, respectively

In order to primarily discriminate the backscatter within an angular sector of the incident angle θ , a Doppler filter, whose normalized transfer function is given by $H(\omega)$, is employed (see Figure 3.1). The portion of the return power appearing at the output of the filter is therefore given by

$$W_r = \frac{\lambda^2 W_t}{(4\pi)^3} \sum_{i=1}^N \sum_{j=1}^M G_{tp}(\theta_i, \phi_j) G_{rq}(\theta_i, \phi_j) \sigma_{pq}(\theta_i, \phi_j) |H(\omega_{ij})|^2 / R_i^4 \quad (3.2)$$

where

$$\omega_{ij} = 4\pi v \sin\theta_i \cos\theta_j / \lambda \quad (3.3)$$

is the radian frequency associated with the patch in direction (θ_i, ϕ_j) .

When the normalized scattering coefficient $\sigma_{pq}^0(\theta, \phi)$ is introduced, the double summation may be replaced with a double integral given by

$$W_r = \frac{\lambda^2 W_t}{(4\pi)^3} \iint \frac{G_{tp} G_{rq} \sigma_{pq}^0 |H(\omega)|^2 dA}{R^4} \quad (3.4)$$

In the interpretation of W_r it is important to realize that $|H(\omega)|^2$

participates within the integration since ω is dependent on (θ, ϕ) . In an ideal fan beam scatterometer, ϕ ranges over a small interval about zero since the crosstrack antenna beam width is small. As a consequence

$$\omega \approx 4\pi v \sin \theta / \lambda \quad (3.5)$$

It is then observed that $|H(\omega)|^2$ plays the role of a normalized antenna pattern having discriminatory power in the θ dimension.

When the bandwidth of the Doppler filter is sufficiently narrow, σ_{pq}^0 may be regarded as constant over the area A spanned by the Doppler bandwidth and the crosstrack beamwidth. The scatterometer equation (3.4) then reduces to the form

$$W_r(\theta_0) \cong \frac{\lambda^2 W_t}{(4\pi)^3} \frac{\sigma_{pq}^0(\theta_0)}{h^4} \iint G_{tp} G_{rq} |H|^2 \cos^4 \theta dA \quad (3.6)$$

where θ_0 is the incident angle corresponding to the center frequency ω_0 of the Doppler filter, i.e.,

$$\omega_0 = 4\pi v \sin \theta_0 / \lambda \quad (3.7)$$

The recovery of $\sigma(\theta_0)$ is, therefore, dependent on measurement of W_r , W_t and h and on an estimate of the double integral. In this regard the integral is often approximated by introducing an effective area A_{eff} so that

$$\iint G_t G_r |H|^2 \cos^4 \theta dA = G_t(\theta_0, 0) G_r(\theta_0, 0) \cos^4 \theta_0 A_{\text{eff}} \quad (3.8)$$

3.3 Precision in Estimating σ_{pq}^0

In order to estimate σ_{pq}^0 at a set of incident angles θ_{oi} , $i=1,2,\dots,n$, a bank of filters is required. The center frequency of each filter is chosen in accord with equation (3.7). If $S(\omega)$ is the spectral density of the return signal, then the output of the i th filter H_i is given by

$$W_r(\theta_{oi}) = 2 \int_0^{\infty} |H_i(\omega)|^2 S(\omega) d\omega \quad (3.9)$$

An alternative method could, instead, measure (estimate) the power spectral density (PSD) of the return signal and then form the return power through an integration, viz.,

$$W_r = 2 \int_{f_{li}}^{f_{ui}} S(\omega) d\omega \quad (3.10)$$

where f_{ui} and f_{li} are the upper and lower corner frequencies associated with the i th angle. In the latter case the measurement of σ_{pq}^0 has been reduced to a problem in estimating the power spectral density.

Regardless of the approach, it is necessary to reduce the variance in the estimate of the mean power return W_r to assure a good estimate of the average scattering coefficient σ_{pq}^0 . As is well known, the radar return is characterized by heavy fading since the signal has a Rayleigh-like amplitude distribution [3]. As a consequence, it is difficult to estimate the mean squared statistic of such a signal. The theory for the precision in the estimating the mean squared statistic appears in references [3], [4] and others. The extrapolation of this theory to the case where the PSD is to be estimated can be established on an intuitive basis and is made precise in reference [5]. The PSD is estimated from the periodogram and is

defined by

$$S_N(k) = \frac{1}{N} \left| \sum_{n=0}^{N-1} s(n) e^{-j 2\pi kn/N} \right|^2 \quad (3.11)$$

for the k th spectral line for a signal represented in sampled form $\{s(0), s(1), \dots, s(N-1)\}$.

In the case where analog filtering, detection and integration is performed, it is well known that the standard deviation σ in the estimate of W_r is given by [4]

$$\sigma = W_r / \sqrt{BT} \quad (3.12)$$

where B is the pre-detection (effective) bandwidth and T is the integration period. The dependence of the variance ratio σ^2/W_r^2 on the BT product is illustrated in Figure 3.3.

Although the variance reduction is a good indication of the improvement in the estimate of W_r , the precision of a system is better conveyed by a statistical 90% confidence interval. The 90% confidence interval for a BT product of 10 or better can be approximated by [6]

$$\bar{W}_r - 1.645\sigma \leq \bar{W}_r \leq \bar{W}_r + 1.645\sigma \quad (3.13)$$

where \bar{W}_r is the estimate of W_r . When the span of this confidence interval is expressed in dB, it can be written as

$$R = 10 \text{ Log } \frac{1 + 1.645/\sqrt{BT}}{1 - 1.645/\sqrt{BT}} \quad (3.14)$$

The dependence of this precision factor on the time bandwidth product is illustrated in Figure 3.4. It is observed that a ± 1 dB confidence interval

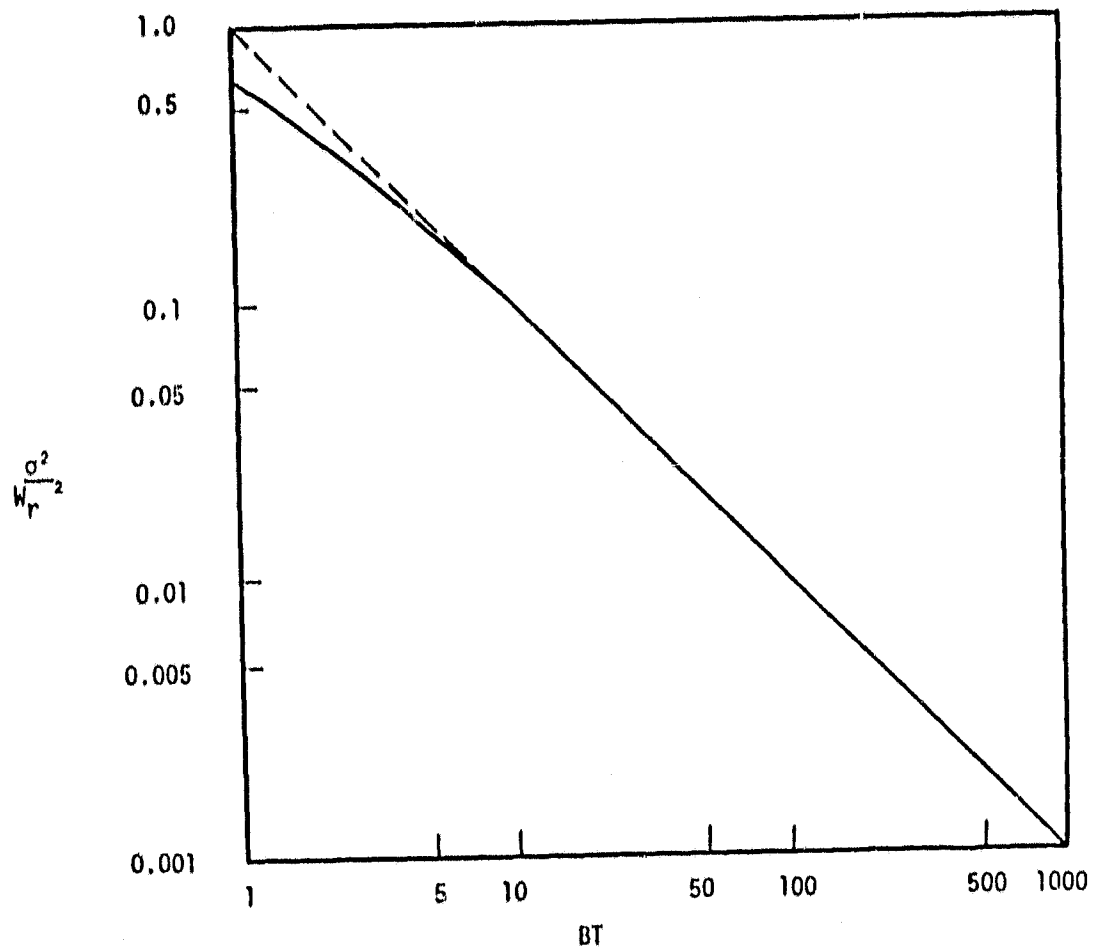


FIGURE 3.3 THE VARIANCE RATIO AS A FUNCTION OF INTEGRATION TIME-BANDWIDTH PRODUCT (FROM REFERENCE [2])

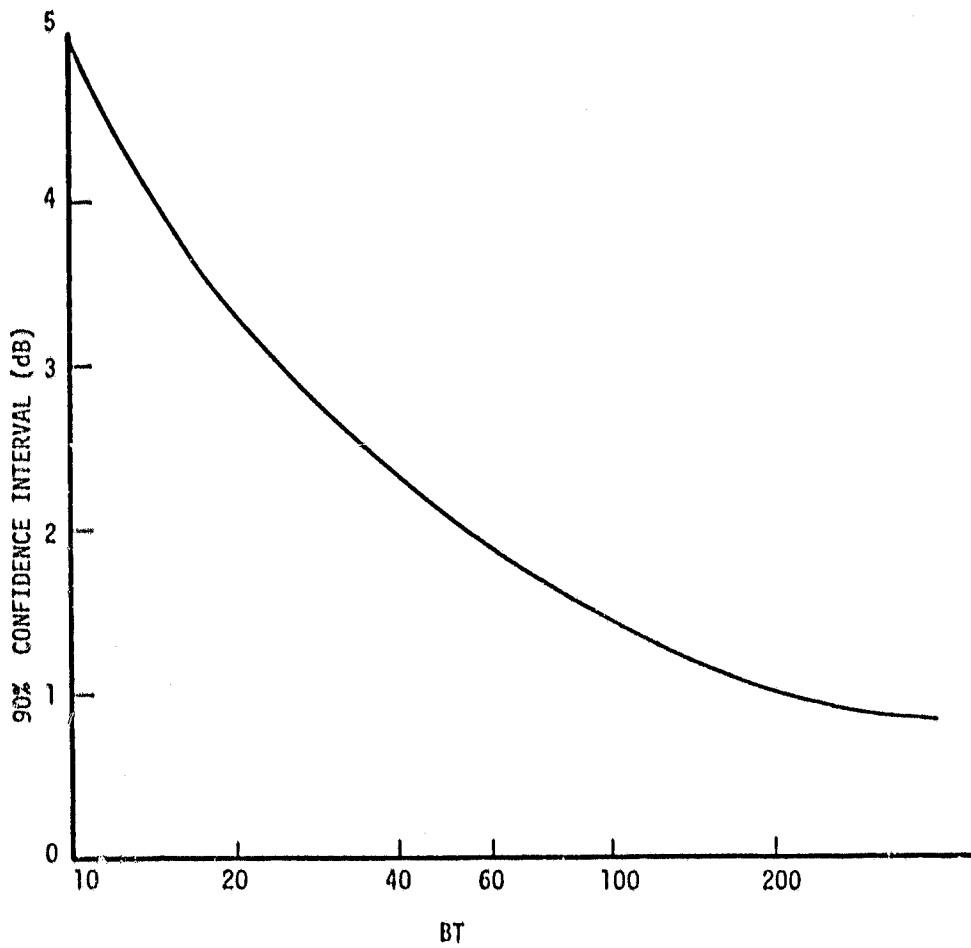


FIGURE 3.4 THE IMPROVEMENT IN PRECISION WITH THE TIME BANDWIDTH PRODUCT

requires a BT product of 50. Furthermore, it is noted that the improvement in precision becomes less rapid when $BT > 100$.

The above theory can also be applied to the case where the PSD is determined from the periodogram. To obtain precision in the estimate of the PSD, the periodogram must be averaged. This can be accomplished by averaging spectral estimates from a sequence of records and by smoothing adjacent spectral lines from a single record. To make the association between the precision for an analog processor with that for a PSD processor, it should be noted that the PSD estimates are based on the discrete Fourier transform (DFT) of the return signal $s(t)$ (see equation (3.18)). The effective bandwidth associated with DFT processing (filtering) is given by

$$\Delta f = \frac{1}{T} \quad (3.15)$$

where T is the duration of the signal record. It has been assumed that the record is unweighted. Therefore, the time bandwidth product associated with a single line from a single record is simply

$$BT = \Delta f T \quad (3.16)$$

or 1. Since the spectral estimates from a single record are poorly correlated, the BT product can be enhanced by averaging over a window of adjacent spectral lines. The BT product can be further enhanced by averaging line estimates from a sequence of non-overlapping records. So, if N_f adjacent lines from each record are smoothed and N_R non-overlapping records are processed, the resulting BT product for a filter of

width $N_F \Delta f$ is given by

$$BT = (N_F \Delta f)(N_R T) \quad (3.17)$$

or simply $N_F N_R$. This result indicates that the precision improvement is identical for analog and PSD processing.

3.4 Power Spectral Estimation Using the Chirp Z-Transform

As indicated above, the estimate of the PSD can be based on the DFT of a radar return record. If $s(n)$, $n=0,1,2,\dots,N-1$ is an N point sequence representing the return signal $s(t)$ over a time interval T , then the DFT of $s(t)$ is defined as

$$F(k) = \sum_{n=0}^{N-1} s(n) e^{-j2\pi kn/N} \quad (3.18)$$

where $k \in \{0,1,2,\dots,N-1\}$. $F(k)$ is interpreted as the spectral amplitude of $s(t)$ at a frequency of k/T when $0 \leq k \leq \frac{N}{2}$ and at a frequency of $-(N-k)/T$ for $\frac{N}{2} < k < N-1$. It has been assumed that N is even. The DFT formulation may be modified through the use of the identity

$$2nk = n^2 + k^2 - (k-n)^2 \quad (3.19)$$

to permit an implementation of the DFT by hardware. The identity results in a DFT given by

$$F(k) = e^{-j\pi k^2/N} \sum_{n=0}^{N-1} s(n) e^{-j\pi n^2/N} e^{j\pi(k-n)^2/N} \quad (3.20)$$

The implication of the above result is that $s(n)$ must be first down-chirped with $e^{-j\pi n^2/N}$, convolved with an up-chirp $e^{j\pi n^2/N}$ and then post multiplied by a down-chirp $e^{-j\pi k^2/N}$. The pre and post multiplications may be

performed by analog methods and the convolution may be formed with transversal filters using charge coupled devices. The transversal filter requires $2N-1$ stages to implement the DFT. When forming the PSD, the post chirp may be discarded since it does not affect the amplitude.

A more efficient means for implementing the transversal filter, requiring only N stages in the transversal filter, is based on the sliding DFT. The sliding transform is defined as

$$F_S(k) = \sum_{n=k}^{k+N-1} s(n)e^{-j2\pi nk/N} \quad (3.21)$$

and differs from the non-sliding version in that the input sequence is shifted forward one sample for each new spectral estimate. The transform consequently operates continuously. Spectral estimates on the same line are updated every N samples since $e^{-j2\pi nk/N}$ is modulo N in the parameter k . As a result of the sliding action, phase information is destroyed; nevertheless, the magnitude information important to the PSD estimation is preserved.

It can be shown that through the use of the identity of equation (3.19), the sliding transform can be rewritten as

$$F_S(k) = e^{-j\pi k^2/N} \sum_{m=1}^N e^{j\pi(m-N)^2/N} s(k-m+N)e^{-j\pi(k-m+N)^2/N} \quad (3.22)$$

This result implies that the input must be pre-chirped by a factor $e^{-j\pi(m-N)^2/N}$, convolved with $e^{j\pi(m-N)^2/N}$ and then post multiplied with $e^{-j\pi k^2/N}$ to form the sliding DFT. When the PSD is required, the post

multiplication may be replaced with a squaring operation to yield $|F_s(k)|^2$. Since the pre-multiplication is periodic in m , the CZT filter can operate on the input signal continuously with the transversal filter only requiring N stages.

3.5 The Application of the CZT to Doppler Filtering

3.5.1 Signal Processing Theory

A simplified block diagram of the CW fan beam scatterometer is illustrated in Figure 3.5. The transmitter illuminates the terrain at a radian frequency of ω_0 . The backscattered signal arriving at the receiver may be denoted as

$$s(t) = a(t) \cos [\omega_0 t + \phi(t)] \quad (3.23)$$

where $a(t)$ and $\phi(t)$ may be regarded as random variables. The spectrum of $s(t)$ is depicted symbolically in Figure 3.6a. The fore and aft spectra are distinguished from one another by "coloring" the fore spectrum as a rectangle and the aft spectrum as a triangle. As indicated in the receiver chain, the return signal is split equally into two channels. This signal in the upper channel is coherently demodulated with $\cos \omega_0 t$ and low pass filtered to yield

$$x(t) = \frac{1}{2} a(t) \cos \phi(t) \quad (3.24)$$

The upper channel is commonly called the cosine channel or the in-phase (I) channel. Demodulation and low pass filtering in the lower channel yields

$$y(t) = \frac{1}{2} a(t) \sin \phi(t) \quad (3.25)$$

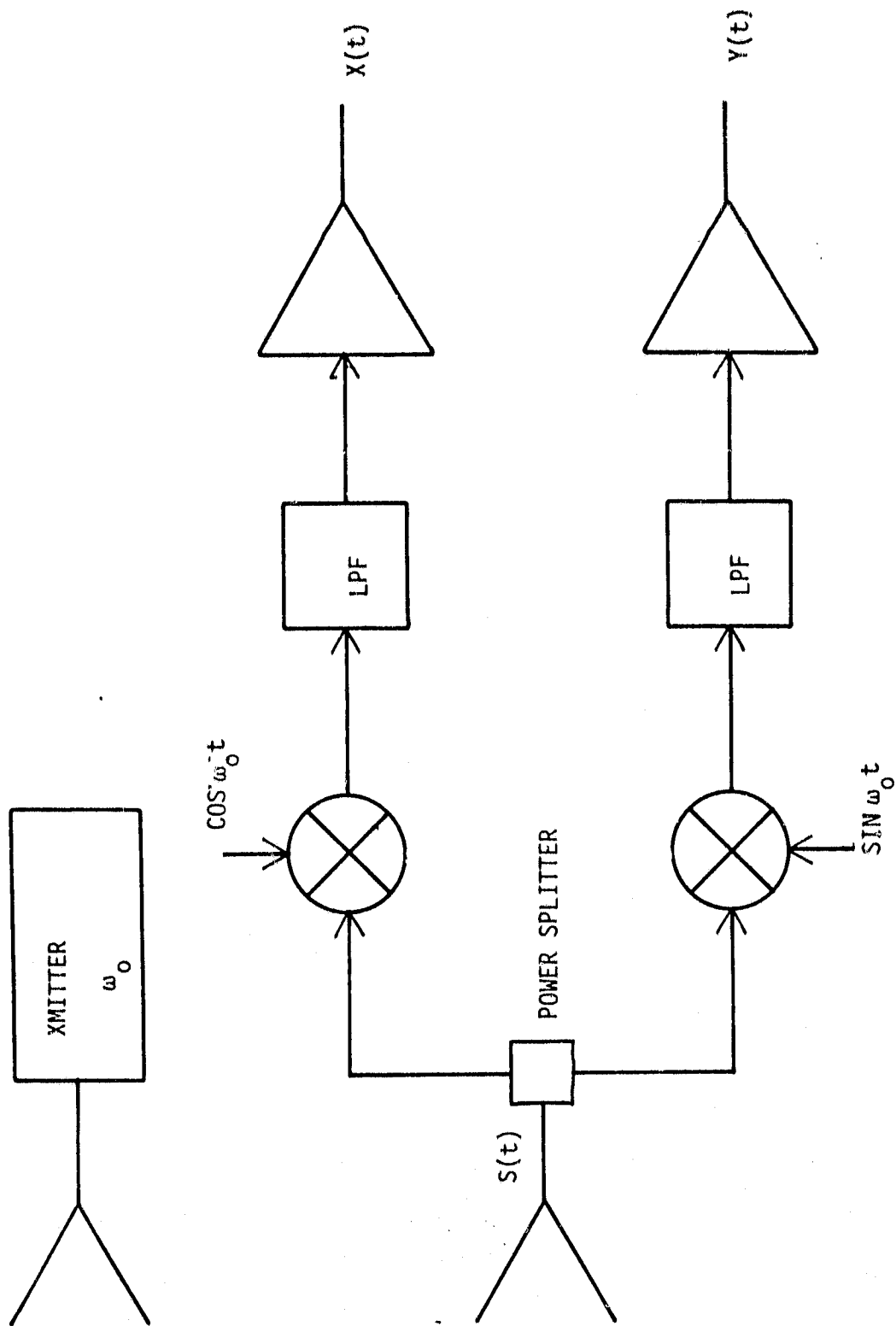


FIGURE 3.5 A SIMPLIFIED BLOCK DIAGRAM OF THE CW FAN BEAM SCATTEROMETER

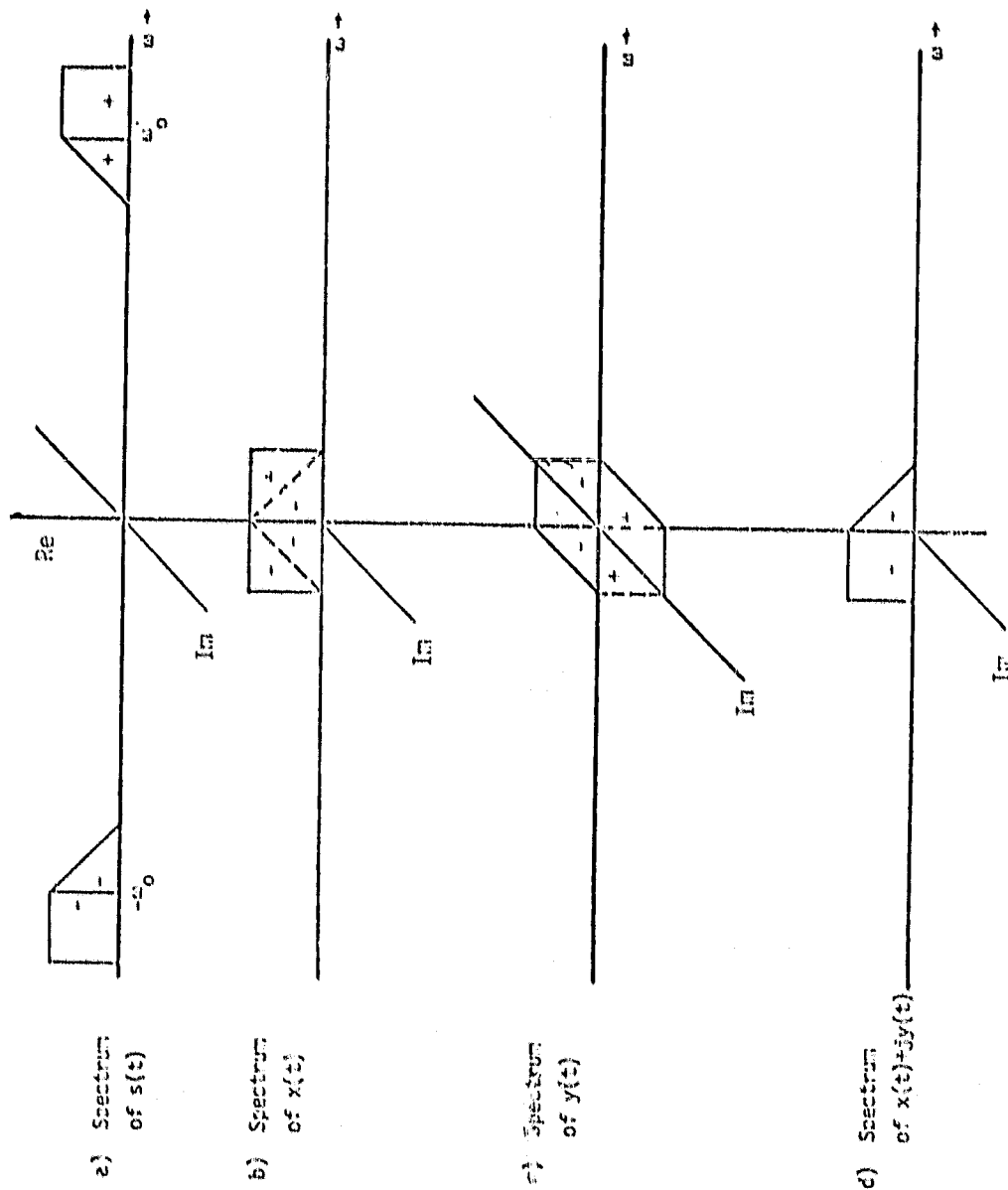


FIGURE 3.6 SPECTRA OF INTEREST

where product modulation with the quadrature reference $\sin \omega_0 t$ has occurred. The lower channel is commonly called the sine channel or the quadrature (Q) channel. The spectra of $x(t)$ and $y(t)$ are illustrated in Figures 3.6b and 3.6c, respectively.

A comparison of Figures 3.6b and 3.6c shows that the aft and fore spectra can be retrieved simultaneously if a complex signal

$$z(t) = x(t) + j y(t) \quad (3.26)$$

is formed. An examination of the spectrum of $z(t)$ indicates that the aft spectrum occurs for $\omega > 0$ and the fore spectrum for $\omega < 0$.

Since the DFT can also be applied to complex signals as well as real signals, the above result shows that the fore and aft spectra can be simultaneously filtered using CZT techniques to implement the DFT. The configuration for implementing the CZT using charge coupled devices and analog multipliers is described below.

3.5.2 The Implementation Technique

It is advantageous to use the sliding version of the CZT, since many sequential measurements are required to estimate the PSD. The actual implementation method is best understood by separating the real and imaginary parts of the sliding CZT. When the sliding CZT of $z(t)$ is taken, $Z_s(k)$ can be rewritten in the form

$$|Z_s(k)| = \sum_{m=1}^N [\cos \pi(m-N)^2/N + j \sin \pi(m-N)^2/N] [x(k-m+N) + jy(k-m+N)] \cdot [\cos \pi(k-m+N)^2/N - j \sin \pi(k-m+N)^2/N] \quad (3.27)$$

A careful interpretation of the arguments within the above magnitude suggests the implementation technique shown in Figure 3.7. Both $x(t)$ and $y(t)$ are pre-chirped and appropriately summed to form the real and imaginary valued entries into the transversal filter bank. Four transversal filters are required to form the cross multiplication products of the complex valued signal entering the filters. The real and imaginary parts of the CZT without the post-multiplication are formed by differencing the outputs of the transversal filters as illustrated in Figure 3.7.

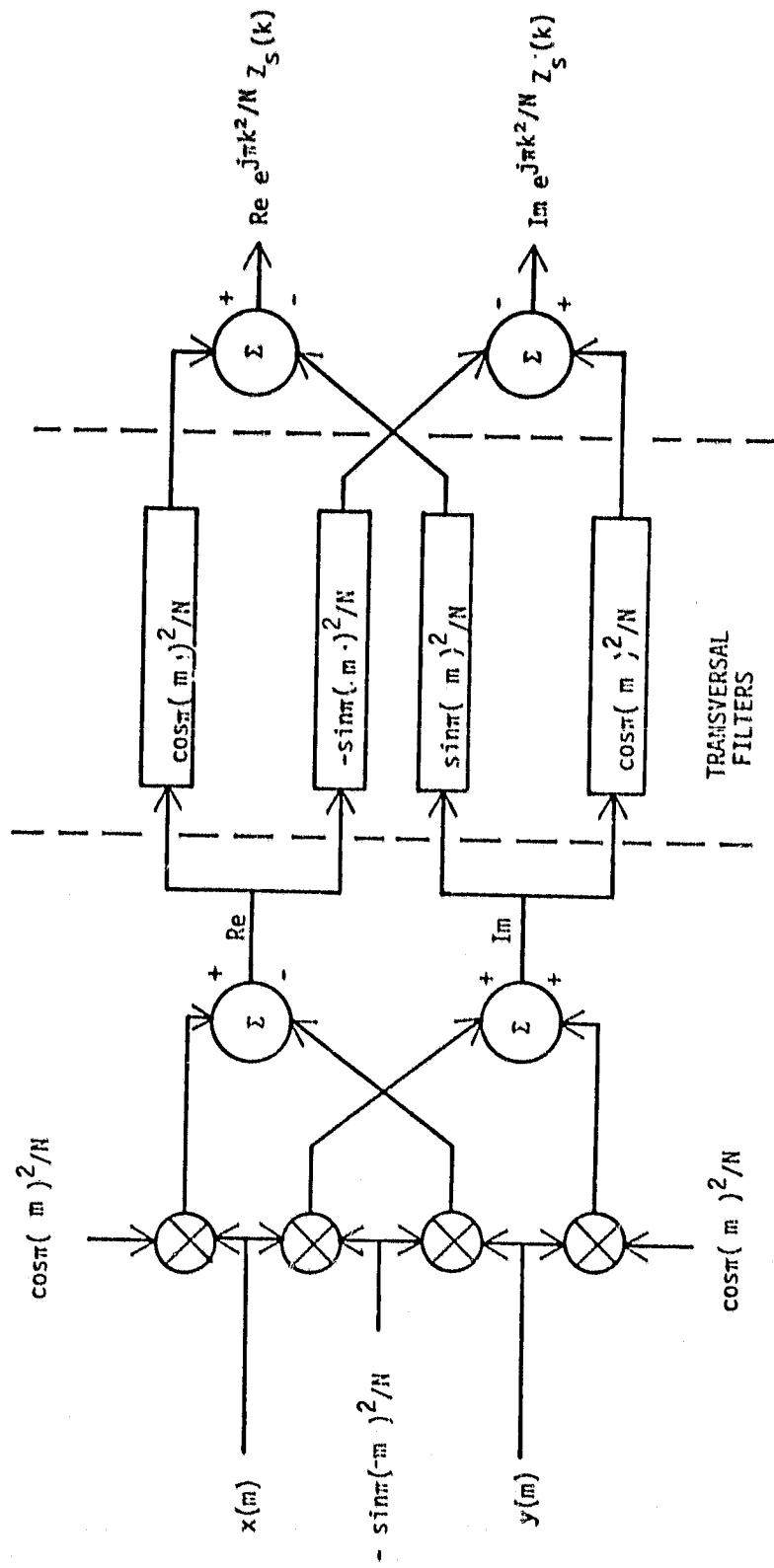


FIGURE 3.7 A TECHNIQUE FOR IMPLEMENTING THE CZT WITHOUT POST MULTIPLICATION

4.0 SYSTEM DESIGN RATIONALE

4.1 Introduction

As is the case with any system, the actual design is a compromise between user requirements and system constraints. The design of real time processors for the NASA scatterometers is no exception. Ideally the scatterometer should provide high precision estimates of σ° with infinitesimal angular and ground resolutions. However, as will be shown below, the precision, angular resolution and ground resolution interact in such a way to prevent maximizing all three parameters simultaneously. In addition to maximizing the precision and resolution parameters, the user is also interested in achieving a reasonable accuracy to permit comparative analysis of the processed data at different view angles and polarizations and with scatterometer data from other sources (also presumably calibrated).

A list of the factors which can potentially influence the performance of the scatterometer/processor system is shown in Table 1. The source of error is described in the left-hand column. The system performance factors most influenced by the error source is reflected in the middle column. The origin of the error within the system is identified in the right-hand columns. Most of the performance parameters are manageable by the processor provided that the scatterometer has been appropriately designed. Those that are manageable are treated below as well as in subsequent chapters to develop the processor design rationale.

TABLE 4.1 FACTORS INFLUENCING SYSTEM PERFORMANCE

Error Source	Affected Performance Factor	Origin in System			
		Scatterometer	Processor	Aircraft	Target
1. Fading Signal	Precision				X
2. Finite Doppler Bandwidth	Precision & Angular Resolution		X		
3. Finite Record Length	Ground Resolution & Precision		X		
4. Filter Sidelobe Level	Accuracy		X		
5. Bit Truncation	Precision		X		
6. Inversion Approximation	Accuracy		X		
7. Uncertainty in Altitude	Accuracy			X	
8. Aircraft Attitude a) Illuminated area b) Polarization decomposition	Accuracy			X X	
9. Transmitted Power	Accuracy	X			
10. Polarization	Accuracy	X			
11. Non-Stationary Return	Accuracy				X
12. Beamwidth	Accuracy & Angular Resolution	X			
13. Pattern Sidelobes	Accuracy	X			
14. Pattern Gain	Accuracy	X			

4.2 Definition of System Design Parameters

To identify the processor's mode of operation it is important to define various parameters associated with fan beam systems. In this regard such terms as angular resolution, ground resolution, scan length, beam resolution, ground track coverage, etc. must be clarified to arrive at the impact of these parameters on the system design.

Angular Resolution

As indicated in Section 3.2 the angular resolution of a fan beam scatterometer is dictated by the physical beamwidth in the crosstrack dimension and by the bandwidth of the Doppler filter in the intrack dimension. If $H(\omega)$ denotes the normalized voltage transfer function of the Doppler filter, then an effective bandwidth may be defined as

$$B = \frac{1}{2\pi} \int_{-\infty}^{\infty} |H(\omega)|^2 d\omega \quad (4.1)$$

where the normalization has been applied so that $\max_{\omega} \{|H(\omega)|\} = 1$ when $H(\omega)$ represented in low pass form. An effective intrack beamwidth may be related to the effective bandwidth through the Doppler relationship

$$\Delta\theta = \frac{\lambda B}{2vc\cos\theta} \quad (4.2)$$

when $\phi \approx 0$. $\Delta\theta$ is defined to be the angular resolution.

Beam Resolution

The beam resolution in the cross track dimension is given by

$$\rho_c = (h \tan\theta) \Delta\phi \quad (4.3)$$

where $\Delta\phi$ is the crosstrack angle subtended by the two way beam when projected on the ground plane. The beam resolution in the intrack dimension is

$$\rho_B = \frac{h}{\cos^2\theta} \Delta\theta \quad (4.4)$$

See Figure 4.1 to clarify these definitions.

Scan Length

Scan length is simply the ground track distance vT traversed by the aircraft during a single integration period T where v is the ground velocity. See Figure 4.2.

Ground Track Coverage

The ground track coverage L_c is that entire length over which a radar return was observed. The coverage includes the scan length as well as the initial coverage within the beam, ie.,

$$L_c = \rho_B + vT$$

See Figure 4.2.

Ground Resolution

The ground resolution may be defined in several ways. However, for the purposes of this design effort the ground resolution is defined as an effective ground length over which radar returns have primarily contributed to the measurement as implied in Figure 4.2. It consequently emphasizes that portion of the ground track which is repeatedly in view within the beam subtended by $\Delta\theta$.

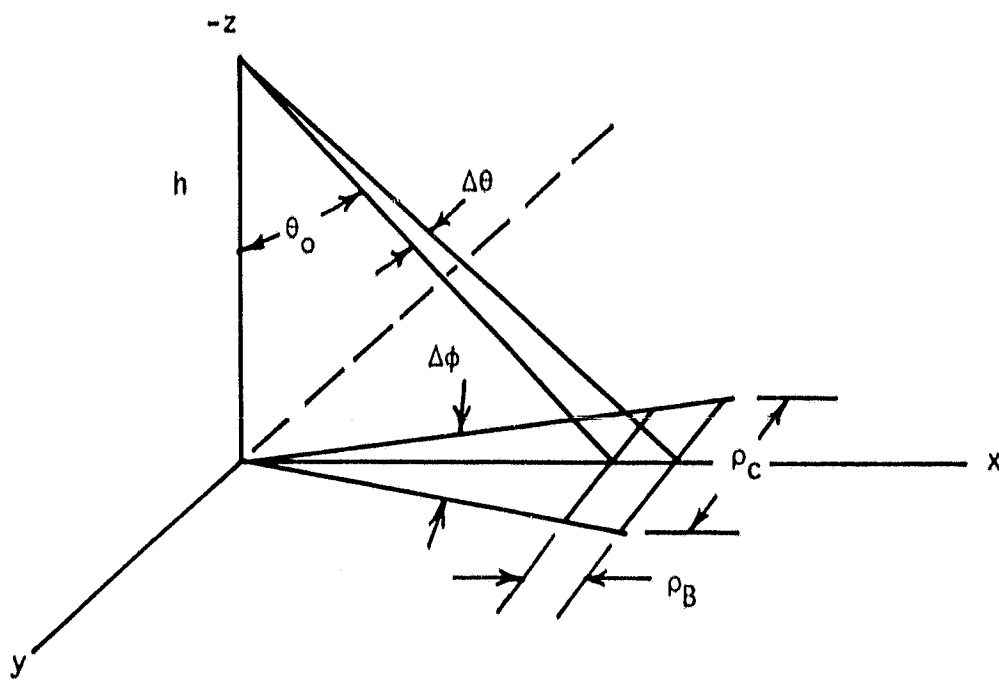


FIGURE 4.1 VARIOUS RESOLUTION PARAMETERS

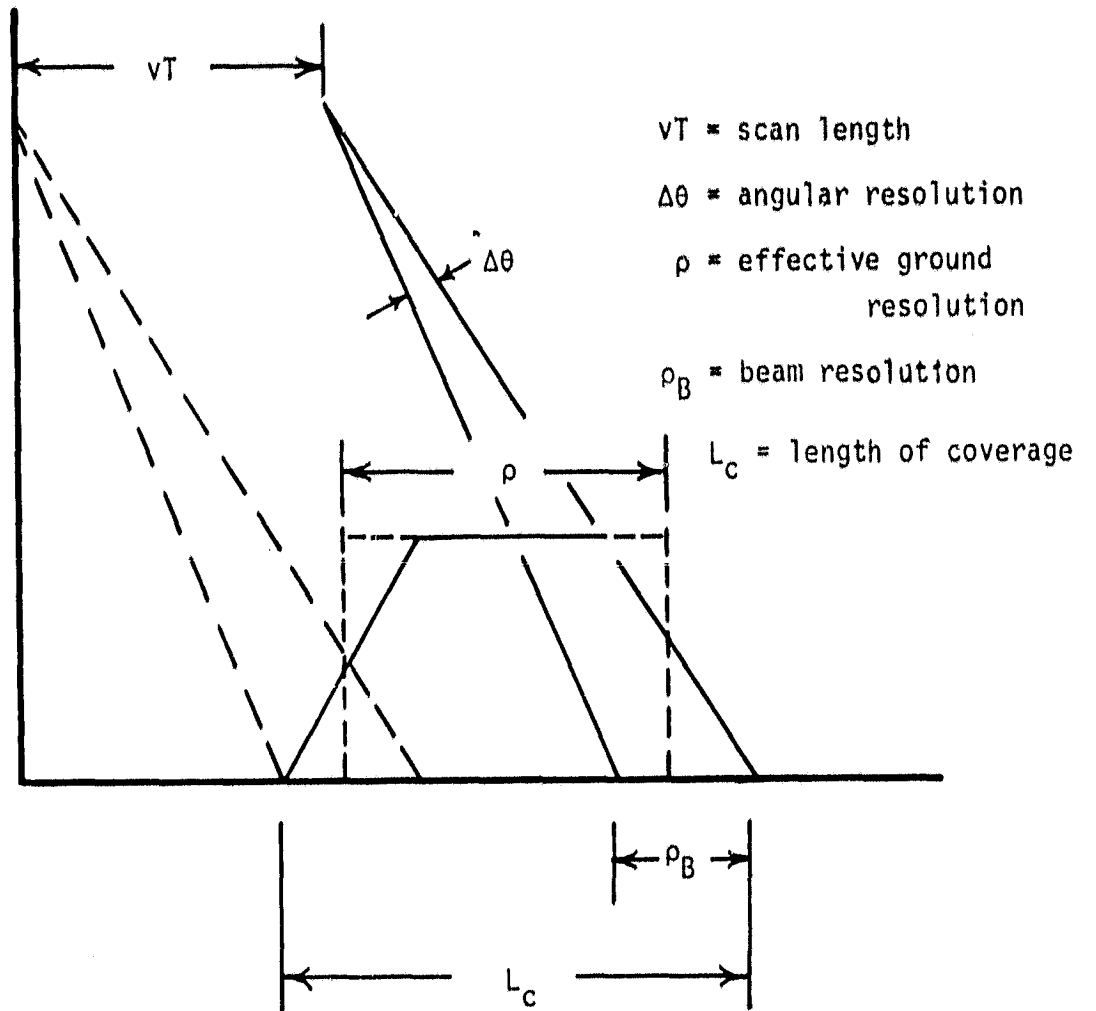


FIGURE 4.2 VARIOUS CELL PARAMETERS

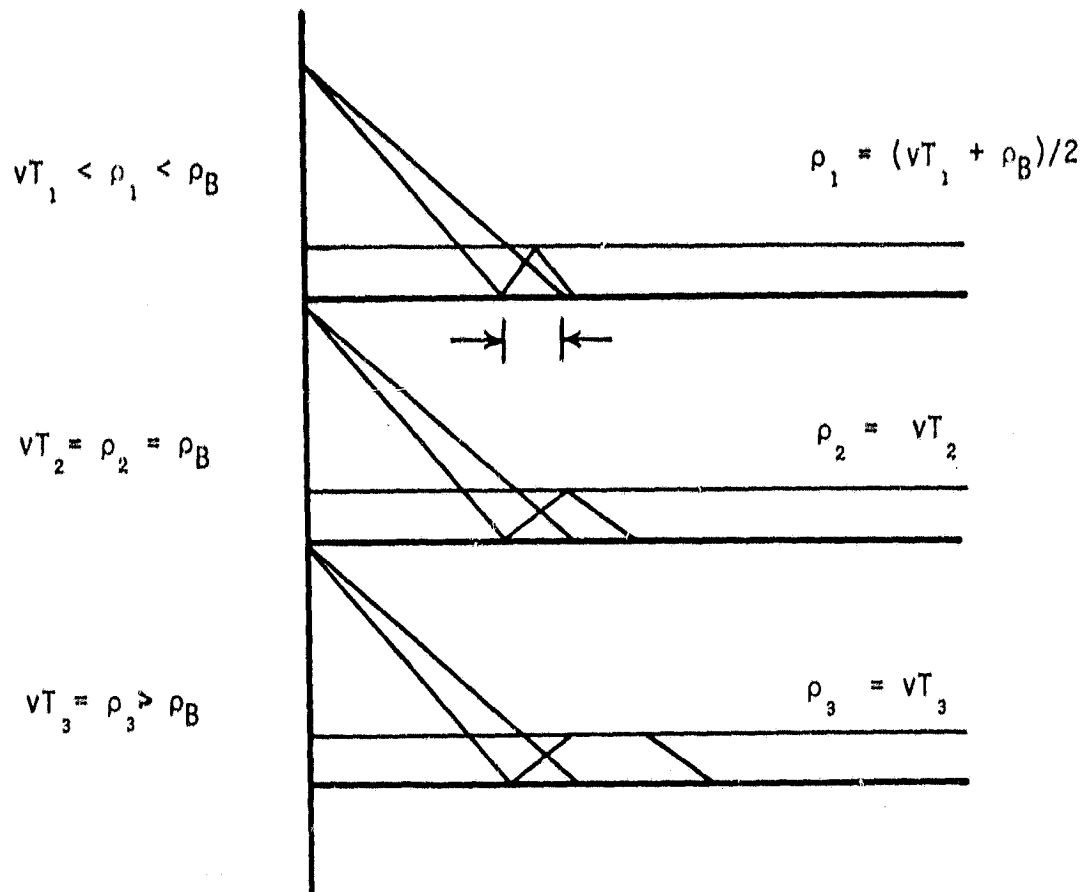


FIGURE 4.3 COMPARISON OF THE RESOLUTION PARAMETERS WHEN
 (a) $vT_1 < \rho_B$, (b) $vT_2 = \rho_B$ and (c) $vT_3 > \rho_B$

Three cases may be identified as illustrated in Figure 4.3. The ground resolution is therefore defined as

$$\rho = \begin{cases} \frac{vT + \rho_B}{2} & \text{if } vT < \rho_B \\ \rho_B & \text{if } vT = \rho_B \\ vT & \text{if } vT > \rho_B \end{cases} \quad (4.5)$$

From the above definitions an important observation can be withdrawn. When the three ground resolution cases are ordered as implied in Figure 4.3 it may be shown that

$$\frac{T_2}{T_1} \geq \frac{\rho_2}{\rho_1} \quad (4.6)$$

and

$$\frac{T_2}{T_3} = \frac{\rho_2}{\rho_3} \quad (4.7)$$

where ρ_j is the ground resolution corresponding to T_j . For a constant Doppler bandwidth these results imply the following conclusion:

Case 2 maximizes the ground resolution consistent with maximizing the precision. It is also interesting to note that Case 2 maximizes the precision for a given coverage interval L_c as demonstrated in Appendix A. In this case the design criterion requires that

$$\rho_B = vT \quad (4.8)$$

at any incident angle. This conclusion will be used in identifying appropriate design theories below.

4.3 The Theory for Constant Precision, Constant Angular Resolution and Constant Ground Resolution Designs

Among the many system designs which could be considered it is helpful to limit considerations to three basic design approaches: (1) constant precision, (2) constant angular resolution and (3) constant ground resolution. The theory for each is presented below and their characteristics are compared in a final subsection.

4.3.1 A Constant Precision Design

A constant precision design approach requires a constant BT product at each incident angle to be processed. The implication deduced from Section 4.2 is helpful in assigning B and T so as to achieve an acceptable BT product. At the smallest incident angle θ_1 the BT product may be maximized for a given coverage L_c by requiring

$$\rho_{B1} = vT \quad (4.9)$$

Then

$$T = \frac{h \Delta\theta_1}{v \cos^2\theta_1} \quad (4.10)$$

and

$$B = 2v \cos\theta_1 \Delta\theta_1/\lambda \quad (4.11)$$

The viewing window at θ_1 is therefore specified as

$$\Delta\theta_1 = \sqrt{\frac{BT \lambda \cos^2\theta_1}{2h}} \quad (4.12)$$

where BT is chosen to achieve the desired precision. When $\Delta\theta_1$ is withdrawn from equation (4.12), T and B are uniquely assigned by equations

(4.10) and (4.11), respectively. On pragmatic grounds the integration time must be constant at all viewing angles. Therefore B is also constant. As a result, the angular resolution, beam resolution, and ground resolution at the remaining view angles θ_k become

$$\Delta\theta_k = \frac{\lambda B}{2v \cos\theta_k} \quad (4.13)$$

$$\rho_{Bk} = \frac{\lambda h B}{2v \cos^3\theta_k} \quad (4.14)$$

and

$$\rho_k = \frac{\lambda h B}{4v \cos^3\theta_1} \left\{ 1 + \frac{\cos^3\theta_1}{\cos^3\theta_k} \right\} \quad (4.15)$$

From the above results it is noted that the angular resolution is inversely proportional to $v \cos\theta_k$, the beam resolution is proportional to $h/v \cos^3\theta_k$ and the ground resolution is proportional to $h \left\{ 1 + \frac{\cos^3\theta_1}{\cos^3\theta_k} \right\}$

4.3.2 Constant Angular Resolution

A constant angular resolution approach requires that $\Delta\theta_k$ be constant at all viewing angles, say $\Delta\theta$. Once $\Delta\theta$ is assigned, the bandwidth at each θ_k is given by

$$B_k = \frac{2v \cos\theta_k \Delta\theta}{\lambda} \quad (4.16)$$

and the beam resolution by

$$\rho_{Bk} = \frac{h \Delta\theta}{\cos^2\theta_k} \quad (4.17)$$

The precision and ground resolution require a rationale to assign T. Once again it is convenient to maximize BT at the smallest incident angle for a given coverage. This requires that

$$\rho_{B_1} = vT \quad (4.18)$$

The remaining parameters then become

$$T = \frac{h\Delta\theta}{v \cos^2\theta_1} \quad (4.19)$$

$$(BT)_k = \frac{2h\cos\theta_k \Delta^2\theta}{\lambda\cos^2\theta_1} \quad (4.20)$$

and

$$\rho_k = \frac{h\Delta\theta}{2\cos^2\theta_1} \left\{ 1 + \frac{\cos^2\theta_1}{\cos^2\theta_k} \right\} \quad (4.21)$$

at each viewing angle $\theta_k > \theta_1$.

4.3.3 Constant Ground Resolution

A constant ground resolution approach requires that ρ be constant at each incident angle. With ρ specified, the precision may be maximized at each incident angle by requiring $\rho = \rho_B = vT$ in accord with Section 4.2. As a result of this imposition

$$\Delta\theta_k = \rho\cos^2\theta_k/h \quad (4.22)$$

$$T = \rho/v \quad (4.23)$$

and

$$(BT)_k = 2\rho^2 \cos^3\theta_k / \lambda h \quad (4.24)$$

4.4 A Comparison of the Design Approaches

To evaluate the three design approaches, the nominal design guidelines shown in Table 4.2 were employed for C band and L band systems.

The guidelines were primarily applied at the smallest incident angle and were allowed to vary at the larger incident angles depending on the design approach. The results of this evaluation are shown graphically in Figures 4.4 and 4.11.

The first four graphs apply to the C band processor whereas the latter four apply to the L band processor. Each figure identifies and compares a single system performance parameter over the entire range of incident angles for the three design approaches. The graphs are parametrically identified by the design approach: CP = constant precision, CGR = constant ground resolution and CAR = constant angular resolution.

Table 4.2 Nominal Design Guidelines

Parameter	Value	Units
Aircraft velocity	150	knots
Aircraft altitude	1500	feet
Angular resolution (nominal)		
C band	3	degrees
L band	6	degrees
Ground resolution(nominal)		
C band	25	meters
L band	50	meters
Precision factor (nominal)		
C band	50	
L band	50	

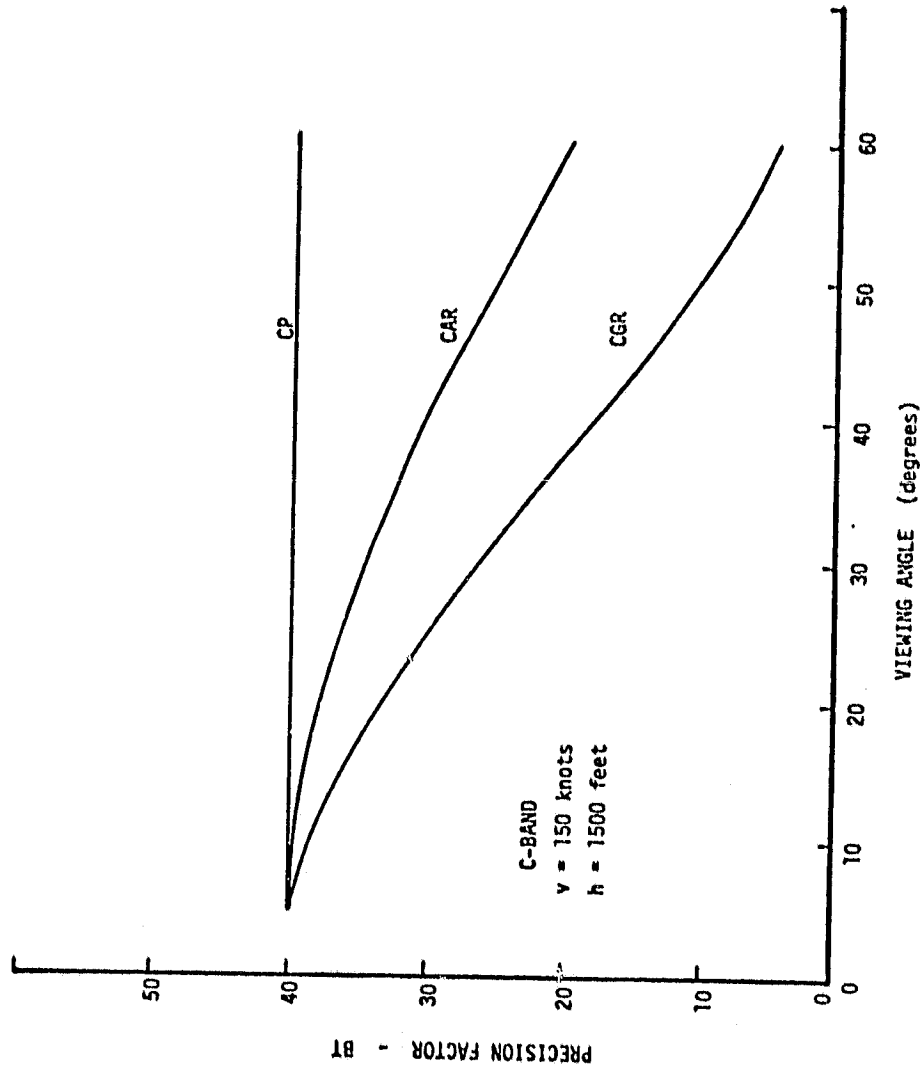


FIGURE 4.4 THE DEPENDENCE OF PROCESSING PRECISION ON VIEW ANGLE FOR THE THREE DESIGN APPROACHES

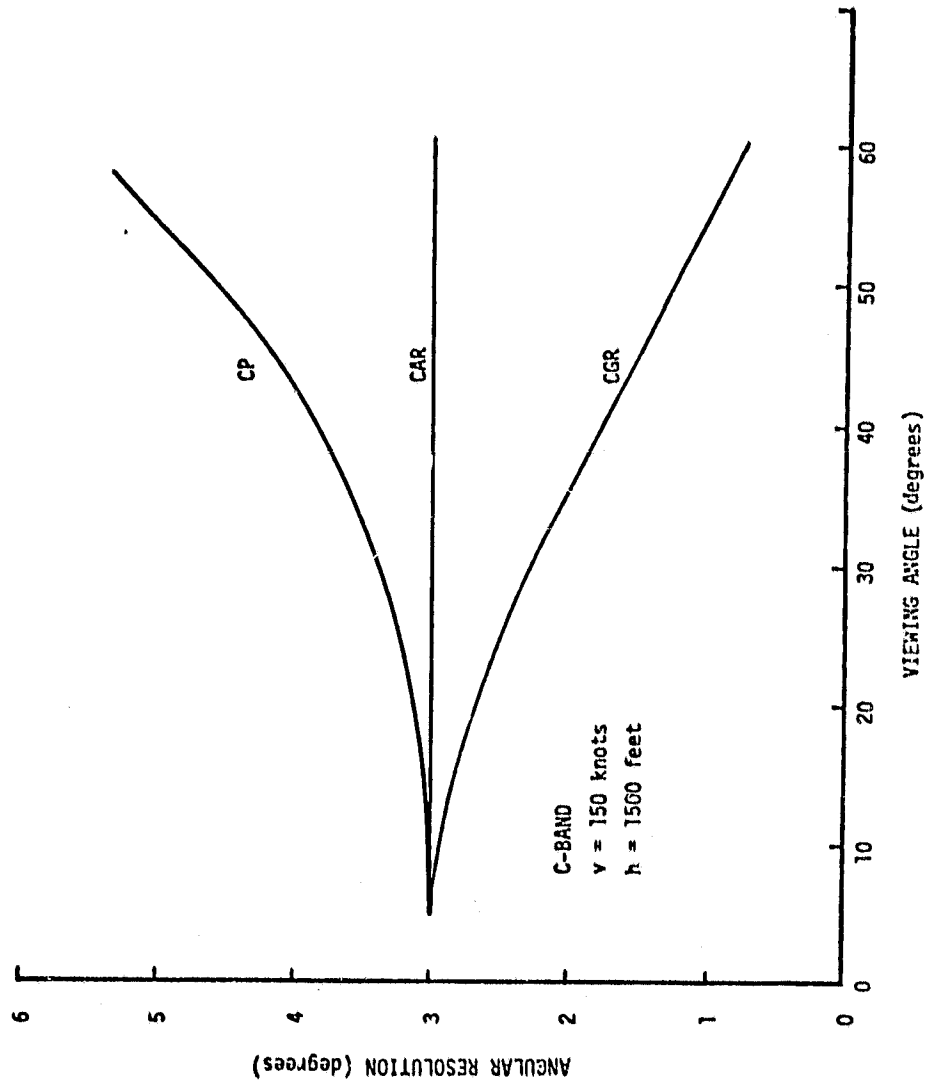


FIGURE 4.5 THE DEPENDENCE OF ANGULAR RESOLUTION ON VIEWING ANGLE FOR THE THREE DESIGN APPROACHES

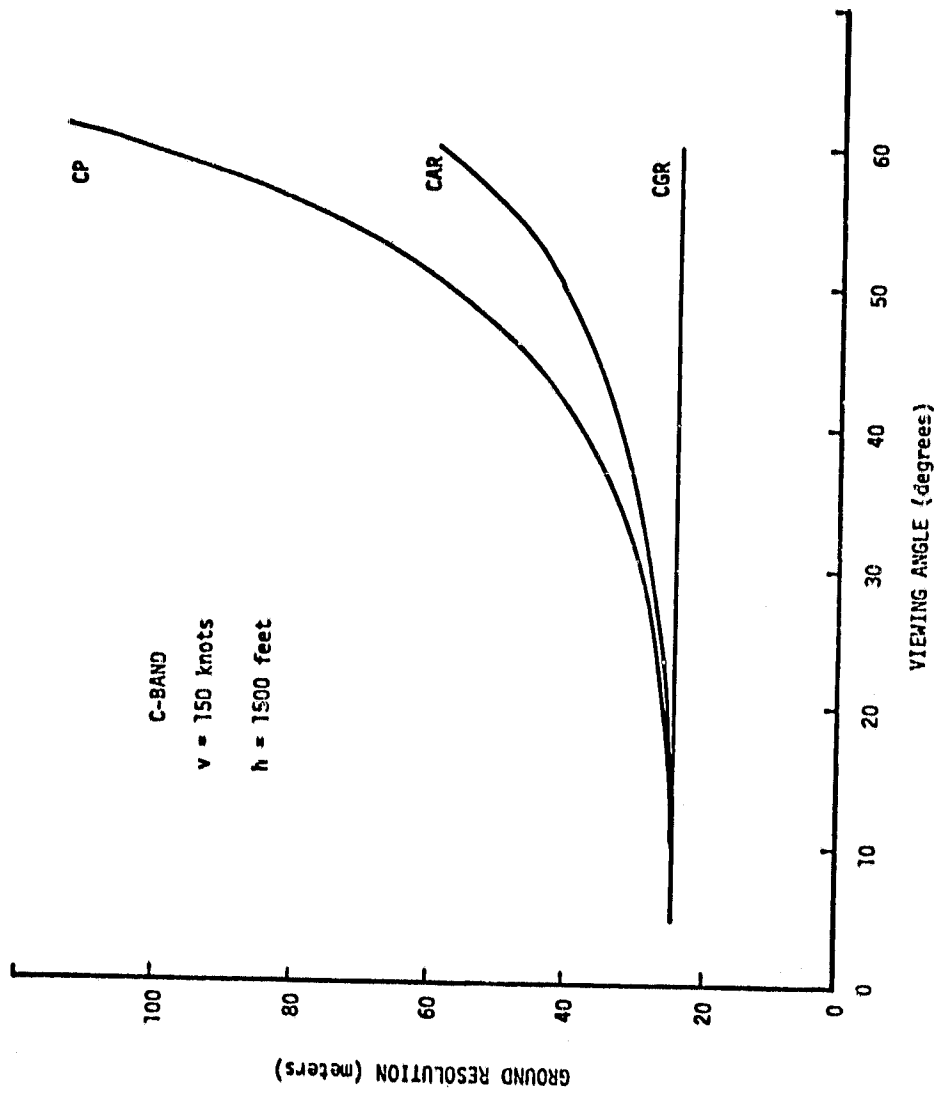


FIGURE 4.6 THE DEPENDENCE OF GROUND RESOLUTION ON VIEWING ANGLE FOR THE THREE DESIGN APPROACHES

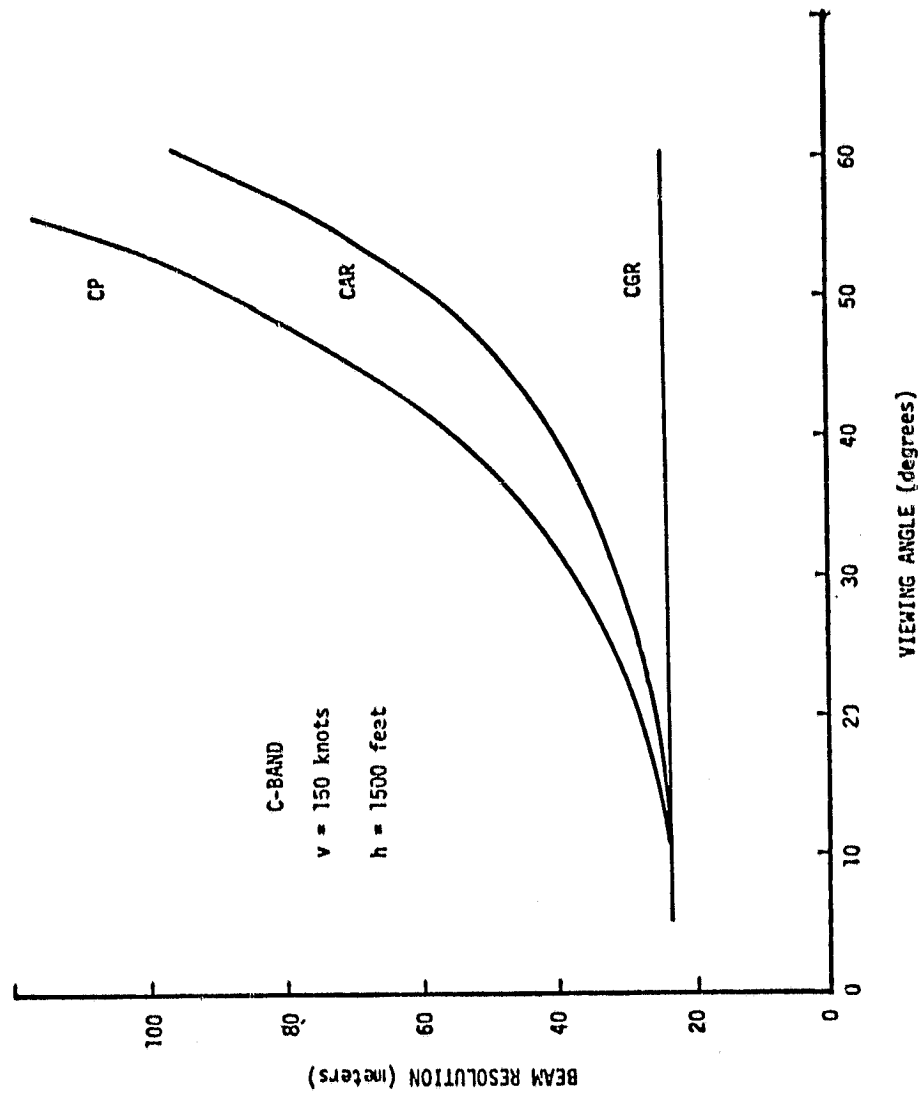


FIGURE 4.7 THE DEPENDENCE OF BEAM RESOLUTION ON VIEWING ANGLE FOR THE THREE DESIGN APPROACHES

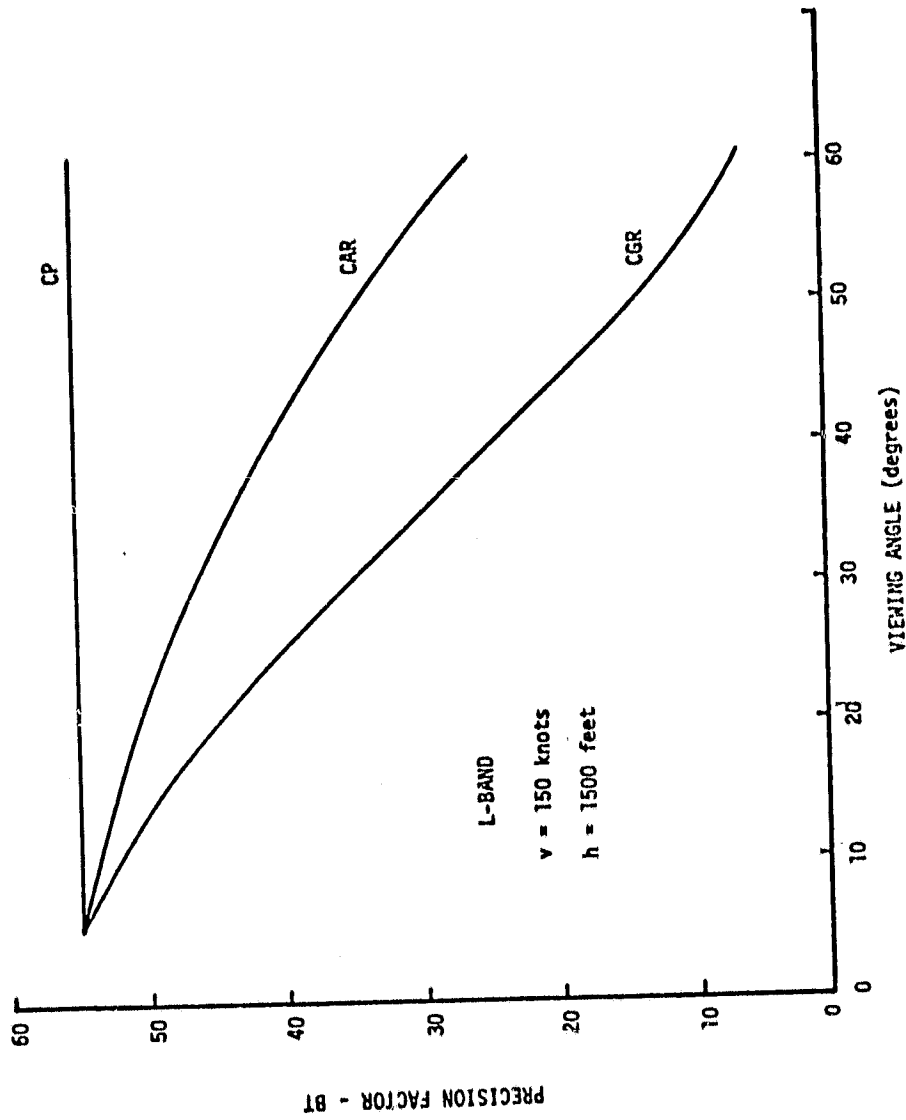


FIGURE 4.8 THE DEPENDENCE OF PROCESSING PRECISION ON VIEW ANGLE FOR THE THREE DESIGN APPROACHES

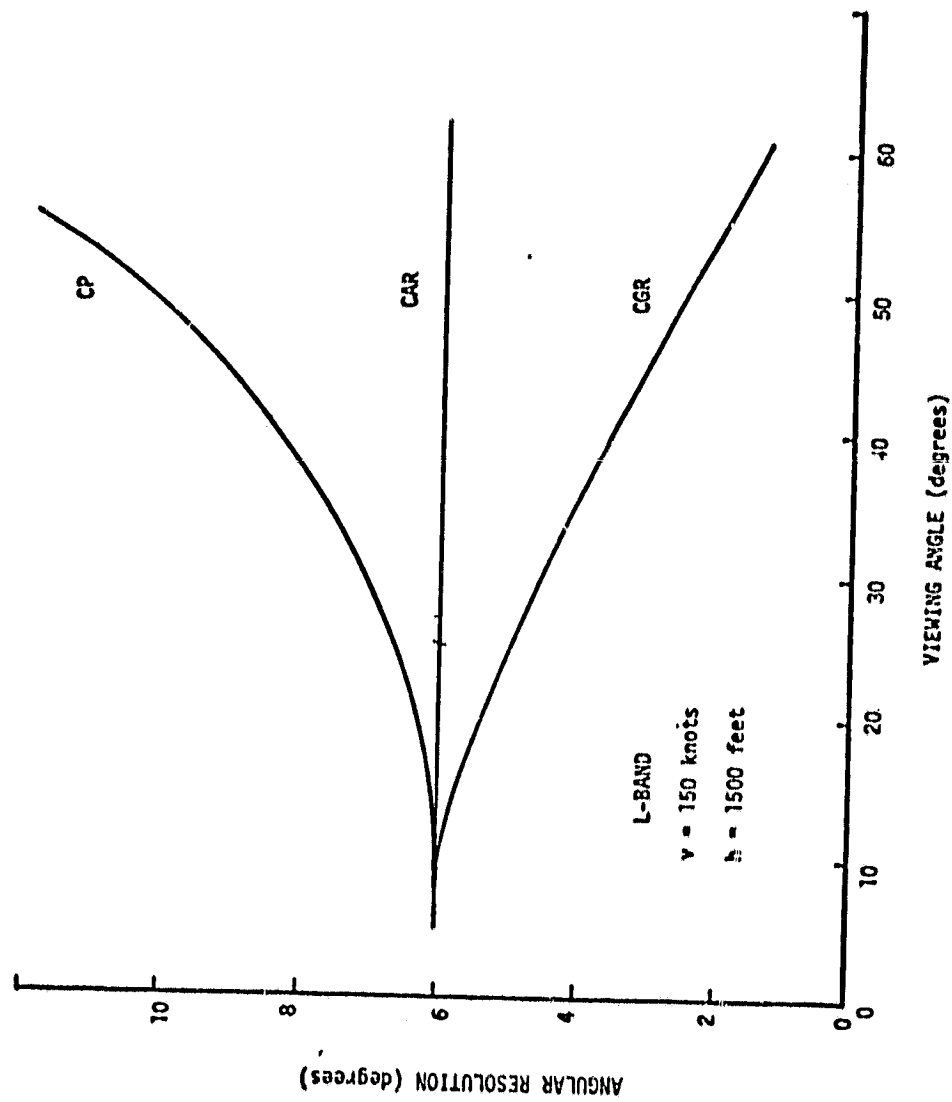


FIGURE 4.9 THE DEPENDENCE OF ANGULAR RESOLUTION ON VIEWING ANGLE FOR THE THREE DESIGN APPROACHES

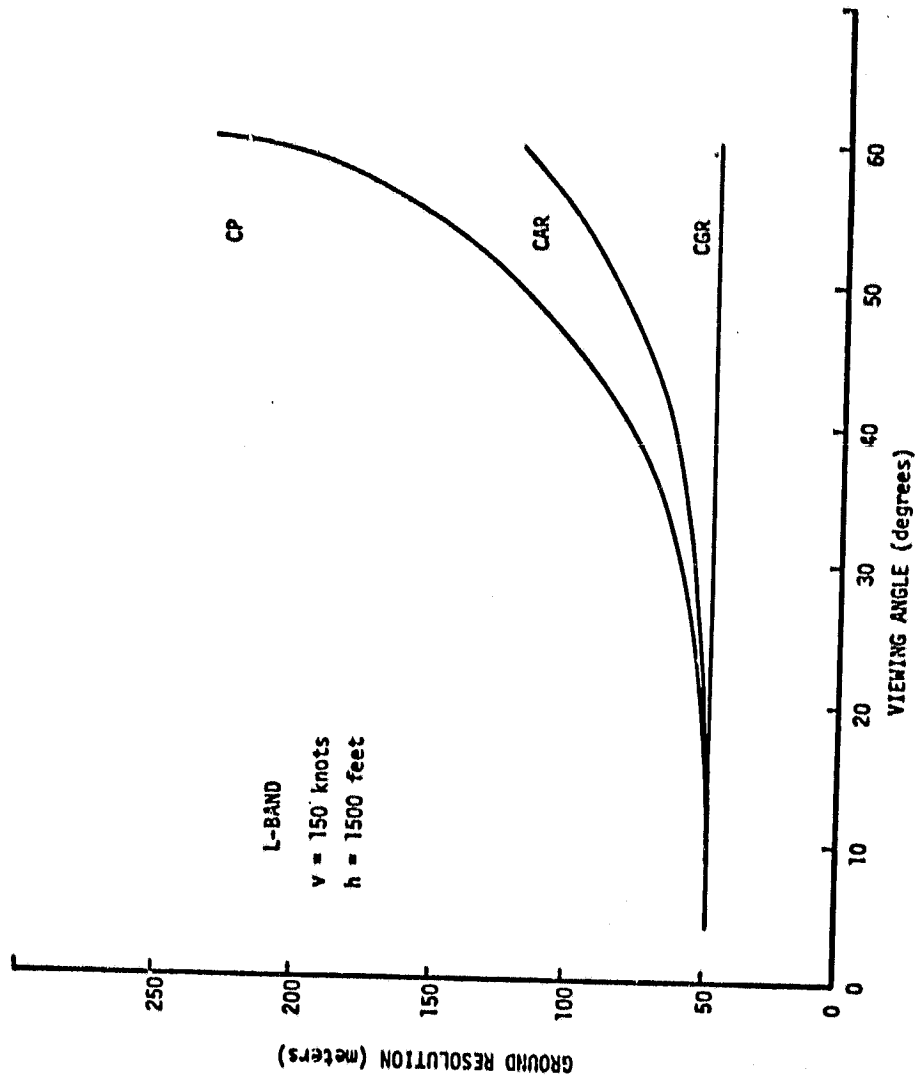


FIGURE 4.10 THE DEPENDENCE OF THE GROUND RESOLUTION ON VIEWING ANGLE FOR THE THREE DESIGN APPROACHES

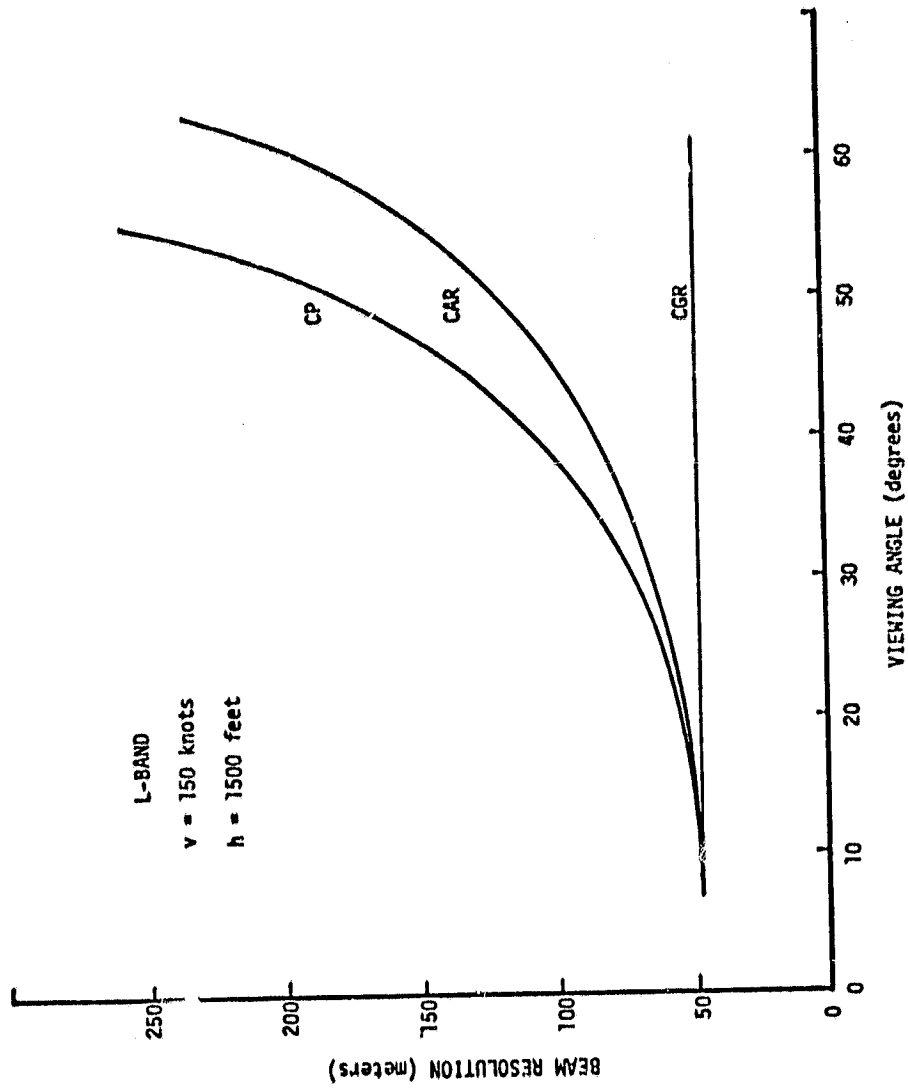


FIGURE 4.11 THE DEPENDENCE OF BEAM RESOLUTION ON VIEWING ANGLE FOR THE THREE DESIGN APPROACHES

From these graphs the following features are noted:

1). When constant precision is imposed, the angular, beam and ground resolutions degrade with incident angle; however, the degradations are only significant for angles greater than 45° .

2). When constant ground resolution is imposed, constant beam resolution is also realized. The precision, however, degrades at the larger incident angles but is useable to 50° . The angular resolution increases rapidly at the large incident angles. The very high angular resolution at the large incident angles may preclude selecting a correct pattern gain at the large incident angles when converting to σ° .

3). When constant angular resolution is imposed, the beam resolution and ground resolution degrade at large incident angles. The performance, in general, lies between the constant ground resolution and constant precision design approaches.

If a single design approach were to be selected among the three, it is apparent that the constant angular resolution approach represents a good compromise between constant precision and constant ground resolution. The constant precision and constant ground resolution designs may, however, suit some experiments better. The constant precision design is attractive in those cases where high precision is required on a single cell, particularly at the larger viewing angles. The constant precision approach may also be helpful in those cases where a large spatial average is required. The constant ground resolution approach is attractive for those applications where well metered - high resolution data are required along the intrack dimension. This design approach is consequently attrac-

tive for those targets which are highly nonhomogeneous.

Since the scatterometer processor is under software control, it is conceivable to provide an experimenter with any design option. The constant angular resolution design represents a good compromise among the approaches, however, when a single approach must be taken.

5.0 SYSTEM ARCHITECTURE AND OVERVIEW

5.1 Target and Development System Architectures

The objective of the scatterometer processing system is to provide real and post time conversion of two channels of scatterometer data, like and cross polarized signals, into σ^0 estimates at eight (8) viewing angles: 5° , 10° , 15° , 20° , 30° , 40° , 50° and 60° . Processor designs were to be developed for the NASA 4.75 GHz and 1.6 GHz fan beam scatterometers. The efforts were 1) to emphasize a standardize design approach suitable for use with these scatterometers as well as future scatterometer systems and 2) to utilize design experience from the previous 13.3 GHz scatterometer processor project. An appropriate target system architecture using the CZT approach to Doppler processing and meeting the above stated objectives is illustrated in Figure 5.1. The system consists of two major subsystems, viz., the PSD estimation subsystem and the micro-processing subsystem. A number of interface units are also provided to permit control of the system, entry of data and storage of processed data. Also a special alignment generator not required during operation but helpful in aligning the CZT filtering unit prior operation is shown.

The PSD estimation subsystem has two channels to handle like and cross polarized return signals. Each channel converts the quadrature signals into discrete PSD estimates over the fore and aft Doppler spectra simultaneously. The estimation technique is based on the CZT technique which may be regarded as an analog technique of implementing the discrete Fourier transform (DFT). The detection (squaring) and accumulation (averaging) of the spectral amplitudes is performed digitally by a

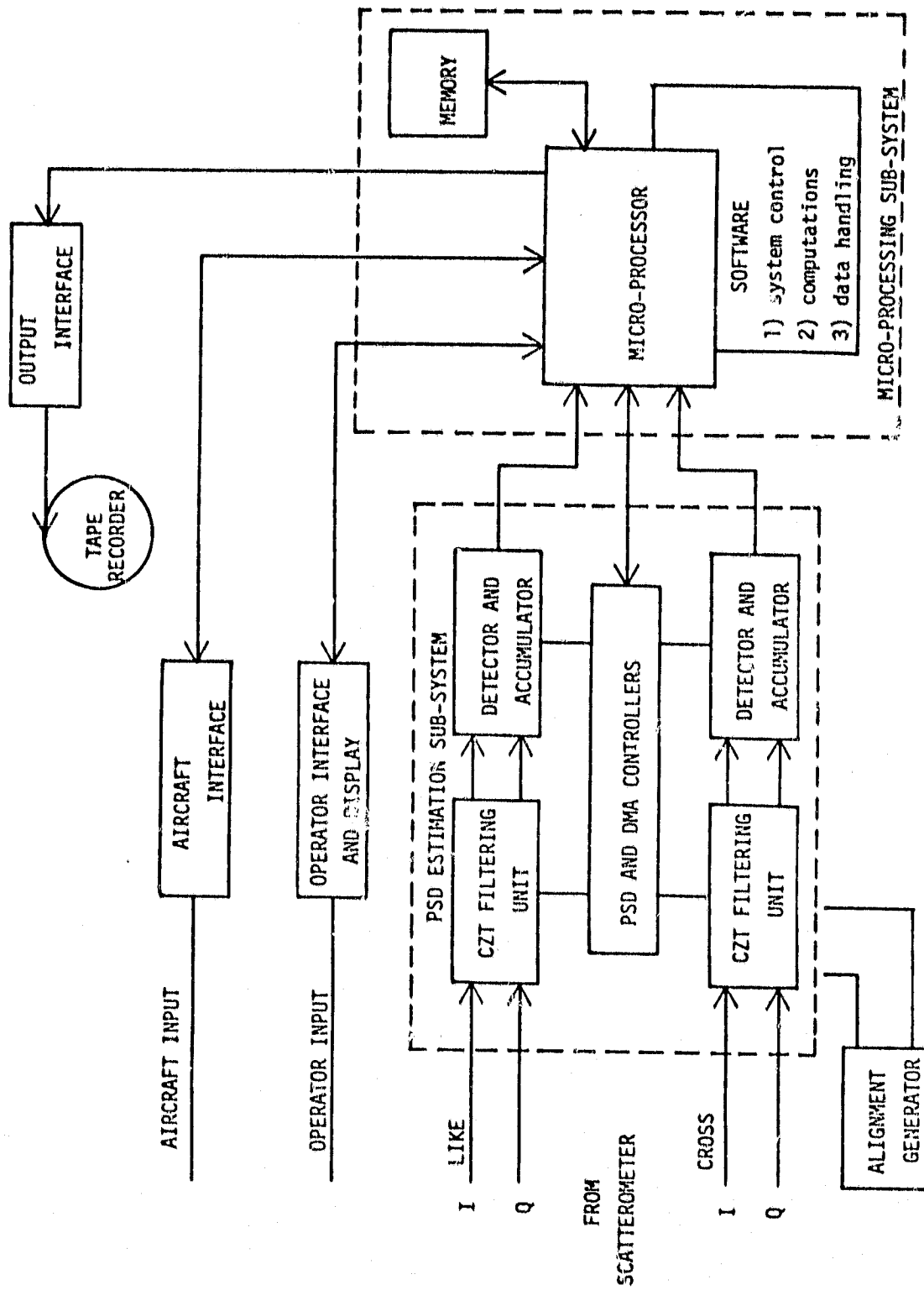


FIGURE 5.1 ARCHITECTURE OF THE TARGET SYSTEM

dedicated processor.

The micro-processing subsystem is composed of a digital micro-computer, memory and associated software. The role of the micro-processing subsystem is 1) to control the scatterometer processor through a software operating system, 2) to select appropriate PSD estimates and to use them within the radar scatterometer equation to invert for σ^0 values on the eight viewing angles and 3) to accept, store and transfer various data required by the processor or the experimenter.

Communication between the micro-processor and various peripheral systems is provided by the interface units indicated in Figure 5.1. Among these interface units is the operator interface and display. All system functions are initiated through this interface by the operator. The display is also used by the micro-processor.

The system of Figure 5.1 represents but a single architecture for the target system. It may not be the architecture of the final system but it will be close. Implied in Figure 5.1 is a single micro-processor to service both scatterometer channels. However, in view of the amounts of data and the number of computations involved, a single micro-processor may be unable to convert both channels of scatterometer to σ^0 values at eight angles without large gaps between successive ground cells. A fast floating point processor such as the Advanced Micro Computer 95/4000 may permit use of a single processor. However, it may be necessary to dedicate a micro-processor to each scatterometer channel. To make an assessment of the number of processors requires that the software design be reduced to machine coding and that the machine be specified.

In view of this uncertainty and with full realization that a complete design effort requires an iterative effort between paper design (called for in this contract) and laboratory evaluation, an engineering model which is expandable to the target system was actually designed. The architecture of the proposed design is illustrated by the block diagram of Figure 5.2. The development model is restricted to single PSD estimation channel and a single micro-processor. The software is sufficiently general to permit processing of like or cross polarized data for any transmit polarization. However, the processor must be cued externally as to which polarization channel the PSD estimation subsystem is connected. A single channel processor of this type is sufficiently simple to fully evaluate the total system design. It can be readily expanded to a two channel system, terminating in either a single micro-processor or two micro-processors. The expansion simply requires that the controller signals be routed to two channels and that simple modifications be incorporated into the software whether one or two micro-processors be required.

5.2 An Overview of the Operation of the System

As indicated above, the scatterometer processing system consists of two major subsystems together with the necessary interfaces to permit communication with various peripheral devices. The relationships among the major subsystems and interface units was shown in Figure 5.2. An alignment generator, although not required during operation, was also shown.

When the system is powered, software control of the system is assumed by the micro-processor within the so-called RESET mode. Within

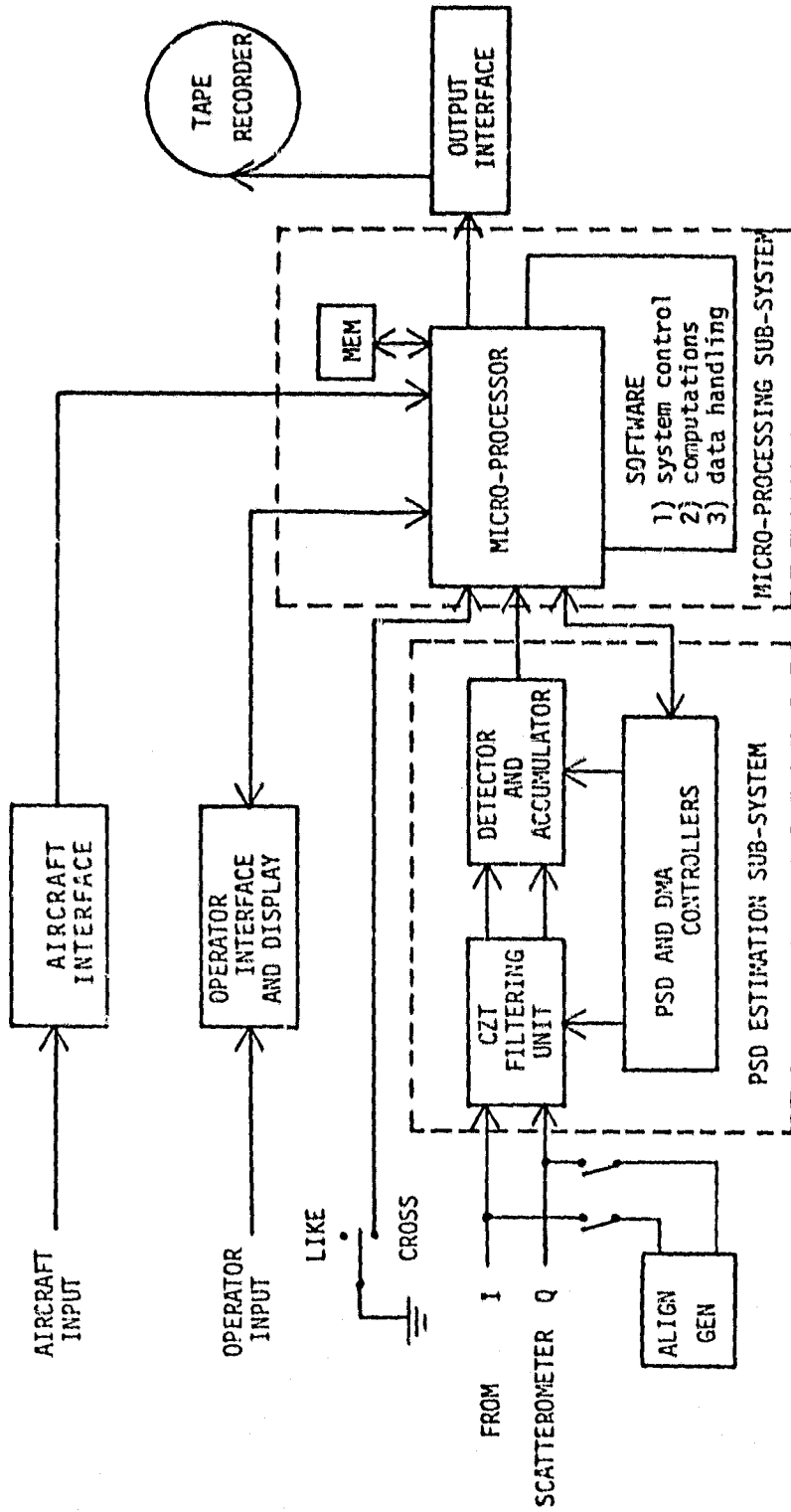


FIGURE 5.2 ARCHITECTURE OF AN ENGINEERING MODEL EXPANDABLE TO THE TARGET SYSTEM

this mode the aircraft, operator and output interface units, and the CZT filtering unit within the PSD estimation subsystem are active (see Figure 5.2). When the system is within this mode, the operator may override certain aircraft parameters arriving at the aircraft interface should they be inaccurate or missing in the aircraft data stream. From the RESET mode, scatterometer data may be processed on a flight line by entering the RUN mode.

Within the RUN mode, the software controller determines the number of 512 point sub-records to be processed to form an average return from the 512 beam resolution cells in view by the CZT filters. The number of sub-records N_R and the address to which the accumulated reserves are to be stored are transferred to the PSD controller (see Figure 5.2). Upon this initialization, the accumulator and detector are activated at the beginning of a CZT cycle. The PSD estimation subsystem then accumulates N_R estimates in each spectral channel. At the end of the N_R sub-records the detector and accumulator are halted and a DMA (direct memory access) transfer is made from the accumulator to the micro-processor memory. If the processor has not been halted by the operator, the detection and accumulation cycle is re-initiated by the micro-processor once the DMA has been completed. It should be noted that the filtering sub-section operates continuously between accumulations to avoid start up transients.

A closer look at the PSD estimation subsystem will indicate that this subsystem filters the fore and aft Doppler spectra of the radar return to form estimates of the return power in 512 parallel spectral channels. Of these channels, 256 are available to characterize the fore spectrum and 255 the aft spectrum. The remaining channel monitors

the return from the nadir point (this return is suppressed by the radar and the processor). A sliding CZT algorithm, as discussed in Section 3.5, is employed to form repeated estimates of the power return within each channel. Several estimates are summed to form an improved estimate of the average power.

The format of the summed spectral estimates in the 512 channels is illustrated in Figure 5.3. When the forward CZT is formed on the complex signal $x + jy$, the aft spectrum appears in the first 257 accumulation bins (channels) and the fore spectrum in reverse order in the latter 255 accumulation bins. Some of the spectral channels among the 512 channels are reserved for the calibration and polarization tones. The effective bandwidth of each channel is given by $B = f_s/512$ where f_s is the sampling frequency. The frequency resolution, i.e., the separation between spectral estimates is also equal to B . The normalized frequency response of each filter in dB is illustrated in Figure 5.4. The filter efficiency to the first sidelobes is 90.3% and through the first sidelobes is 95.5%*. The effective bandwidth is also equivalent to the width of the mainlobe.

The average spectral estimates together with calibration and polarization tone levels are transferred from the PSD estimation subsystem to the micro-processing subsystem through a DMA process initiated by the

*It is possible to increase the mainlobe efficiency by weighting the tap points on the CZT transversal filter. Such a device is available. However, this design is not recommended since 1) the precision will be reduced by a factor of two (the adjacent channel outputs are highly correlated) and 2) the S/N ratio referred to the output of the weighted transversal filter degrades by a factor of two (the noise is primarily governed by the post amplifier and signal output is reduced when weighting is used).

W = BANDWIDTH CORRESPONDING TO DESIRED
ANGULAR RESOLUTION

B = FREQUENCY RESOLUTION AND EFFECTIVE BANDWIDTH

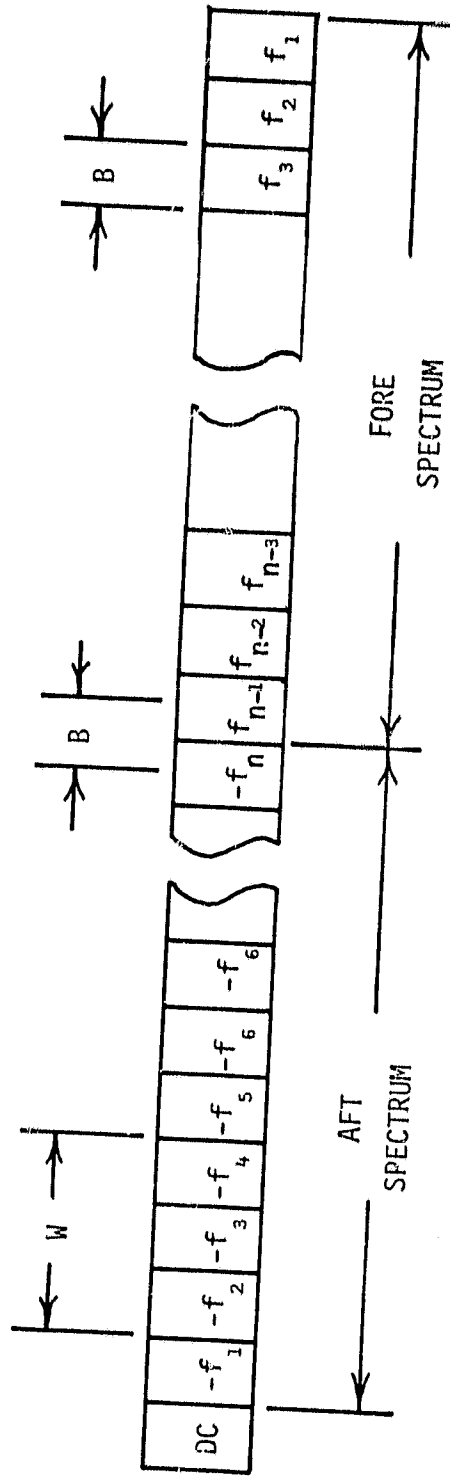


FIGURE 5.3 SPECTRAL DATA FORMAT AND RELATED PARAMETERS

EQUIVALENT BANDWIDTH = $1/T$

FILTER EFFICIENCY = 90.28%

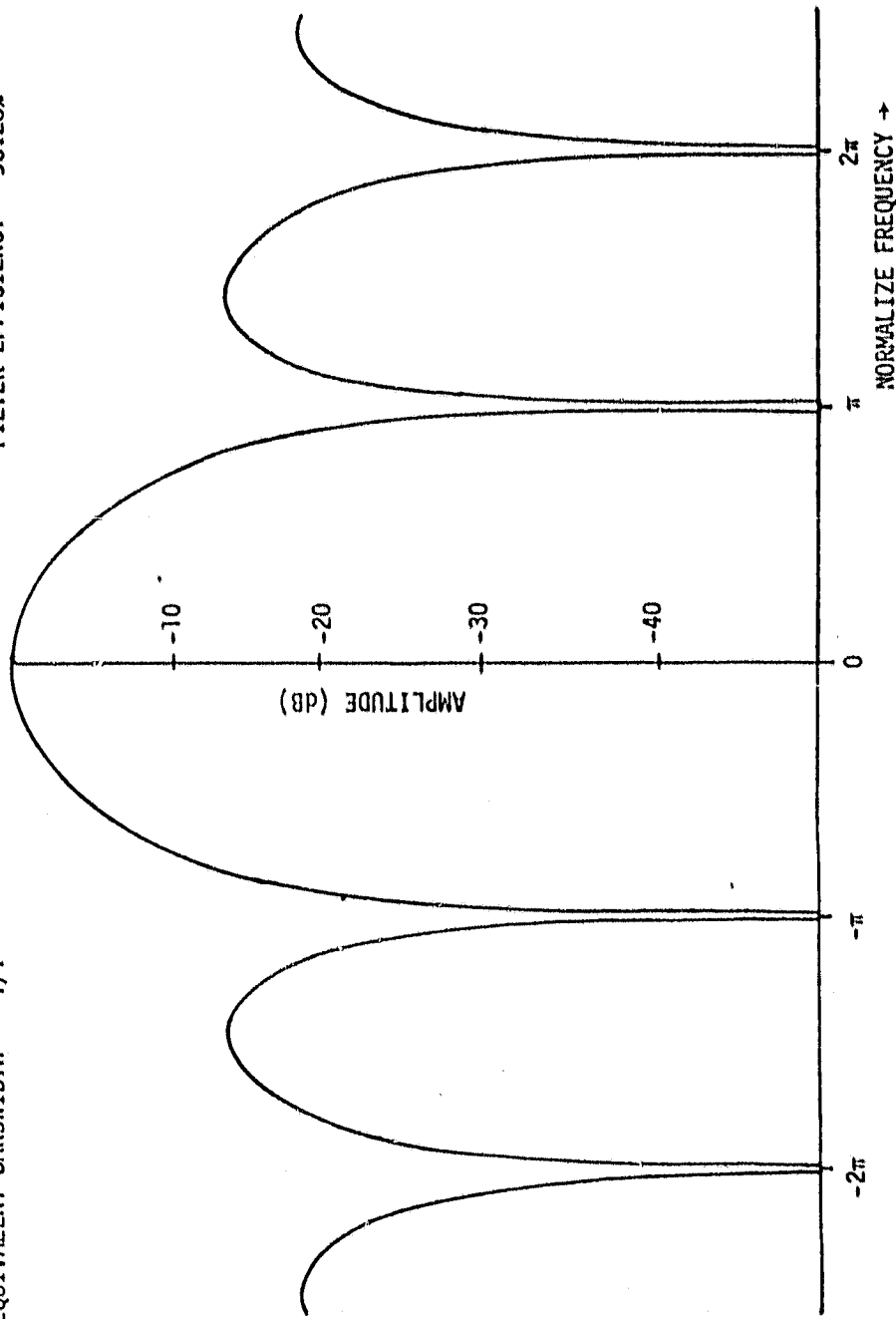


FIGURE 5.4 FILTER POWER SPECTRAL RESPONSE

PSD controller. Once the detector and accumulator are re-initiated, the micro-processor, using data from the aircraft interface together with invariant calibration constants stored in ROMs (Read Only Memories), converts the spectral and calibration data to σ^0 estimates at the required eight viewing angles. Adjacent spectral estimates are summed about each viewing angle to form a total return within the desired angular resolution. The spectral lines are chosen dynamically to track the specified viewing angle.

The processed scatterometer data and other parameters are stored in an array of the micro-processor memory for each of the viewing angles. The σ^0 estimates at the eight angles are stored in memory in an askewed fashion to provide near collocation of the returns on a single cell. When the re-ordered data is available on a single cell at all angles, it is written out to the output interface which in turn transfers it to magnetic tape. The transfer of the data continues until the operator halts the processor. When halted, the processor places an end of file indicator within the output array, transfers the partially filled array to tape and enters the RESET mode to await instructions from the operator. At the beginning of each accumulation cycle, appropriate parameters are withdrawn from the aircraft data channel for use in the computations and for transmittal to the output tape.

6.0 Software Development For C-Band/L-Band Real-Time Radar Data Processor

6.1 Background

During 1979, under NASA Contract NAS9-15311, the analysis and design of a 'real-time' software algorithm for processing fan-beam radar data was completed. The product of the 1979 work was a new technique for producing aligned, digital scattering coefficient values for up to eight aft viewing angles at the same time the raw data was being acquired. The increased capability of this data reduction technique was due mainly to the use of a hardware chirp Z-transform (CZT) to simultaneously produce all the filtered frequency domain data from the analog I and Q channel inputs. A previous hardware processor used individual analog filters which were sampled sequentially. This processor was designed to perform only a quick look, data validation function, rather than provide fully reduced cross section data. The sequential filtering technique was inadequate for full real-time processing in that it could not provide adequate along track coverage while maintaining an acceptable time bandwidth product [1].

The current production techniques used for processing radar data to scattering coefficients utilize several stages of computer processing, all of which require expensive digital Fourier transforms to convert the data from the time domain to the frequency domain. The objective of the effort during this contract period was to demonstrate

that this entire process could be accomplished with micro-processor technology along with a relatively inexpensive CCD transversal filter to accomplish the normally expensive domain transformation of the input data.

6.2 Software Algorithm

6.2.1 General Concept

The computational sequence used for sigma-zero calculations is essentially the same as that described in Section 8.0 of reference [1]. Some minor changes were made in the way some of the individual parameters are developed, and in the order and refresh rate of the aircraft data input. The most significant difference in the overall system is in the number of options available to the operator when setting the mode of operation.

6.2.2 System Initialization and Set-Up

In its simplest form the system logic is as illustrated in Figure 6.1. The initialize task opens the I/O port for the CRT, the serial port for the Bi-Phase L, and sets the CZT board for mapping the frequency domain data to micro-processor memory. Also initialized are the NERDAS port and the Interrupt Controller on the SBC 80/20. Table 6.1 defines the I/O Ports used by the system.

After initialization, the software begins a series of queries to the operator to determine the exact set-up and run mode options to be used. The system software is designed to process either C-Band or

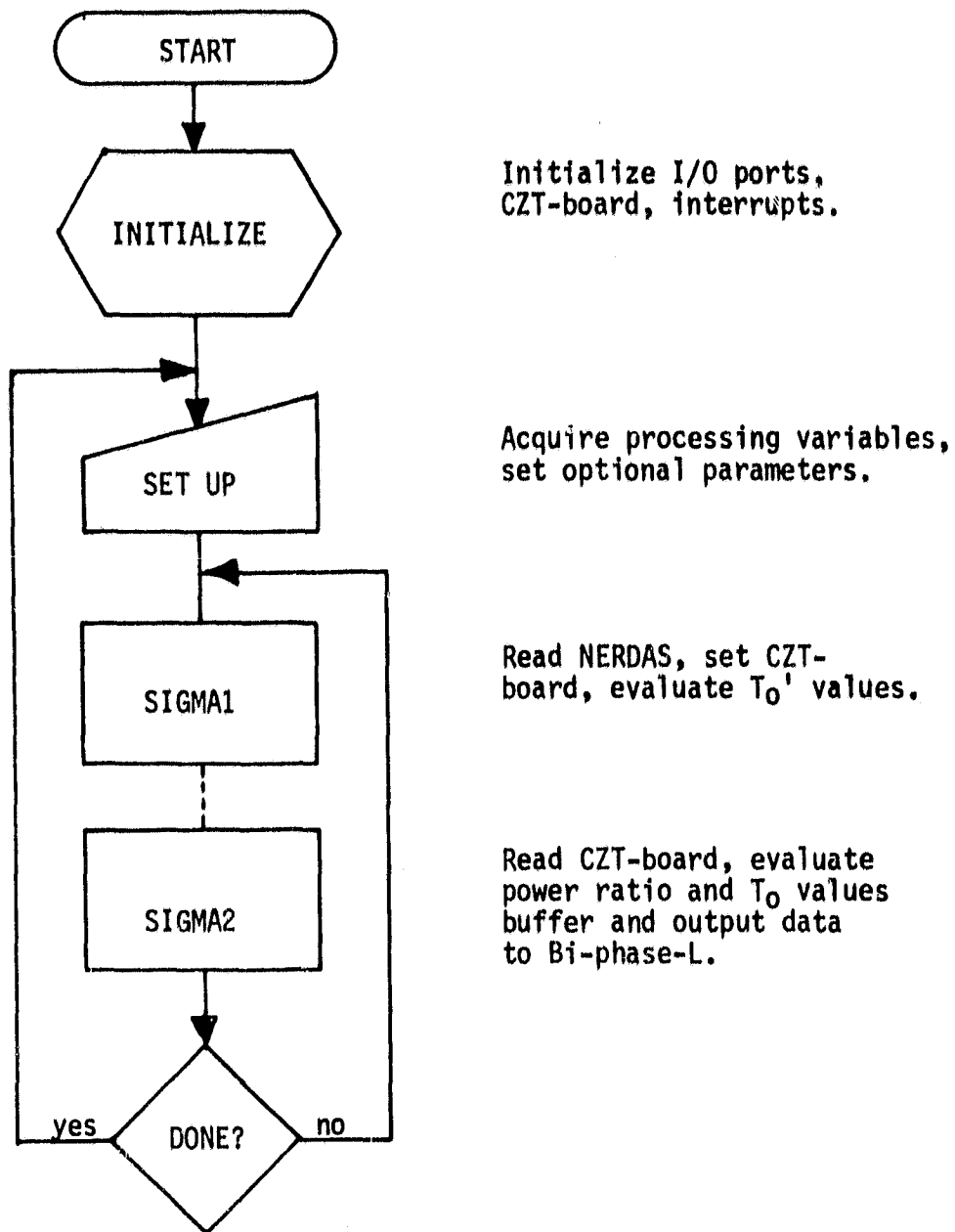


FIGURE 6.1 Overall System Structure.

TABLE 6.1

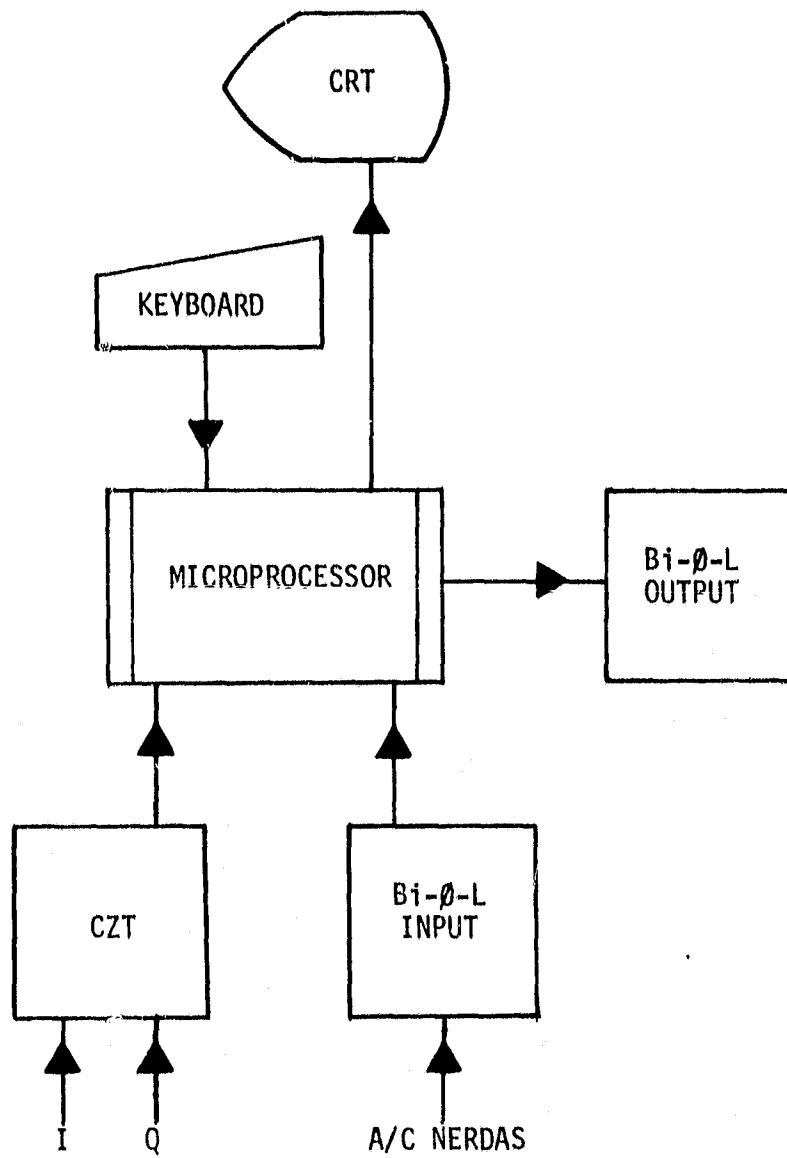
I/O PORTS USED BY NASA-CZT PROCESSOR

<u>PORT ADDR (HEX)</u>	<u>LOCATION (BOARD)</u>	<u>FUNCTION</u>
08	CZT BOARD	Load NBR RCDS
09	CZT BOARD	Begin DMA Transfer
0A	CZT BOARD	DMA ADDR (LSB)
0B	CZT BOARD	DMA ADDR (MSB)
CC	SBC 116	CRT Data, I/O
CD	SBC 116	USRT Control Port
D8	SBC 80/20	Interrupt Controller, Control
D9	SBC 80/20	Interrupt Controller, Control
E4	SBC 80/20	Bi-phase-L, Data Port A
E6	SBC 80/20	Bi-phase-L, Data Port C
E7	SBC 80/20	Bi-phase-L, Control PI/O
EC	SBC 80/20	NERDAS, Data IN
ED	SBC 80/20	USRT Control Port

L-Band input data of any desired polarization combination. Considerable flexibility has been given to the operator regarding use of system constants and aircraft data inputs. Further, the operator is allowed to change the ground cell resolution and thus may make interpretive analysis of resolution effects on sigma-zero for repeated passes through the same data set.

Figure 6.2 illustrates the general system layout from the operator's point of view. System communication and control is provided through the keyboard/CRT. One type of input data may be selected and processed; e.g., vertical, cross-polarized, L-band. The system may be operated in various combinations of 'Open-Loop' or 'Timed' modes. During 'Open-Loop' operations, no defined start time is given and no defined stop time is given. The system begins processing on permission of the operator and continues until the operator gives the stop command. In the 'Timed' mode the system is given a data set start time; e.g., 14:15:21, and a data set length in seconds. The system begins processing on permission of the operator and on detection of the desired start time in the NERDAS Bi-phase-L input frame. Processing continues until the specified number of seconds of data have been acquired.

Specific options given the operator during operation vary according to whether the first data set is about to be taken or subsequent sets are being acquired. The Flow charts in Appendix A give complete details on all available options and exactly how they are provided. Some highlights of the system flexibility are:



Polariz: (HH/HV/VV/VH)
 Band: (L/C)

FIGURE 6.2 General System Layout,
 Real Time Processor.

- a. Band {L/C}, select L or C
- b. Polarization {HH/HV/VV/VH}, select one combination
- c. Cell Resolution Over-ride {YES/NO}
- d. System Constant Over-ride {YES/NO}
- e. NERDAS DATA Over-ride {YES/NO}
 - If 'YES':
 - ALTITUDE Over-ride {YES/NO}
 - DRIFT Over-ride {YES/NO}
 - ROLL Over-ride {YES/NO}
 - PITCH Over-ride {YES/NO}
 - VELOCITY Over-ride {YES/NO}
- f. Start Time Select {YES/NO}
- g. Run Time Select {YES/NO}
- h. Display Set-up Data {YES/NO}

On the second and subsequent data sets the operator may choose whether the above set-up is retained or whether a new set-up is used. All necessary data for calculating the sigma-zero values is requested from the operator during the set-up phase of the program.

As each operation is initiated, appropriate pointer settings are computed for accessing the correct beamwidth, wavelength, filter size, calibration frequency, noise frequency band, antenna gain table, roll-off function, processing constant, and aircraft data. In addition, flags are set (which are later put in the output data frame) that indicate to the data user which options were selected and which over-rides were used. Finally, after the start and stop times are set (if selected) and the option to display the set-up has been made, the operator is given the option to start taking data. If the operator response is 'Y', the system begins reading and processing the raw input data. As each set is processed it is placed in the output data buffer for transfer to the Bi-phase-L output port.

To ease the formatting burden, especially in the set-up module, the main driver program was coded in FORTRAN, using an INTEL MDS230 to

enter and prepare the source code for the INTEL FORT80 compiler. Although the FORTRAN was used extensively for internal data formatting, all I/O is handled by special device service routines, each written in assembler language.

In the set-up module all mathematical expression were accomplished using the INTEL floating point software library routines. Since speed in this portion of the system was not critical, the time saved in coding was considered a good trade-off for additional core-usage and slower execution. However, all floating-point evaluations in the data processing and output modules are done using the AMC 95/6011 hardware arithmetic unit. Special device service routines were written in assembler language to perform all the required operations. Appendix D lists all AM9511 routines and defines their particular function.

6.2.3 'RUN' Mode of Operation

After all set-up data has been acquired, the 'RUN' mode is initiated by an operator 'Y' response to the system query. The 'RUN' mode reads NERDAS, sets integration time and starts integration by the CZT-board. Initial data is processed to calculate σ^0_i values for each of the eight viewing angles.

6.2.3.1 Calculating σ^0_i

The expression which must be evaluated for each of the eight viewing angles is

$$\sigma^0_i = \frac{(4\pi)^3}{\lambda^2} \frac{CL}{K} \frac{Zw(fdc_i)}{GG(\theta'_{ij})} \frac{R^4(\theta_{Li})}{A(\theta_{Li})} \frac{PR(\theta_{Di})}{P_T} \quad (6.1)$$

where: σ^0_i = estimated scattering coefficient.

λ = wavelength.

C_L = cable loss term.

K = system constant.

$Z_w(f_{dc_i})$ = system roll-off, function of doppler frequency,
 f_{dc_i} , for each filter center.

$GG(\theta_{L_i})$ = antenna gain, function of viewing angle through
the antenna.

$R^4(\theta_{L_i})$ = target cell range, function of viewing angle.

$A(\theta_{L_i})$ = cell area, also function of viewing angle.

$PR(\theta_{D_i})$ = received power, function of each doppler angle.

P_T = transmitted power.

i = index for viewing angle, $i=1, 8$.

Note that after integration is started, all the terms in the above expression can be evaluated except the power ratio, P_R/P_T . After arranging the equation so that

$$\sigma^0_i = \sigma^{0'}_i (P_{Ri}/P_T) \quad (6.2)$$

where

$$\sigma^{0'}_i = \frac{(4\pi)^3}{\lambda^2} \frac{C_L}{K} \frac{Z_w}{GG} \frac{R^4}{A} \quad (6.3)$$

it can be seen that $\sigma^{0'}$ may be evaluated while the CZT-band is performing the integration. This is done in the SIGMA1 code module for each of the eight viewing angles. The value $(4\pi)^3/\lambda^2$ is a stored-constant value, one for each band. Similarly, C_L/K are stored

values. However, there are eight values in all, four for each band. The roll-off, Z_w , is calculated using an appropriate expression for the band and polarization being processed. The antenna gain, G_G , is acquired by table look-up using an appropriate table of values for the band and polarization in use. The L-band tables have values for angles ranging over -70 to $+10$ degrees, while the C-band tables range from -60 to 0 degrees.

The R^4/A term is evaluated as two separate values, R^4 and A . The range, R , is evaluated using the expression

$$R_i = H/\cos\theta_i \quad (6.4)$$

where H = altitude, and
 θ_i = viewing angle

The above expression is derived from the system geometry shown in Figure 6.3. The range expression is not exact, but is sufficiently accurate as justified in Reference [1], Section 8.0. The area term, A , is evaluated using the expression

$$A = W'(Y_2 - Y_1)/\cos\phi \quad (6.5)$$

where

W' = cell width, function of altitude, beamwidth, viewing angles and aircraft roll.

Y_1 = Y-coordinate of doppler contour, lower frequency.

Y_2 = Y-coordinate of doppler contour, upper frequency.

ϕ = aircraft drift angle.

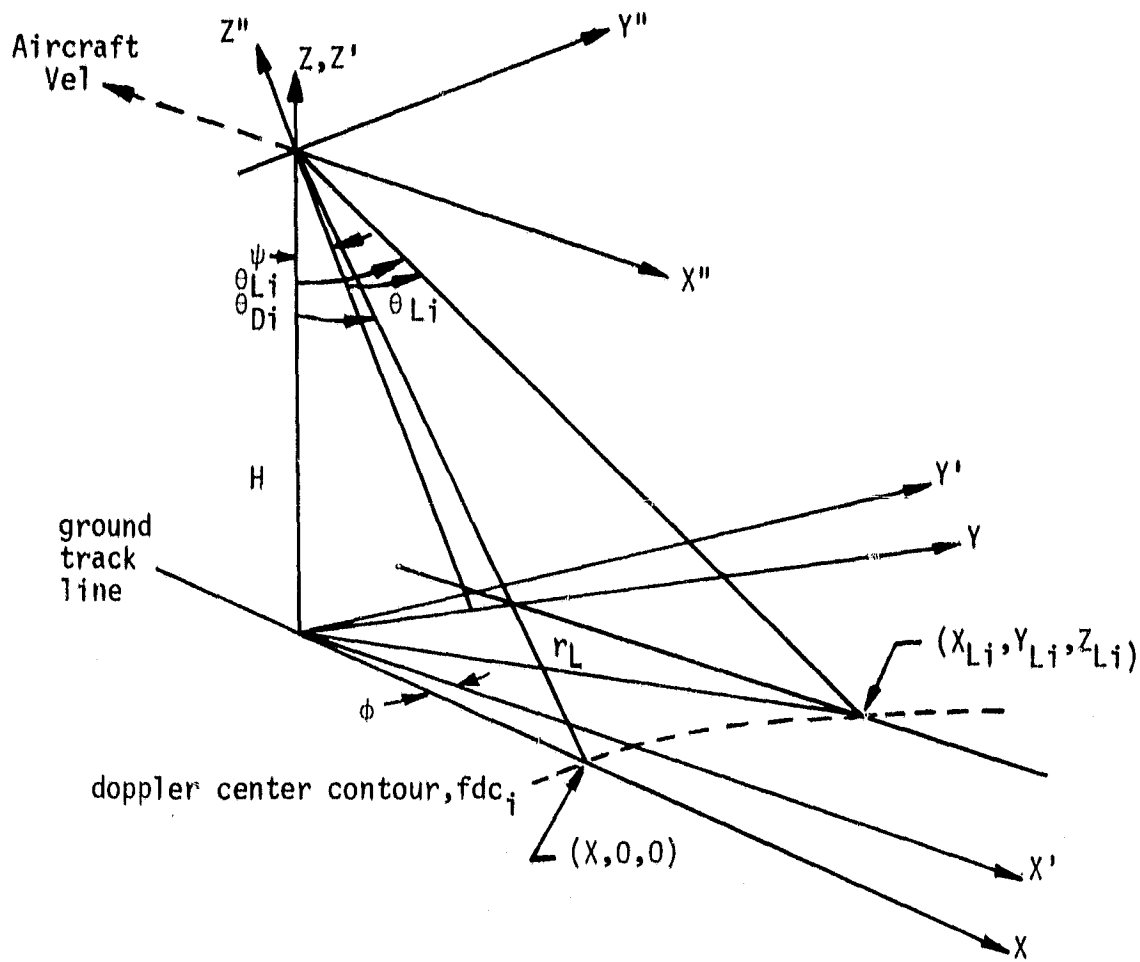


FIGURE 6.3 System Geometry.

Detail justification for the above expression is available in References [7] and [8]. The software encoding of the above expression requires the upper and lower Doppler frequencies that define the cell boundary; therefore, they are evaluated before area, A. The upper and lower Doppler frequency defining each cell require in turn, each cell center coordinate, Doppler angles and bandwidth.

The cell center coordinates are calculated using the two coordinate transformation vectors

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{pmatrix} \begin{pmatrix} X'' \\ Y'' \\ Z'' \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ H \end{pmatrix} \quad (6.6)$$

and

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} \quad (6.7)$$

where ψ = aircraft roll angle and ϕ = aircraft drift angle.

From the above, the viewing point coordinates in Figure 6.4 are

$$X_j = (-H/\cos\psi)\tan\theta'_{Lj} \cos\phi + H\tan\psi\sin\phi \quad (6.8)$$

and

$$Y_j = (-H/\cos\psi)\tan\theta'_{Lj} \sin\phi - H\tan\psi\cos\phi \quad (6.9)$$

where $\tan\theta'_{Lj} = -(\tan^2\theta_{Lj} - \tan^2\psi)^{1/2}\cos\psi$. Software checks are provided to guard against negative frequency solution where $\theta_{Lj} < \psi$; i.e., θ_{Lj} is always set $\geq \psi$. Then each Doppler angle is given by $\theta_{Dj} = \text{TAN}^{-1}(X_j^2/(H^2+Y_j^2))^{1/2}$.

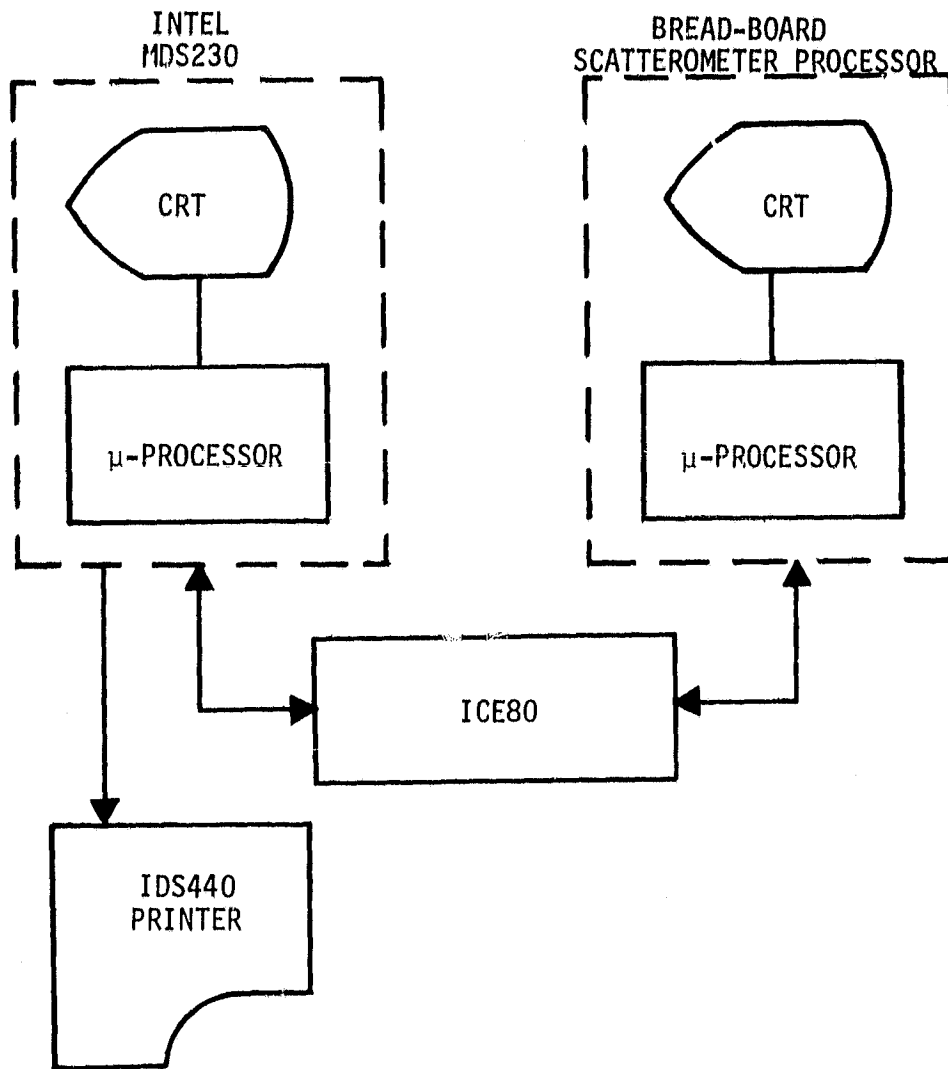


FIGURE 6.4 Module Test Set-Up.

The desired bandwidth is calculated using the expression

$$B_i = (2VL_i \cos^3 \theta_{D_i}) / \lambda H \quad (6.10)$$

where V = aircraft velocity (ground speed)
 L_i = cell length (along ground track vector)
 θ_{D_i} = doppler angle to cell center
 λ = wavelength, and
 H = altitude of aircraft.

Actual bandwidth is then

$$BW_i = \Delta f [B_i / \Delta f + 0.5]_I \quad (6.11)$$

where Δf is the spectral line width of the filters and the nearest integer value is represented by $[B_i / \Delta f + 0.5]_I$.

Finally, actual bandwidth is used to evaluate upper and lower pointers used to derive power from the filter bank.

6.2.3.2 Calculating σ^0

After all σ^0 values have been evaluated and after the CZT integration has been completed, the calculation of the power ratio values, P_{R_i} / P_T may be calculated to complete the solution for each σ^0 . This is done in the SIGMA2 module for each of the eight viewing angles.

First, the results of the CZT band integration are transferred to the memory as 512 integers, 32-bits each. Using the pointers derived in the SIGMA1 module, power ratio values, P_{R_i} / P_T , are calculated for each viewing angle by summing appropriate sets of the 32-bit integers. Next, each σ_{0_i} is calculated using the expression in

Equation (6.2). Finally, the inner execution loop is completed by buffering all the output data quantities and writing the appropriate buffer line to the Bi-Phase-L output port.

The output data frame produced on the Bi-Phase-L output port for each pass through the processing loop is shown in Appendix C. The first 54 bytes (108 4-bit BCD characters) are copies of the aircraft NERDAS data as received by the system. These 108 characters contain all the time, altitude, location and mission data. Characters 109-172 contain the processed viewing angles and scattering coefficients for each of the eight aft angles. Characters 173-176 and 177-180 contain the calibration and noise power respectively. Flag bits and alarm words are grouped into the characters 181-187. Characters 188-256 are fill data, each containing a hexadecimal 'D', binary 12.

6.2.3.3 'RUN' Termination

The inner-most execution loop is terminated only by 1) reset of the micro-processor, 2) completing the specified number of seconds of data, or 3) operator input of the 'ESC' character on the keyboard. The first type of termination is a hardware reset and provides no logical data set close-out functions. The latter two terminations both cause an orderly close-out of the output buffer and the data file being transferred to the Bi-Phase-L output port.

6.2.4 Testing and Evaluation

Testing was performed in two stages. First, as each executable module or subroutine was completed, independent verification was performed. In general, this was accomplished by preparing special

purpose "drivers" which supplied the parameter and/or buffer requirements of the module being tested. This stage constituted the bulk of the project schedule. Secondly, an "all-up" systems test was performed using all modules and sub-routines. This second stage took approximately 1/6th of the total schedule.

Module testing was initially accomplished using the hardware set-up illustrated in Figure 6.4. The ICE80 (in-circuit emulator) and its associated software package in the INTEL MDS230 allowed the software being evaluated to be run at almost real-time while residing in RAM space provided as part of the engineering breadboard system hardware, or within RAM space in the MDS230. Both modes were used, where the selection depended on the particular module being tested.

Over-all system testing was accomplished using the set-up illustrated in Figure 6.5. Aircraft and radar data inputs were provided by the Bi-Phase-L input from the 14-Track AMPEX Tape Unit. Bi-Phase-L output was captured by the TI980, partially decoded and recorded on the 9-track TI979A Tape Unit. Final decoding of the Bi-Phase-L output data was done as a post-run task and the results printed on the printer for analysis. During early May 1980, several demonstration runs were made using the system shown in Figure 6.5. These runs all demonstrated the capability of the system to process C-Band data at a rate of one data frame per 1.5 seconds. The output data quality was compared to data generated by NASA for the same input data set. The results of the over-all system tests showed that the micro-processor based system can produce comparable data ($\pm 2.0\text{db}$) to that produced by much larger and more expensive systems.

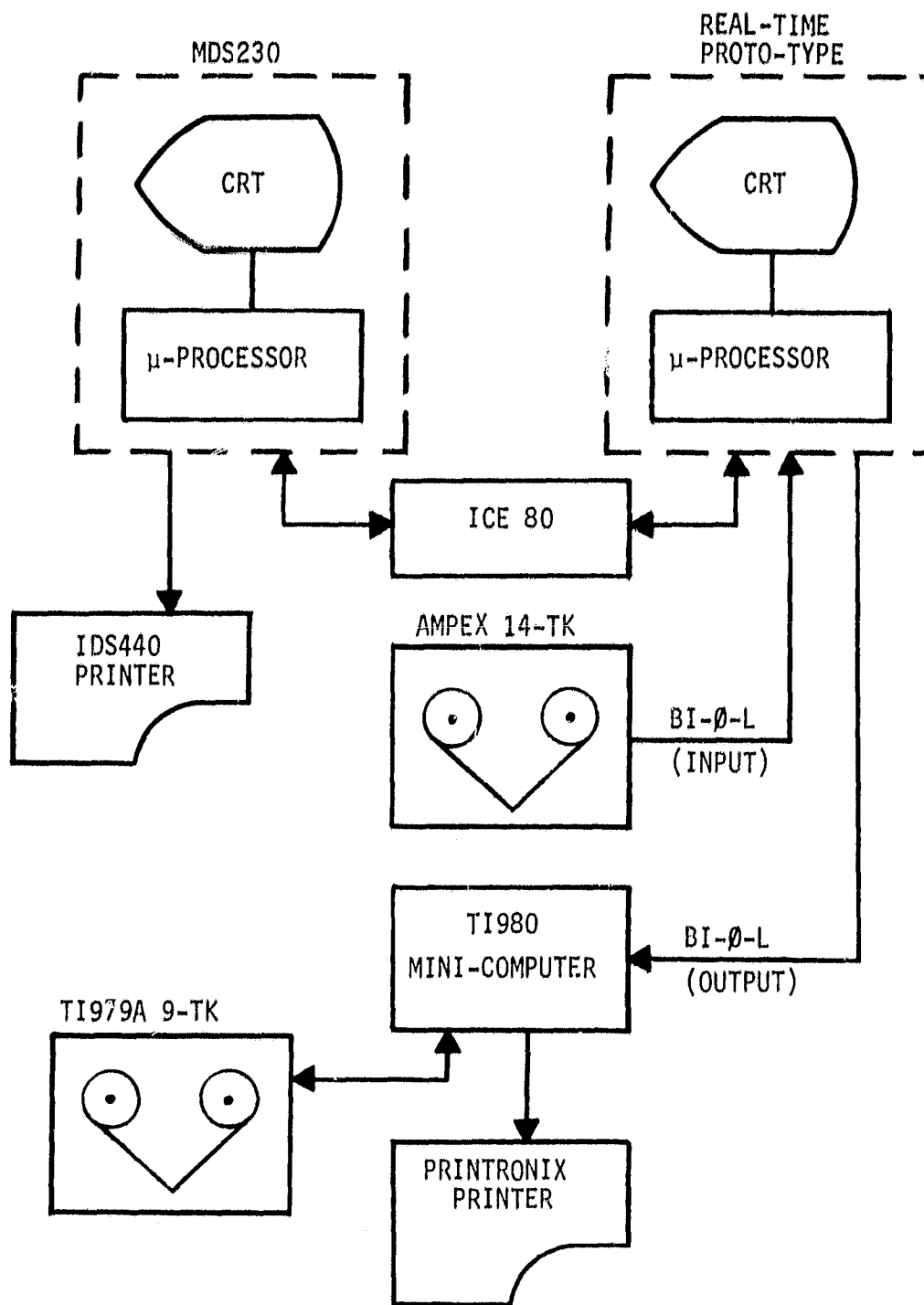


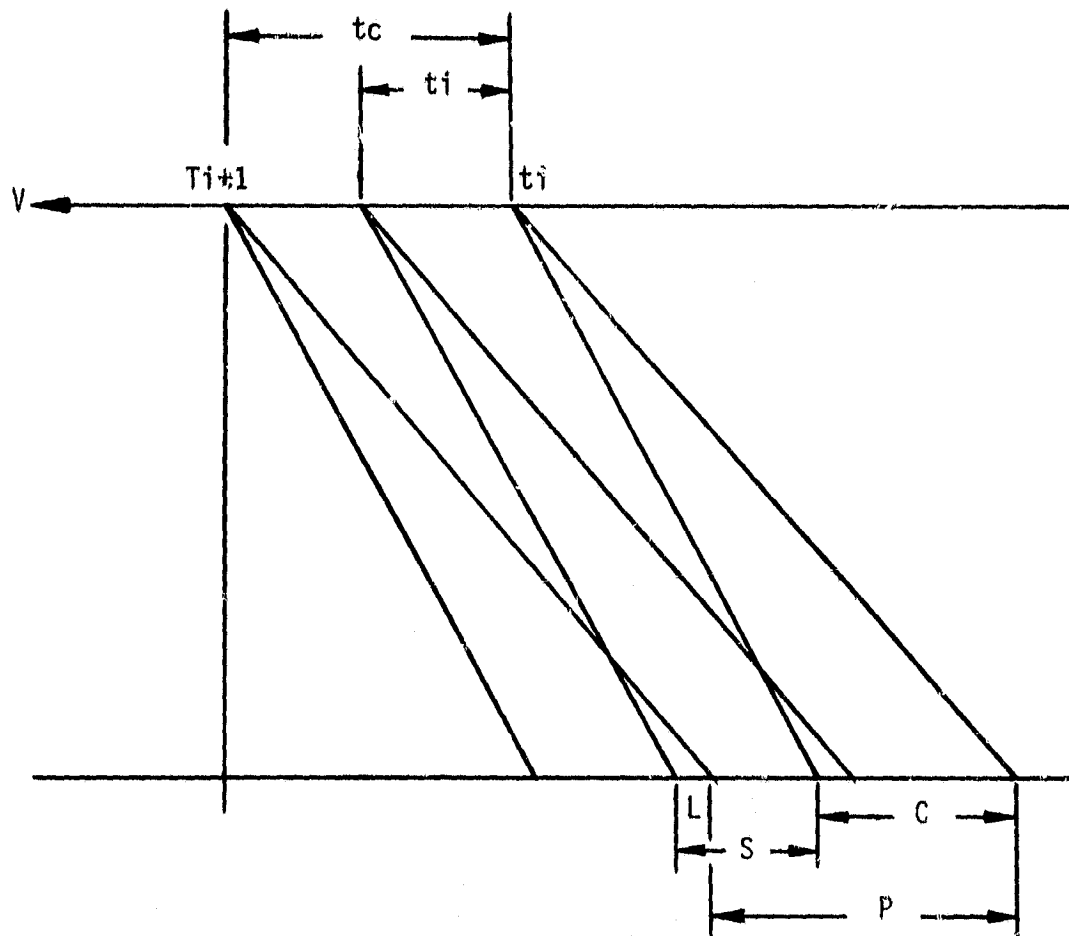
FIGURE 6.5 System Test Set-Up.

Subsequent to the above tests, two tests were run to find out approximately how much time was required to execute each major portion of the software. First, the software inner loop was modified to delete area, range, roll-off, and gain calculations. Those deletions reduced cycle time from approximately 1.5 seconds to 1.05 seconds. Next, the loop was further modified to delete the power-ratio evaluations. This change further reduced the execution loop time to approximately 0.55 seconds. The effect of varying the cycle time, t_c , is only on the effective repetition rate of the viewing cells occurring within the field of view of the radar. This repetition rate also varies with viewing angle, producing greater overlap between successive cells for greater viewing angles. Figure 6.6 shows the geometry of the effect. Using the following expression for cell overlap,

$$L = C + vt_j - vt_c \quad (6.12)$$

where t_j = integration time
 t_c = cycle time
 v = velocity of the aircraft,

the parameter plot in Figure 6.7 can be calculated. Note that a cycle time of very near 0.5 seconds is required to always have positive overlap. Negative overlap is produced when the successive ground cells for a given viewing angle are spaced apart. From the parameter plot it can also be shown that for a cycle time of 0.75 seconds cell overlaps will all be greater than -50%, giving a ground coverage of more than 66%. The above timing tests indicate that if closer cell spacing is required, the present system could be used to produce partially reduced scattering coefficients; i.e., read NERDAS, calculate



- L = overlap between successive ground cells
- S + C = effective ground cell length
- C = instantaneous cell size
- S = Vt_i , smear length
- P = Vt_c , cycle length; i.e., start of integration for cell i to start of integration for cell i+1

FIGURE 6.6 Cell Overlap Geometry.

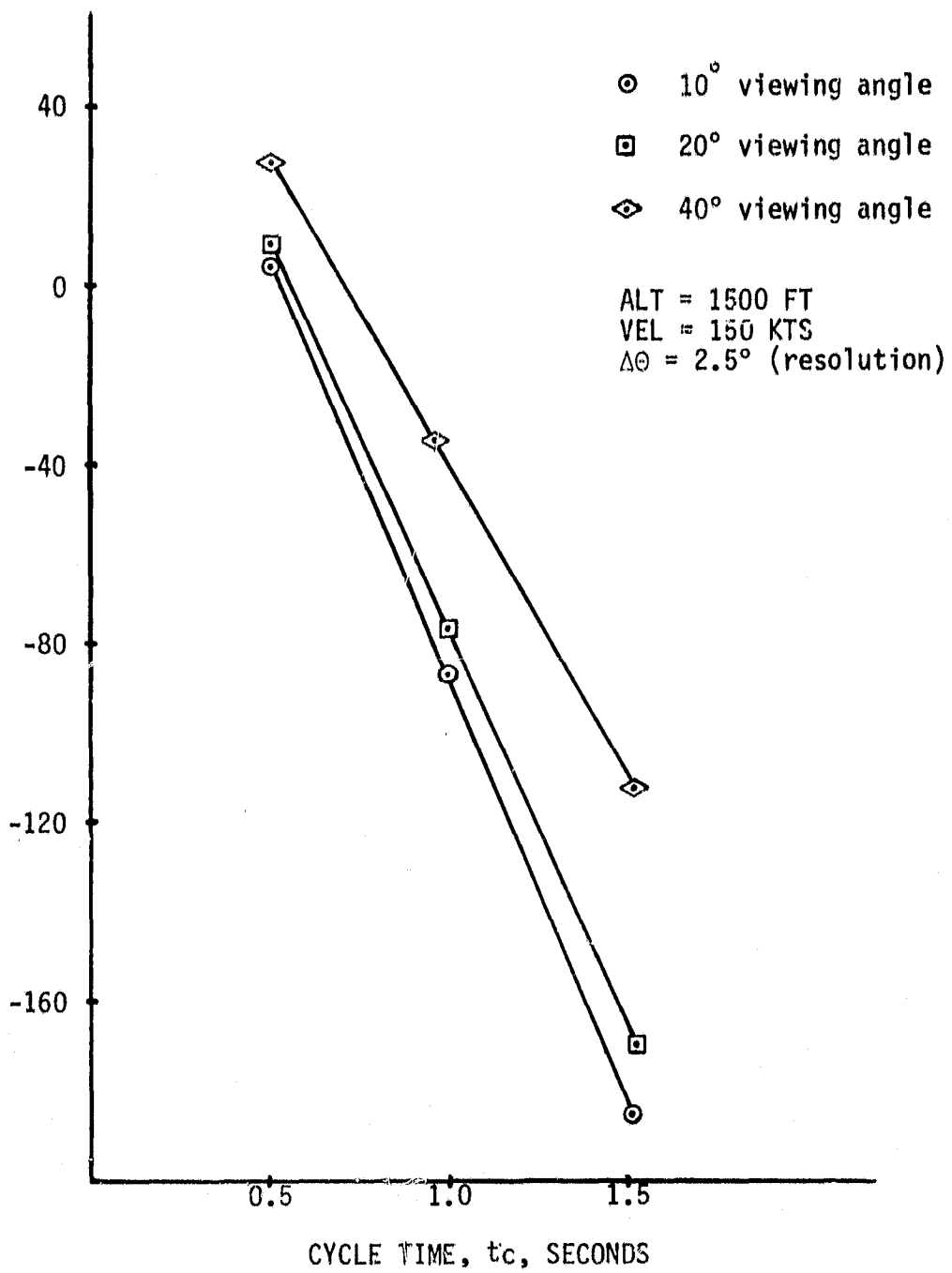


FIGURE 6.7 Cell Overlap vs Cycle-Time

integration times, set and integrate using the CZT board, recover the frequency domain data, place the time correlated power spectral density data on the output port. Also, such a system could operate with a cycle time of approximately 0.6 to 0.7 seconds of which approximately 0.3 seconds is integration time. A small amount of post processing would then be required to yield complete sigma-zero estimates for each ground cell.

Another and perhaps more flexible, alternative to full processing (full or partial) would be to use fewer viewing angles, say 10, 20, 30, and 45 degrees, or a set selectable by the operator. Approximately half the overhead and buffering tasks would be saved. Such a system could likely run at very near 1.0 second cycle time while producing fully reduced sigma-zero data. Using selectable angles and multiple passes, analysis to any depth desired could be made.

6.2.5 Memory Requirements

As illustrated in Figure 6.8, a large amount of ROM (A000H) will be required to store the data tables and code. A large portion of this space (4000H) is required for inter-communications formatting; i.e., the I/O formatting of communications between the operator and the system. Figure 6.5 also shows the suggested board arrangement to accomplish the appropriate program alignment within the ROM/RAM structure.

6.3 Interrupt Structure

The inner-loop (SIGMA1, SIGMA2) execution uses four interrupt lines. The associated software interrupt jump table is located at

0000	Restart and Interrupt jump table	4K ROM SBC 80/20-4
0120		4K ROM SBC 116
		16K ROM
0200		SBC 416, #1
	Code and Data	
		16K ROM
71AA		SBC 416, #2
9FFF		
A000		4K Unused
B000		4K RAM SBC 80/20-4
	Variables	
C000		
	and	16K RAM
D000		
	Stack	SBC 116
E000		
	Space	
F000		

FIGURE 6.8 Program Alignment within ROM and RAM.

4000H. The table is loaded under the module name LDJMPS. A level 0 interrupt is generated when the CZT-board completes a timed integration. The subroutine CZTINT handles these interrupts. It causes a flag, IEOC, to be set in the main program, Figure 6.9. The main program, after completing its SIGMA1 tasks, is held in a wait-loop until the IEOC flag is set. After the flag is set, control passes from the SIGMA1 module to the SIGMA2 module where power ratios are calculated using data from the CZT board.

Level 1 interrupts are used for Bi-Phase-L output. A level 1 interrupt signals acknowledge/receipt by the 8255 of the outgoing byte.

Level 3 and Level 4 interrupts are used in NERDAS data inputs. The level 3 interrupt signals a byte is pending. The level 4 interrupt signals that the byte is on the data base.

6.4 Sub-routine Definitions

Most of the inner-loop operations, SIGMA1 and SIGMA2 modules, are accomplished using the subroutine calls defined below.

AFIX(X,I): converts floating point number, X, to integer and places result in I.

AFLOAT(IN,FP): converts integer value, IN, to floating point format and place result in FP.

AINDIX(IN,I80,TEN,AGL,RAD): calculates an index value, IX, using the expression:

$$IX = IFIX(AGL * RAD - TEN) + I80$$

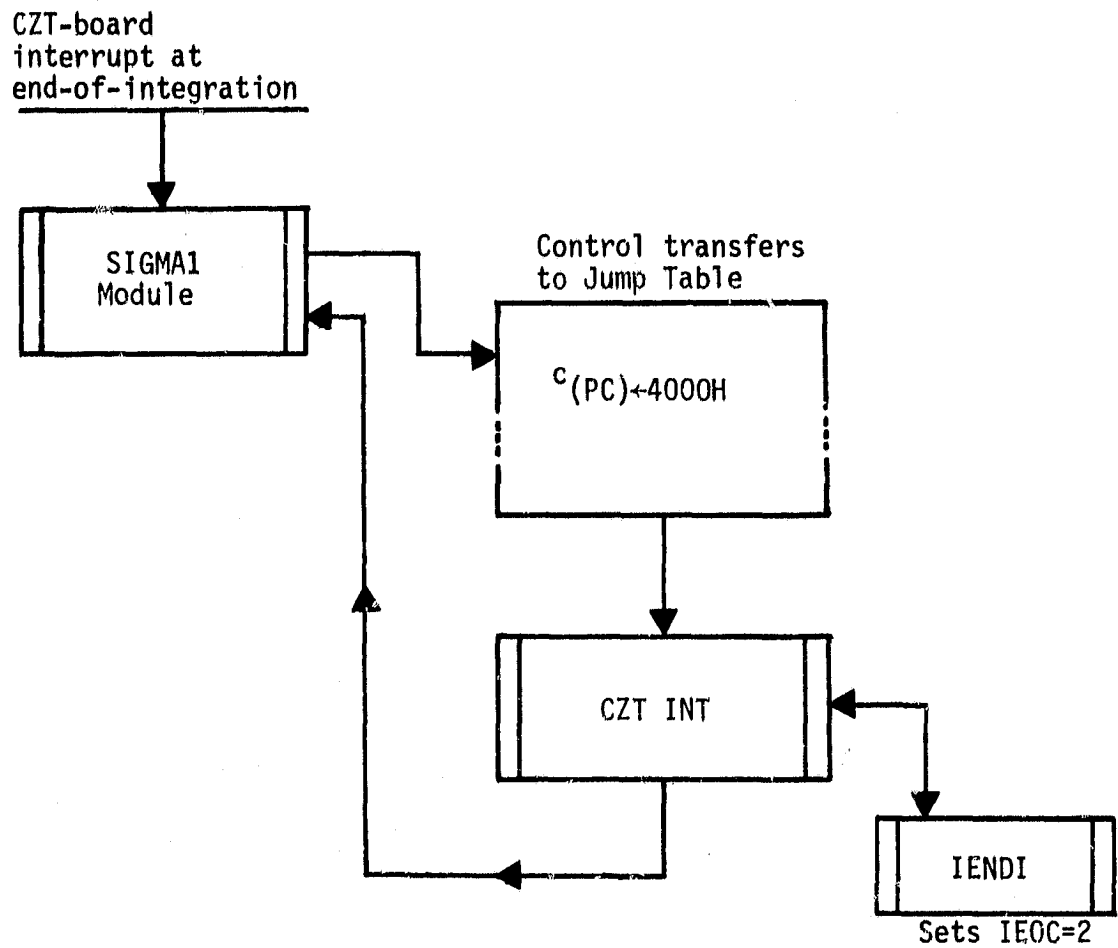


FIGURE 6.9 CZT Interrupt Structure.

where AGL = angle in radians
 RAD = constant, 57.3
 TEN = constant, 10.0
 180 = constant, 80

Index value is used as a pointer into the L-Band antenna table.

ALTFP(ALT, CALT, ITN, IHD, ITH, IUN): evaluates altitude of the aircraft in meters, using the following expression

$$ALT = \text{FLOAT}(IUN + 10*(ITN + 10*(IHD + 10*ITH)))*CALT$$

where IUN = NERDAS UNITS digit
 ITN = NERDAS TENS digit
 IHD = NERDAS HUNDREDS digit
 ITH = NERDAS THOUSANDS digit
 CALT = constant, 0.3048

Routine is called by DECODA module when decoding NERDAS data.

AMDADD(R,A1,A2): evaluates the result of addition of two floating point numbers, places result in R:

$$R = A1 + A2$$

AMDIV(R,A1,A2): evaluates the result of dividing two floating point numbers:

$$R = A1/A2$$

AMDGN(DGN,TWO,DG,TEN,I1,I2): calculates interpolation value DGN according to the expression:

$$DGN = (\text{FLOAT}(I1 - I2)/TEN)*DG/TWO$$

where I1 = upper table index
 I2 = lower table index
 TEN = 10.0, constant
 DG = ABS(AGL*57.3) - FLOAT(IFIX((AGL*57.3 + 0.5)/2.))*2
 where AGL = viewing angle through the antenna
 TWO = 2.0, constant

AMDGSQ(GSQ,DGN,IGT,TEN): evaluates final gain value by adding interpolation value DGN to table value using the expression:

$$GSQ = \text{FLOAT}(IGT)/TEN + DGN$$

where IGT = table value of gain
 TEN = constant, 10
 DGN = interpolation adjustment

AMDMUL(R,A1,A2): computes product of floating point numbers
A1 and A2 according to the expression:

$$R = A1 * A2$$

AMDSUB(R,A1,A2): evaluates difference of two floating point
numbers:

$$R = A1 - A2$$

CBNDW(BNDW,NFEL,DELFL): calculates actual filter bandwidth
according to the expression:

$$BNDW = \text{FLOAT}(NFEL, DELFL) * DELFL$$

where NFEL = number of filter elements in the band
DELFL = spectral line width of the filter

CCELL(CELL,DIFF,SUM,ALT): evaluates ground cell length according
to the expression:

$$CELL = ALT * (\text{TAN}(SUM) - \text{TAN}(DIFF))$$

where ALT = aircraft altitude, meters
SUM = viewing angle plus one-half the resolution
angle
DIFF = viewing angle minus one-half the resolution
angle

CEL CNT(ICNT,VEL,TC,ALT,THET8): computes the number of cells in
view (required buffer size) for a given altitude and
viewing angle range.

$$ICNT = \text{IFIX}(ALT * \text{TAN}(THET8)) / (VEL * TC) + 0.5$$

where ALT = aircraft altitude, meters
VEL = velocity of aircraft (ground speed), meters/sec
TC = processor cycle time, seconds

CINDX(IX,I1,TWO,HALF,D): calculates the index value IX using the
the expression:

$$IX = \text{IFIX}((D + 0.5) / TWO) + I1$$

where D = ABS(AGL*57.3)
AGL = viewing angle through antenna
TWO = constant, 2
I1 = constant, 1
HALF = constant, 0.5

Routine is called by GAIN when evaluating C-Band antenna gain.

CLNPT(IBPT,LT,PC,PN,XT,YI,ANGTD,ALT): calculates buffer loading pointers for viewing angle according to the procedure:

ANGTD = ASIN(ANGTD)
XT = (ALT*ALT + YI*YI)*TAN(ANGTD)**2
IBPT = LT - IFIX(PC*(TAN(ANGTD) - PN))

where

ANGTD = doppler angle
XT = X-AXIS INTERCEPT OF DOPPLER CONTOUR
IBPT = number of cells aft of nadir to place the σ_0 for the given viewing angle
LT = current nadir position in buffer
YI = viewing point y-coordinate
PC = (ALT/VEL)/TC
PN = (TI/PC)*0.5
TI = integration time

CNFILT(NFEL,DEL,BNDW,CELL,FDOP,ANGLD,YI,XI,XT,S): calculates number of filter elements required to represent each viewing angle band. Routine implements the following:

XT = (2.*VEL)/SLMDA
ANGLD = ATAN(SQRT((XI*XI)/(ALT*ALT + YI*YI)))
FDOP = XI*SIN(ANGLD)
BNDW = (CELL*XT/ALT)*COS(ANGLD)**3
NFEL = IFIX(BNDW/DEL + 0.5)

where

SLMD = wavelength
ANGLD = doppler angle
XI,YI = viewing point coordinates
FDOP = doppler center frequency

CRG4(RG4,FRTY,ALT,ANGTL): evaluates the range of the cell center from the antenna. Value, RG4, is returned in db according to the expression:

RG4 = 40.*ALOG10(ALT/COS(ANGTL))

where

FRTY = constant, 40.0
ALT = altitude, meters
ANGTL = viewing angle, radians

CZT(N): set and start integration of N sub-records

CZTR: start DMA transfer of 512 filter elements from the CZT board

DAREA(AL,TEN,TRM1,TRM2,TRM3): computes the area of a ground cell according to the expression:

AL = 10.*ALOG10(TRM1*(TRM2 - TRM3))

where

AL = area in db
TRM1 = width of ground cell
(TRM2 - TRM3) = length of ground cell

DECODA(IOV,NERZ,T1,PARM): Decodes NERDAS frame. Uses subroutines as follows:

ALTFP: decodes altitude up to 9999. feet
DRPFP: decodes drift, roll, pitch up to 99.9 degrees
VELFP: decodes velocity up to 999. knots
MINHR: decodes minutes, hours
TSECS: decodes seconds

DRPFP(DRP,CRAD,ISGN,ITN,IHD,IUN): converts NERDAS digits to drift, roll, or pitch in radians:

$$DRP = \text{FLOAT}((IUN + 10*(ITN + 10*IHD))*ISGN)/CRAD$$

where

CRAD = 57.3
IUN = units digit of NERDAS
ITN = TENS digit of NERDAS
IHD = HUNDREDS digit of NERDAS
ISGN = sign digit of NERDAS

DVERIF(ICNT,IBUFF): edits a buffer, IBUFF, containing ICNT bytes for ASCII decimal characters. Any character, other than decimal or leading unary, is converted to a blank.

DWAIT: a linkage routine for FORTRAN error recovery. Assumes control of program counter until a machine reset.

FDLEV(IFDL,NFEL,NEV,FDOP,DELF): evaluates starting pointer for summing spectral power when the number of filter elements in the filter band is even and uses the expression:

$$IFDL = \text{IFIX}(FDOP/DELF) + NEV - NFEL/Z$$

where

FDOP = doppler center frequency
DELF = spectral line width
NEV = constant, 3
NFEL = number of filter elements in the filter

FDLOD(IFDL,NFEL,NOD,FDOP,DELF): evaluates starting pointer for summing spectral power when the number of filter elements in the filter band is odd. It uses the expression:

$$IFDL = \text{IFIX}(FDOP/DELF + 0.5) + NOD - (NFEL - 1)/2$$

where NOD = constant, 2.

GAIN(ANG,PCH,IGTBL,IGTBC,NPZN,IBND,GSQ): calculates antenna gain, GSQ, for either L-Band or C-Band using a table look-up and interpolation procedure. Input parameters are:

ANG = antenna view angle in radians
PCH = aircraft pitch, in radians
IGTBL = L-Band gain table
IGTBC = C-Band gain table
NPZN = polarization identifier, W=1, VH=2, HH=3, HV=4
IBND = band identifier, L=1, C=2

GAML(Z,C1,C4,C5,CZ,F,C3): evaluates the value of Z, roll-off, in db for L-Band using the expression:

$$Z = -(C1 + F*(CZ + C3*F) + (C4 + C5/F)/F)$$

where $F = \text{frequency}/10.$
 $C1, \dots, C5 = \text{constants}$

GAMC(Z,C1,CZ,C3,C4,C5,C6,C7,F): evaluates the value of Z, roll-off, in db for C-Band using the expression:

$$Z = C1 + F*(CZ + F*(C3 + F*(C4 + C5*F))) + (C6 + C7/F)/F$$

where $F = \text{frequency}/10.$
 $C1, \dots, C7 = \text{constants}$

GAMMA(FRQ,IB,NPZ,Z): evaluates roll-off, Z, using subroutine calls to GAML or GAMC depending on band identifier, IB. Computes $F = \text{FRQ}/10.$ before calling subroutine. NPZ is the polarization identifier.

GETVLU(CIOBUF,IOBUFF,N10,N20,IERR,XNUM): acquires from console the floating point number, XNUM. CIOBUF and IOBUFF are the input buffers, N10 = constant, 10, and N20 = constant, 20.

GXI(XI,XT,ALT,RL,DR): computes X-coordinate, XI, using the expression:

$$IX = XT*\text{COS}(DR) + \text{ALT}*\text{TAN}(RL)*\text{SIN}(DR)$$

where $XT = \text{SQRT}(\text{ALT}*\text{TAN}(\text{ANGL}))**Z - (\text{ALT}*\text{TAN}(RL)**Z),$
calculated in subroutine GXT, below.
DR = drift in radians
ALT = altitude, meters
RL = roll

GXT(XT,RL,ANGL,ALT): evaluates XT, used in the above subroutine, using the expression:

$$XT = \text{SQRT}(\text{ALT} * \text{TAN}(\text{ANGL})) ** 2 - (\text{ALT} * \text{TAN}(\text{RL})) ** 2$$

where ANGL = viewing angle.

GYI(YI,XT,ALT,RL,DR): calculates the Y-coordinate of the viewing point, YI, using the expression:

$$YI = XT * \text{SIN}(\text{DR}) - \text{ALT} * \text{TAN}(\text{RL}) * \text{COS}(\text{DR})$$

where XT is as calculated by the subroutine GXT above.

IBELL: sends bell character to CRT. No calling parameters.

IBFILL(L,IBUFF): fills 128 byte buffer lines with 99H, L = line count and IBUFF = buffer starting address.

IBIPHL(IBUFF): move 128 bytes beginning at IBUFF to the 8255 serial port.

ICRLF: routine to output a carriage return and line-feed to the CRT.

IKEYI: sets flag for keyboard 'ESC' input. Called by KEYCK.

INIPIO: initializes parallel I/O port on the SBC 80/20 board.

INICZT(IPSD): initializes CZT board with DMA address, IPSD.

INI259: initializes the 8259 interrupt controller for the vested mode at 4 byte intervals. Call table set at 4000H. Leaves interrupts enabled but all masked.

INTGT(ICNT,TI,DELF): calculates the number of subrecords over which to integrate. Uses expression:

$$ICNT = \text{IFIX}(\text{TI} * \text{DELF} + 0.5)$$

where

TI = integration time, seconds
DELF = spectral line width of CZT

IUSART: initializes the USART for CRT/keyboard I/O.

I32SUM(PX,NV,IV): calculates the sum of a set of 'IV' 32-bit integers beginning at NV(1), and extending to NV(IV). The result in db is placed in PX, and uses the expression:

$$PX = 10. * \text{ALOG10}(\text{FLOAT}(\text{NV}(1) + \text{NV}(2) + \dots + \text{NV}(\text{IV})))$$

KBPHAL(IOUT): control routine for handing Bi-Phase-L output. Sends out a 128 byte line from IOUT then backfills

the line with 99H. Calls IBIPHL to write the line to the serial interface.

KEYCHK: senses the keyboard for an escape character. If no character is pending or if character is not an 'ESC', no action is taken. If 'ESC' is found, a branch is taken to subroutine IKEYI to set a main program flag, KESC.

KEYIN(N,IB): key board input routine. Reads keyboard to place N characters in IB, left justified. As each character is read, it is echoed to the CRT; 'ESC' is echoed as a '\$'.

MSKSET(MSK): sets 8259 interrupt mask to the value MSK.

MINHR(IMH,NU,NT): evaluates minutes or hours from NERDAS digits using the expression:

$$IMH = NT*10 + NU$$

where NT = tens digit
 NU = units digit

MFLAG(IC,IOF,IOUT,IB,NZ): moves flag data words to output buffer line.

IC = column count in buffer
IOF = flag word buffer
IOUT = output buffer
IB = band identifier
NZ = polarization identifier

Seven flag words are placed in the output line then the line is finished by filling it with DDH.

MFPNUM(FPNBR,ICOL,IBPT,IOUT,IBCD,NK): moves up to 8 floating point numbers into the output buffer after they have been converted to BCD (binary coded decimal). NK = number values to convert and move. FPNBR = buffer of floating point numbers. The numbers, starting at FPNBR, are converted one at a time into a BCD buffer IBCD, then moved to a location in the output buffer. ICOL and IB are pointers into the output buffer, IOUT.

NERD(ICNT): reads Bi-Phase-L for ICNT bytes of NERDAS data.

OUTPUT(N,IBUFF): outputs a character string of N bytes beginning at IBUFF. String is sent to CRT or logging device at port CCH.

PCPN(PNTI,PC,TC,ALT,VEL): calculates the parameters PC and PN used in evaluating the required buffer length. Implements:

$$PC = (ALT/VEL)/PC$$
$$PN = (TI/(ALT/VEL))*0.5$$

RUNLMT(PREV,IOVER,PARM): checks aircraft data (altitude, drift, roll, pitch, velocity) to verify that it is within prescribed limits. The prescribed limits are set by monitoring a limited number of previous values. The previous values have also been monitored against a preset standard.

SIGSUM(SGMA,ZW,AL,GSQ,SYSK,RG4,FP): computes the final value of σ_0 . Uses the expression:

$$SGMA = FP + RG4 + SYSK - GSQ - AL - ZW$$

where all parameters are in db.

$$FP = (4\pi)^3/\lambda^2$$
$$RG4 = R^4, \text{ range parameter}$$
$$SYSK = C_L/K, \text{ system constant}$$
$$GSQ = GG(L_i), \text{ antenna gain}$$
$$AL = \text{area of ground cell}$$
$$ZW = \text{roll-off function value}$$

TIMEFP(TI,ITS,IM,IH): computes floating point value of time in seconds by:

$$TI = \text{FLOAT}(IH*60 + IM)*60. + \text{FLOAT}(ITS)/10.$$

where

$$IH = \text{NERDAS hours}$$
$$IM = \text{NERDAS minutes}$$
$$ITS = \text{NERDAS tenths of seconds}$$

TMP12(TMP1,VEL,X12): evaluates the ground cell parameter TMP1 using the expression:

$$TMP1 = X12*(VEL**2)$$

where X12 = XK1 or XK2, as defined in Reference (2).

TRM1(T1BMW,ANG,ALT,RL): evaluates the ground cell area term T1 using the expression:

$$T1 = 2.*\text{TAN}(BMW/2.)*(ALT**2)/\text{COS}(RL)/\text{COS}(ANG)$$

where

$$BMW = \text{beamwidth}$$
$$ANG = \text{viewing angle}$$
$$RL = \text{roll}$$

TRM23(TRM,TMP1,RL,DR): calculates the ground cell area term TRM according to the expression:

$$\text{TRM} = (-\text{TAN}(\text{DR}) * \text{TAN}(\text{RL}) + \text{SQRT}(\text{TAN}(\text{RL}) ** 2 * (\text{TMP1} - 1.) \\ + \text{TMP1} * \text{COS}(\text{DR}) ** 2 - (1. - \text{TMP1} * \text{COS}(\text{DR}) ** 2))$$

where TRM = TRM2 or TRM3 of the area equation.

TECS(ITS,ITN,IHD,IUN): evaluates seconds from NERDAS date by

$$\text{ITS} = 10 * (\text{ITN} + 10 * \text{IHD}) + \text{IUN}$$

where
ITN = tens digit
IHD = hundreds digit
IUN = units digit

UNPACK(INN,INB): unpacks a frame of 58 NERDAS NIBS at INN (2 BCD values per byte) into a buffer at INB with 1 BCD value per byte. INB is 55 bytes in length. First 2 NIBS are skipped.

VALID(DL,DF,IOV,IVAL,PARM): validates NERDAS data values unless the override flag is set. If value is outside preset limits, the default value is used. Whenever default values are used an appropriate flag is set in the output buffer. Calling parameters are

DL = preset limit values (buffer of 10 values)
DF = preset default values (buffer of 5 values)
IOV = override flag buffer
IVAL = flag byte
PARM = buffer of NERDAS parameters; i.e., alt., pitch, roll, drift, velocity

VELFP(VEL,CVEL,ITN,IHD,IUN): computes velocity of the aircraft in meters per second using the expression

$$\text{VEL} = \text{FLOAT}(\text{IUN} + 10 * (\text{ITN} + 10 * \text{IHD})) * \text{CVEL}$$

where
CVEL = 0.514
IHD = NERDAS hundreds digit
ITN = NERDAS tens digit
IUN = NERDAS units digit

XK12(XK2,XK1,SLMDA,BNDW,FDOP): evaluates the two area terms XK1, XK2, using the expressions:

$$\text{XK1} = 4. / (\text{SLMDA} * (\text{FDOP} - \text{BNDW} / 2.) ** 2) \\ \text{XK2} = 4. / (\text{SLMDA} * (\text{FDOP} + \text{BNDW} / 2.) ** 2)$$

where SLMDA = wavelength, meters.
 DFOP = doppler center frequency
 BNDW = filter bandwidth

6.5.0 AMC95/6011 Arithmetic Unit Operations

All inner-loop arithmetic, except indexing, is accomplished using the hardware floating point arithmetic board. These operations are all accomplished using the following set of device calls:

AMDCMD: routine to send a command in the A-register to the AMC 95/6011.

AMDLOD: loads a floating point number at the address in register-pair BC into the AMC 95/6011.

AMDSTR: stores a floating point number in INTEL format from the AMC 95/6011 internal register. Storage address is expected to be in register pair BC.

GET: gets a floating point number in AMC 95/6011 internal register and stores it at the address in register pair DE.

GIVE: gives a floating point number in AMC format located at the address in register pair DE to the AMC 95/6011 internal register.

INTLOD: loads a 16-bit integer at address in register pair BC into the AMC 95/6011 internal register.

INTSTR: stores a 16-bit integer at the address in register pair BC. Integer is retrieved from the AMC 95/6011 internal register.

I32LOD: loads a 32-bit integer at the address in register pair BC into the AMC 95/6511 internal register.

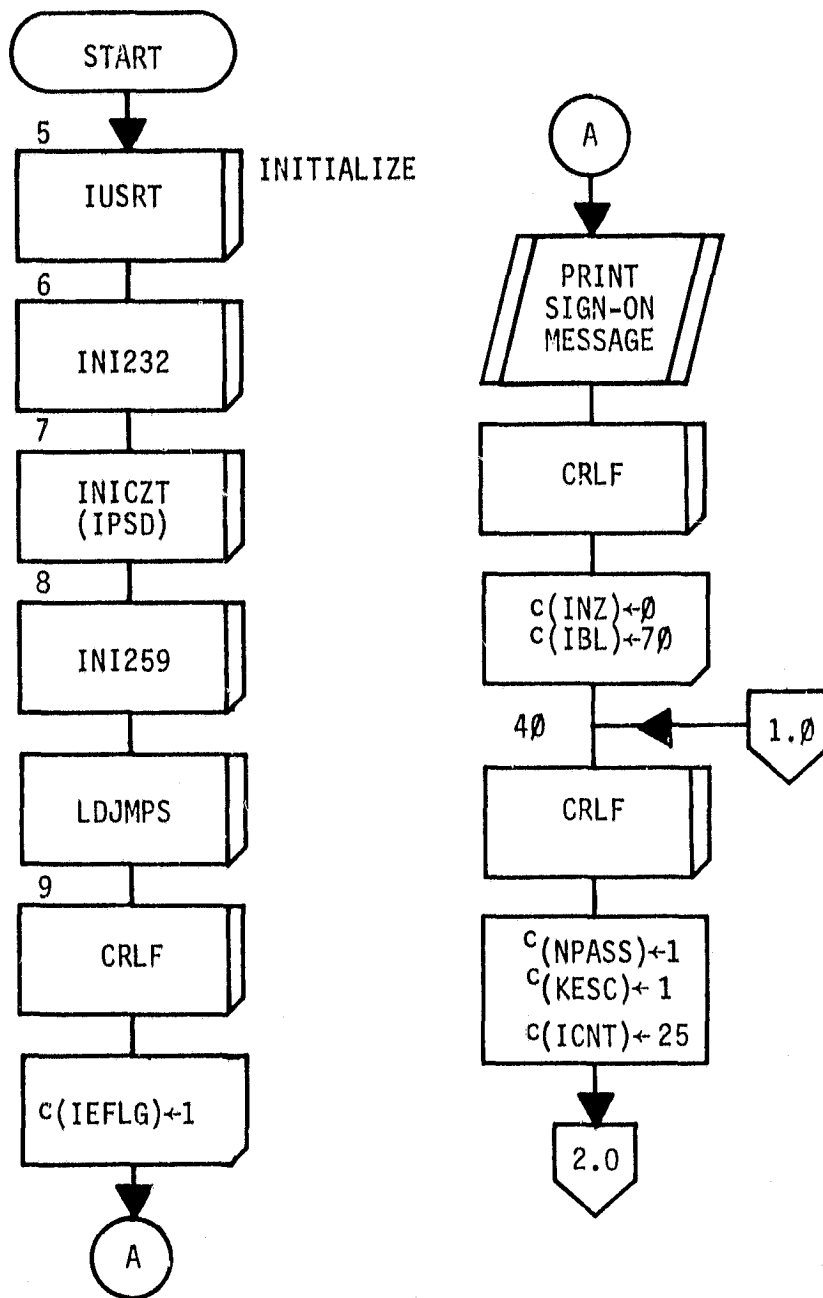
REFERENCES

1. John P. Claassen, et al., "The System and Hardware Design of Real-Time Fan Beam Scatterometer Data Processors," Final Report RSC3556, Remote Sensing Center, Texas A&M University, College Station, Texas. March 1979.
2. "Definition and Fabrication of an Air borne Scatterometer Radar Signal Processor," Technical Report RSC3182-2, Remote Sensing Center, Texas A&M University, NASA Contract NAS9-14493, December 1976.
3. Reeves, R. G., et al., Manual of Remote Sensing, American Society of Photogrammetry, 1975.
4. Claassen, J. P., "Accuracy Criterion for Estimating the Mean Squared Signal," Rsl-TM 186-1, Remote Sensing Laboratory, University of Kansas, 1970.
5. Oppenheim, A. V. and R. W. Schaffer, Digital Signal Processing, Prentice-Hall, Inc., 1975.
6. Claassen, J. P., "The Design of the RADSCAT Experiments," RSL-TR-186-2, Remote Sensing Laboratory, University of Kansas, 1971.
7. B. V. Clark and R. W. Newton, "JSME Scatterometer Data Processing," Texas A&M University, Remote Sensing Center, August 1979.
8. _____, "An Airborne Radar Scatterometer Signal Processing System," Progress Report, February - April 1975.
9. Advanced Micro Computers, "AMC 95/6011 Arithmetic Processing Unit Board User's Manual," Revision A, 1978.

APPENDIX A
System Flow Charts

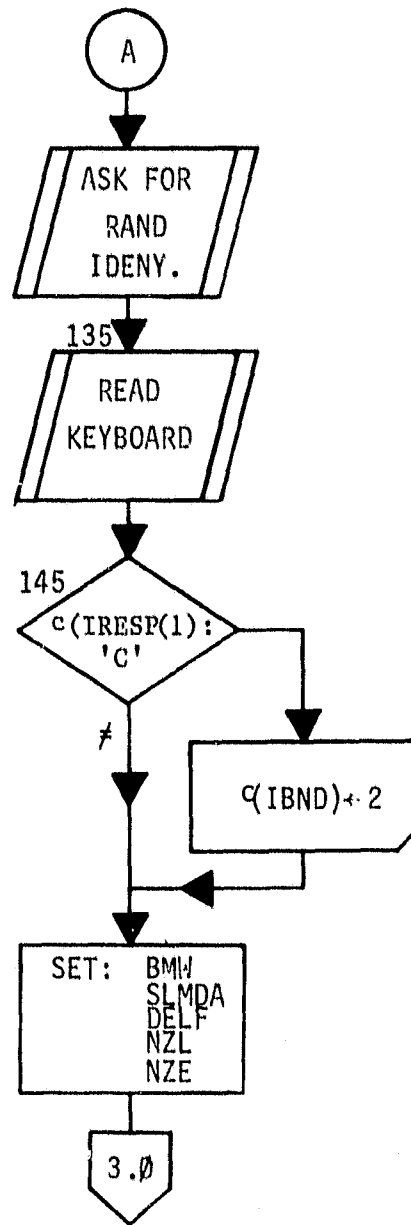
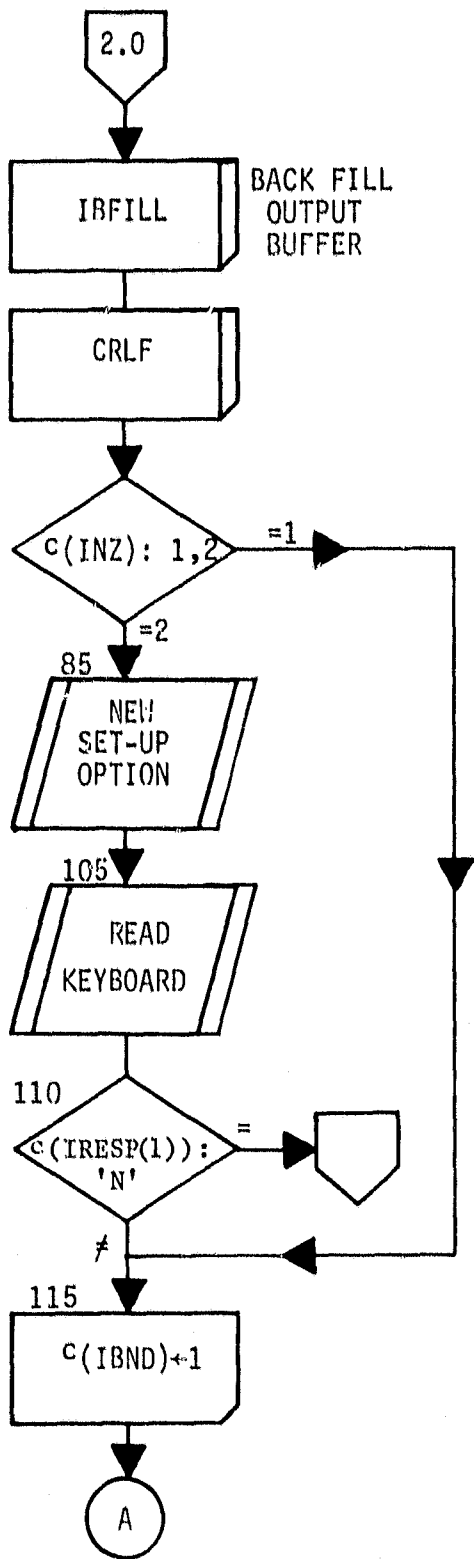
PRECEDING PAGE BLANK NOT FILMED

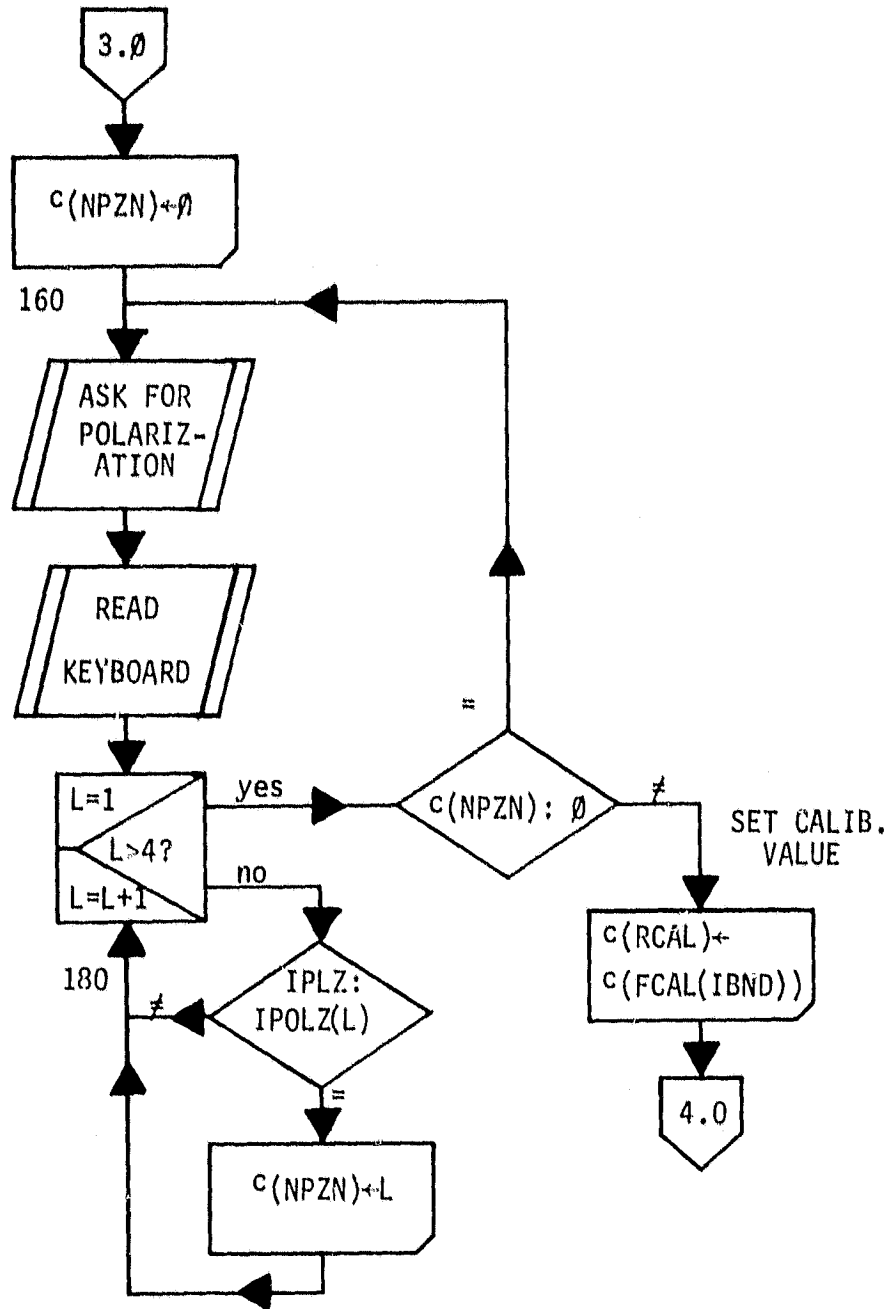
OPERATOR COMMUNICATIONS MODULE

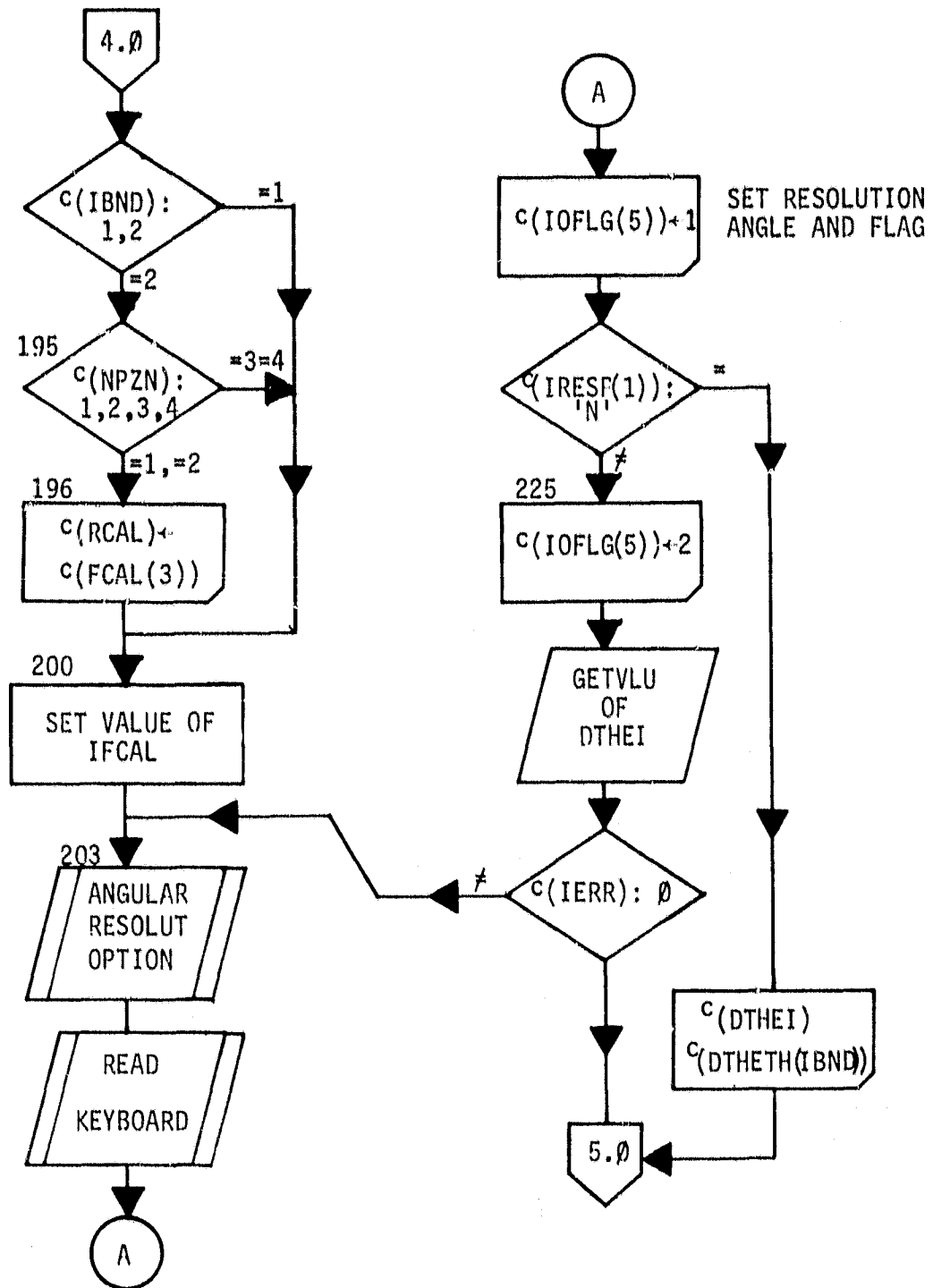


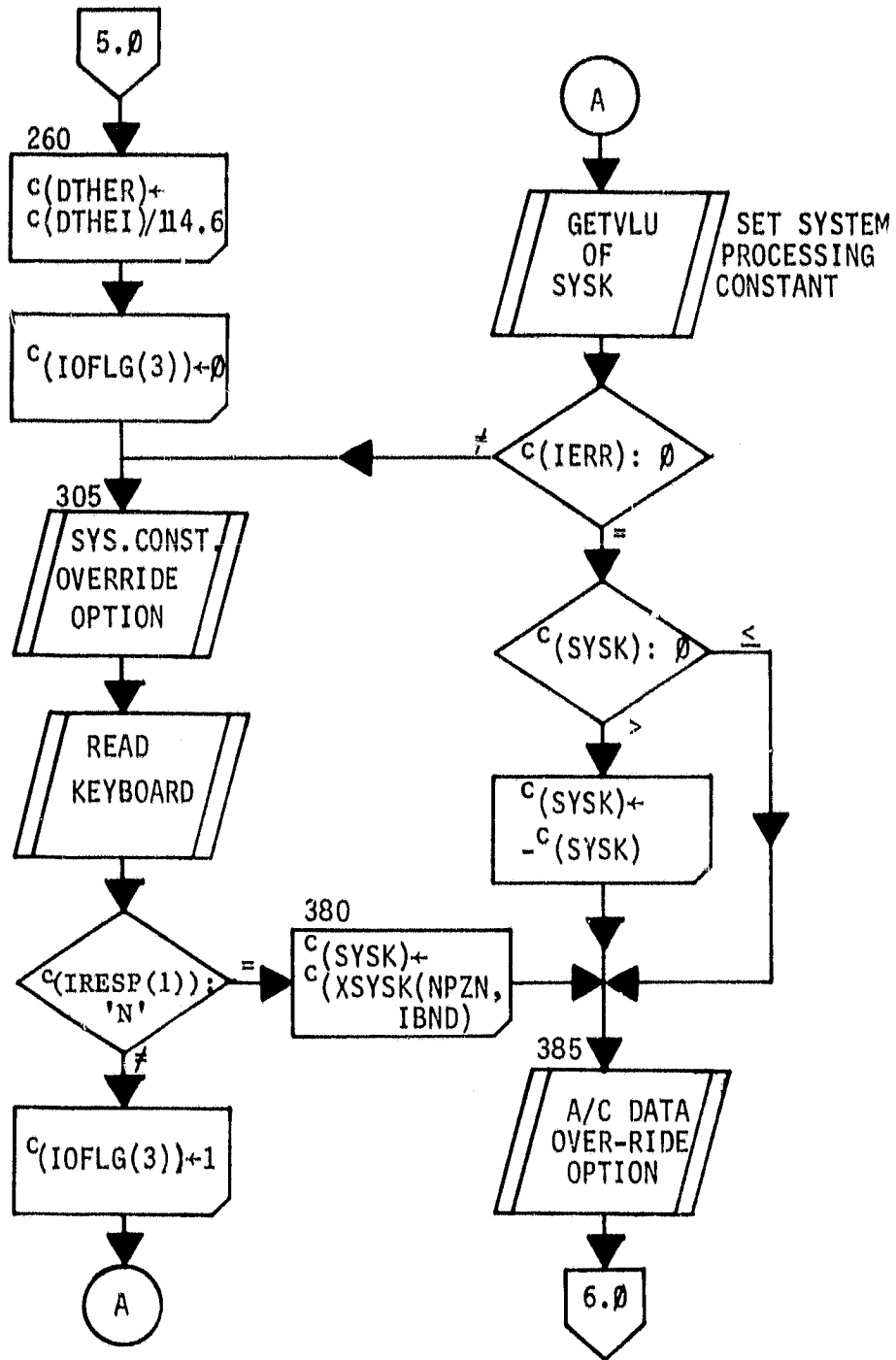
p.1

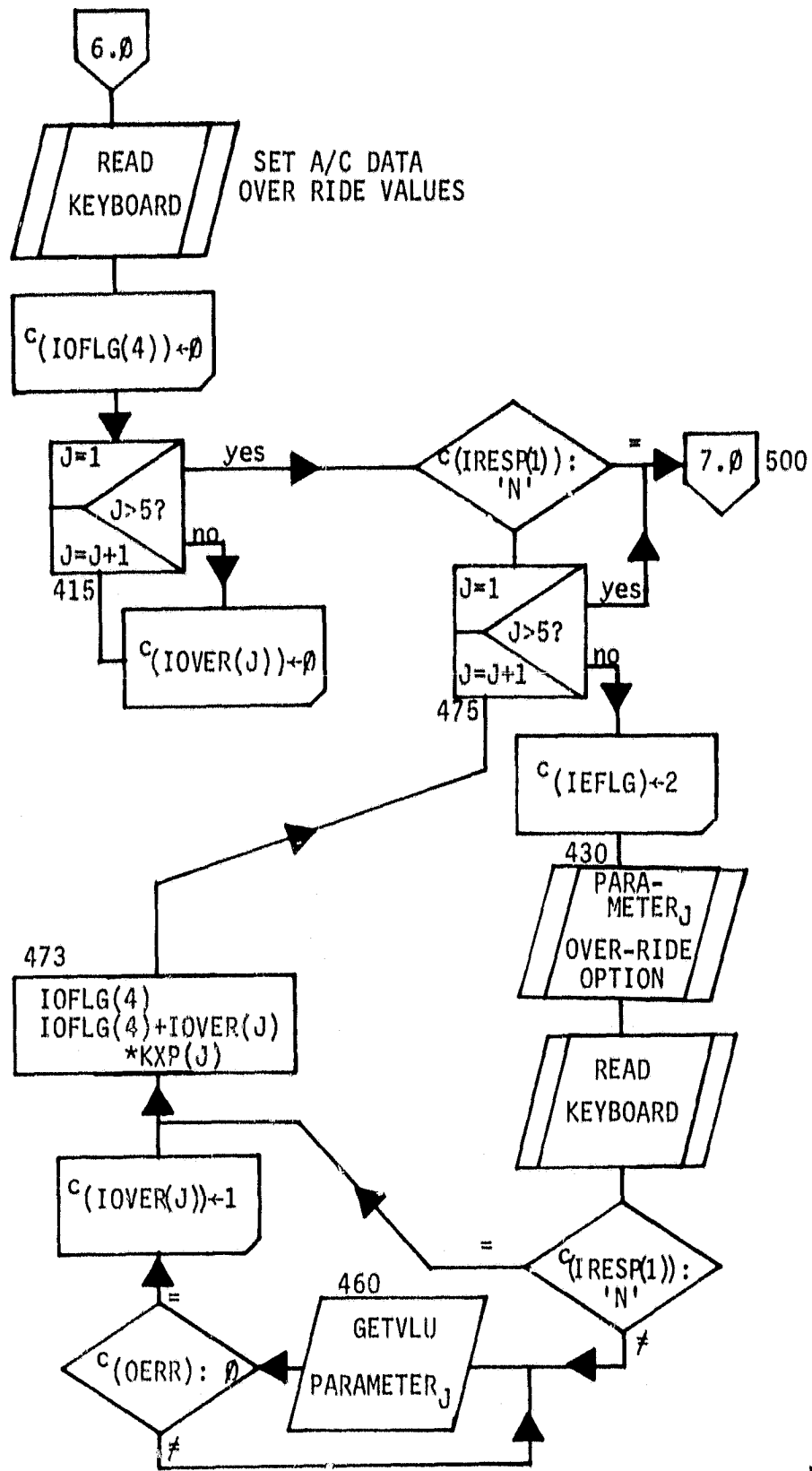
PRECEDING PAGE BLANK NOT FILMED

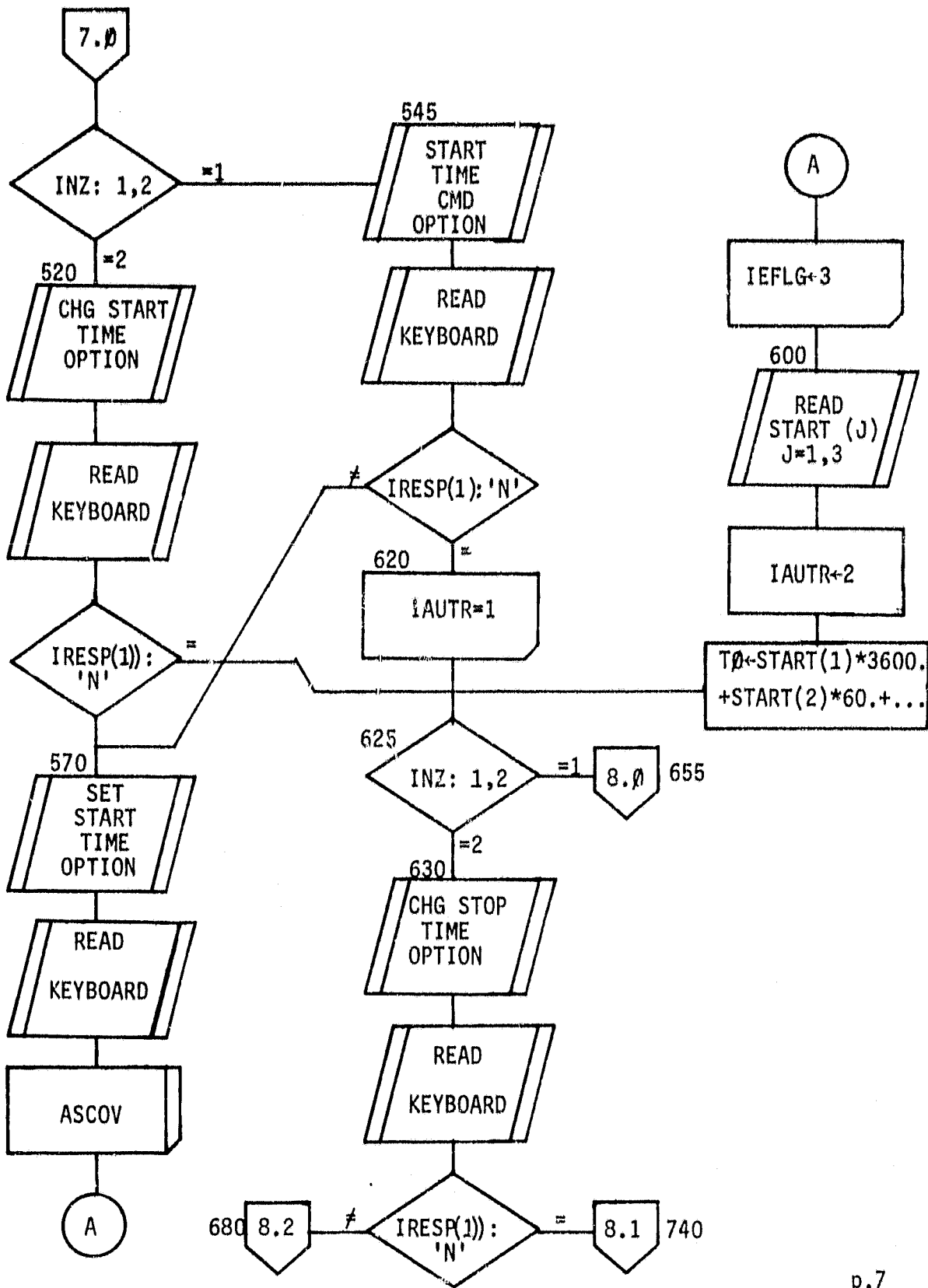


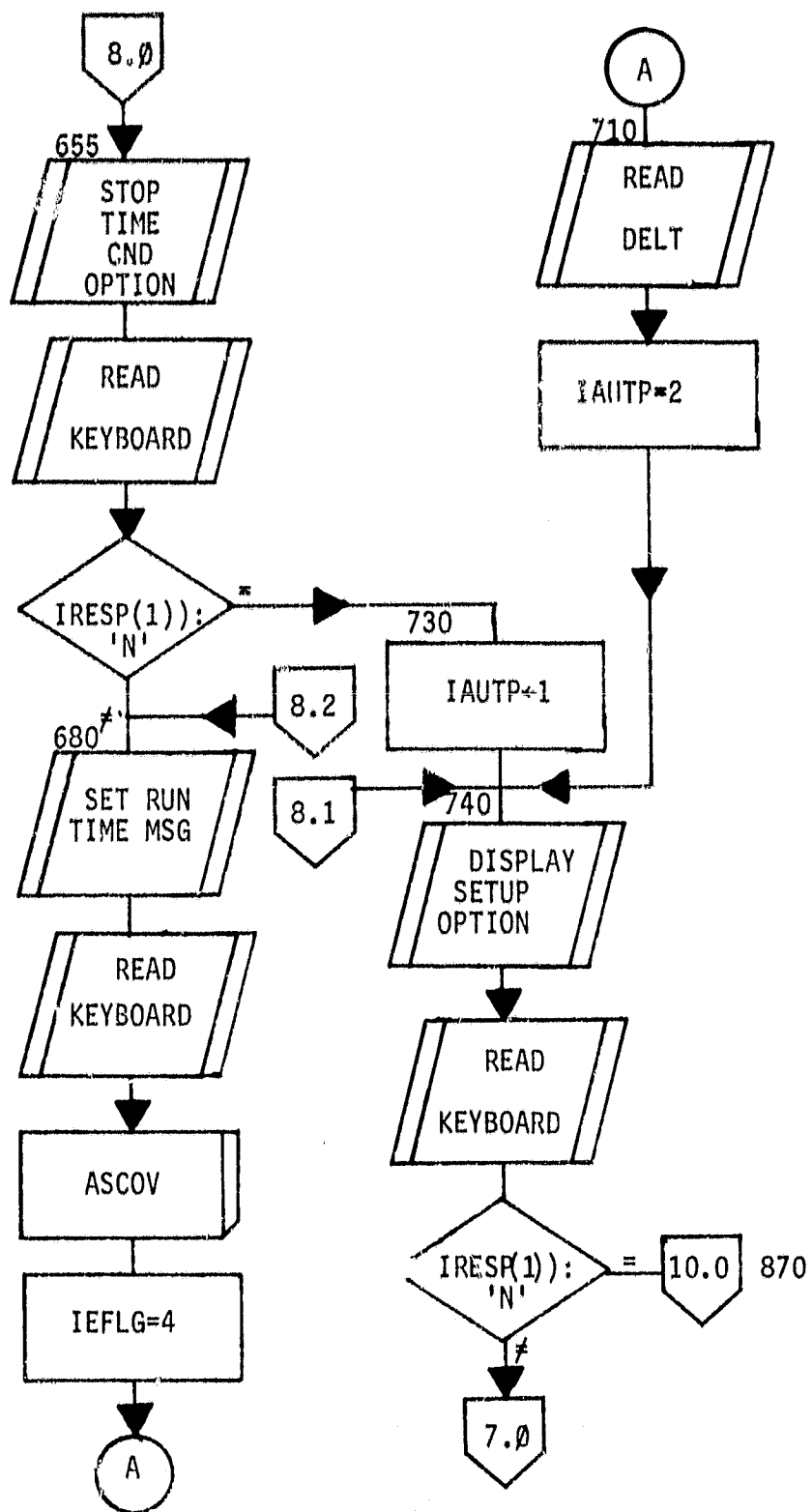


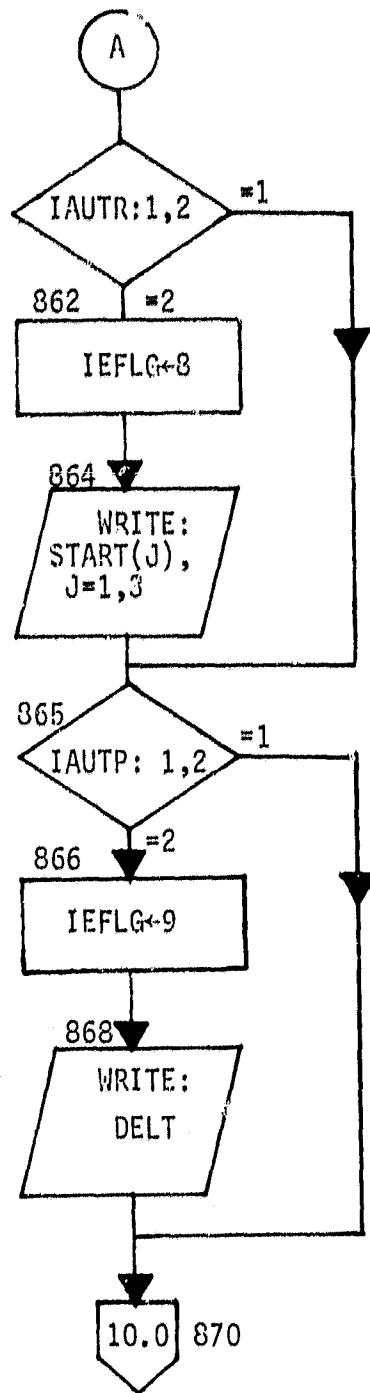
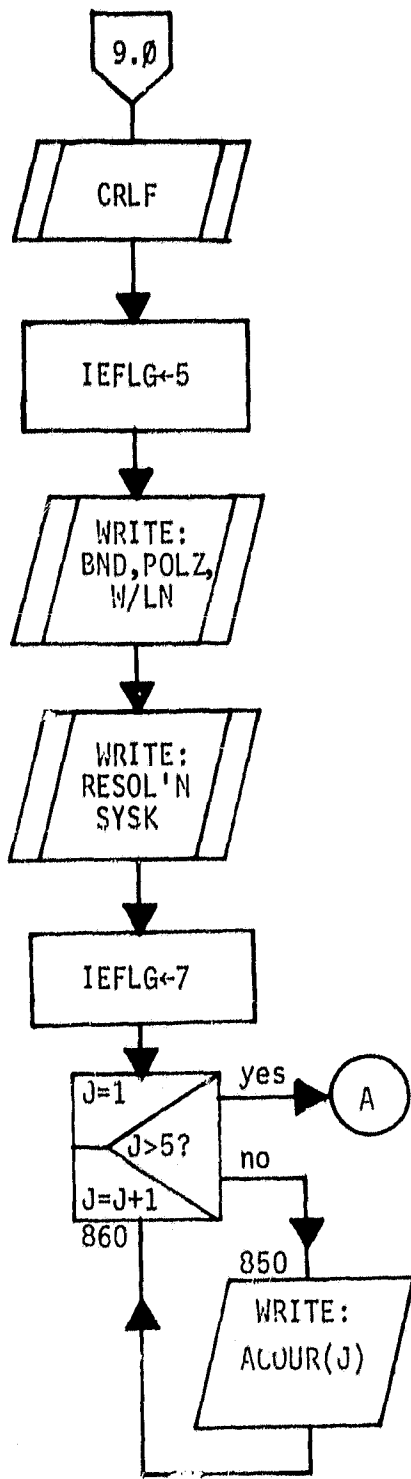


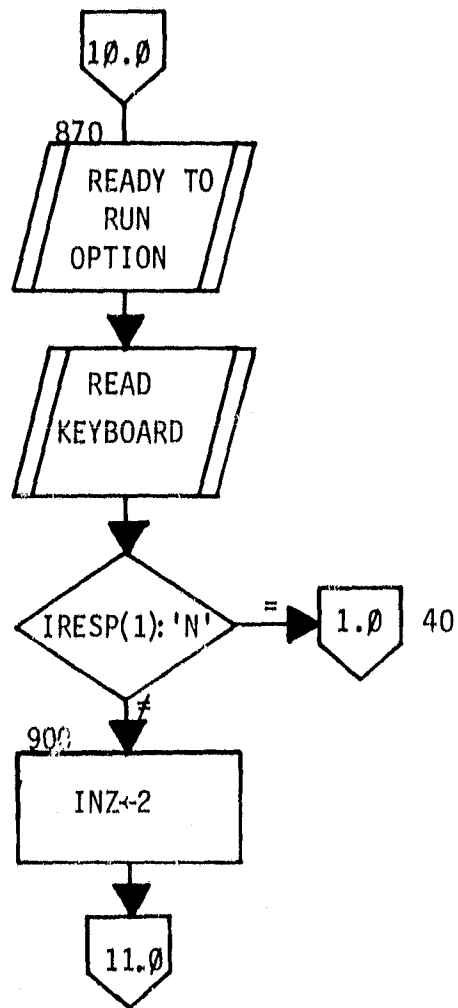




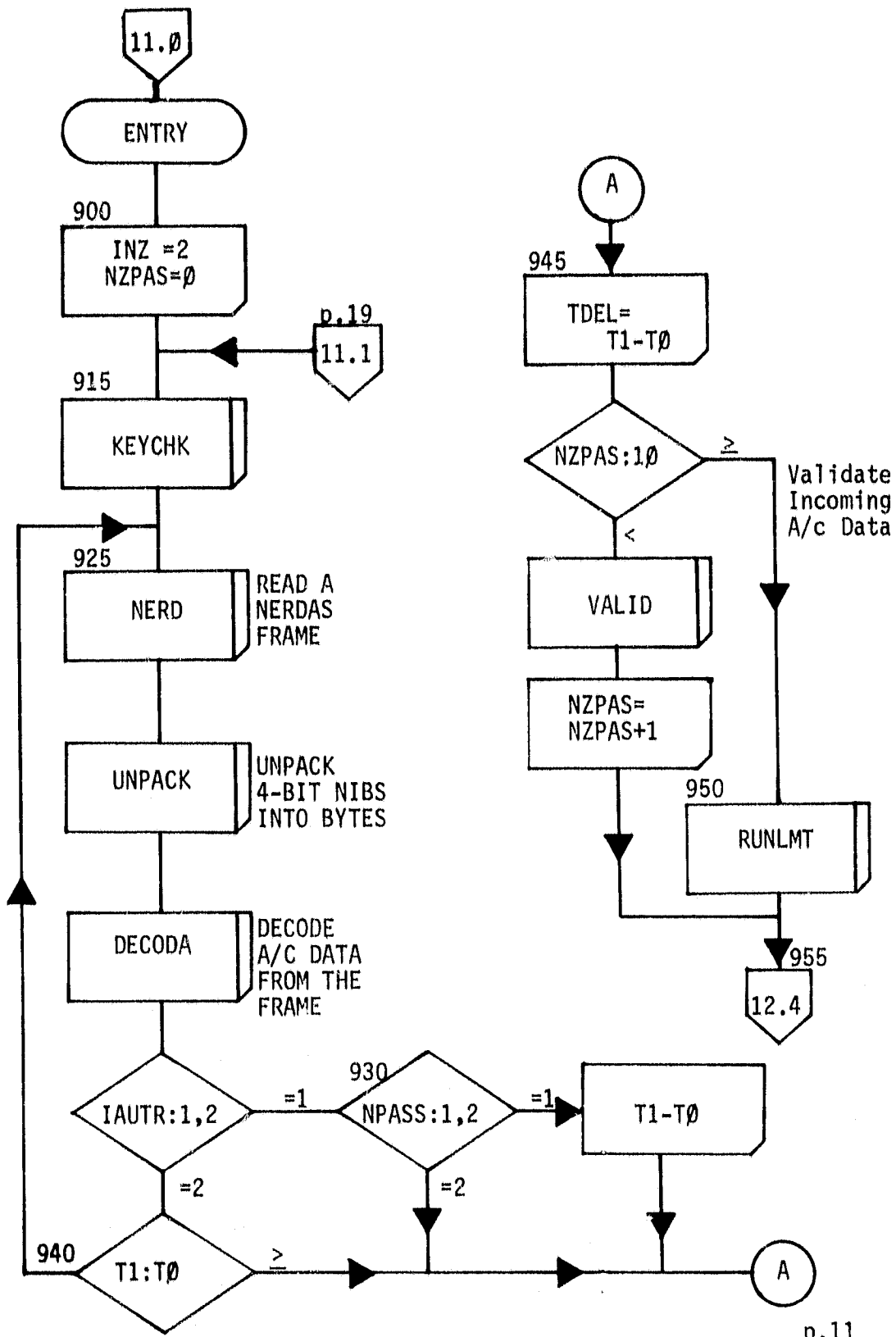




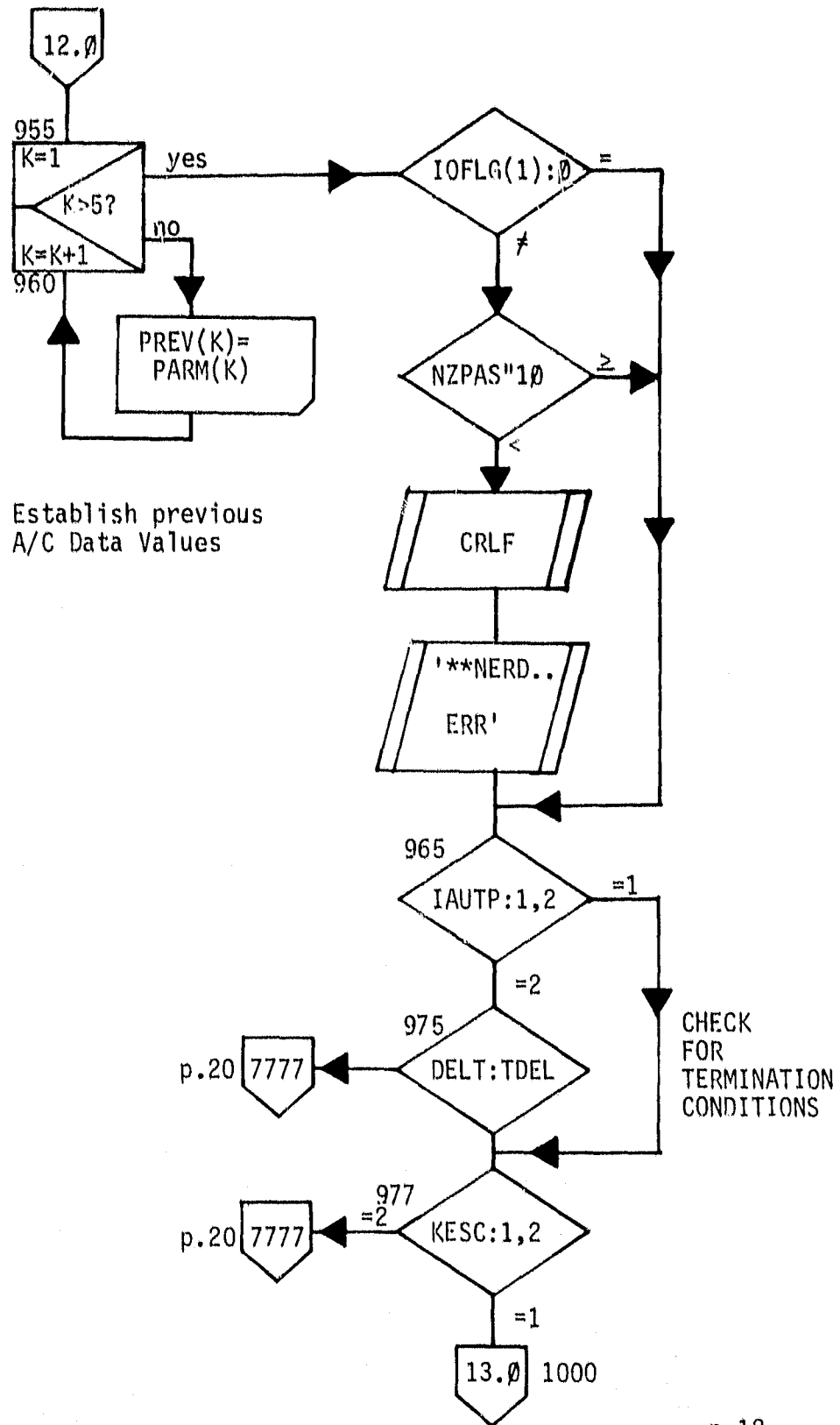




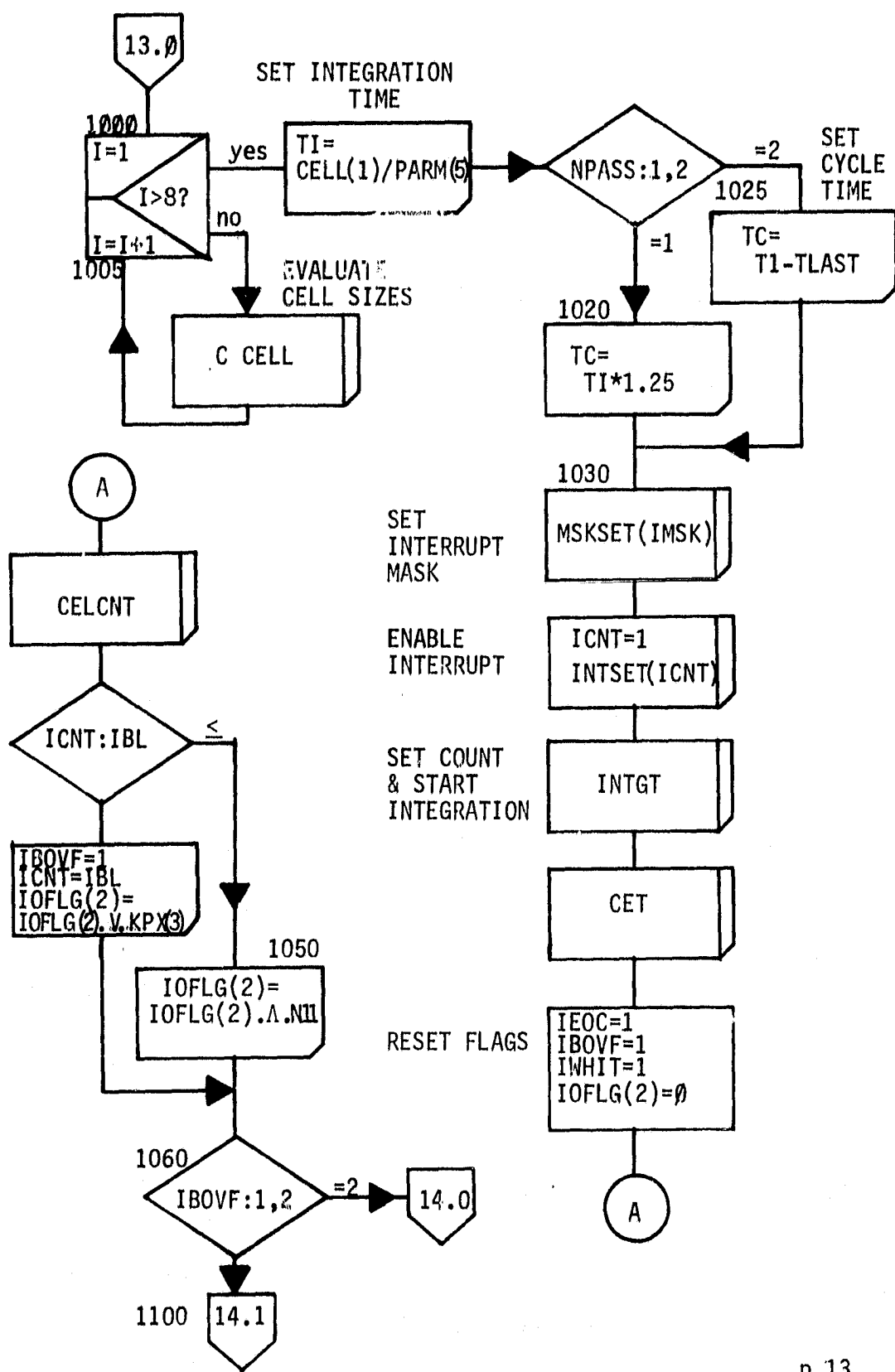
SIGMA1 CALCULATIONS

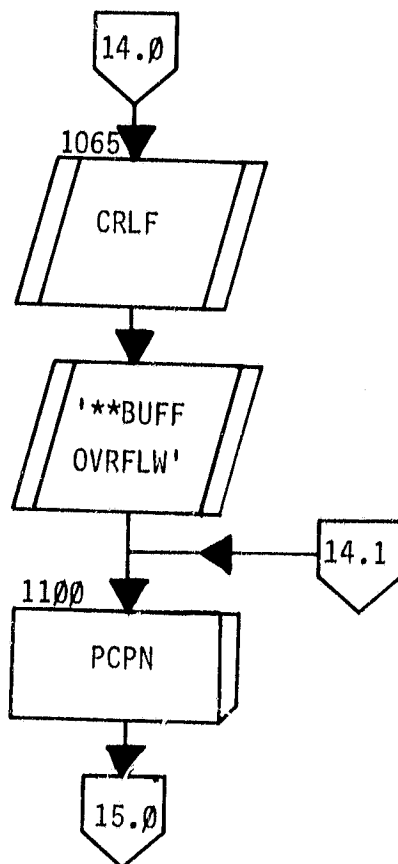


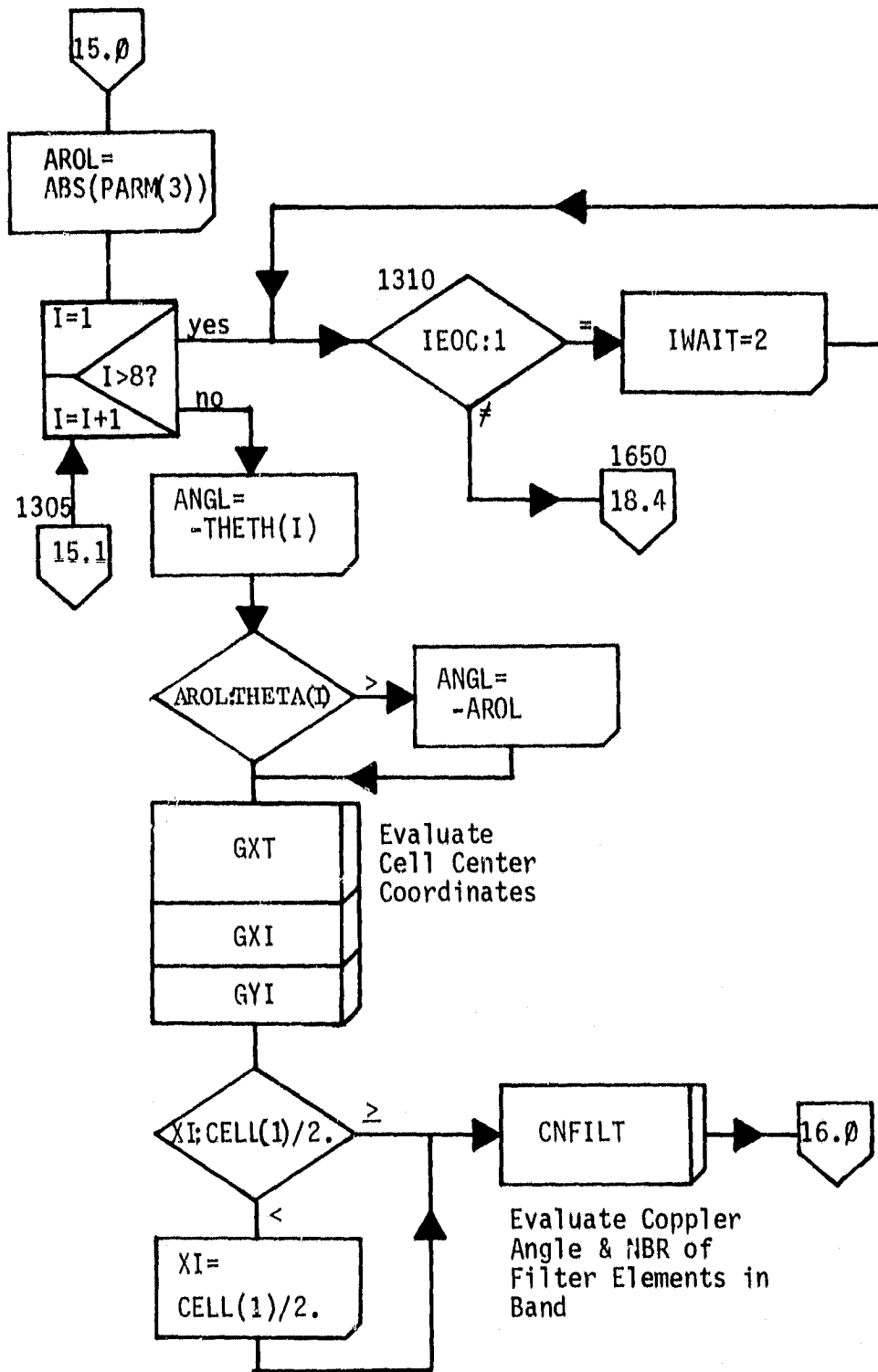
p.11

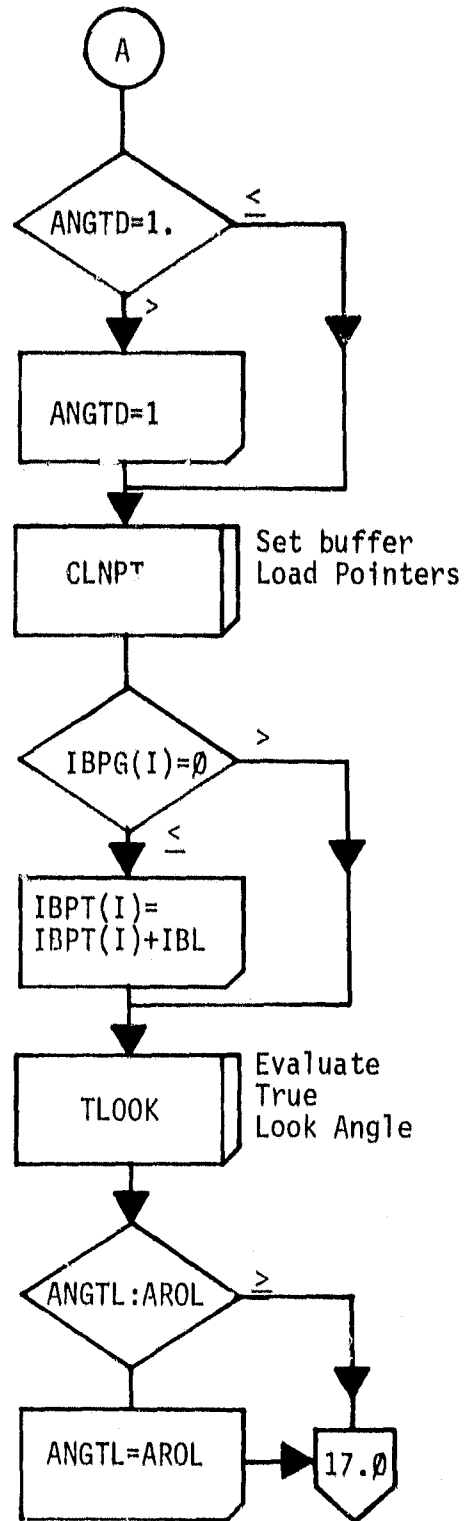
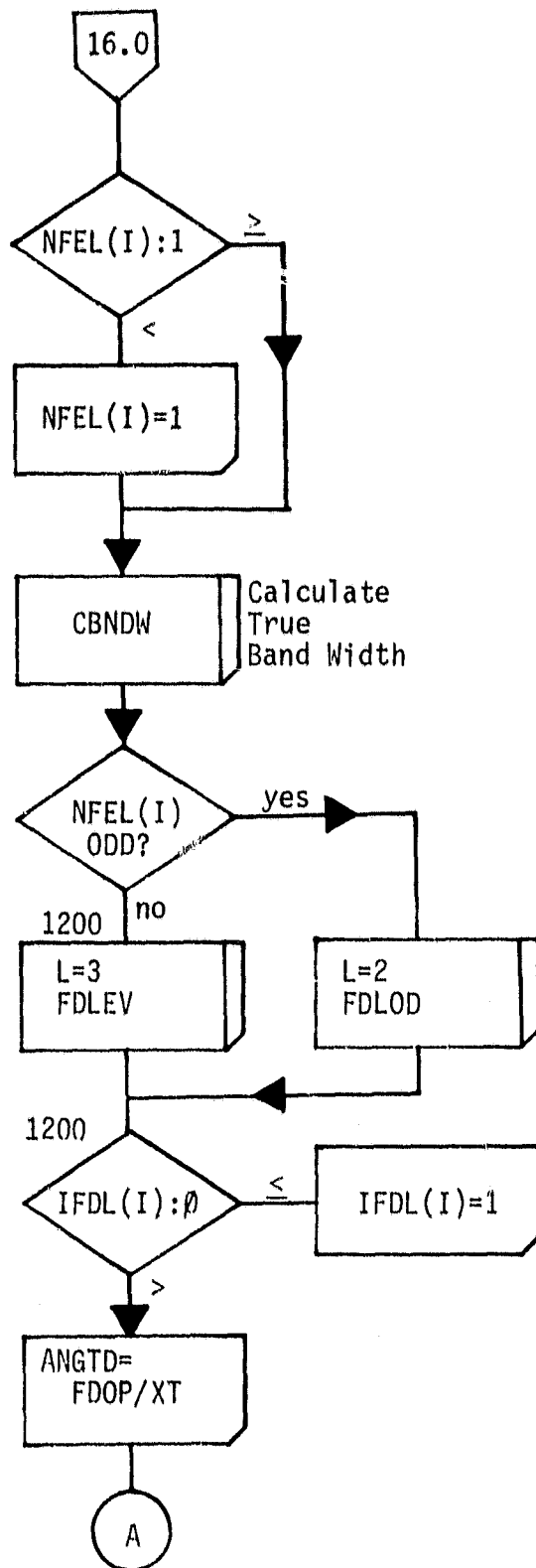


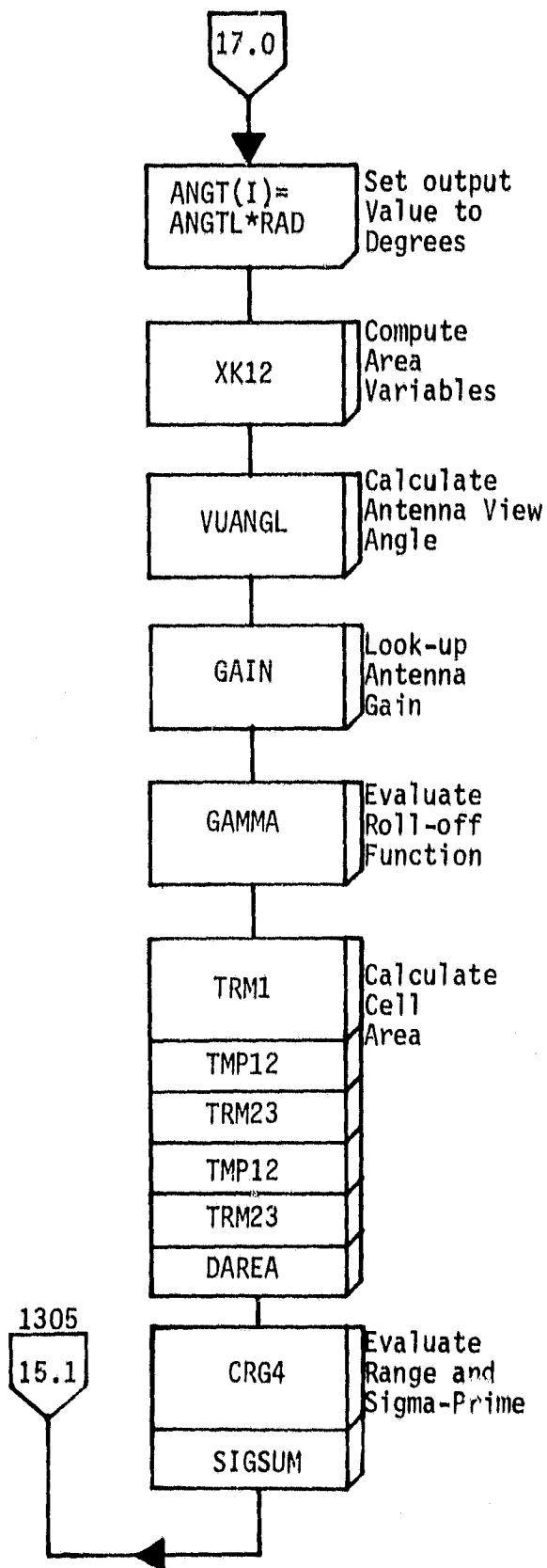
p.12



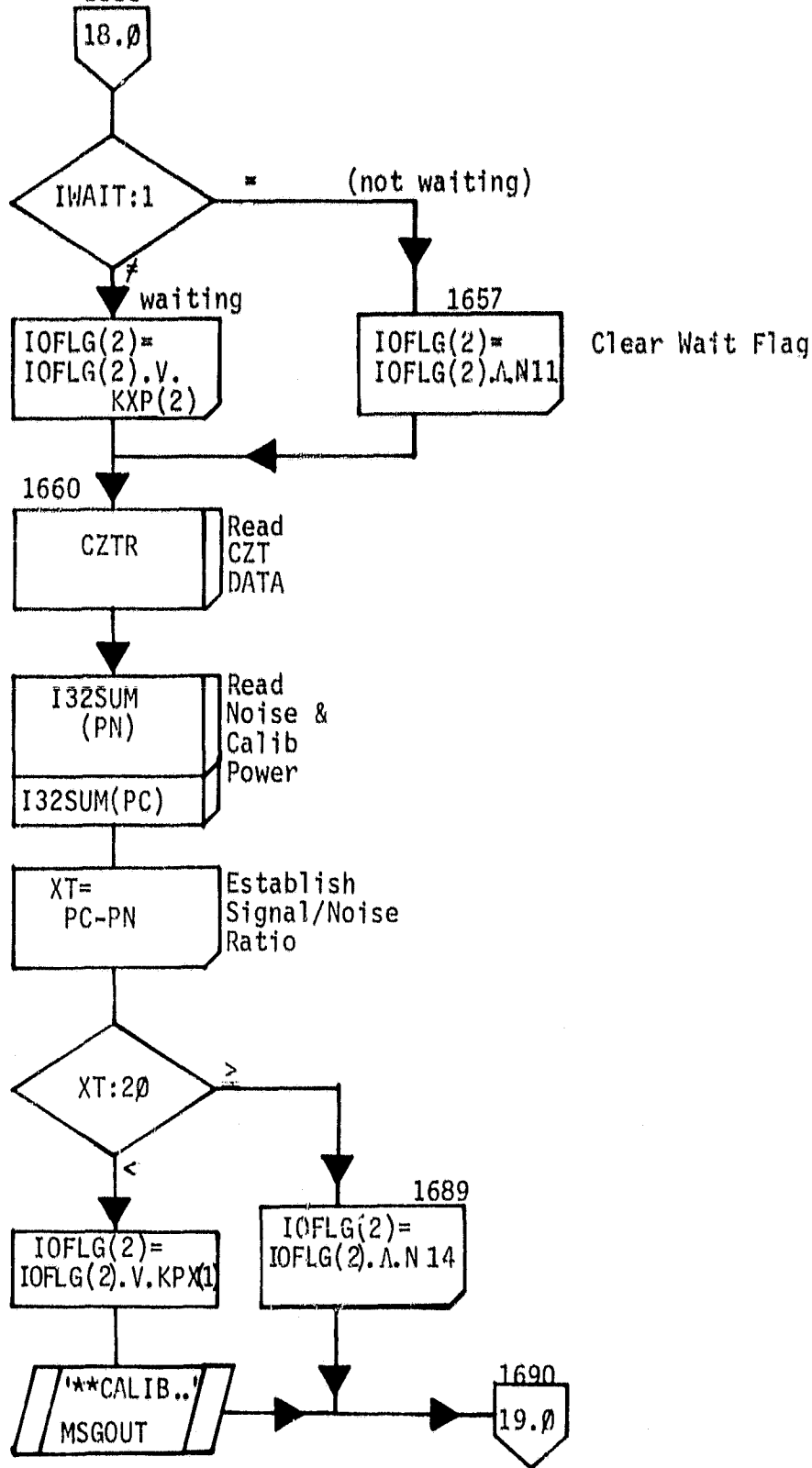


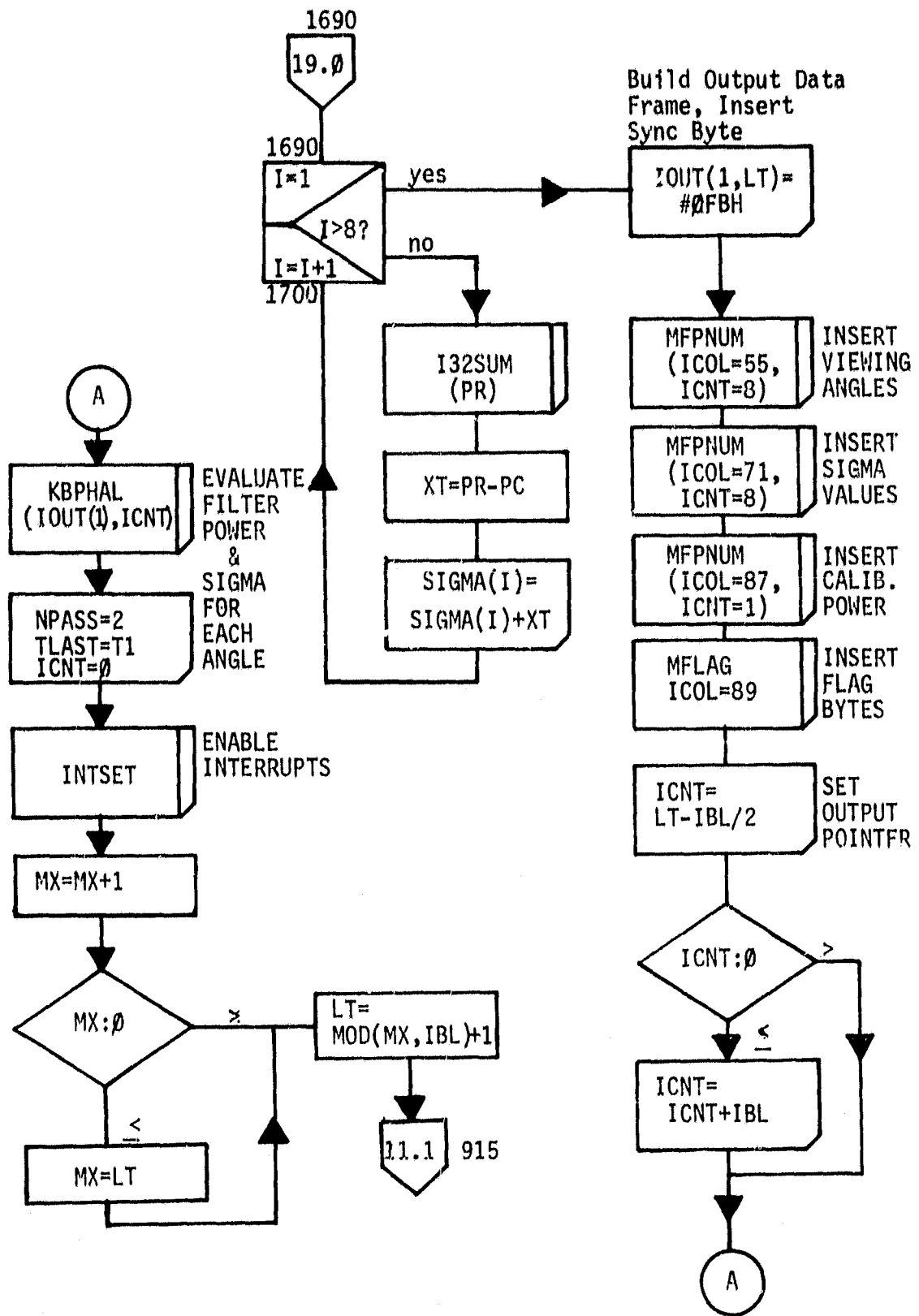




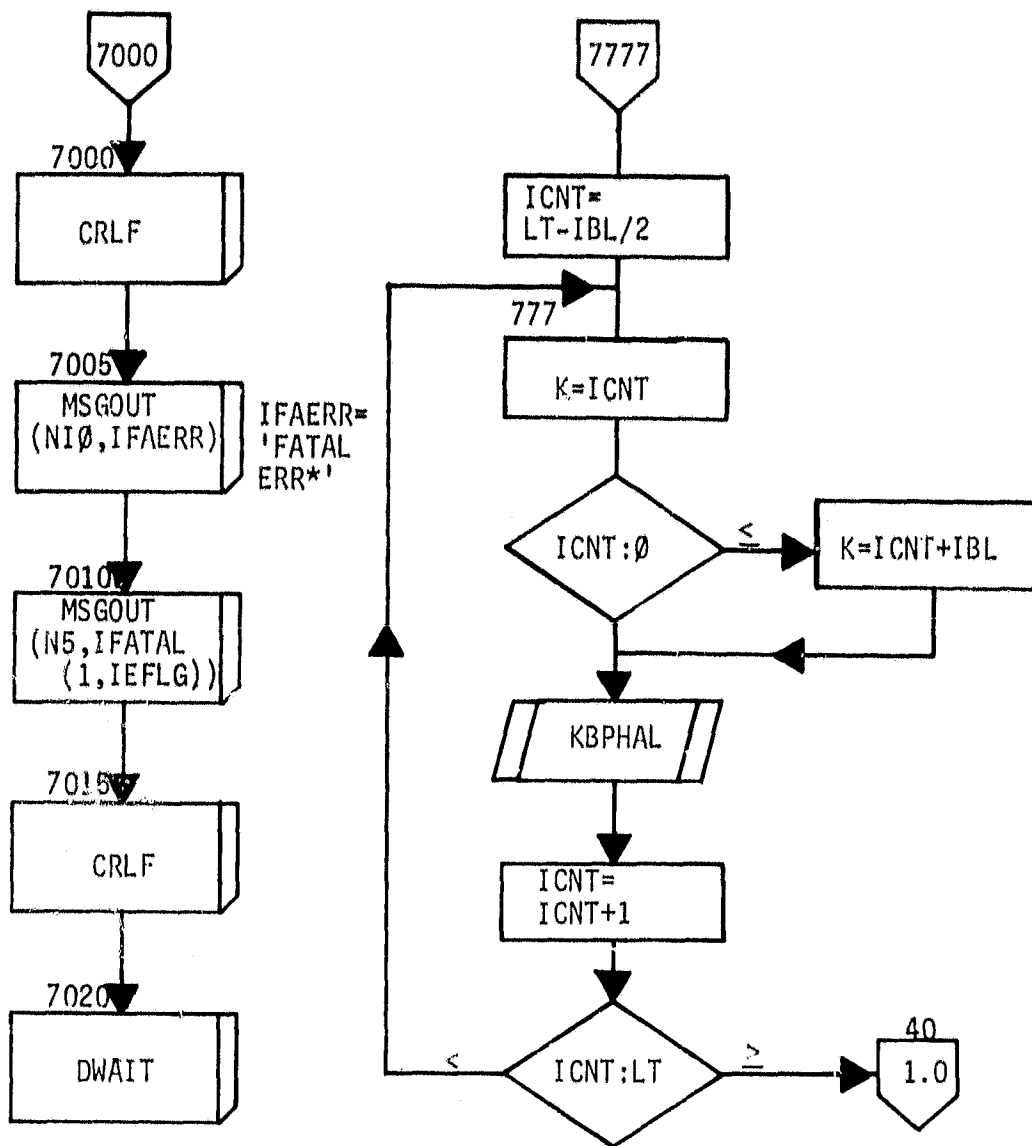


1650 SIGMA2 CALCULATIONS

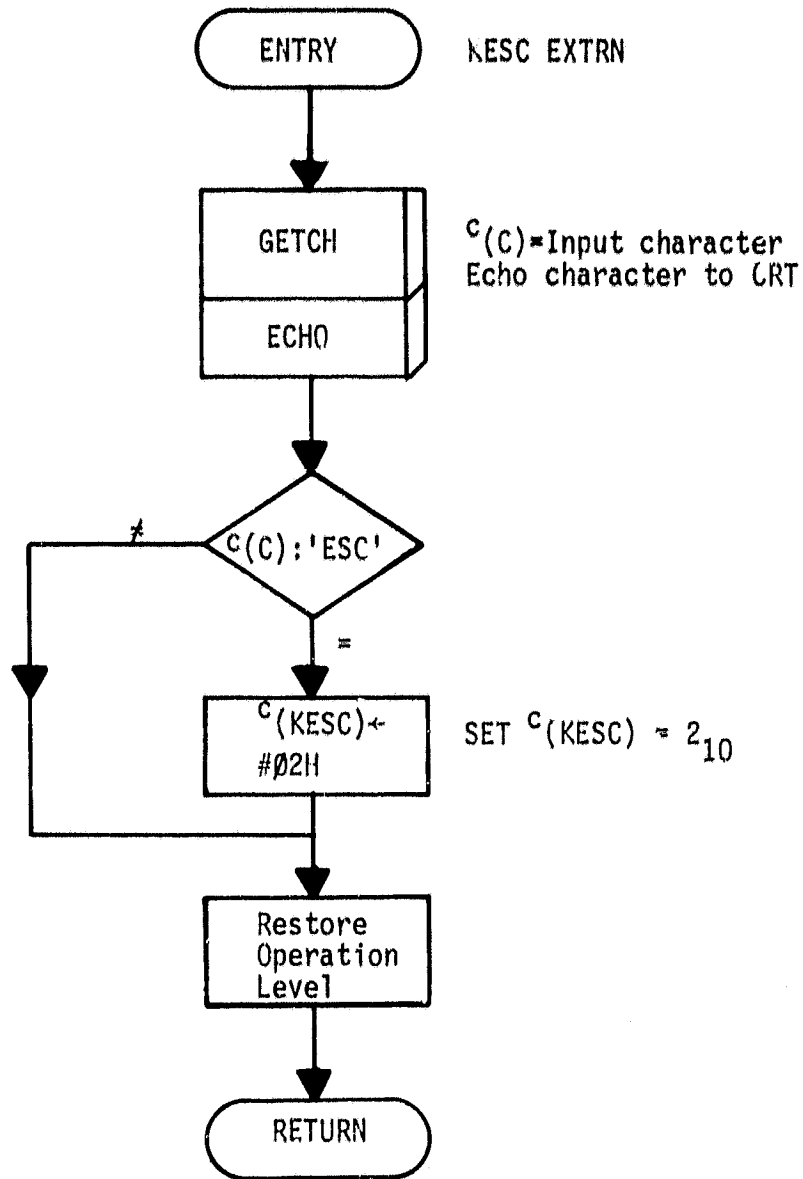




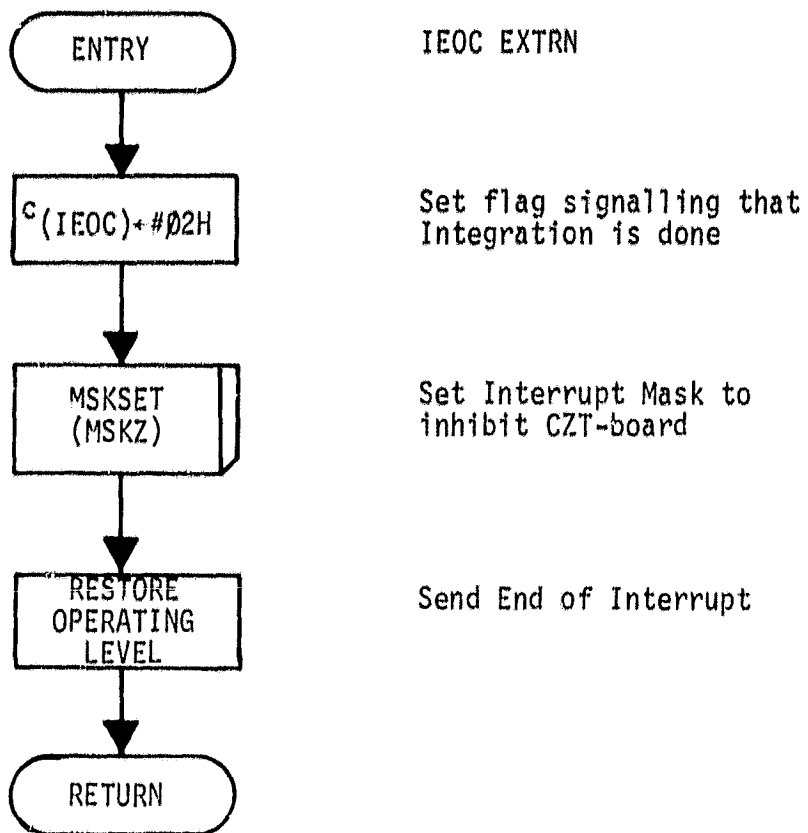
ERROR MESSAGE HANDLER, IINTCOM MODULE



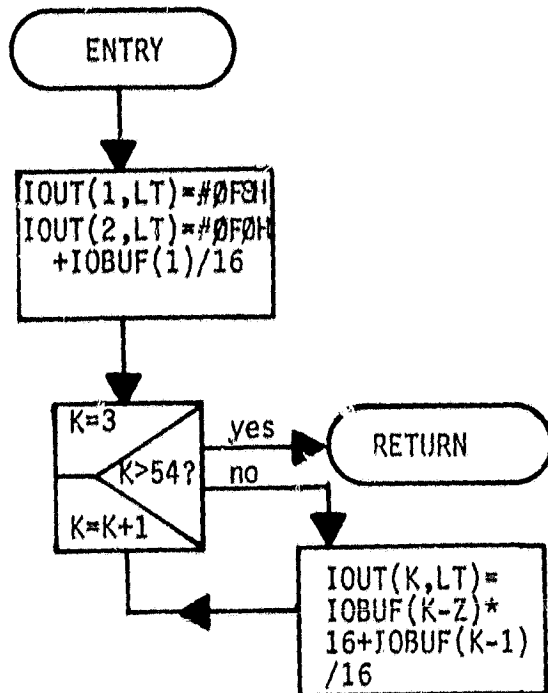
KEYBOARD INTERRUPT HANDLER



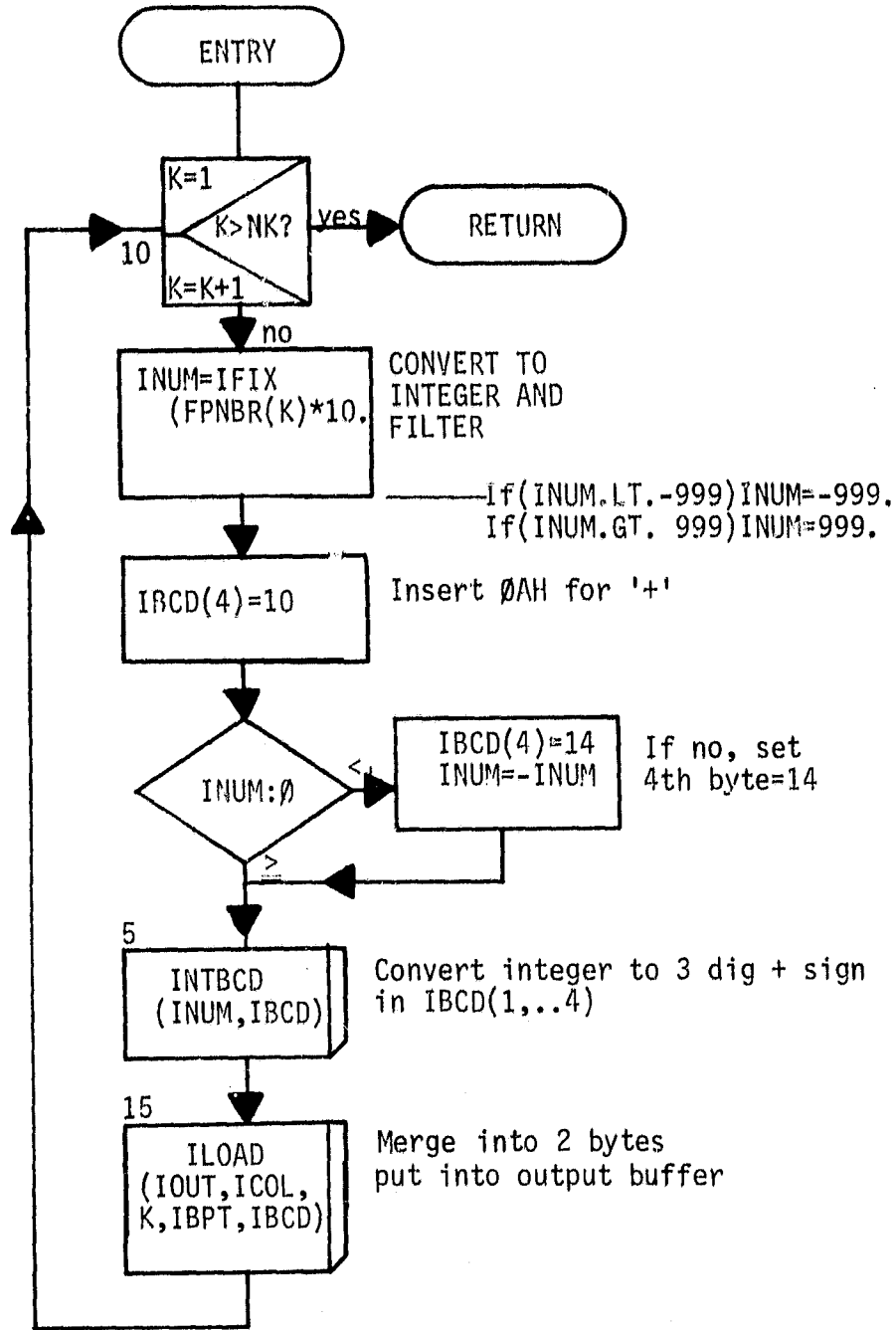
CZT Board Interrupt Handler



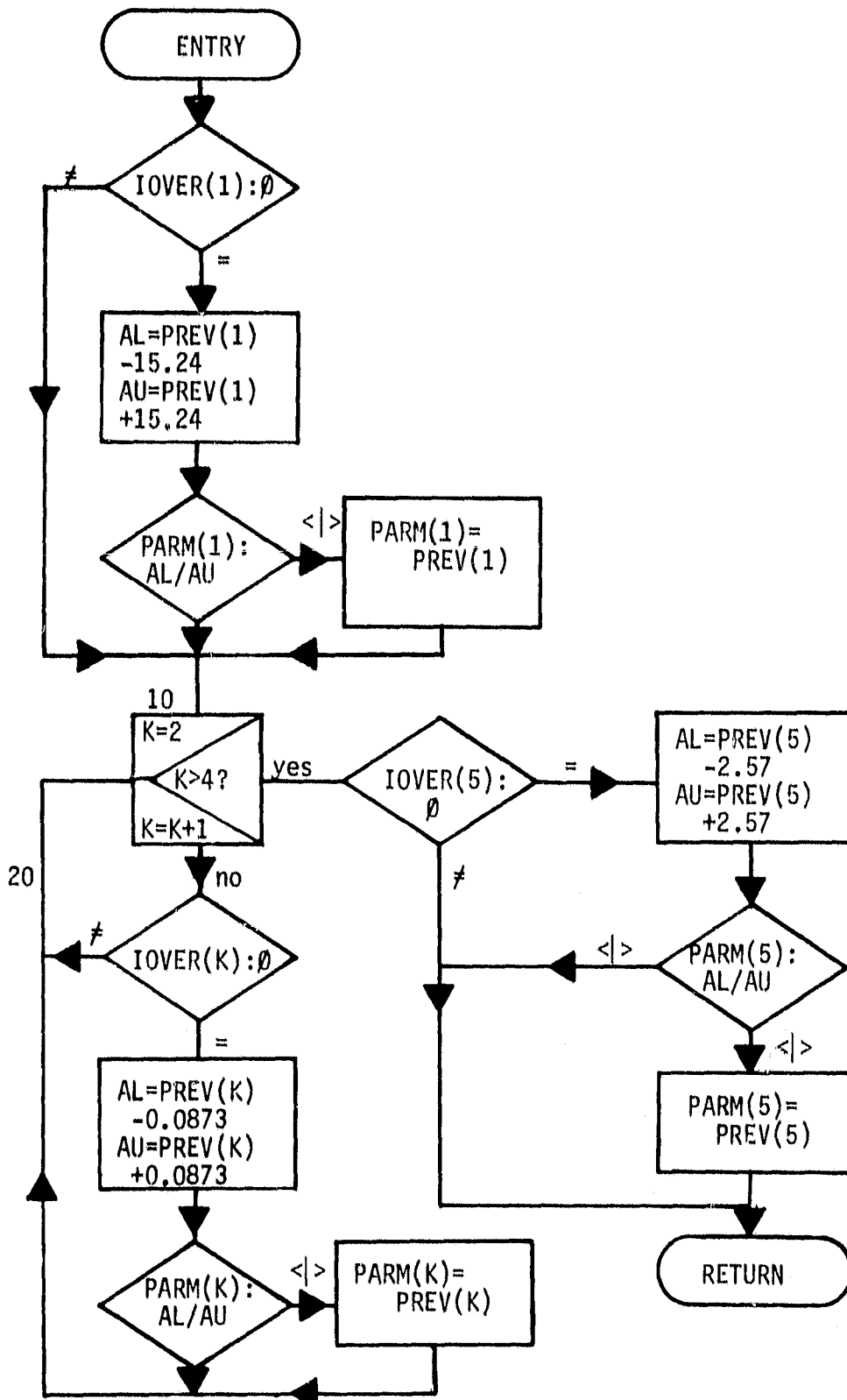
SUBROUTINE MVNERD(IOUT, IOBUF)
where IOUT(128,70) and IOBUF(60)
are integer*1



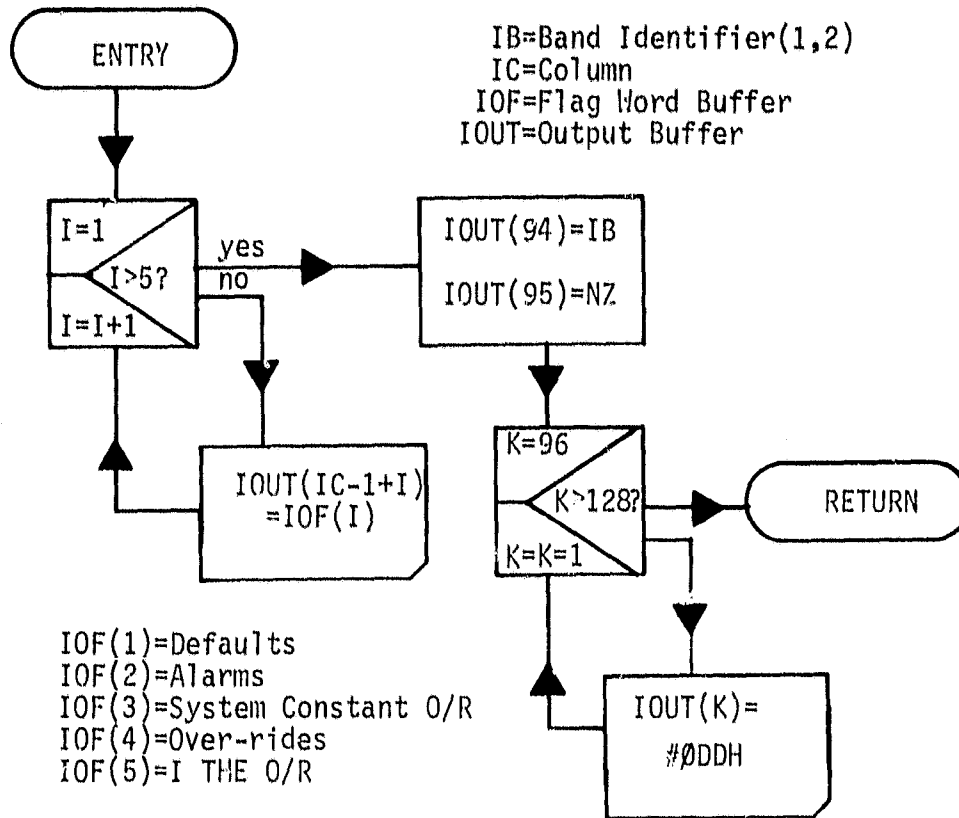
MFPNUM



SUBROUTINE RUNLMT(PREV,PARM,IOVER)
 PREV(5), PARM(5), IOVER(5)



MFLAG SUBROUTINE



BIT ASSIGNMENT:

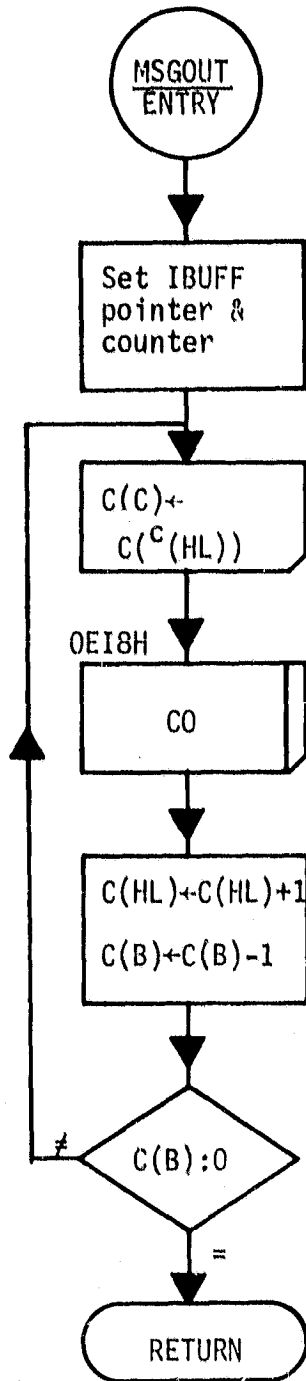
IOF(1), IOF(3)

BIT	
0	VEL
1	PITCH
2	ROLL
3	DRIFT
4	ALT
5	X
6	X
7	X

IOF(2)

BIT	
0	BUFFER OVERFLOW
1	WPIT FLAG
2	CALIBRATION ALARM
3	X
4	X
5	X
6	X
7	X

Message Output to Console, Fortran Callable
MSGOUT(ICNT, IBUFF)



Register:

C(BC)=adds of char.count
C(DE)=adds of output strings

C(HL) C(DE), sets pointer
C(A) C(C(BC)), gets count
C(B) C(A), set counter (07FH)

Get a character from the buffer.

Send character in Reg C to console.

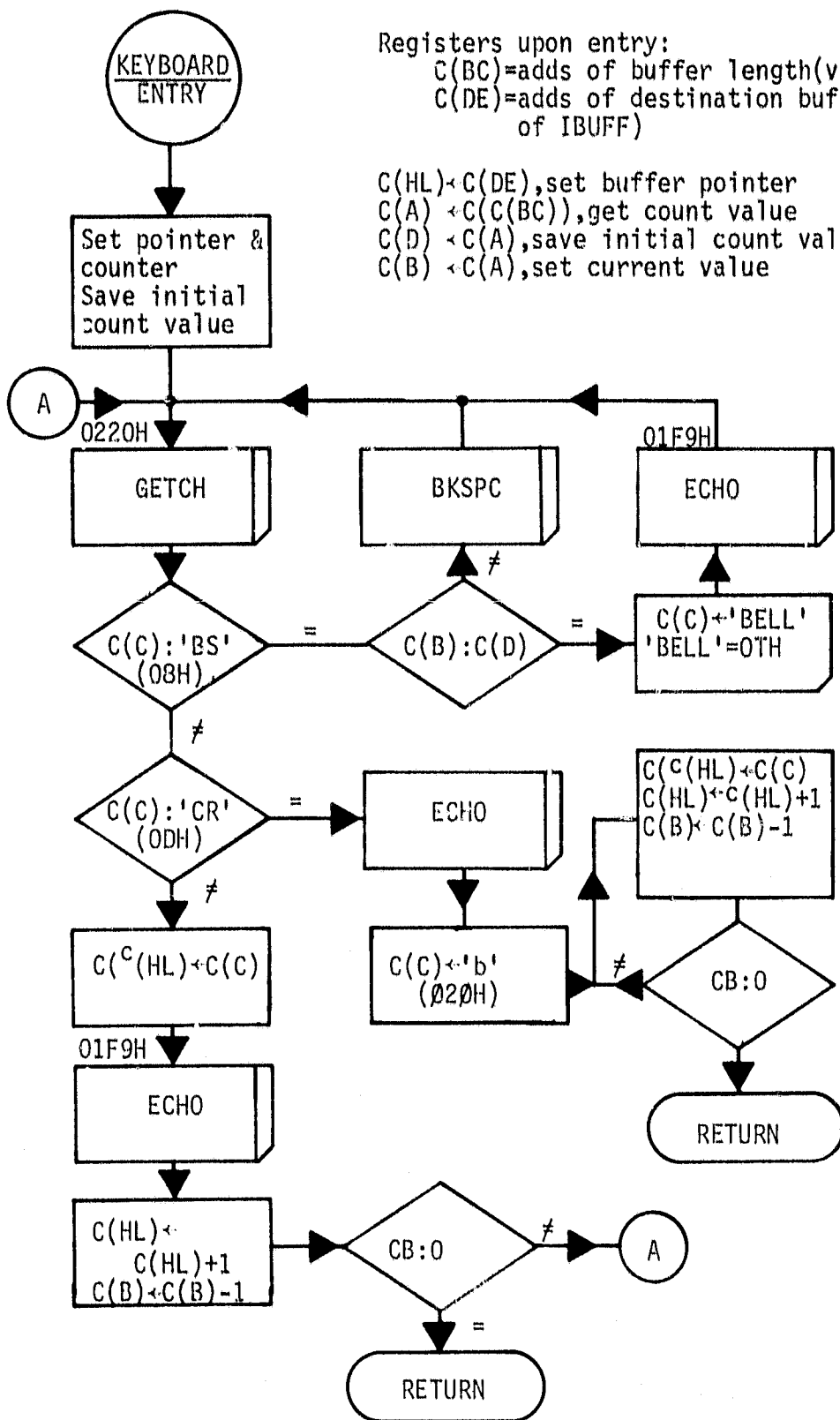
Step adds pointer
Decrement count

Last of String?

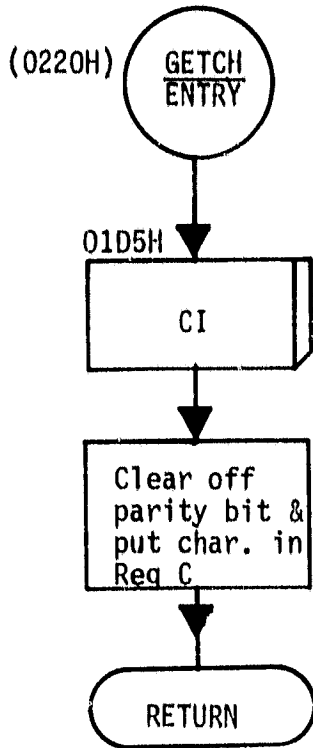
CONSOLE INPUT ROUTINE: KEYBRD(NBR, Ibuff)

Registers upon entry:
 C(BC)=adds of buffer length(value of NBR)
 C(DE)=adds of destination buffer (value of Ibuff)

C(HL)←C(DE),set buffer pointer
 C(A) ←C(C(BC)),get count value
 C(D) ←C(A),save initial count value
 C(B) ←C(A),set current value



Get Character from Input Terminal



Registers;
 C(HL) = Input buffer address pointer
 C(B) = current char. count

Call char. input routine
 C(A) ← 'char'

ANI 07FH
 C(C) ← C(A)

Registers:
 C(HL) = Input buffer address
 C(B) = Current char. count

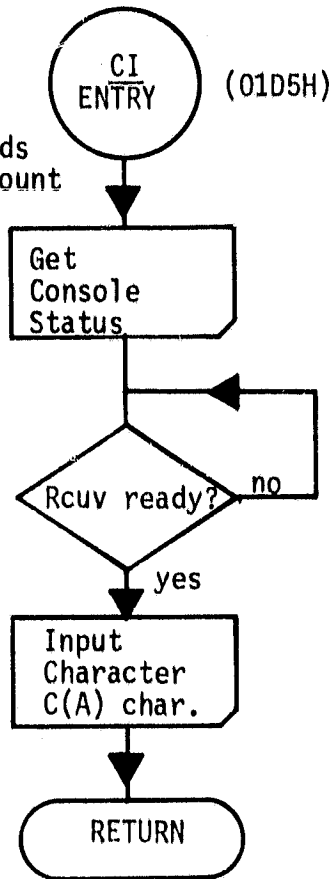
IN OCDH

ANI 0ZH
 JZ CI

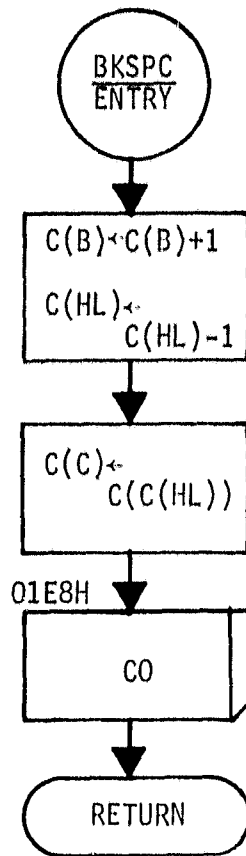
IN CC

RET

CONSOLE INPUT



BACKSPACE OR RUBOUT



Registers:

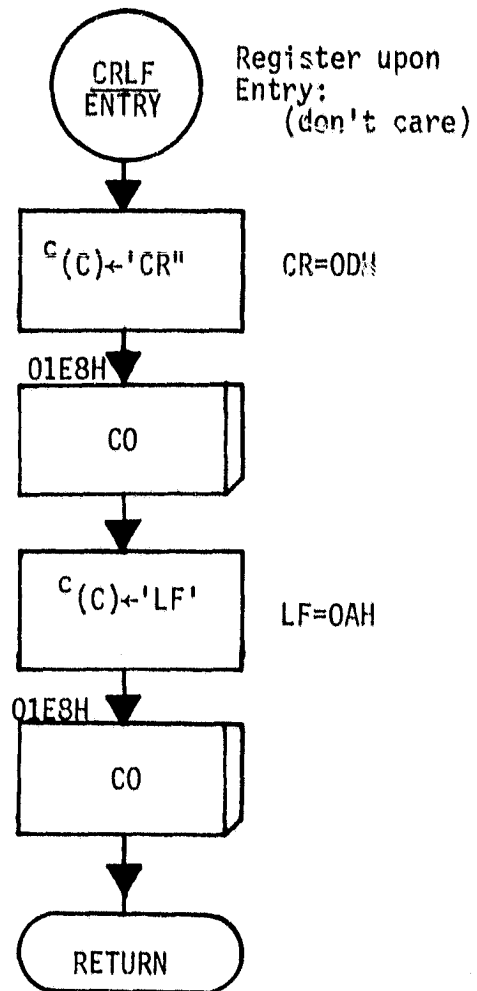
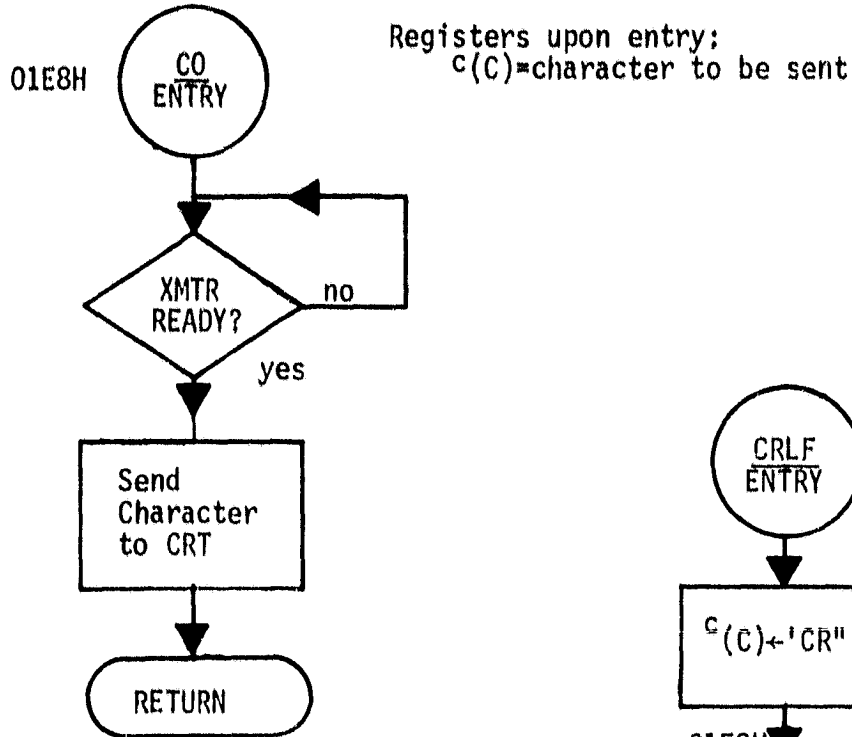
C(HL) = pointer to input buffer
C(IBNT) = initial character count
C(B) = current initial character count
C(C) = input char. value ('BS')

Increment character counter by one.
Set pointer to prior input buffer character.

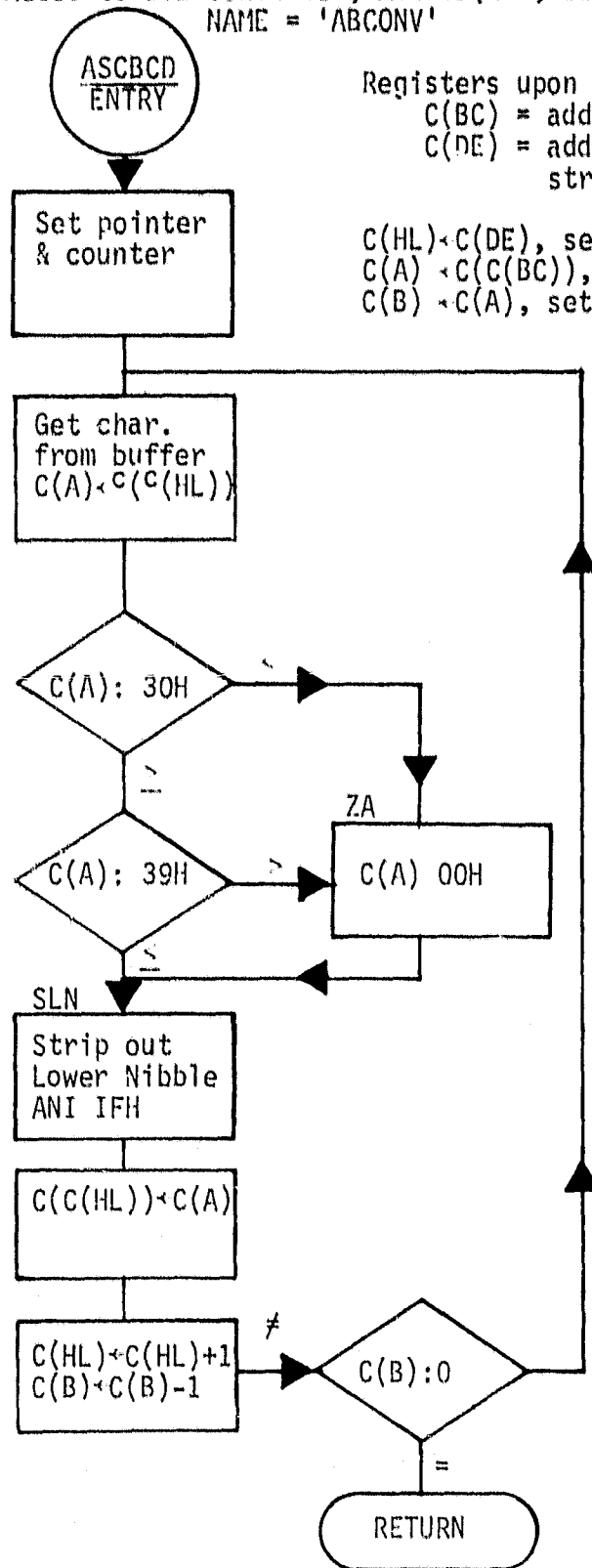
Get prior char. from buffer.

Send previous character to CRT.

Console Output Routine



ASCII to BCD Converter, ASCBCD(NBR, Ibuff)
 NAME = 'ABCONV'



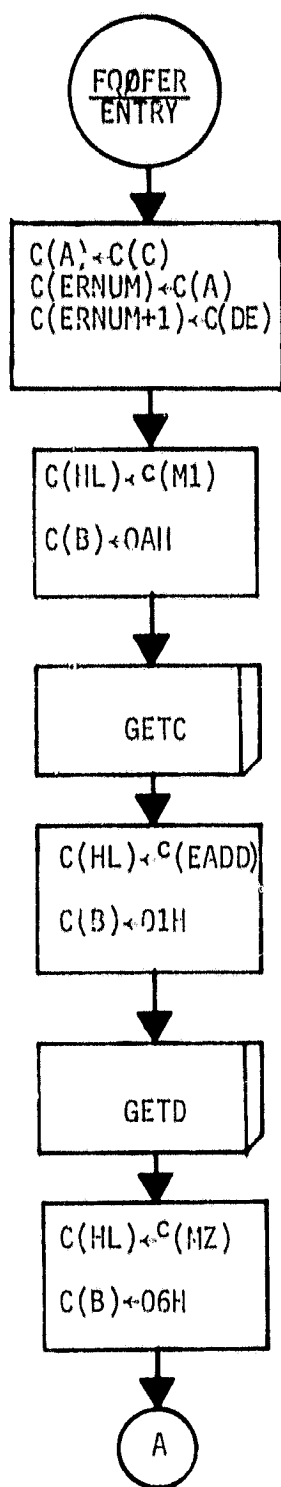
Registers upon entry:
 C(BC) = adds of counts value
 C(DE) = adds of buffer containing string

C(HL) ← C(DE), set pointer
 C(A) ← C(C(BC)), get count value
 C(B) ← C(A), set counter

If character is < '0' or if character is > '9' replace it with BCD '0'.

Replace ASCII character in the string with BCD value.

FORTRAN: Run-Time Error Recovery



Registers:

C(BC) = Error Nbr, ≤255

C(DE) = Adds of statement nearest where error occurred.

Save value of Error Nbr @ ERNUM
and adds @ ERNUM+1

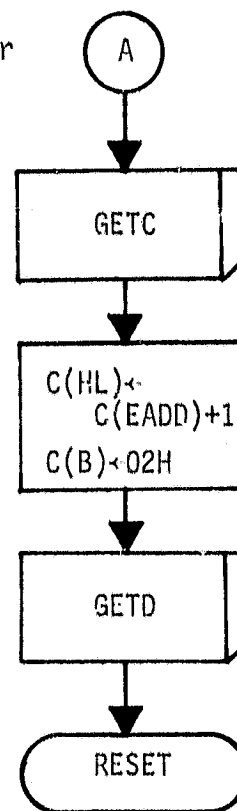
Set string pointer
Set string(character) counter

Send string to CRT

Set byte pointer
Set byte counter

Send to CRT

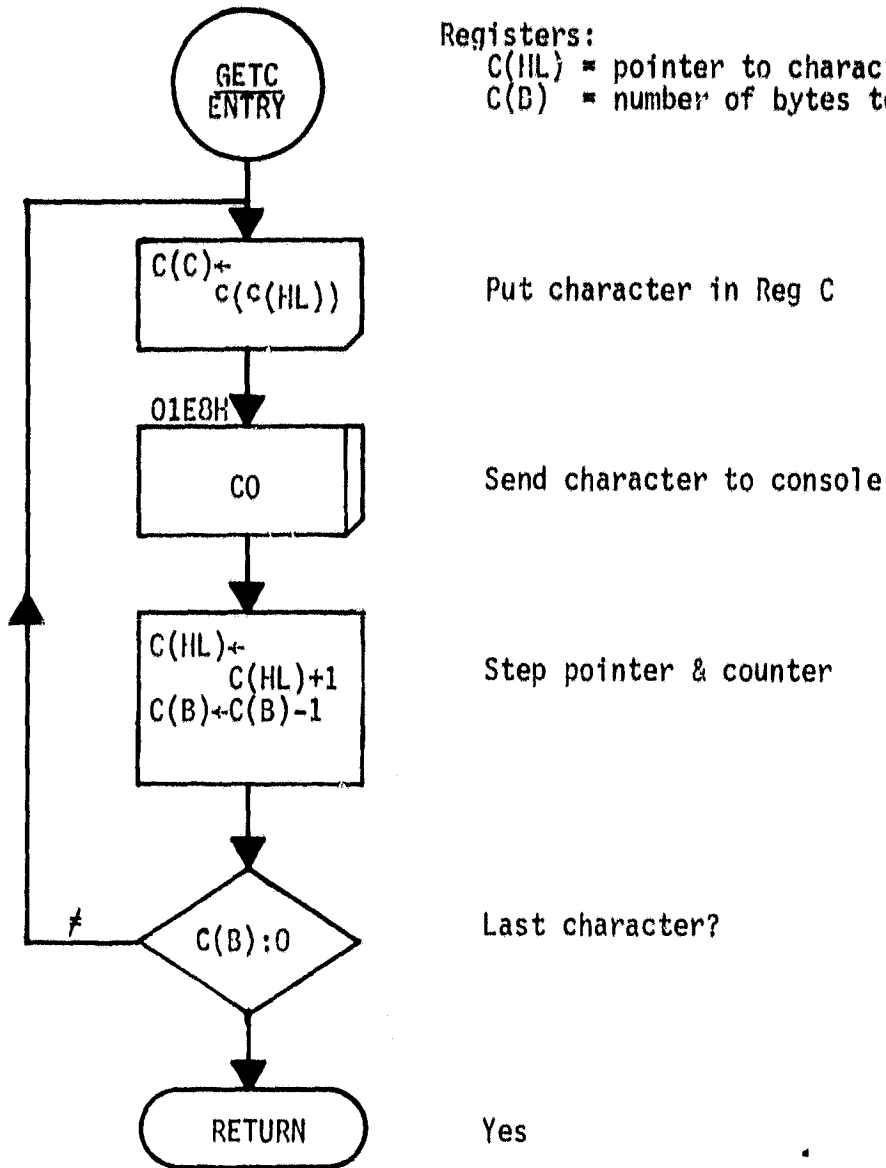
Set pointer
Set counter



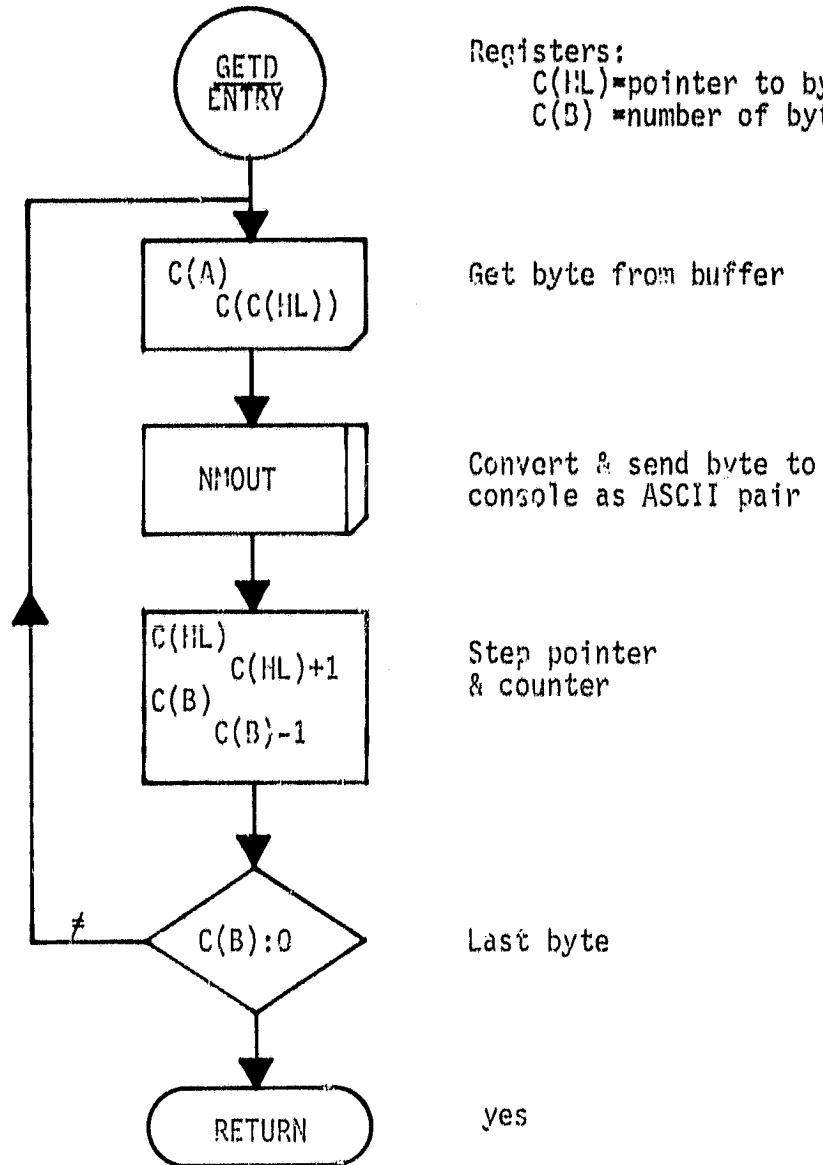
FORTTRAN Run-Time Error Recovery (cont.)
 GETC, sends character string to terminal.
 co routine resides in Monitor.

Registers:

C(HL) = pointer to character buffer
 C(B) = number of bytes to send



FORTRAN Run-Time Error Recovery (cont.)
 GETD, converts byte to ASCII character pair
 NMOUT routine resides in Monitor



Registers:
 C(HL) = pointer to byte buffer
 C(B) = number of bytes to process

Get byte from buffer

Convert & send byte to console as ASCII pair

Step pointer & counter

Last byte

yes

APPENDIX B
INTCOM Listing (Main Driver)

```

0001 C:.....PARAMETER DEFINITIONS:.....C
0002 C
0003 C          C/L-BAND SCATTEROMETER PROCESSOR          C
0004 C
0005 C          TEXAS A&M UNIVERSITY REMOTE SENSING CENTER      C
0006 C
0007 C:.....C
0008 C
0009 C          ANGL          VIEWING ANGLE, ALWAYS AFT, ANTENNA TO GROUND CELL.
0010 C          ANGT          VIEWING ANGLE, IN DEGREES, USED IN OUTPUT FRAME.
0011 C          AROL          ABSOLUTE VALUE OF AIRCRAFT ROLL, RADIAN.
0012 C          BMW           BEAM WIDTH OF ANTENNA, BAND DEPENDENT, RADIAN.
0013 C          BMWID        BEAM WIDTH CONSTANT, BAND DEPENDENT, DEGREE.
0014 C          BNDW        BAND WIDTH VARIABLE USED FOR EACH FILTER.
0015 C          CANT1       ANTENNA GAIN CONSTANT, USED FOR C-BAND.
0016 C          CANT2       " " " " " "
0017 C          CANT3       " " " " " "
0018 C          CANT4       " " " " " "
0019 C          CC          TWO DIMENSIONAL CONSTANT BUFFER USED FOR ROLL-OFF.
0020 C          CCLC        CHARACTER BUFFER FOR 'L' AND 'C'.
0021 C          CELL        CELL SIZE, METERS, 8 VALUES.
0022 C          CFAERR      ERROR MESSAGE BUFFER.
0023 C          CFATAL      " " "
0024 C          CILC        CHARACTER BUFFER FOR 'L', 'C'.
0025 C          CIOBUF      I/O BUFFER, EQUIVALENCED TO CRESP.
0026 C          CIOVRG      CHARACTER BUFFER USED FOR OVER-RIDE INPUTS.
0027 C          CIPOLZ      " " " " POLARIZATION ".
0028 C          CIYN        " " " " OPERATOR RESPONSES.
0029 C          CL          L-BAND ROLL-OFF POLYNOMIAL CONSTANTS.
0030 C          CNERD       CHARACTER BUFFER, EQUIVALENCED TO CRESP.
0031 C          CRESP      " " , GENERAL I/O USAGE.
0032 C          CST         GENERAL CONSTANT DATA IUFFER.
0033 C          DELF        FILTER SPECTRAL LINE-WIDTH, BAND DEPENDENT.
0034 C          DELT        DATA ACQUISITION AND RUN TIME, OPERATOR SETS.
0035 C          DFALT       AIRCRAFT NERDAS DEFAULT VALUES.
0036 C          DIFF        DIFFERENCE BETWEEN VIEWING AND RESOLUTION ANGLE.
0037 C          DLIM        LOWER AND UPPER LIMITS FOR INPUT NERDAS DATA.
0038 C          DTHEI      INPUT VALUE OF RESOLUTION ANGLE, OPERATOR SETS.
0039 C          DTHETA     RADIAN VALUE OF " " USED IN EXECUTION.
0040 C          DTHETA     DEFAULT VALUE OF RESOLUTION ANGLE, BAND DEPENDENT.
0041 C          FCAL        CALIBRATION TONE FREQUENCIES, BAND DEPENDENT.
0042 C          FDOP        DOPPLER CENTER FREQUENCY FOR FILTER.
0043 C          FNOIZ       NOISE FILTER FREQUENCY, BAND DEPENDENT.
0044 C          FPI3L       CONSTANT, BAND DEPENDENT, (4*PI)**3/LAMDA**2.
0045 C          GSQ         GAIN VALUE IN DB, ANTENNA ANGLE RELATED.
0046 C          IAUTP       FLAG SIGNIFYING AUTO-STOP IF SET TO VALUE OF 2.
0047 C          IAUTR      " " " -START WHEN SET TO 2.
0048 C          IBCD        BUFFER USED WHEN CONVERTING TO BINARY-CODED DECIMAL.
0049 C          IBFLW       MESSAGE BUFFER OF OVER-FLOW OF OUTPUT BUFFER.
0050 C          IBL         ALLOWABLE BUFFER LENGTH, SET TO 70.
0051 C          IBND        BAND IDENTIFIER, L-BAND=1, C-BAND=2.

```


0052	C	IBOVF	FLAG USED TO SIGNAL BUFFER OVERFLOW.
0053	C	IBPT	POINTER FOR POSITIONING DATA IN OUTPUT BUFFER.
0054	C	ICALW	MESSAGE BUFFER FOR CALIBRATION WARNING.
0055	C	ICLC	CHARACTER BUFFER FOR 'L', 'C'.
0056	C	ICNT	GENERAL PURPOSE INTEGER VARIABLE.
0057	C	IEFLG	POINTER USED TO GET ERROR MESSAGES FOR I/O.
0058	C	IEOC	FLAG USED TO SIGNAL END OF INTEGRATION, CZT BOARD.
0059	C	IERR	FLAG USED TO SIGNAL I/O ERRORS.
0060	C	IFAERR	ERROR MESSAGE BUFFER.
0061	C	IFATAL	" " "
0062	C	IFCAL	POINTER TO CALIBRATION TONE FILTER.
0063	C	IFDL	LEFT FILTER POINTER FOR EACH ANGLE.
0064	C	IGTBC	GAIN TABLE FOR C-BAND ANTENNA, INTEGERS.
0065	C	IGTBL	" " " L-BAND " "
0066	C	ILC	CHARACTER BUFFER FOR 'L', 'C'.
0067	C	IMSK	MASK BUFFER FOR SETTING INTERRUPTS
0068	C	INERD	CHARACTER BUFFER FOR NERDAS I/O.
0069	C	INZ	INITIALIZATION FLAG.
0070	C	INZERR	MESSAGE BUFFER FOR NERDAS WARNING.
0071	C	IOBUF	GENERAL PURPOSE I/O BUFFER.
0072	C	IOFLG	BUFFER OF FLAG WORDS USED IN OUTPUT DATA SET.
0073	C	IOUT	OUTPUT DATA BUFFER.
0074	C	IOVER	FLAGS USED TO SIGNAL OVER-RIDE CONDITIONS.
0075	C	IDVRQ	MESSAGE BUFFERS USED IN ACQUIRING OVER-RIDE VALUES.
0076	C	IPLZ	CHARACTER BUFFER USED TO IDENTIFY POLARIZATION.
0077	C	IPOLZ	POLARIZATION IDENTIFIERS.
0078	C	IPSD	POWER SPECTRAL DATA BUFFER, 512 INTEGER*4.
0079	C	IRESP	GENERAL PURPOSE I/O BUFFER.
0080	C	IWAIT	FLAG TO SIGNAL COMPLETION OF SIGMA1 TASKS.
0081	C	IYN	CHARACTER BUFFER.
0082	C	I, J, K	DO LOOP INDEXES.
0083	C	KESC	CHARACTER, 'ESCAPE'.
0084	C	KXP	BINARY POWER BUFFER.
0085	C	LANT11	L-BAND ANTENNA GAIN BUFFER.
0086	C	LANT12	" " " "
0087	C	LANT13	" " " "
0088	C	LANT14	" " " "
0089	C	LANT21	" " " "
0090	C	LANT22	" " " "
0091	C	LANT23	" " " "
0092	C	LANT24	" " " "
0093	C	LT	LOAD POINTER FOR OUTPUT BUFFER.
0094	C	MX	OUTPUT POINTER FOR DATA BEING SENT OUT.
0095	C	NC	INTEGER CONSTANT TABLE.
0096	C	NFEL	NUMBER OF FILTER ELEMENTS TO SUM.
0097	C	NPASS	FLAG TO SIGNAL PASS NUMBER.
0098	C	NPZN	POLARIZATION IDENTIFIER NUMBER.
0099	C	NZE	NOISE FILTER ELEMENT COUNTER.
0100	C	NZL	LEFT MOST NOISE FILTER ELEMENT.
0101	C	NZPAS	PASS COUNTER USED FOR CHECKING NERDAS DATA.
0102	C	PARM	BUFFER OF NERDAS AIRCRAFT DATA PARAMETERS.

```

0103 C      PC      SUBPARAMETER USED INT EVALUATING SIGMAO.
0104 C      PN      " " " " " "
0105 C      PR      " " " " " "
0106 C      PC=CALIB POWER, PN=NOISE POWER, PR=RECEIVE POWER
0107 C      PREV    BUFFER TO SAVE PREVIOUS VALUES OF AIRCRAFT DATA
0108 C      RCAL    CALIBRATION FREQUENCY USED, BAND DEPENDENT.
0109 C      RG4     RANGE VALUE IN DB
0110 C      SIGMA   SIGMA-O VALUES FOR EACH VIEWING ANGLE.
0111 C      SLMDA   WAVE
0112 C      START  TIME IN SECONDS AT WHICH DATA SET STARTS.
0113 C      SUM     POWER SPECTRAL SUM OVER FILTER SET FOR ANGLE.
0114 C      SYSK   SYSTEM PROCESSING CONSTANT.
0115 C      TO     TIME IN SECONDS AT WHICH DATA SET STARTS.
0116 C      T1     CURRENT NERDAS TIME IN SECONDS.
0117 C      T2     VARIABLE USED IN AREA CALCULATION.
0118 C      T3     " " " " " "
0119 C      TANTL   VARIABLE USED IN AREA CALCULATION.
0120 C      TC     MICRO-PROCESSOR CYCLE TIME IN SECONDS.
0121 C      TDEL   ELAPSED TIME COUNTER.
0122 C      THETA  DESIRED VIEWING ANGLES, RADIANS.
0123 C      TI     INTEGRATION TIME, SECONDS.
0124 C      TM1    AREA CALCULATION VARIABLE.
0125 C      XDEL   FILTER LINE WIDTH, BAND DEPENDENT.
0126 C      XI     X-COORD. OF GROUND CELL POSITION.
0127 C      XK1    AREA CALCULATION VARIABLE.
0128 C      XK12   " " " "
0129 C      XK2    " " " "
0130 C      XLMDA  WAVE-LENGTH, BAND DEPENDENT.
0131 C      XSYSK  SYSTEM CONSTANT VALUES, BAND/POLARIZATION DEP.
0132 C      XI     VARIABLE USED IN EVALUATION OF CELL LOCATION.
0133 C      YI     Y-COORD. OF GROUND CELL POSITION.
0134 C      ZW     ROLL-OFF VALUE IN DB.
0135 C
0136 C

```

NO END CARD

SUBROUTINE

#F4END

PROGRAM ALLOCATION

TI-980 FORTRAN V4L1

PROGRAM END

COMPILER MEMORY USED = 6960

THERE ARE 0001 ERRORS IN THIS COMPILATION.


```

C
26 C   ::: VIEWING ANGLES (5,10,15,20,25,30,40,50) IN RADIANS
      DATA THETA/0.0873,0.1745,0.2618,0.3491,
      &          0.4363,0.5236,0.6091,0.6727/
C
27 C   DATA CIYN/'YN'/,CILC/'LC'/,CIPOLZ/'VV','VH','HH','HV'/,
      & KXP/1,2,4,8,16/,IMSK/#OFEH/,
      & NC/5,10,11,13,14,20,30,40/
C
C C C C   ::: ANTENNA GAIN TABLE FOR 1.6GHZ :::, EQUIVALENCED TO IGTBL :::
C C C C   VALUES ARE ORDERED SETS, 80 BY 4 MATRIX, -70 DEG TO +10 DEG EACH
C
28 C   *** 1.6VV :::
      DATA LANT11/147,152,154,155,157,161,164,169,174,177,
      &          179,183,187,191,194,197,199,203,206,211,
      &          216,218,219,219,219,222,224,226,227,228,
      &          229,227,226,227,229,227,224,223,222,221/
29 C   DATA LANT21/219,217,214,214,214,213,210,210,209,208,
      &          207,211,214,214,214,216,217,219,221,221,
      &          222,222,222,222,221,221,219,217,214,213,
      &          212,211,209,208,207,204,201,199,197,194/
C
30 C   *** 1.6VH :::
      DATA LANT12/214,218,222,224,227,229,232,236,239,240,
      &          241,243,246,247,249,250,251,255,259,260,
      &          261,261,262,262,262,263,264,263,262,262,
      &          262,264,256,256,256,251,246,244,242,249/
31 C   DATA LANT22/236,233,229,226,222,219,216,213,209,215,
      &          201,201,201,199,197,195,194,195,197,198,
      &          199,200,201,205,209,209,209,211,214,216,
      &          217,219,222,222,222,222,222,221,219,216/
C
32 C   *** 1.6HH :::
      DATA LANT13/231,234,237,239,242,243,244,244,244,244,
      &          244,245,246,246,246,246,246,248,249,248,
      &          246,245,244,244,244,244,244,242,239,238,
      &          237,234,231,230,229,226,224,222,219,217/
33 C   DATA LANT23/214,211,209,206,204,202,199,195,191,190,
      &          189,188,186,185,184,184,184,184,184,186,
      &          189,189,189,193,197,197,197,197,197,197,
      &          197,197,197,193,189,188,187,184,181,179/
C
34 C   *** 1.6HV :::
      DATA LANT14/164,168,172,173,174,175,177,179,182,183,
      &          184,185,187,191,194,195,196,197,199,200,
      &          201,201,202,203,204,204,204,204,204,204,
      &          204,205,206,205,204,203,202,202,201,201/
35 C   DATA LANT24/202,201,199,199,199,199,199,196,194,194,
      &          194,193,192,193,194,194,194,197,199,200,
      &          202,203,204,205,207,207,207,206,204,203,
      &          202,200,197,193,189,186,184,180,176,175/
C
C C C C   *** 4.75GHZ ANTENNA GAIN :::, EQUIVALENCED TO IGTBC :::
C C C C   ORDERED SET, 31 BY 4, 0 DEG TO 60 DEG
C
36 C   ::: 4.75VV :::
      DATA CANT1/300,310,320,330,340,350,360,365,370,375,
      &          380,385,380,370,360,370,380,380,380,390,
      &          400,400,390,380,370,360,350,330,310,290,270/
C
37 C   ::: 4.75VH :::
      DATA CANT2/300,318,336,354,372,390,397,404,411,418,
      &          420,425,425,420,420,420,420,420,420,420,
      &          420,418,411,404,390,380,370,360,350,330,310/
C
38 C   ::: 4.75HH :::
      DATA CANT3/330,340,350,360,370,380,390,400,410,415,
      &          420,423,427,431,433,435,437,435,435,430,
      &          425,420,415,410,400,390,380,370,360,350,340/
C
39 C   ::: 4.75HV :::
      DATA CANT4/315,322,330,345,360,385,395,400,395,405,
      &          410,410,410,410,410,410,410,410,405,405,
      &          390,405,395,400,395,385,360,345,330,322,315/
C
40 C   DATA CFATAL/'W0020','W0430','EF600','EF710','W0790',

```

```

      &          'W0815','W0850','W0864','W0868','UNDEF'/
41  C          DATA CFAERR/'FATAL ERR','**NERD ERR','**BUFF OVR','**CAL LOW'/
42  C          DATA CCLC/' L',' C'/,CST/1.25,2.0,57.3,10.0,40.0,0.5,573.,0.3048,0.514/
43  C          DATA CL/7.71094,-21.6914,14.941,1.11182,-0.0186787,1000./
44  C          DATA CC/5.4637,9.61506,-15.2809,3.90396,-1.05396,3.2055,-0.305786,
      &          18.5575,-16.5981,8.64097,-3.44443,0.653213,-0.25549,0.001/
C C C *****INTERCOM*****
C C C *** INITIALIZE I/O BOARD FOR CRT/KEY
45  C          CALL IUSRT
C C C ::: INITIALIZE BI-PHASE-L I/O BOARD, CZT BOARD, AND INTERRUPT CONTROLLER
46  C          CALL INIPIO
47  C          CALL INICZT(IPSD)
48  C          CALL INI259
49  C          CALL LDJMPS
C C C *** SEND OUT SIGN-ON MESSAGE
50  C          5 CALL CRLF
51  C             IEFLG=1
52  C          WRITE(CIOBUF,21,ERR=7000)
53  C          21 FORMAT(10X,'NASA L/C-BAND PROCESSOR',17X)
54  C          CALL MSGOUT(NC(8),IOBUF)
55  C          CALL CRLF
C C C *** INITIALIZE FLAG SET *****
56  C          INZ=1
C C C ***** SET OUTPUT BUFFER LINE LIMIT *****
57  C          IBL=70
C C C *** ** * ** * ** RE-CYCLE LOOP ENTRY ** * **
58  C          40 CONTINUE
59  C             CALL IBELL
60  C             CALL CRLF
61  C             NPASS=1
62  C             KESC=1
C C C ::: SET STARTING POINTERS INTO OUTPUT BUFFER :::
63  C             MX=IBL-10
64  C             LT=MOD(MX,IBL)+1
C C C ***** PRE-LOAD OUTPUT BUFFER W/099H *****
65  C             ICNT=25
66  C             CALL IBFILL(ICNT,IOUT(1,1))
C C C *** ** * ** * ** SET UP OPTIONS ** * **
67  C             CALL CRLF
68  C             GO TO (115,85),INZ
69  C          85 CONTINUE
70  C             CRESP=' NEW SET-UP(Y/N)?'
71  C             CALL IBELL
72  C             CALL MSGOUT(NC(7),IRESP)
73  C             CALL KEYBRD(NC(1),IRESP)
74  C             IF(IRESP(1).EQ.#1BH) GO TO 5
75  C             IF(IRESP(1).EQ.IYN(2))GO TO 740
C C C *****SET BAND IDENTIFIER*****
76  C          115 CONTINUE
77  C             IBND=1
78  C             CRESP=' TYPE OF SCAT(L/C)?'
79  C             CALL MSGOUT(NC(7),IRESP)
80  C             CALL KEYBRD(NC(1),IRESP)
81  C             IF(IRESP(1).EQ.ILC(2))IBND=2
C C C *****SET BAND RELATED VARIABLES*****
82  C             BMW=BMWID(IBND)/CST(3)
83  C             SLMDA=XLMDA(IBND)

```

```

84      DELF=XDELFF( IBND)
85      NZL=IFIX(FNOIZ( IBND)/DELF)-6/IBND+1
86      NZE=( 6/IBND)*2
      C
      C ***** SET POLARIZATION IDENTIFIER *****
160     NPZN=0
      CRESP='          POLARITY(UV,UH,HH,HV)?'
88     CALL MSGOUT( NC(7), IRESP)
89     CALL KEYBRD( NC(1), IRESP)
90     DO 180 L=1,4
91     IF( IPLZ.EQ. IPOLZ( L)) NPZN=L
92     180 CONTINUE
93     IF( NPZN.EQ.0) GO TO 160
94
      C
      C ***** SET POLARIZATION VARIABLES *****
95     RCAL=FCAL( IBND)
96     GO TO (200,195), IBND
97     195 GO TO (196,196,200,200), NPZN
98     196 RCAL=FCAL(3)
      C
      C ***** ANGULAR RESOLUTION OPTION *****
99     200 IFCAL=512-IFIX( RCAL/DELF)-6/IBND+1
100    203 CRESP='          ANG RESOL O/RIDE(Y/N)?'
101    CALL MSGOUT( NC(7), IRESP)
102    CALL KEYBRD( NC(1), IRESP)
103    IOFLG(5)=1
104    IF( IRESP(1).EQ. IYN(2)) GO TO 255
      C
      C ***** READ OVER-RIDE RESOLUTION *****
105    IOFLG(5)=2
106    CALL GETVLK( CRESP, IRESP, NC(2), NC(6), IERR, DTHEI)
107    IF( IERR.NE.0) GO TO 203
108    GO TO 260
109    255 DTHEI=DTHETA( IBND)
110    C !!! SET VALUE OF DTHETA/2, TO RADIANs !!!
110    260 DTHETA=( DTHEI/CST(3))/CST(2)
      C
      C ***** SYSTEM CONSTANT OPTION *****
111    IOFLG(3)=0
112    305 CRESP='          SYS CONST O/RIDE(Y/N)?'
113    CALL MSGOUT( NC(7), IRESP)
114    CALL KEYBRD( NC(1), IRESP)
115    IF( IRESP(1).EQ. IYN(2)) GO TO 380
116    IOFLG(3)=1
      C
      C ***** RESET SYS CONSTANT *****
117    CALL GETVLK( CRESP, IRESP, NC(2), NC(6), IERR, SYSK)
118    IF( IERR.NE.0) GO TO 305
119    IF( SYSK.GT.0. ) SYSK=-SYSK
120    GO TO 385
121    380 SYSK=XSYSK( NPZN, IBND)
      C
      C ***** NERDAS VALUE OVER-RIDE OPTION *****
122    385 CONTINUE
123    CRESP='          ACFT DATA O/RIDES(Y/N)?'
124    CALL MSGOUT( NC(7), IRESP)
125    CALL KEYBRD( NC(1), IRESP)
126    IOFLG(4)=0
127    DO 415 J=1,5
128    IOVER( J)=0
129    415 CONTINUE
130    IF( IRESP(1).EQ. IYN(2)) GO TO 500
      C
      C ***** AIRCRAFT DATA OVER-RIDE ENTRIES *****
131    DO 475 J=1,5
132    IEFLG=2
133    WRITE( CRESP, 430, ERR=7000) CIOVRQ( J)
134    430 FORMAT(5X, 'O/RIDING ', 1A5, ' (Y/N)?', 4X)
135    CALL MSGOUT( NC(7), IRESP)
136    CALL KEYBRD( NC(1), IRESP)
137    IF( IRESP(1).EQ. IYN(2)) GO TO 473
138    460 CALL GETVLK( CRESP, IRESP, NC(2), NC(6), IERR, ACOVR( J))

```

```

139      IF(IERR.NE.0) GO TO 460
140      IOVER(J)=1
141      473  IOFLG(4)=IOFLG(4)+IOVER(J)*KXP(J)
142      475  CONTINUE
C
C      ***** INITIALIZE PARM VALUES *****
143      DO 490 J=1,5
144      IF(IOVER(J).EQ.0)GO TO 490
145      GO TO (481,482,482,482,483),J
146      481  PARM(J)=ACDVR(J)*CST(8)
147      GO TO 490
148      482  PARM(J)=ACDVR(J)/CST(3)
149      GO TO 490
150      483  PARM(J)=ACDVR(J)*CST(9)
151      490  CONTINUE
C
C      ***** START/STOP TIME OPTIONS *****
152      500  GO TO (545,520), INZ
153      520  CONTINUE
154      CRESP='      CHG START TIME(Y/N)?'
155      CALL MSGOUT(NC(7),IRESP)
156      CALL KEYBRD(NC(1),IRESP)
157      IF(IRESP(1).EQ.IYN(2)) GO TO 625
158      GO TO 570
C
C      ***** START TIME OPTION *****
159      545  CONTINUE
160      CRESP='      START TIME CMD(Y/N)?'
161      CALL MSGOUT(NC(7),IRESP)
162      CALL KEYBRD(NC(1),IRESP)
163      IF(IRESP(1).EQ.IYN(2)) GO TO 620
C
C      ***** SET START TIME *****
164      570  CONTINUE
165      CRESP='      SET START TIME(MHMMSS)'
166      CALL MSGOUT(NC(7),IRESP)
167      CALL KEYBRD(NC(2),IRESP)
168      CALL ASCDV(NC(2),IRESP)
169      IEFLG=3
170      READ(CRESP,600,ERR=570,END=7000)(START(J),J=1,3)
171      600  FORMAT(3F2.0)
172      IAUTR=2
173      TO=START(1)*3600.+START(2)*60.+START(3)
174      GO TO 625
175      620  IAUTR=1
C
C      ***** STOP TIME CHANGE OPTIONS *****
176      625  GO TO (655,630),INZ
177      630  CONTINUE
178      CRESP='      CHG STOP TIME(Y/N)?'
179      CALL MSGOUT(NC(7),IRESP)
180      CALL KEYBRD(NC(1),IRESP)
181      IF(IRESP(1).EQ.IYN(2)) GO TO 740
182      GO TO 680
C
C      ***** STOP TIME OPTION *****
183      655  CONTINUE
184      CRESP='      STOP TIME CMD(Y/N)?'
185      CALL MSGOUT(NC(7),IRESP)
186      CALL KEYBRD(NC(1),IRESP)
187      IF(IRESP(1).EQ.IYN(2)) GO TO 730
C
C      ***** SET NUMBER OF SECONDS TO RUN *****
188      680  CONTINUE
189      CRESP='      SET RUN TIME,SECS(F5.0)'
190      CALL MSGOUT(NC(7),IRESP)
191      CALL KEYBRD(NC(2),IRESP)
192      CALL ASCDV(NC(1),IRESP)
193      IEFLG=4
194      READ(CRESP,710,ERR=680,END=7000)DELT
195      710  FORMAT(F5.0)
196      IAUTP=2

```



```

245  C 925 CONTINUE
      C
246  C *****READ IN A NERDAS FRAME****
      C CALL NERD(J,IOUT(1,LT))
      C
      C *****DECODE A/C DATA; T1,ALT,DRF,ROL,PCH,VEL
      C ::: NOTE::: PARM(1), . . . , (5) = ALT,DRF,ROL,PCH,VEL :::
247  C
248  C CALL IBELL
249  C CALL UNPACK(IOUT(1,LT),INERD)
250  C CALL DECODA(IOVER,INERD,T1,PARM)
251  C GO TO (930,940),IAUTR
252  C 930 GO TO (935,945),NPASS
253  C 935 TO=T1
254  C 940 IF(T1,LT,TO) GO TO 925
255  C 945 CALL AMSUB(TDEL,T1,TO)
256  C IF(NZPAS,GE,10) GO TO 950
257  C CALL VALID(DLIM,DFALT,IOVER,IOFLG(1),PARM)
258  C NZPAS=NZPAS+1
259  C GO TO 955
260  C 950 CALL RUNLMT(PREV,IOVER,PARM)
261  C 955 DO 960 K=1,5
262  C PREV(K)=PARM(K)
263  C 960 CONTINUE
264  C ::: :::: CHECK VALIDITY OF NERDAS DATA :::
      C IF(IOFLG(1),EQ,0) GO TO 965
      C IF(NZPAS,GE,10) GO TO 965
      C ***** SEND OUT NERDAS WARNING MESSAGE *****
265  C
266  C CALL CRLF
267  C CALL IBELL
268  C CALL MSGOUT(NC(2),INZERR)
269  C ***** SKIP TIME CHECK WHEN NERDAS IN ERROR *****
270  C GO TO 977
271  C 965 GO TO (977,975), IAUTP
272  C 975 IF(DELT,LE,TDEL) GO TO 7777
273  C 977 GO TO (1000,7777),KESC
274  C 1000 DO 1005 I=1,8
275  C ***** EVAL CELL SIZES (DOTHER=DTHETA/2, IN RAD)
276  C CALL AMDADD(SUM,THETA(I),DOTHER)
277  C CALL AMSUB(DIFF,THETA(I),DOTHER)
278  C CALL CCELL(CELL(I),DIFF,SUM,PARM(1))
279  C 1005 CONTINUE
280  C
281  C ***** COMPUTE INTEGRATION TIME *****
282  C CALL AMDIV(TI,CELL(1),PARM(5))
283  C ***** SET INTERRUPT MASK FOR LEVEL 0 *****
284  C
285  C 1030 CALL MSKSET(IMSK)
286  C
287  C ***** START INTEGRATION *****
288  C CALL INTGT(ICNT,TI,DELF)
289  C CALL CZT(ICNT)
290  C IEOC=1
291  C IBOVF=1
292  C IWAIT=1
293  C IOFLG(2)=0
294  C ::: SET ICNT = NBR OF CELLS IN VIEW :::
295  C CALL CELCNT(ICNT,PARM(5),TC,PARM(1),THETA(8))
296  C IF(ICNT,LE,IBL)GO TO 1050
297  C IBOVF=2
298  C ICNT=IBL
299  C IOFLG(2)=IOFLG(2),OR,KXP(3)
300  C GO TO 1060
301  C 1050 IOFLG(2)=IOFLG(2),AND,NC(3)
302  C 1060 GO TO (1100,1065),IBOVF
303  C 1065 CALL CRLF
304  C CALL MSGOUT(NC(2),IBFLW)
305  C 1100 CONTINUE
306  C ***** SET VARIABLES PC & PN FOR POINTER EVALUATIONS ::::

```

```

296 = CALL PCPN(FN, TI, PC, TC, PARM(1), PARM(5))
    C
    C ***** COMPUTE A/C RELATED VALUES *****
297 = AROL=ABS(PARM(3))
    C ***** COMPUTE ANGLE RELATED VALUES *****
298 = DO 1305 I=1,8
299 = ANGL=-THETA(I)
300 = IF(AROL.GT.THETA(I))ANGL=-AROL
301 = CALL GXT(XT, PARM(3), ANGL, PARM(1))
302 = CALL GXI(XI, XT, PARM(1), PARM(3), PARM(2))
303 = CALL GYI(YI, XT, PARM(1), PARM(3), PARM(2))
304 = CALL AMDIV(XT, CELL(1), CST(2))
305 = IF(XI.LT.XT)XI=XT
    C
    C ***** COMPUTE DOPPLER ANGLE AND NBR OF FILTER ELEMENTS *****
306 = CALL CNFILT(NFEL(I), DELF, BNDW, CELL(I), FDOP, ANGL, YI, XI, XT, SLMDA,
    & PARM(5), PARM(1))
307 = IF(NFEL(I), LT, 1)NFEL(I)=1
    C
    C ***** EVALUATE TRUE BANDWIDTH *****
308 = CALL CBNDW(BNDW, NFEL(I), DELF)
309 = IF(MOD(NFEL(I), 2), EQ, 0) GO TO 1200
    C ::: ODD NBR OF ELEMENTS :::
310 = L=2
311 = CALL FIDOD(IFDL(I), NFEL(I), L, FDOP, DELF)
312 = GO TO 1210
    C ::: EVEN NBR OF ELEMENTS :::
313 = 1200 L=3
314 = CALL FIDLEV(IFDL(I), NFEL(I), L, FDOP, DELF)
315 = 1210 CONTINUE
316 = IF(IFDL(I), LE, 0)IFDL(I)=1
    C
    C ***** COMPUTE TRUE DOPPLER ANGLE AND LOAD POINTERS *****
317 = CALL AMDIV(ANGL, FDOP, XT)
318 = IF(ANGL.GT.1.)ANGL=1.
319 = CALL CLNPT(IBPT(I), LT, PC, FN, XT, YI, ANGL, PARM(1))
320 = IF(IBPT(I), LE, 0)IBPT(I)=IBL+IBPT(I)
    C ***** ESTABLISH OUTPUT LOOK ANGLE *****
321 = ANGL=THETA(I)
322 = IF(ANGL.LT.AROL)ANGL=AROL
323 = CALL ANDMUL(ANGT(I), ANGL, CST(3))
    C
    C ***** ESTABLISH ANTENNA VIEWING ANGLE *****
324 = ANGL=-ANGTL
    C
325 = SIGMA(I)=0.0
326 = 1305 CONTINUE
    C
327 = 1310 IF(IEOC.NE.1) GO TO 1650
328 = IWAIT=2
329 = GO TO 1310
    C
    C ***** END OF SIGMA1 MODULE *****
    C ***** SIGMA2 *****
330 = $INCLUDE( :F1: SIGMA2.SRC )
    C ***** SIGMA2 MODULE *****
    C ***** ENTRY CONDITIONAL ON IEOC FLAG SET BY CZT-BOARD INTERRUPT
    C ***** CZTINT ROUTINE WILL HAVE MASKED ALL INTERRUPTS *****
331 = 1650 IF(IWAIT, EQ, 1)GO TO 1657

```

```

332 * C SET ALARM WORD BIT FOR TIME
333 * IOFLG(2)=IOFLG(2).OR.KXP(2)
333 * GO TO 1660
334 * C*****
334 * 1657 IOFLG(2)=IOFLG(2).AND.NC(4)
335 * C READ IN CZT-BOARD DATA VIA DMA
335 * 1660 CALL CZTR
336 * C
336 * C EVALUATE NOISE POWER
336 * CALL I32SUM(PN,IPSD(1,NZL),NZE)
337 * C EVALUATE CALIBRATION POWER
337 * CALL I32SUM(PC,IPSD(1,IFCAL),NZE)
338 * C CHECK FOR CALIB TONE >> NOISE
338 * CALL AHDSUB(XT,PC,PN)
339 * IF(XT,GE,CC(1,1))GO TO 1689
340 * IOFLG(2)=IOFLG(2).OR.KXP(1)
341 * CALL CRLF
342 * CALL MSGOUT(NC(2),ICALW)
343 * GO TO 1690
344 * 1689 IOFLG(2)=IOFLG(2).AND.NC(5)
345 * C PERFORM POWER SUMS OVER EACH BAND
345 * C AND CALCULATE SIGMA VALUES
346 * 1690 DO 1700 I=1,8
346 * CALL I32SUM(PR,IPSD(1,IFDL(I)),NFEL(I))
347 * CALL AHDSUB(XT,PR,PC)
348 * CALL AHDAIDK(SIGMA(I),SIGMA(I),XT)
349 * 1700 CONTINUE
350 * C***** MOVE ANGLES,SIGMA,NOISE & CAL-PWR,& FLAGS TO OUTPUT *****
350 * ICNT=1
351 * DO 1800 K=1,8
352 * I=55+(K-1)*2
353 * CALL MFPNUM(ANGT(K),I,LT,IOUT,IBCD,ICNT)
354 * 1800 CONTINUE
355 * ICNT=8
356 * I=71
357 * 1900 CALL MFPNUM(SIGMA,I,IBPT,IOUT,IBCD,ICNT)
358 * I=87
359 * ICNT=1
360 * CALL MFPNUM(PC,I,LT,IOUT,IBCD,ICNT)
361 * I=89
362 * CALL MFPNUM(PN,I,LT,IOUT,IBCD,ICNT)
363 * I=91
364 * CALL MFLAG(I,IOFLG,IOUT(I,LT),IBND,NPZN)
365 * C***** SET OUTPUT POINTER *****
365 * ICNT=LT-IBL/2
366 * IF(ICNT.LE.0)ICNT=ICNT+IBL
367 * C
367 * C ::::::::::: REPLACE SYNC BYTE IN OUTPUT LINE :::::::::::
368 * IOUT(1,ICNT)=#OFBH
368 * IOUT(2,ICNT)=#OF1H
369 * C
369 * C ::::::::::: WRITE TO BI-PHASE-L :::::::::::
369 * C
369 * CALL KBPHAL(IOUT(1,ICNT))
370 * C***** SET LOOP CONTROLS *****
370 * NPASS=2
371 * C :: SET LOAD POINTER FOR NEXT LINE ::
371 * MX=MX+1
372 * IF(MX.LE.0)MX=LT
373 * LT=MOD(MX,IBL)+1
373 * C ::: LOOP BACK TO NEXT FRAME :::
374 * C
374 * GO TO 915
375 * C***** READ - WRITE ERROR RECOVERY *****
375 * C
375 * C 7000 CALL CRLF
376 * CALL MSGOUT(NC(2),IFAERR)
377 * CALL MSGOUT(NC(1),IFATAL(1,IEFLG))
378 * CALL CRLF

```


CROSS-REFERENCE LISTING

DEFN	ADDR	SIZE	NAME, ATTRIBUTES, AND REFERENCES
		1248	COMMON-BLOCK 16 17 18
272	0E28H	1000	LABEL 271 272
276	0E7BH	1005	LABEL 272 276
278	0E92H	1030	LABEL 278
291	0F01H	1050	LABEL 286 291
292	0F0BH	1060	LABEL 290 292
293	0F1AH	1065	LABEL 292 293
295	0F26H	1100	LABEL 292 295
76	0352H	115	LABEL 68 76
313	10ABH	1200	LABEL 309 313
315	10D0H	1210	LABEL 312 315
326	11BAH	1305	LABEL 298 326
327	11C4H	1310	LABEL 327 329
87	03ECH	160	LABEL 87 94
331	11DBH	1650	LABEL 327 331
334	11FAH	1657	LABEL 331 334
335	1204H	1660	LABEL 333 335
344	126BH	1689	LABEL 339 344
345	1275H	1690	LABEL 343 345
349	12D1H	1700	LABEL 345 349
93	0440H	180	LABEL 91 93
354	131BH	1800	LABEL

		351 354
357 1331H	1900	LABEL 357
97 0474H	195	LABEL 96 97
98 0484H	196	LABEL 97 98
99 048DH	200	LABEL 96 97 99
100 04B5H	203	LABEL 100 107
53 017AH	21	LABEL 52 53
109 0520H	255	LABEL 104 109
110 052FH	260	LABEL 108 110
112 054CH	305	LABEL 112 118
121 05CDH	380	LABEL 115 121
122 05E3H	385	LABEL 120 122
58 02CEH	40	LABEL 58 238 385
129 062BH	415	LABEL 127 129
134 019DH	430	LABEL 133 134
138 06RDH	460	LABEL 138 139
141 06F9H	473	LABEL 137 141
142 0715H	475	LABEL 131 142
146 075AH	481	LABEL 145 146
148 077CH	482	LABEL 145 148
150 079EH	483	LABEL 145 150
151 07BDH	490	LABEL 143 144 147 149 151
50 027BH	5	LABEL 50 74
152 07C7H	500	LABEL 130 152
153 07D3H	520	LABEL

		152 153
159 0808H	545	LABEL 152 159
164 083AH	570	LABEL 158 164 170
171 018EH	600	LABEL 170 171
175 090EH	620	LABEL 163 175
176 0913H	625	LABEL 157 174 176
177 091FH	630	LABEL 176 177
183 0954H	655	LABEL 176 183
188 0986H	680	LABEL 182 188 194
375 144AH	7000	LABEL 52 133 170 194 206 211 218 224 230 375
195 01C5H	710	LABEL 194 195
198 0A04H	730	LABEL 187 198
199 0A09H	740	LABEL 75 181 197 199
381 1488H	777	LABEL 381 386
380 146EH	7777	LABEL 270 271 380
207 01CRH	790	LABEL 206 207
212 01F4H	815	LABEL 211 212
69 0312H	85	LABEL 68 69
219 0219H	850	LABEL 218 219
221 0B9AH	860	LABEL 215 216 221
223 0BB3H	862	LABEL 222 223
225 022EH	864	LABEL 224 225
228 0C25H	865	LABEL 222 228
229 0C34H	866	LABEL 228 229
231 024BH	868	LABEL

			230	231				
234	OC82H	870	LABEL					
			203	228	234			
244	OC02H	915	LABEL					
			244	374				
245	OC05H	925	LABEL					
			245	253				
251	OD23H	930	LABEL					
			250	251				
252	OD32H	935	LABEL					
			251	252				
253	OD3BH	940	LABEL					
			250	253				
254	OD4DH	945	LABEL					
			251	254				
257	OD8BH	950	LABEL					
			255	259				
260	OD95H	955	LABEL					
			258	260				
262	ODBBH	960	LABEL					
			260	262				
269	ODF8H	965	LABEL					
			263	264	269			
270	OE07H	975	LABEL					
			269	270				
271	OE19H	977	LABEL					
			268	269	271			
2C8EH	18	@IOPB	INTEGER*2 DIMENSIONED					
			52					
	3	A	COMMON-BLOCK					
			18 19					
		ARS	INTRINSIC					
			297					
0060H	20	ACDVR	REAL*4 DIMENSIONED					
			2 138 146 148 150 218					
		AMDADD	EXTERNAL SUBROUTINE					
			273 348					
		AMDIV	EXTERNAL SUBROUTINE					
			277 304 317					
		AMDHUL	EXTERNAL SUBROUTINE					
			323					
		AMDSUB	EXTERNAL SUBROUTINE					
			254 274 338 347					
2CFEH	4	ANGL	REAL*4					
			299 300 301 306 317 318 319 324					
0040H	32	ANGT	REAL*4 DIMENSIONED					
			2 323 353					
2D16H	4	ANGTL	REAL*4					

			321 322 323 324
2CFAH	4	AROL	REAL*4 297 300 322
		ASCDV	EXTERNAL SUBROUTINE 168 192
2CA4H	4	EMW	REAL*4 82
0044H	8	EMWID	REAL*4 DIMENSIONED COMMON 2 16 20 82
2D0EH	4	ENDW	REAL*4 306 308
00EBH	62	CANT1	INTEGER*2 DIMENSIONED COMMON EQUIVALENCED 5 14 36
0129H	62	CANT2	INTEGER*2 DIMENSIONED COMMON EQUIVALENCED 5 14 37
0167H	62	CANT3	INTEGER*2 DIMENSIONED COMMON EQUIVALENCED 5 14 38
01A5H	62	CANT4	INTEGER*2 DIMENSIONED COMMON EQUIVALENCED 5 14 39
		CBNDW	EXTERNAL SUBROUTINE 308
0018H	56	CC	REAL*4 DIMENSIONED COMMON 17 44 339
		CCELL	EXTERNAL SUBROUTINE 275
04BDH	4	CCLC	CHARACTER*2 DIMENSIONED COMMON EQUIVALENCED 9 13 16 42 206
		CELCNT	EXTERNAL SUBROUTINE 285
0000H	32	CELL	REAL*4 DIMENSIONED 2 275 277 304 306
0495H	40	CFAERR	CHARACTER*10 DIMENSIONED COMMON EQUIVALENCED 10 13 16 41
0463H	50	CFATAL	CHARACTER*5 DIMENSIONED COMMON EQUIVALENCED 8 13 16 40
00D3H	2	CILC	CHARACTER*2 COMMON EQUIVALENCED 9 13 16 27
00C8H	60	CIOBUF	CHARACTER*60 EQUIVALENCED 12 13 52
00B8H	25	CIOVRQ	CHARACTER*5 DIMENSIONED COMMON EQUIVALENCED 8 13 16 25 133 218
00D5H	8	CIPOLZ	CHARACTER*2 DIMENSIONED COMMON EQUIVALENCED 9 13 16 27 206
00D1H	2	CIYN	CHARACTER*2 COMMON EQUIVALENCED 9 13 16 27
0000H	24	CL	REAL*4 DIMENSIONED COMMON 17 43
		CLNPT	EXTERNAL SUBROUTINE

```

319
0104H  60  CNERD  CHARACTER*60 EQUIVALENCED
          12  13
          CNFILT  EXTERNAL SUBROUTINE
          306
0104H  30  CRESP  CHARACTER*30 EQUIVALENCED
          11  13  70  78  88  100  106  112  117  123  133
          138  154  160  165  170  178  184  189  194  200  206
          211  218  224  230  235
          CRLF    EXTERNAL SUBROUTINE
          50  55  60  67  199  204  209  217  226  232  234
          265  293  341  375  378
0050H  36  CST    REAL*4 DIMENSIONED COMMON
          17  42  82  110  146  148  150  304  323
          CZT     EXTERNAL SUBROUTINE
          280
          CZTR    EXTERNAL SUBROUTINE
          335
          DECODA  EXTERNAL SUBROUTINE
          249
2CACH  4  DELF   REAL*4
          84  85  99  279  306  308  311  314
2CDOH  4  DELT   REAL*4
          194  230  270
0084H  20  DFALT  REAL*4 DIMENSIONED COMMON
          2  16  24  256
2CEAH  4  DIFF   REAL*4
          274  275
005CH  40  DLIM   REAL*4 DIMENSIONED COMMON
          2  16  23  256
2CBEH  4  DTHEI  REAL*4
          106  109  110  211
2CC2H  4  DOTHER REAL*4
          110  273  274
0008H  8  DTHETA  REAL*4 DIMENSIONED COMMON
          2  16  21  109
          DWAIT  EXTERNAL SUBROUTINE
          379
0038H  12  FCAL   REAL*4 DIMENSIONED COMMON
          2  16  23  95  98
          FDLEV  EXTERNAL SUBROUTINE
          314
          FILOD  EXTERNAL SUBROUTINE
          311
2D12H  4  FDOP   REAL*4
          306  311  314  317
004CH  8  FNOIZ  REAL*4 DIMENSIONED COMMON
          2  16  20  85
0054H  8  FPI3L  REAL*4 DIMENSIONED COMMON

```

```

                2  16  20
GETVLU          EXTERNAL SUBROUTINE
                106 117 138
116 GG          COMMON-BLOCK
                17
GXI             EXTERNAL SUBROUTINE
                302
GXT             EXTERNAL SUBROUTINE
                301
GYI             EXTERNAL SUBROUTINE
                303
2CD4H          2  I          INTEGER*2
                224 272 273 274 275 298 299 300 306 307 308
                309 311 314 316 319 320 321 323 325 345 346
                348 352 353 356 357 358 360 361 362 363 364
I32SUM          EXTERNAL SUBROUTINE
                336 337 346
244DH          1  IAUTP      INTEGER*1
                3  196 198 228 269
244CH          1  IAUTR      INTEGER*1
                3  172 175 222 250
2445H          4  IRCD       INTEGER*1 DIMENSIONED
                3  353 357 360 362
IBELL          EXTERNAL SUBROUTINE
                59  71 247 266
IRFILL         EXTERNAL SUBROUTINE
                66
04A9H          2  IBFLW      INTEGER*2 COMMON EQUIVALENCED
                13 294
244BH          1  IBL        INTEGER*1
                3  57  63  64 286 288 320 365 366 373 380
                382
2C56H          2  IRND       INTEGER*2
                7  77  81  82  83  84  85  86  95  96  99
                109 121 206 364
244EH          1  IBOVF      INTEGER*1
                3  282 287 292
2C7AH          16 IBPT       INTEGER*2 DIMENSIONED
                7  319 320 357
04B3H          2  ICALW      INTEGER*2 COMMON EQUIVALENCED
                13 342
04BDH          4  ICCLC      INTEGER*2 DIMENSIONED COMMON EQUIVALENCED
                7  13
2C58H          2  ICNT       INTEGER*2
                7  65  66 279 280 285 286 288 350 353 355
                357 359 360 362 365 366 367 368 369 380 381
                382 384 385
2C8CH          2  IEFLG      INTEGER*2
                51 132 169 193 205 210 214 223 229 377
0000H          2  IEOD       INTEGER*2 COMMON

```

			18 281 327
2CBCH	2	IERR	INTEGER*2 106 107 117 118 138 139
0495H	10	IFAERR	INTEGER*1 DIMENSIONED COMMON EQUIVALENCED 3 13 376
0463H	125	IFATAL	INTEGER*1 DIMENSIONED COMMON EQUIVALENCED 3 13 377
2CBAH	2	IFCAL	INTEGER*2 99 337
2C6AH	16	IFDL	INTEGER*2 DIMENSIONED 7 311 314 316 346
		IFIX	INTRINSIC 85 99
00EBH	248	IGTBC	INTEGER*2 DIMENSIONED COMMON EQUIVALENCED 6 14 16
01E3H	640	IGTBL	INTEGER*2 DIMENSIONED COMMON EQUIVALENCED 6 15 16
00D3H	2	ILC	INTEGER*1 DIMENSIONED COMMON EQUIVALENCED 3 13 81
00E2H	1	IMSK	INTEGER*1 COMMON 3 16 27 278
0104H	60	INERD	INTEGER*1 DIMENSIONED EQUIVALENCED 3 13 248 249
		INI259	EXTERNAL SUBROUTINE 48
		INICZT	EXTERNAL SUBROUTINE 47
		INIPIO	EXTERNAL SUBROUTINE 46
		INTGT	EXTERNAL SUBROUTINE 279
2CSAH	2	INZ	INTEGER*2 7 56 68 152 176 239
049FH	10	INZERR	INTEGER*1 DIMENSIONED COMMON EQUIVALENCED 3 13 267
00CBH	60	IOBUF	INTEGER*1 DIMENSIONED EQUIVALENCED 3 13 54
244FH	5	IOFLG	INTEGER*1 DIMENSIONED 3 103 105 111 116 126 141 256 263 284 289 291 332 334 340 344 364
0145H	8960	IOUT	INTEGER*1 DIMENSIONED 3 66 246 248 353 357 360 362 364 367 368 369 383
0140H	5	IOVER	INTEGER*1 DIMENSIONED 3 128 140 141 144 216 249 256 259
00BBH	25	IOVRQ	INTEGER*1 DIMENSIONED COMMON EQUIVALENCED 3 13
0104H	2	IPLZ	INTEGER*2 EQUIVALENCED 6 13 92

```

00D5H      8  IPOLZ      INTEGER*2 DIMENSIONED COMMON EQUIVALENCED
              6  13  92

2454H     2048  IPSD      INTEGER*2 DIMENSIONED
              6  47 336 337 346

0104H     30  IRESP      INTEGER*1 DIMENSIONED EQUIVALENCED
              3  13  72  73  74  75  79  80  81  89  90
              101 102 104 106 113 114 115 117 124 125 130
              135 136 137 138 155 156 157 161 162 163 166
              167 168 179 180 181 185 186 187 190 191 192
              201 202 203 208 213 220 227 233 236 237 238

              IUSRT      EXTERNAL SUBROUTINE
              45

2449H     1  IWAIT      INTEGER*1
              3 283 328 331

00D1H     2  IYN        INTEGER*1 DIMENSIONED COMMON EQUIVALENCED
              3  13  75 104 115 130 137 157 163 181 187
              203 238

2CCA8     2  J          INTEGER*2
              127 128 131 133 138 140 141 143 144 145 146
              148 150 170 215 216 218 243 246

2CE4H     2  K          INTEGER*2
              260 261 351 352 353 381 382 383

              KBPHAL     EXTERNAL SUBROUTINE
              369 383

0002H     1  KESC       INTEGER*1 COMMON
              3  18  62 271

              KEYBRD     EXTERNAL SUBROUTINE
              73  80  90 102 114 125 136 156 162 167 180
              186 191 202 237

              KEYCHK     EXTERNAL SUBROUTINE
              244

00DDH     5  KXP        INTEGER*1 DIMENSIONED COMMON
              3  16  27 141 289 332 340

2CB4H     2  L          INTEGER*2
              91  92 310 311 313 314

01E3H     80  LANT11     INTEGER*2 DIMENSIONED COMMON EQUIVALENCED
              4  15  28

0283H     80  LANT12     INTEGER*2 DIMENSIONED COMMON EQUIVALENCED
              4  15  30

0323H     80  LANT13     INTEGER*2 DIMENSIONED COMMON EQUIVALENCED
              4  15  32

03C3H     80  LANT14     INTEGER*2 DIMENSIONED COMMON EQUIVALENCED
              4  15  34

0233H     80  LANT21     INTEGER*2 DIMENSIONED COMMON EQUIVALENCED
              4  15  29

02D3H     80  LANT22     INTEGER*2 DIMENSIONED COMMON EQUIVALENCED
              4  15  31

0373H     80  LANT23     INTEGER*2 DIMENSIONED COMMON EQUIVALENCED
              4  15  33

0413H     80  LANT24     INTEGER*2 DIMENSIONED COMMON EQUIVALENCED
              4  15  35
    
```

```

LDJMPS      EXTERNAL SUBROUTINE
              49
2CA2H      2  LT      INTEGER*2
              64 246 248 319 353 360 362 364 365 372 373
              380 385
MFLAG      EXTERNAL SUBROUTINE
              364
MFPNUM     EXTERNAL SUBROUTINE
              353 357 360 362
MOD        INTRINSIC
              64 309 373
MSGOUT     EXTERNAL SUBROUTINE
              54 72 79 89 101 113 124 135 155 161 166
              179 185 190 201 208 213 220 227 233 236 267
              294 342 376 377
MSKSET     EXTERNAL SUBROUTINE
              278
2CA0H      2  MX      INTEGER*2
              63 64 371 372 373
00E3H      8  NC      INTEGER*1 DIMENSIONED COMMON
              3 16 27 54 72 73 79 80 89 90 101
              102 106 113 114 117 124 125 135 136 138 155
              156 161 162 166 167 168 179 180 185 186 190
              191 192 201 202 208 213 220 227 233 236 237
              267 291 294 334 342 344 376 377
NERD      EXTERNAL SUBROUTINE
              246
2C5AH      16  NFEL   INTEGER*2 DIMENSIONED
              7 306 307 308 309 311 314 346
244AH      1  NPASS  INTEGER*1
              3 61 251 370
2C54H      2  NPZN   INTEGER*2
              7 87 92 94 97 121 206 364
2CB2H      2  NZE    INTEGER*2
              86 336 337
2CBOH      2  NZL    INTEGER*2
              85 336
2CDAH      2  NZPAS  INTEGER*2
              242 255 257 264
0074H      20  FARM  REAL*4 DIMENSIONED
              2 146 148 150 249 256 259 261 275 277 285
              296 297 301 302 303 306 319
2CF6H      4  PC     REAL*4
              296 319 337 338 347 360
              PCPN   EXTERNAL SUBROUTINE
              296
2CF2H      4  PN     REAL*4
              296 319 336 338 362
2D1AH      4  FR     REAL*4
              346 347
0094H      20  PREV  REAL*4 DIMENSIONED

```

```

                2 259 261
2CB6H      4  RCAL      REAL*4
                95  98  99
                RUNLMT  EXTERNAL SUBROUTINE
                259
00ABH     32  SIGMA    REAL*4 DIMENSIONED
                2 325 348 357
2CABH      4  SLMDA    REAL*4
                83 206 306
008BH     12  START    REAL*4 DIMENSIONED
                2 170 173 224
2CE6H      4  SUM      REAL*4
                273 275
2CC6H      4  SYSK     REAL*4
                117 119 121 211
2CCCH      4  TO       REAL*4
                173 252 253 254
2CDCH      4  T1       REAL*4
                249 252 253 254
0020H     32  TANTL    REAL*4 DIMENSIONED
                2
2CD6H      4  TC       REAL*4
                241 285 296
2CE0H      4  TDEL     REAL*4
                254 270
0098H     32  THETA    REAL*4 DIMENSIONED COMMON
                2 16 26 273 274 285 299 300 321
2CEEH      4  TI       REAL*4
                277 279 296
                UNPACK  EXTERNAL SUBROUTINE
                248
                VALID  EXTERNAL SUBROUTINE
                256
0010H      8  XDSELF   REAL*4 DIMENSIONED COMMON
                2 16 21 84
2D06H      4  XI       REAL*4
                302 305 306
0000H      8  XLMDA    REAL*4 DIMENSIONED COMMON
                2 16 21 83
0018H     32  XSYSK    REAL*4 DIMENSIONED COMMON
                2 16 22 121
2D02H      4  XT       REAL*4
                301 302 303 304 305 306 317 319 338 339 347
                348
2D0AH      4  YI       REAL*4
                303 306 319

```

MODULE INFORMATION:

CODE AREA SIZE = 14CFH 5327D
VARIABLE AREA SIZE = 2D1EH 11550D
MAXIMUM STACK SIZE = 0016H 22D
356 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT INTCOM

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

ISIS-II OBJECT LOCATER V3.0 INVOKED BY:
 -LOCATE :F1:SIGMAO.LNK TO :F1:SIGMAO CODE(4100H) COLUMNS(2) SYMBOLS &
 **ORDER(CODE,DATA,/,/,/GG/,/A/,MEMORY) MAP PRINT(,LP,)

SYMBOL TABLE OF MODULE SIGMAO
 READ FROM FILE :F1:SIGMAO.LNK
 WRITTEN TO FILE :F1:SIGMAO

VALUE TYPE SYMBOL

4363H	MOD	INTCOM
D1D1H	SYM	INTCOM
D1D1H	SYM	FNOIZ
9E4DH	SYM	ANGT
D18DH	SYM	DTHEA
9E6DH	SYM	ACOUR
D209H	SYM	DFALT
D18DH	SYM	FCAL
9E95H	SYM	START
D1D9H	SYM	FPI3L
9EA1H	SYM	PREV
9ED5H	SYM	IOBUF
9F4DH	SYM	IOVER
9F52H	SYM	IOUT
D268H	SYM	NC
D258H	SYM	ILC
D23DH	SYM	IOVRQ
C256H	SYM	IWAIT
C257H	SYM	NPASS
C258H	SYM	TBL
C25AH	SYM	IAUTP
D267H	SYM	IMSK
D624H	SYM	INZERR
D3B8H	SYM	LANT21
D458H	SYM	LANT22
D4F8H	SYM	LANT23
D598H	SYM	LANT24
D2AEH	SYM	CANT2
D32AH	SYM	CANT4
D25AH	SYM	IFOLZ
D368H	SYM	IGTBL
D642H	SYM	ICLC
CA63H	SYM	IBND
CA67H	SYM	NFEL
CA87H	SYM	IBPT
D23DH	SYM	CIORVQ
D256H	SYM	CIYN
D25AH	SYM	CIPOLZ
D61AH	SYM	CFAERR
9ED5H	SYM	CIORVQ
D62EH	SYM	IBFLW
D645H	SYM	CL
D685H	SYM	CST
437BH	SYM	?5
CA9BH	SYM	QIOPB
554AH	SYM	?7000
CAADH	SYM	MX
4452H	SYM	?115
4B09H	SYM	?740
CAB5H	SYM	SLMDA
CABDH	SYM	NZL
44ECH	SYM	?160
CAC1H	SYM	L
458DH	SYM	?200
4584H	SYM	?196
4585H	SYM	?203
CAC9H	SYM	IERR
462FH	SYM	?260
464CH	SYM	?305
CAD3H	SYM	SYSK
4728H	SYM	?415
48C7H	SYM	?500
429DH	SYM	?430
47BDH	SYM	?460
485AH	SYM	?481
489EH	SYM	?483
48D3H	SYM	?520
493AH	SYM	?570
42BEH	SYM	?600
4A54H	SYM	?655

VALUE TYPE SYMBOL

9E0DH	SYM	CELL
9E2DH	SYM	TANTL
D185H	SYM	XLMDA
D19DH	SYM	XSYSK
D1E1H	SYM	OLIM
9E81H	SYM	PARM
D21DH	SYM	THETA
D195H	SYM	XDRLF
D1C9H	SYM	BHWID
9EB5H	SYM	SIGMA
9F11H	SYM	INERD
9F11H	SYM	IRESP
D5E8H	SYM	IFATAL
D256H	SYM	IYN
C252H	SYM	IBC
D262H	SYM	KXP
D61AH	SYM	IFAERR
D6DBH	SYM	KESC
C257H	SYM	IAUTR
C25BH	SYM	IOBUF
C25CH	SYM	IOFLG
D368H	SYM	LANT11
D408H	SYM	LANT12
D4A8H	SYM	LANT13
D548H	SYM	LANT14
D270H	SYM	CANT1
D2ECH	SYM	CANT3
9F11H	SYM	IPLZ
C261H	SYM	IPSD
D270H	SYM	IGTBC
CA61H	SYM	NPZN
CA65H	SYM	ICNT
CA77H	SYM	IFDL
CA97H	SYM	INZ
D5E8H	SYM	CFATAL
D258H	SYM	CILC
D642H	SYM	CCLC
9F11H	SYM	CRSP
9F11H	SYM	CNERD
D638H	SYM	ICALW
D67DH	SYM	CC
D6D9H	SYM	IEOC
CA99H	SYM	IEFLG
427AH	SYM	?21
43CEH	SYM	?40
CAAFH	SYM	LT
4412H	SYM	?85
CAB1H	SYM	BMW
CAB9H	SYM	DELF
CABFH	SYM	NZE
4540H	SYM	?180
CAC3H	SYM	RCAL
4574H	SYM	?195
CAC7H	SYM	IFCAL
4620H	SYM	?255
CACBH	SYM	DTHEI
CACFH	SYM	DTHER
46CDH	SYM	?380
46E3H	SYM	?385
CAD7H	SYM	J
4815H	SYM	?475
47F9H	SYM	?473
48BDH	SYM	?490
487CH	SYM	?482
4908H	SYM	?545
4A13H	SYM	?625
4A0EH	SYM	?620
CAD9H	SYM	TO
4A1FH	SYM	?630

4A86H	SYM	7680	4B04H	SYM	7730
42C5H	SYM	7710	CADDH	SYM	DELT
4D82H	SYM	7870	42CBH	SYM	7790
42F4H	SYM	7815	4C9AH	SYM	7860
4319H	SYM	7850	4D25H	SYM	7865
4CB3H	SYM	7862	432EH	SYM	7864
CAE1H	SYM	I	4D34H	SYM	7866
434BH	SYM	7868	CAE3H	SYM	TC
CAE7H	SYM	NZPAS	4DD2H	SYM	7915
4DD5H	SYM	7925	CAE9H	SYM	T1
4E23H	SYM	7930	4E3BH	SYM	7940
4E32H	SYM	7935	4E4DH	SYM	7945
CAEDH	SYM	TDEL	4E88H	SYM	7950
4E95H	SYM	7955	4ERBH	SYM	7940
CAF1H	SYM	K	4EF8H	SYM	7965
4F19H	SYM	7977	4F07H	SYM	7975
556EH	SYM	77777	4F28H	SYM	71000
4F7BH	SYM	71005	CAF3H	SYM	SUM
CAF7H	SYM	DIFF	CAFBH	SYM	TI
4F92H	SYM	71030	5001H	SYM	71050
500BH	SYM	71060	5026H	SYM	71100
501AH	SYM	71065	CAFFH	SYM	PN
CB03H	SYM	PC	CB07H	SYM	AROL
52BAH	SYM	71305	CB0BH	SYM	ANGL
CB0FH	SYM	XT	CB13H	SYM	XI
CB17H	SYM	YI	CB1BH	SYM	BNDW
CB1FH	SYM	FDOP	51ABH	SYM	71200
51DOH	SYM	71210	CB23H	SYM	ANGTL
52C4H	SYM	71310	52DBH	SYM	71650
52FAH	SYM	71657	5304H	SYM	71660
536BH	SYM	71689	5375H	SYM	71690
53D1H	SYM	71700	CB27H	SYM	PR
541BH	SYM	71800	5431H	SYM	71900
558BH	SYM	7777			
	MOD	IUSART			
0027H	SYM	CMD	00CDH	SYM	CNCTL
00CEH	SYM	MODE	55CFH	SYM	IUSRT
	MOD	OVERIF			
55D8H	SYM	ASCDU	5628H	SYM	CO
5602H	SYM	CCO	5611H	SYM	CP0
561FH	SYM	CP1	561AH	SYM	CP2
55FOH	SYM	DC	55DDH	SYM	GC
55EAH	SYM	STP	55E8H	SYM	STP0
	MOD	KEYIN			
0008H	SYM	BACK	0007H	SYM	BELL
000DH	SYM	CR	564AH	SYM	BKSPC
5674H	SYM	CI	565CH	SYM	FILL
566DH	SYM	GETCH	5631H	SYM	KEYBRD
5635H	SYM	NEXT	5652H	SYM	SKTPI
5663H	SYM	SKIP2			
	MOD	OUTPUT			
5681H	SYM	GBYT	567EH	SYM	MSGOUT
5693H	SYM	XIT			
	MOD	ICRLF			
000DH	SYM	CR	001BH	SYM	ESC
000AH	SYM	LF	56B4H	SYM	CO
5694H	SYM	CRLF	56C1H	SYM	DELAY
56A3H	SYM	ECHO5	56B4H	SYM	ECH10
569AH	SYM	ECHO	56C5H	SYM	LOOP
	MOD	GETVLU			
56ECH	SYM	GETVLU	CB2BH	SYM	CIOBUF
CB2FH	SYM	IOBUFF	CB31H	SYM	N10
CB33H	SYM	N20	CB35H	SYM	IERR
CB37H	SYM	XNUM	CB2DH	SYM	CIOBUF@
571DH	SYM	?5	CB39H	SYM	CIOPB
56CFH	SYM	?10	57B9H	SYM	?50
5750H	SYM	?15	575CH	SYM	?20
5768H	SYM	?25	5774H	SYM	?30
56E6H	SYM	?35	57C5H	SYM	?51
57D1H	SYM	?52	57B8H	SYM	?40
	MOD	IBFILL			
57DDH	SYM	IBFILL	57E0H	SYM	LL1
57E2H	SYM	LL2			
	MOD	DWAIT			
57EEH	SYM	DWAIT	57F3H	SYM	LOOP
5804H	SYM	LP2			
	MOD	RUNLMT			
5816H	SYM	RUNLMT	CB4BH	SYM	PREV
CB4DH	SYM	IOVER	CB4FH	SYM	PARM
CB51H	SYM	LZ	CB52H	SYM	UZ

5898H SYM ?10
 CB57H SYM AU
 CB58H SYM L
 MOD UNPACK
 59E2H SYM RR4B
 59C9H SYM UNPACK
 MOD LDJMPS
 4020H SYM LDJMPS
 MOD DECODA
 59E7H SYM DECODA
 CB5FH SYM NERZ
 CB63H SYM FARM
 D665H SYM CL
 D685H SYM CST
 5A63H SYM ?30
 CB6FH SYM K
 5B1FH SYM ?50
 5A94H SYM ?35
 CB73H SYM L
 5B93H SYM ?80
 5BRFH SYM ?85
 MOD KEYCHK
 5C78H SYM KEYCHK
 MOD VALID
 5C96H SYM VALID
 CB77H SYM DF
 CB7BH SYM IVAL
 CB7FH SYM LZ
 CB81H SYM KK
 CB83H SYM K
 5D6BH SYM ?10
 MOD MFLAG
 5DA0H SYM MFLAG
 CB89H SYM IOF
 CB8DH SYM IB
 5DEBH SYM ?10
 5E43H SYM ?20
 MOD MFPNUM
 5E56H SYM MFPNUM
 CB97H SYM ICOL
 CB9BH SYM IOUT
 CB9FH SYM NK
 CBA1H SYM K
 CBA7H SYM INUM
 5F82H SYM ?10
 CBA8H SYM IH
 601FH SYM ?20
 CBAFH SYM INDX
 MOD KBPHAL
 6034H SYM KBPHAL
 CBB5H SYM J
 MOD INIT
 00EBH SYM C8255
 00ECH SYM D8251
 00E9H SYM DATA
 0009H SYM DMA
 0009H SYM FILB
 001FH SYM ICW1
 000BH SYM INTEA
 000BH SYM LOADN
 00FFH SYM MASK
 0020H SYM DCW2
 00E7H SYM UNSTR
 60D0H SYM BPLISR
 60C3H SYM CZTR
 60FFH SYM FN5H
 60A2H SYM INI259
 60E2H SYM INIPIO
 60CBH SYM LOAD
 60DBH SYM NERD
 MOD IKEYI
 610AH SYM IKEYI
 D6D9H SYM IECC
 MOD CZTINT
 6110H SYM CZTINT
 MOD IENDI
 6125H SYM IENDI
 D6D9H SYM IECC
 MOD AMDMUL
 0012H SYM MULT

CB53H SYM AL
 593CH SYM ?20
 59C8H SYM ?30
 59CEH SYM STEP
 CB5DH SYM IOV
 CB61H SYM T1
 CB62H SYM II
 CB63H SYM CC
 59FDH SYM ?10
 5A40H SYM ?20
 5A4AH SYM ?25
 CB71H SYM J
 5AD4H SYM ?40
 5B29H SYM ?60
 5B70H SYM ?70
 5C95H SYM XIT
 CB75H SYM DL
 CB79H SYM IOL
 CB7DH SYM PARM
 CB80H SYM UZ
 5D95H SYM ?20
 CB85H SYM IDFG
 CB87H SYM IC
 CB8BH SYM IOUT
 CB8FH SYM NZ
 CB91H SYM I
 CB93H SYM K
 CB95H SYM FPNBR
 CB9DH SYM IBPT
 6029H SYM IBCD
 CBA3H SYM ?30
 5F2AH SYM FP
 CBA9H SYM ?5
 5F9AH SYM ?15
 CBADH SYM L
 CBB3H SYM IOUT
 6077H SYM ?10
 00A6H SYM CW8255
 00EBH SYM D8255
 0008H SYM DATB
 0020H SYM EOI
 00D0H SYM FILD
 0040H SYM ICW2
 0005H SYM INTER
 006AH SYM LSR
 000BH SYM MSR
 00FDH SYM UMASK
 00EFH SYM UNSYN
 60BFH SYM CZT
 60RDH SYM DSABLE
 60C6H SYM IBIPHL
 607BH SYM INICZT
 60B5H SYM INTSET
 60B1H SYM MSGSET
 60E6H SYM STR
 D6DBH SYM KESC
 D6DBH SYM KESC
 612CH SYM AMDMUL

ORIGINAL PAGE IS
 OF POOR QUALITY

0013H	MOD	AMDIV	613FH	SYM	AMDIV
	SYM	FDIV			
	MOD	FDLEV			
0013H	SYM	FDIV	001FH	SYM	FIX
006CH	SYM	IAD	006FH	SYM	IDIV
006DH	SYM	ISUB	6152H	SYM	FDLEV
6187H	SYM	TWO			
	MOD	CBNDW			
001DH	SYM	FLOT	0012H	SYM	MULT
6189H	SYM	CBNDW			
	MOD	GXI			
0003H	SYM	COS	0019H	SYM	EXCHG
0010H	SYM	FADD	0012H	SYM	MULT
0002H	SYM	SIN	0004H	SYM	TAN
61A1H	SYM	GXI			
	MOD	GYI			
0003H	SYM	COS	0019H	SYM	EXCHG
0011H	SYM	FSUB	0012H	SYM	MULT
0002H	SYM	SIN	0004H	SYM	TAN
61DFH	SYM	GYI			
	MOD	CLNPT			
0005H	SYM	ASIN	0010H	SYM	FADD
0012H	SYM	FMUL	0011H	SYM	FSUB
0074H	SYM	ICHS	001FH	SYM	IFIX
006DH	SYM	ISUB	0017H	SYM	PUSHT
0004H	SYM	TAN	6222H	SYM	CLNPT
C8E7H	SYM	HOLD			
	MOD	AMDSUB			
0011H	SYM	SUBT	62A8H	SYM	AMDSUB
	MOD	AMDADD			
0010H	SYM	RADD	62BBH	SYM	AMDADD
	MOD	INTGT			
0010H	SYM	FADE	001FH	SYM	FIX
0012H	SYM	MULT	62F1H	SYM	HALF
62CEH	SYM	INTGT			
	MOD	FDLDD			
0010H	SYM	FADD	0013H	SYM	FDIV
001FH	SYM	FIX	006CH	SYM	IAD
006FH	SYM	IDIV	006DH	SYM	ISUB
62F5H	SYM	FDLDD	6340H	SYM	HALF
6344H	SYM	ONE	6346H	SYM	TWO
	MOD	GIVE			
00F0H	SYM	PORT	6348H	SYM	GIVE
	MOD	GET			
00F0H	SYM	PORT	6364H	SYM	BSY
6361H	SYM	GET			
	MOD	INTLDD			
00F0H	SYM	PORT	637AH	SYM	INTLDD
	MOD	INTSTR			
00F0H	SYM	PORT	6388H	SYM	INTSTR
	MOD	AMDLOD			
00F0H	SYM	PORT	6397H	SYM	AMDLOD
63BFH	SYM	LABEL1	63CBH	SYM	LABEL2
63CEH	SYM	LABEL3			
	MOD	ALTFF			
001DH	SYM	FLOAT	006CH	SYM	IAD
006EH	SYM	IMULT	0012H	SYM	MULT
63D1H	SYM	ALTFF	6425H	SYM	TEN
	MOD	DRPFF			
0013H	SYM	DIV	001DH	SYM	FLOAT
006CH	SYM	IAD	006EH	SYM	IMULT
0012H	SYM	MULT	6427H	SYM	DRPFF
6470H	SYM	TEN			
	MOD	VELFP			
0013H	SYM	DIV	001DH	SYM	FLOAT
006CH	SYM	IAD	006EH	SYM	IMULT
0012H	SYM	MULT	64B2H	SYM	TEN
6472H	SYM	VELFP			
	MOD	TSECS			
006CH	SYM	IAD	006EH	SYM	IMULT
64E6H	SYM	TEN	64B4H	SYM	TSECS
	MOD	PCPN			
0013H	SYM	DIV	0017H	SYM	ENTR
0019H	SYM	EXCHG	0012H	SYM	MULT
00F0H	SYM	PORT	6526H	SYM	HALF
64E8H	SYM	PCPN			
	MOD	CELCNT			
0010H	SYM	AD	0013H	SYM	DIV
001FH	SYM	FIX	0012H	SYM	MULT
00F0H	SYM	PORT	0004H	SYM	TAN

652AH	SYM	CCLCNT	6564H	SYM	HALF
	MOD	ANDCMD	0018H	SYM	PULL
00F0H	SYM	PORT	6573H	SYM	BUSY
6568H	SYM	ANDCMD	6595H	SYM	LOOP
6569H	SYM	RZY	65B0H	SYM	NOERR
65AAH	SYM	LOOP2			
65A6H	SYM	UNDFLO	0018H	SYM	PULL
	MOD	ANDSTR	65DDH	SYM	BACK
00F0H	SYM	PORT	65EBH	SYM	NOERR
65B3H	SYM	ANDSTR	65FAH	SYM	SKIP2
65B7H	SYM	BUSY			
65F8H	SYM	SKIP1	0003H	SYM	COS
65E5H	SYM	UNDFLO	0013H	SYM	FDIV
	MOD	CNFILT	0012H	SYM	MULT
0007H	SYM	ATAN	0002H	SYM	SIN
0010H	SYM	FADD	660DH	SYM	CNFILT
001FH	SYM	FIX	CBBBH	SYM	ISAVE
0017H	SYM	PUSHT			
0001H	SYM	SQRT	0012H	SYM	MULT
6706H	SYM	HALF	0001H	SYM	SQRT
	MOD	IBELL	6710H	SYM	GXT
670AH	SYM	IBELL	006EH	SYM	IMULT
	MOD	GXT	6759H	SYM	HINHR
0011H	SYM	FSUB			
0017H	SYM	PUSHT	0013H	SYM	DIV
0004H	SYM	TAN	006CH	SYM	IAD
	MOD	MINHR	0012H	SYM	MULT
006CH	SYM	IAD	67C2H	SYM	F60
6777H	SYM	I10	6779H	SYM	TIMEFF
	MOD	TIMEFF			
0010H	SYM	AD	00F0H	SYM	PORT
001DH	SYM	FLOAT	0004H	SYM	TAN
006EH	SYM	IMULT			
67C6H	SYM	F10	002CH	SYM	IADD32
67C0H	SYM	I60	0012H	SYM	MULT
	MOD	CCELL	67F0H	SYM	I32SUM
0012H	SYM	MULT	681FH	SYM	TEN
0011H	SYM	SUBT			
67CAH	SYM	CCELL	6823H	SYM	I32L0D
	MOD	I32SUM			
001CH	SYM	FLOT32			
000BH	SYM	LOG			
6805H	SYM	FLOAT			
67F6H	SYM	LOOP			
	MOD	I32L0D			
00F0H	SYM	PORT			

MEMORY MAP OF MODULE SIGMA0
 READ FROM FILE :F1:SIGMA0.LNK
 WRITTEN TO FILE :F1:SIGMA0
 MODULE START ADDRESS 4363H

START	STOP	LENGTH	REL	NAME
4000H	4020H	21H	A	ABSOLUTE
4100H	9E0CH	5D0BH	B	CODE
9E0DH	D184H	3378H	B	DATA
D185H	D664H	4E0H	B	//
D665H	D6DBH	74H	B	/GG/
D6D9H	D6DBH	3H	B	/A/
D6DCH	F6BFH	1FE4H	B	MEMORY
F6C0H	F854H	195H	B	STACK

APPENDIX C
Bi-Phase-L Output Frame

Bi-Phase L Output Data Format

<u>CONTENT</u> (H means Hex)	<u>WORD Nbr</u> (4 Bits Each)
SYNC FH	1
SYNC BH	2
SYNC FH	3
FRAME	4
TIME, SEC. (Tenths)	5
TIME, SEC. (Units)	6
TIME, SEC. (Tens)	7
TIME, MIN. (Units)	8
TIME, MIN. (Tens)	9

 (Same As Input NERDAS)

FLIGHT Nbr (Units)	103
FLIGHT Nbr (Tens)	104
LINE Nbr (Units)	105
LINE Nbr (Tens)	106
RUN Nbr (Units)	107
LINE START (Units)	108
ANGLE 1 (Tenths)	109
ANGLE 1 (Units)	110
ANGLE 1 (Tens)	111

Bi-Phase L Output Data Format (continued)

<u>CONTENT</u> (H means Hex)	<u>WORD Nbr</u> (4 Bits Each)
ANGLE 1 (Sign)	112
ANGLE 2 (Tenths)	113
ANGLE 2 (Units)	114
ANGLE 2 (Tens)	115
ANGLE 2 (Sign)	116

ANGLE 8 (Tenths)	137
ANGLE 8 (Units)	138
ANGLE 8 (Tens)	139
ANGLE 8 (Sign)	140
SIGMA 1 (Tenths)	141
SIGMA 1 (Units)	142
SIGMA 1 (Tens)	143
SIGMA 1 (Sign)	144
SIGMA 2 (Tenths)	145
SIGMA 2 (Units)	146
SIGMA 2 (Tens)	147
SIGMA 2 (Sign)	148

SIGMA 8 (Tenths)	169
SIGMA 8 (Units)	170
SIGMA 8 (Tens)	171

Bi-Phase L Output Data Format (continued)

<u>CONTENT</u> (H means Hex)	<u>WORD Nbr</u> (4 Bits Each)
SIGMA 8 (Sign	172
CPWR (Tenths)	173
CPWR (Units)	174
CPWR (Tens)	175
CPWR (Sign)	176
NPWR (Tenths)	177
NPWR (Units)	178
NPWR (Tens)	179
NPWR (Sign)	180
NPZN (Units)	181
IVALB (Units)	182
SALARM (Units)	183
ISYSK (Units)	184
IOVRB (Units)	185
IBND (Units)	186
ITHE (Units)	187
DDH Frame End Fill	188

DDH Identified	225
DDH DD _{HEX}	256

APPENDIX D
Subroutine Listings (Alphabetical)

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME AFLOAT
		2	SUBROUTINE AFLOAT(IN,FP)
		3	;
		4	;
		5	;;; FP=FLOAT(IN)
001D		6	FLOAT EQU 1DH
		7	;
		8	EXTRN INTLOD,AMDSTR,AMDCMD
		9	;
		10	PUBLIC AFLOAT
		11	;
		12	CSEG
		13	;
0000	E1	14	AFLOAT: POP H ;SAVE RETURN ADDR
0001	CD0000	15	CALL INTLOD ;IN ↑
0004	3E1D	16	MVI A,FLOAT
0006	CD0000	17	CALL AMDCMD ;FLOAT(IN)
0009	42	18	MOV B,D
000A	4B	19	MOV C,E
000B	CD0000	20	CALL AMDSTR ;FP=FLOAT(IN)
000E	E9	21	PCHL ;RETURN
		22	;
		23	END

PUBLIC SYMBOLS
AFLOAT C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDSTR E 0000 INTLOD E 0000

USER SYMBOLS
AFLOAT C 0000 AMDCMD E 0000 AMDSTR E 0000 FLOAT, A 001D INTLOD E 0000

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME AIDX
		2	SUBROUTINE AIDX(IX, I80, TEN, AGL, RAD)
		3	;
		4	;
		5	::: IX=IFIX(AGL*RAD-TEN)+I80
0012		6	MULT EQU 12H
0011		7	SUBT EQU 11H
001F		8	FIX EQU 1FH
006C		9	IAD EQU 6CH
		10	;
		11	EXTRN AMDL0D, INTL0D, INTSTR, AMDCHD
		12	;
		13	PUBLIC AIDX
		14	;
		15	CSEG
		16	;
0000	E1	17	AIDX: POP H ;SAVE RTN ADDR
0001	CD0000	18	CALL AMDL0D ;AGL ↑
0004	42	19	MOV B, D
0005	4B	20	MOV C, E
0006	CD0000	21	CALL AMDL0D ;RAD ↑
0009	3E12	22	MVI A, MULT
000B	CD0000	23	CALL AMDCHD ;AGL*RAD
000E	C1	24	POP B
000F	CD0000	25	CALL AMDL0D ;TEN ↑
0012	3E11	26	MVI A, SUBT
0014	CD0000	27	CALL AMDCHD ;AGL*RAD-TEN
0017	3E1F	28	MVI A, FIX
0019	CD0000	29	CALL AMDCHD ;IFIX(AGL*RAD-TEN)
001C	C1	30	POP B
001D	CD0000	31	CALL INTL0D ;I80 ↑
0020	3E6C	32	MVI A, IAD
0022	CD0000	33	CALL AMDCHD ; + I80
0025	C1	34	POP B
0026	CD0000	35	CALL INTSTR ;IX=RESULT
0029	E9	36	PCHL ;RETURN
		37	;
		38	END

PUBLIC SYMBOLS
AIDX C 0000

EXTERNAL SYMBOLS
AMDCHD E 0000 AMDL0D E 0000 INTL0D E 0000 INTSTR E 0000

USER SYMBOLS
AIDX C 0000 AMDCHD E 0000 AMDL0D E 0000 FIX A 001F IAD A 006C
INTL0D E 0000 INTSTR E 0000 MULT A 0012 SUBT A 0011

ASSEMBLY COMPLETE, NO ERRORS

ORIGINAL PAGE IS
OF POOR QUALITY

LOC	OBJ	LINE	SOURCE STATEMENT
		1 ;	SUBROUTINE ALTFP(ALT,CALT,ITN,IHD,ITH,IUN)
		2 ;	
		3 ;	NAME ALTFP
		4 ;	
		5 ;	
		6 ;	EVALUATES ALTITUDE IN METERS:
		7 ;	
		8 ;	ALT=FLOAT(IUN+10*(ITN+10*(IHD+10*ITH)))*CALT
		9 ;	
0012		10	MULT EQU 12H
006C		11	IAD EQU 6CH
006E		12	IMULT EQU 6EH
001D		13	FLOAT EQU 1DH
		14 ;	
		15 ;	EXTRN INTLOD,AMDCMD,AMDSTR,AMDLOD
		16 ;	PUBLIC ALTFP
		17 ;	
		18 ;	CSEG
		19 ;	
0000	E1	20	ALTFP: POP H ;SAVE RETURN ADDRESS
0001	CD0000	21	CALL INTLOD ;ITH ↑
0004	015400	22	LXI B,TEN
0007	CD0000	23	CALL INTLOD ;10 ↑
000A	3E6E	24	MVI A,IMULT
000C	CD0000	25	CALL AMDCMD ;X
000F	C1	26	POP B
0010	CD0000	27	CALL INTLOD ;IHD
0013	3E6C	28	MVI A,IAD
0015	CD0000	29	CALL AMDCMD ;↑
0018	015400	30	LXI B,TEN
001B	CD0000	31	CALL INTLOD ;10 ↑
001E	3E6E	32	MVI A,IMULT
0020	CD0000	33	CALL AMDCMD ;X
0023	C1	34	POP B
0024	CD0000	35	CALL INTLOD ;ITN ↑
0027	3E6C	36	MVI A,IAD
0029	CD0000	37	CALL AMDCMD ;↑
002C	015400	38	LXI B,TEN
002F	CD0000	39	CALL INTLOD ;10 ↑
0032	3E6E	40	MVI A,IMULT
0034	CD0000	41	CALL AMDCMD ;X
0037	42	42	MOV R,D
0038	4B	43	MOV C,E
0039	CD0000	44	CALL INTLOD ;IUN ↑
003C	3E6C	45	MVI A,IAD
003E	CD0000	46	CALL AMDCMD ;↑
0041	3E1D	47	MVI A,FLOAT
0043	CD0000	48	CALL AMDCMD ;FLOAT
0046	C1	49	POP B
0047	CD0000	50	CALL AMDLOD ;CALT ↑
004A	3E12	51	MVI A,MULT
004C	CD0000	52	CALL AMDCMD ;X
004F	C1	53	POP B
0050	CD0000	54	CALL AMDSTR ;SAVE IN ALT
0053	E9	55	PCHL ;RETURN
0054	0A00	56	TEN: DW 1C
		57	END

PUBLIC SYMBOLS
ALTFP C 0000

EXTERNAL SYMBOLS

AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000 INTLDD E 0000

USER SYMBOLS

ALTFP	C 0000	AMDCMD	E 0000	AMDLDD	E 0000	AMDSTR	E 0000	FLOAT	A 001D
IAD	A 006C	IMULT	A 006E	INTLDD	E 0000	MULT	A 0012	TEN	C 0054

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME AMDADD
		2	;
		3	;
		4	SUBROUTINE AMDADD(R,A1,A2)
		5	;
		6	!!! R=A1+A2
0010		7	RADD EQU 10H
		8	;
		9	EXTRN AMDLDD,AMDCMD,AMDSTR
		10	;
		11	PUBLIC AMDADD
		12	;
		13	CSEG
		14	;
0060	E1	15	AMDADD: POP H ;SAVE RETURN ADDR
0001	CD0000	16	CALL AMDLDD ;A1 ↑
0004	42	17	MOV B,D
0005	4B	18	MOV C,E
0006	CD0000	19	CALL AMDLDD ;A2 ↑
0009	3E10	20	MVI A,RADD
000B	CD0000	21	CALL AMDCMD ;A1+A2
000E	C1	22	POP B
000F	CD0000	23	CALL AMDSTR ;R=A1+A2
0012	E9	24	PCHL ;RETURN
		25	;
		26	END

PUBLIC SYMBOLS
AMDADD C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000

USER SYMBOLS
AMDADD C 0000 AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000 RADD A 0010

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;
		2	SUBROUTINE AMDGSQ(GSQ,DGN,IGT,TEN)
		3	;
		4	;;; GSQ=FLOAT(IGT)/TEN+DGN
		5	;
		6	NAME AMDGSQ
001D		7	AFLOAT EQU 1DH
0013		8	ADIV EQU 13H
0010		9	FADD EQU 10H
		10	;
		11	PUBLIC AMDGSQ
		12	EXTRN AMDLOD,INTLOD,AMDSTR,AMDCMD
		13	;
		14	CSEG
		15	;
0000	E1	16	AMDGSQ: POP H ;SAVE RTN ADDR
0001	CD0000	17	CALL INTLOD ;IGT ↑
0004	3E1D	18	MVI A,AFLOAT
0006	CD0000	19	CALL AMDCMD ;FLOAT(IGT)
0009	42	20	MOV B,B
000A	4B	21	MOV C,E
000B	CD0000	22	CALL AMDLOD ;TEN ↑
000E	3E13	23	MVI A,ADIV
0010	CD0000	24	CALL AMDCMD ;FLOAT(IGT)/TEN
0013	C1	25	POP B
0014	CD0000	26	CALL AMDLOD ;DGN ↑
0017	3E10	27	MVI A,FADD
0019	CD0000	28	CALL AMDCMD ;+
001C	C1	29	POP B
001D	CD0000	30	CALL AMDSTR ;SAVE RESULT AT GSQ
0020	E9	31	PCHL ;RETURN
		32	END

PUBLIC SYMBOLS
AMDGSQ C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLOD E 0000 AMDSTR E 0000 INTLOD E 0000

USER SYMBOLS
ADIV A 0013 AFLOAT A 001D AMDCMD E 0000 AMDGSQ C 0000 AMDLOD E 0000
AMDSTR E 0000 FADD A 0010 INTLOD E 0000

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;
		2	;
		3	;
		4	;
		5	;
		6	;
		7	NAME AMDGN
001D		7	AFLOAT EQU 1DH
0013		8	ADIV EQU 13H
0012		9	MULT EQU 12H
006D		10	ISUB EQU 6DH
		11	;
		12	EXTRN AMDLDD,INTLDD,AMDSTR,AMDCMD
		13	;
		14	PUBLIC AMDGN
		15	;
		16	CSEG
		17	;
0000	E1	18	AMDGN: POP H ;PICK OFF RETURN ADDR
0001	CD0000	19	CALL INTLDD ;I1 ↑
0004	42	20	MOV B,D
0005	4B	21	MOV C,E
0006	CD0000	22	CALL INTLDD ;I2 ↑
0009	3E6D	23	MVI A,ISUB
000B	CD0000	24	CALL AMDCMD ;I1-I2
000E	3E1D	25	MVI A,AFLOAT
0010	CD0000	26	CALL AMDCMD ;FLOAT(I1-I2)
0013	C1	27	POP B
0014	CD0000	28	CALL AMDLDD ;TEN ↑
0017	3E13	29	MVI A,ADIV
0019	CD0000	30	CALL AMDCMD ;FLOAT(I1-I2)/TEN
001C	C1	31	POP B
001D	CD0000	32	CALL AMDLDD ;DG ↑
0020	3E12	33	MVI A,MULT
0022	CD0000	34	CALL AMDCMD ;FLOAT(I1-I2)/TEN*DG
0025	C1	35	POP B
0026	CD0000	36	CALL AMDLDD ;TWO ↑
0029	3E13	37	MVI A,ADIV
002B	CD0000	38	CALL AMDCMD ;(...)*DG/TWO
002E	C1	39	POP B
002F	CD0000	40	CALL AMDSTR ;SAVE RESULT AT DGN
0032	E9	41	PCHL ;RETURN
		42	END

PUBLIC SYMBOLS
AMDGN C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000 INTLDD E 0000

USER SYMBOLS
ADIV A 0013 AFLOAT A 001D AMDCMD E 0000 AMDGN C 0000 AMDLDD E 0000
AMDSTR E 0000 INTLDD E 0000 ISUB A 006D MULT A 0012

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME AMDMUL
		2	SUBROUTINE AMDMUL(R,A1,A2)
		3	;
		4	;
		5	::: R=A1*A2
0012		6	MULT EQU 12H
		7	;
		8	EXTRN AMDLDD,AMDSTR,AMDCMD
		9	;
		10	PUBLIC AMDMUL
		11	;
		12	CSEG
		13	;
0000	E1	14	AMDMUL: POP H ;SAVE RTN ADDR
0001	CD0000	15	CALL AMDLDD ;A1 ↑
0004	42	16	MOV B,D
0005	4B	17	MOV C,E
0006	CD0000	18	CALL AMDLDD ;A2 ↑
0009	3E12	19	MVI A,MULT
000B	CD0000	20	CALL AMDCMD ;A1*A2
000E	C1	21	POP B
000F	CD0000	22	CALL AMDSTR ;R=A1*A2
0012	E9	23	PCHL ;RETURN
		24	;
		25	END

PUBLIC SYMBOLS
AMDMUL C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000

USER SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDMUL C 0000 AMDSTR E 0000 MULT A 0012

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME AMDSUB
		2	SUBROUTINE AMDSUB(R,A1,A2)
		3	;
		4	;
		5	;;; R=A1-A2
0011		6	SUBT EQU 11H
		7	;
		8	EXTRN AMD CMD,AMD LOD,AMD STR
		9	;
		10	PUBLIC AMDSUB
		11	;
		12	CSEG
		13	;
0000	E1	14	AMDSUB: POP H ;SAVE RTN ADDR
0001	CD0000	15	CALL AMD LOD ;A1 ↑
0004	42	16	MOV R,D
0005	4B	17	MOV C,E
0006	CD0000	18	CALL AMD LOD ;A2 ↑
0009	3E11	19	MVI A,SUBT
000B	CD0000	20	CALL AMD CMD ;A1-A2
000E	C1	21	POP B
000F	CD0000	22	CALL AMD STR ;R=A1-A2
0012	E9	23	PCHL ;RETURN
		24	;
		25	END

PUBLIC SYMBOLS
AMDSUB C 0000

EXTERNAL SYMBOLS
AMD CMD E 0000 AMD LOD E 0000 AMD STR E 0000

USER SYMBOLS
AMD CMD E 0000 AMD LOD E 0000 AMD STR E 0000 AMDSUB C 0000 SUBT A 0011

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME CBNDW
		2	;
		3	;
		4	SUBROUTINE CBNDW(BNDW,NFEL,DELF)
		5	;
		6	::: BNDW=FLOAT(NFEL)*DELF
001D		7	FLOT EQU 1DH
0012		8	MULT EQU 12H
		9	;
		10	EXTRN INTLOD,AMDLOD,AMDSTR,AMDCMD
		11	;
		12	PUBLIC CBNDW
		13	;
		14	CSEG
		15	;
0000	E1	16	CBNDW: POP H ;SAVE RTN ADDR
0001	CD0000	17	CALL INTLOD ;NFEL ↑
0004	3E1D	18	MVI A,FLOT
0006	CD0000	19	CALL AMDCMD ;FLOAT(NFEL)
0009	42	20	MOV B,D
000A	4B	21	MOV C,E
000B	CD0000	22	CALL AMDLOD ;DELF ↑
000E	3E12	23	MVI A,MULT
0010	CD0000	24	CALL AMDCMD ;DELF*FLOAT(NFEL)
0013	C1	25	POP B
0014	CD0000	26	CALL AMDSTR ;BNDW= "
0017	E9	27	PCHL ;RETURN
		28	;
		29	END

PUBLIC SYMBOLS
CBNDW C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLOD E 0000 AMDSTR E 0000 INTLOD E 0000

USER SYMBOLS
AMDCMD E 0000 AMDLOD E 0000 AMDSTR E 0000 CBNDW C 0000 FLOT A 001D
INTLOD E 0000 MULT A 0012

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1 ;	SUBROUTINE CCELL(CELL,DIFF,SUM,ALT)
		2 ;	
		3 ;	NAME CCELL
		4 ;	
00F0		5 ;	
0004		6 PORT	EQU 00F0H
0011		7 TAN	EQU 04H
0012		8 SUBT	EQU 11H
		9 MULT	EQU 12H
		10 ;	
		11	EXTRN AMDLDD
		12	EXTRN AMDSTR
		13	EXTRN AMDCMD
		14 ;	
		15	PUBLIC CCELL
		16 ;	
		17	CSEG
		18 ;	
0000	E1	19 CCELL:	POP H ;SAVE RETURN ADDRESS
0001	CD0000	E 20	CALL AMDLDD ;SUM ↑
0004	3E04	21	MVI A,TAN
0006	CD0000	E 22	CALL AMDCMD ;TANGENT
0009	C1	23	POP B
000A	CD0000	E 24	CALL AMDLDD ;DIFF ↑
000D	3E04	25	MVI A,TAN
000F	CD0000	E 26	CALL AMDCMD ;TANGENT
0012	3E11	27	MVI A,SUBT
0014	CD0000	E 28	CALL AMDCMD ;-
0017	4B	29	MOV B,D
0018	4B	30	MOV C,E
0019	CD0000	E 31	CALL AMDLDD ;ALT ↑
001C	3E12	32	MVI A,MULT
001E	CD0000	E 33	CALL AMDCMD ;X
0021	C1	34	POP B
0022	CD0000	E 35	CALL AMDSTR ;SAVE IT IN CELL
0025	E9	36	PCHL ;RETURN
		37	END

PUBLIC SYMBOLS
CCELL C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000

USER SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000 CCELL C 0000 MULT A 0012
PORT A 00F0 SUBT A 0011 TAN A 0004

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME CELCNT
		2	;
		3	;
		4	;
		5	SUBROUTINE CELCNT(ICNT,VEL,TC,ALT,THETS)
		6	::: ICNT=IFIX((ALT*TAN(THETS))/(VEL*TC)+.5)
00F0		7	PORT EQU 00F0H
0010		8	AD EQU 10H
0012		9	MULT EQU 12H
0013		10	DIV EQU 13H
0004		11	TAN EQU 04H
001F		12	FIX EQU 1FH
		13	;
		14	EXTRN AMDLDD
		15	EXTRN AMDSTR
		16	EXTRN AMDCMD
		17	EXTRN GIVE
		18	EXTRN INTSTR
		19	;
		20	PUBLIC CELCNT
		21	;
		22	CSEG
		23	;
0000	E1	24	CELCNT: POP H ;SAVE RETURN ADDRESS
0001	CD0000	25	CALL AMDLDD ;ALT ↑
0004	42	26	MOV B,D
0005	4B	27	MOV C,E
0006	CD0000	28	CALL AMDLDD ;THETS ↑
0009	3E04	29	MVI A,TAN
000B	CD0000	30	CALL AMDCMD ;TAN(THETS)
000E	3E12	31	MVI A,MULT
0010	CD0000	32	CALL AMDCMD ;X
0013	C1	33	POP H
0014	CD0000	34	CALL AMDLDD ;TC ↑
0017	3E13	35	MVI A,DIV
0019	CD0000	36	CALL AMDCMD ;/
001C	C1	37	POP B
001D	CD0000	38	CALL AMDLDD ;VEL ↑
0020	3E13	39	MVI A,DIV
0022	CD0000	40	CALL AMDCMD ;/
0025	113A00	41	LXI D,HALF
0028	CD0000	42	CALL GIVE ;.5 ↑
002B	3E10	43	MVI A,AD
002D	CD0000	44	CALL AMDCMD ;+
0030	3E1F	45	MVI A,FIX
0032	CD0000	46	CALL AMDCMD ;CONVERT TO INTEGER
0035	C1	47	POP B
0036	CD0000	48	CALL INTSTR ;SAVE RESULT IN ICNT
0039	E9	49	PCHL ;RETURN
		50	;
003A	00	51	HALF: DB 00H,00H,80H,00H
003B	00		
003C	80		
003D	00		

LOC	OBJ	LINE	SOURCE STATEMENT			
		52	END			
PUBLIC SYMBOLS						
CELCNT C 0000						
EXTERNAL SYMBOLS						
AMDCHD	E 0000	AMDLOD	E 0000	AMDSTR	E 0000	GIVE E 0000 INTSTR E 0000
USER SYMBOLS						
AD	A 0010	AMDCMD	E 0000	AMDLOD	E 0000	AMDSTR E 0000 CELCNT C 0000 DIV A 0013
FIX	A 001F					
GIVE	E 0000	HALF	C 003A	INTSTR	E 0000	MULT A 0012 PORT A 00F0 TAN A 0004
ASSEMBLY COMPLETE, NO ERRORS						


```

LOC OBJ          LINE      SOURCE STATEMENT
      1          NAME      CINDX
      2          SUBROUTINE CINDX(IX,I1,TWO,HALF,D)
      3          ;
      4          ;
      5          ;::: IX=IFIX((D+0.5)/TWO)+I1
0010          6 AD        EQU      10H
0013          7 DIV       EQU      13H
001F          8 FIX       EQU      1FH
006C          9 IAD       EQU      6CH
      10         ;
      11         EXTRN    AMDL0D,INTL0D,INTSTR,AMDCHD
      12         ;
      13         PUBLIC  CINDX
      14         ;
      15         CSEG
      16         ;
0000 E1          17 CINDX:  POP      H          $SAVE RTN ADDR
0001 CD0000     E 18      CALL     AMDL0D    $HALF ↑
0004 42          19      MOV      B,D
0005 4B          20      MOV      C,E
0006 CD0000     E 21      CALL     AMDL0D    $D ↑
0009 3E10        22      MVI     A,AD
000B CD0000     E 23      CALL     AMDCHD    $D+0.5
000E C1          24      POP      B
000F CD0000     E 25      CALL     AMDL0D    $TWO ↑
0012 3E13        26      MVI     A,DIV
0014 CD0000     E 27      CALL     AMDCHD    $(D+.5)/2.
0017 3E1F        28      MVI     A,FIX
0019 CD0000     E 29      CALL     AMDCHD    $IFIX((D+.5)/2.)
001C C1          30      POP      B
001D CD0000     E 31      CALL     INTL0D    $I1 ↑
0020 3E6C        32      MVI     A,IAD
0022 CD0000     E 33      CALL     AMDCHD    $IFIX(...)+11
0025 C1          34      POP      B
0026 CD0000     E 35      CALL     INTSTR    $IXC--IFIX(...)+11
0029 E9          36      PCHL     $RETURN
      37         ;
      38         END

```

PUBLIC SYMBOLS
CINDX C 0000

EXTERNAL SYMBOLS
AMDCHD E 0000 AMDL0D E 0000 INTL0D E 0000 INTSTR E 0000

USER SYMBOLS
AD A 0010 AMDCHD E 0000 AMDL0D E 0000 CINDX C 0000 DIV A 0013
FIX A 001F IAD A 006C INTL0D E 0000 INTSTR E 0000

ASSEMBLY COMPLETE, NO ERRORS

ORIGINAL PAGE IS
OF POOR QUALITY

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME CLNPT
		2	;
		3	;
		4	SUBROUTINE CLNPT(IBPT,LT,PC,PN,XT,YI,ANGTD,ALT)
		5	;
		6	::: ANGTD = ASIN(ANGTD)
		7	XT = (ALT*ALT+YI*YI)*TAN(ANGTD)**2
		8	IBPT = LT-IFIX(PC*(TAN(ANGTD)-PN))
0005		9	ASIN EQU 05H
0004		10	TAN EQU 04H
0017		11	PUSHT EQU 17H
0012		12	FMUL EQU 12H
0010		13	FADD EQU 10H
0011		14	FSUB EQU 11H
001F		15	IFIX EQU 1FH
006D		16	ISUB EQU 6DH
0074		17	ICHS EQU 74H
		18	;
		19	EXTRN AMDL0D,AMDSTR,AMDCMD,INTL0D,INTSTR,GIVE,GET
		20	;
		21	PUBLIC CLNPT
		22	;
		23	CSEG
		24	;
0000	E1	25	CLNPT: POP H ;SAVE RETURN ADDRESS
0001	C5	26	PUSH R ;SAVE ADDRESS OF ANGTD
0002	CD0000	27	CALL AMDL0D ;ANGTD ↑
0005	3E05	28	MVI A,ASIN
0007	CD0000	29	CALL AMDCMD ;ASIN(ANGTD):--:--:--
000A	3E17	30	MVI A,PUSHT
000C	CD0000	31	CALL AMDCMD ;ASIN(ANGTD):ASIN(ANGTD):--:--:--
000F	C1	32	POP B
0010	CD0000	33	CALL AMDSTR ;SAVE IN ANGTD
0013	3E04	34	MVI A,TAN
0015	CD0000	35	CALL AMDCMD ;TAN(ANGTD):--:--:--
0018	3E17	36	MVI A,PUSHT
001A	CD0000	37	CALL AMDCMD
001D	CD0000	38	CALL AMDCMD ;TAN(ANGTD):TAN(ANGTD):TAN(ANGTD):--
0020	D5	39	PUSH D
0021	110000	40	LXI D,HOLD
0024	CD0000	41	CALL GET ;SAVE TAN(ANGTD) IN HOLD
0027	3E12	42	MVI A,FMUL
0029	CD0000	43	CALL AMDCMD ;TAN(ANGTD)**2:--:--:--
002C	C1	44	POP B
002D	CD0000	45	CALL AMDL0D ;ALT ↑
0030	3E17	46	MVI A,PUSHT
0032	CD0000	47	CALL AMDCMD
0035	3E12	48	MVI A,FMUL
0037	CD0000	49	CALL AMDCMD ;ALT**2:TAN(ANGTD)**2:--:--:--
003A	C1	50	POP B
003B	CD0000	51	CALL AMDL0D ;YI ↑
003E	3E17	52	MVI A,PUSHT
0040	CD0000	53	CALL AMDCMD
0043	3E12	54	MVI A,FMUL
0045	CD0000	55	CALL AMDCMD ;YI**2:ALT**2:TAN(ANGTD)**2:--
0048	3E10	56	MVI A,FADD
004A	CD0000	57	CALL AMDCMD ;YI**2+ALT**2:TAN(ANGTD)**2:--:--
004D	3E12	58	MVI A,FMUL
004F	CD0000	59	CALL AMDCMD ;(YI**2+ALT**2)*TAN(ANGTD)**2:--:--:--
0052	C1	60	POP B
0053	CD0000	61	CALL AMDSTR ;SAVE RESULT IN XT
0056	110000	62	LXI D,HOLD
0059	CD0000	63	CALL GIVE ;TAN(ANGTD) ↑

LOC	OBJ	LINE	SOURCE STATEMENT
005C	C1	64	POP B
005D	CD0000	E 65	CALL AMDLDD ;PN ↑
0060	3E11	66	MVI A,FSUB
0062	CD0000	E 67	CALL AMDCMD ;TAN(ANGTD)-PN:--:--:--
0065	C1	68	POP B
0066	CD0000	E 69	CALL AMDLDD ;PC ↑
0069	3E12	70	MVI A,FMUL
006B	CD0000	E 71	CALL AMDCMD ;PC*(TAN(ANGTD)-PN):--:--:--
006E	3E1F	72	MVI A,IFIX
0070	CD0000	E 73	CALL AMDCMD ;IFIX(PC*(TAN(ANGTD)-PN)):--:--:--
0073	C1	74	POP B
0074	CD0000	E 75	CALL INTLDD ;LT ↑
0077	3E6D	76	MVI A,ISUB
0079	CD0000	E 77	CALL AMDCMD ;IFIX(PC*(TAN(ANGTD)-PN))-LT:--:--:--
007C	3E74	78	MVI A,ICHS
007E	CD0000	E 79	CALL AMDCMD ;LT-IFIX(PC*(TAN(ANGTD)-PN)):--:--:--
0081	C1	80	POP B
0082	CD0000	E 81	CALL INTSTR ;SAVE RESULT IN IBPT
0085	E9	82	PCHL ;RETURN
		83 ;	
		84 ;	DSEG
		85 ;	
0000		86	HOLD: DS 4
		87	END

PUBLIC SYMBOLS
CLNPT C 0000

EXTERNAL SYMBOLS

AMDCMD E 0000	AMDLDD E 0000	AMDSTR E 0000	GET E 0000	GIVE E 0000
INTLDD E 0000	INTSTR E 0000			

USER SYMBOLS

AMDCMD E 0000	AMDLDD E 0000	AMDSTR E 0000	ASIN A 0005	CLNPT C 0000
FADD A 0010	FMUL A 0012	FSUB A 0011	GET E 0000	GIVE E 0000
HOLD D 0000	ICHS A 007+	IFIX A 001F	INTLDD E 0000	INTSTR E 0000
ISUB A 006D	PUSHT A 0017	TAN A 0004		

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME CNFILT
		2	;
		3	SUBROUTINE CNFILT(NFEL,DEL,BNDW,CELL,FDOP,ANGLD,YI,XI,XT,S
		4	LMDA,VEL,ALT)
		5	;
		6	:: XT=(2.*VEL)/SLMDA
		7	ANGLD=ATAN(SORT((XI*XI)/(ALT*ALT+YI*YI)))
		8	FDOP=XT*SIN(ANGLD)
		9	BNDW=(CELL*XT/ALT)*COS(ANGLD)**3
		10	NFEL=IFIX(BNDW/DEL+0.5)
0010		11	FADD EQU 10H
0012		12	MULT EQU 12H
0013		13	FDIV EQU 13H
0007		14	ATAN EQU 07H
0001		15	SQRT EQU 01H
0002		16	SIN EQU 02H
001F		17	FIX EQU 1FH
0017		18	PUSHT EQU 17H
0003		19	COS EQU 03H
		20	;
		21	EXTRN AMDLDD,AMDCMD,INTSTR,AMDSTR,GET,GIVE
		22	;
		23	PUBLIC CNFILT
		24	;
		25	CSEG
		26	;
0000	E1	27	CNFILT: POP H ;SAVE RTN ADDR
0001	220000	28	SHLD ;SAVE
0004	62	29	MOV H,D ;AT INTERNAL BUFF
0005	6B	30	MOV L,E ;PUT ALT ADDR IN RP(HL)
0006	CD0000	31	CALL AMDLDD ;VEL ↑
0009	3E17	32	MVI A,PUSHT
000B	CD0000	33	CALL AMDCMD ;VEL ↑
000E	3E10	34	MVI A,FADD
0010	CD0000	35	CALL AMDCMD ;2.*VEL
0013	C1	36	POP R
0014	CD0000	37	CALL AMDLDD ;SLMDA ↑
0017	3E13	38	MVI A,FDIV
0019	CD0000	39	CALL AMDCMD ;2.*VEL/SLMDA
001C	3E17	40	MVI A,PUSHT
001E	CD0000	41	CALL AMDCMD ;XT:XT:--
0021	C1	42	POP R
0022	CD0000	43	CALL AMDSTR ;SAVE XT
0025	110200	44	LXI D,ISAVE+2
0028	CD0000	45	CALL GET ;SAVE CY OF XT
002B	C1	46	POP R
002C	CD0000	47	CALL AMDLDD ;XI ↑
002F	3E17	48	MVI A,PUSHT
0031	CD0000	49	CALL AMDCMD ;XI:XI:--
0034	3E12	50	MVI A,MULT
0036	CD0000	51	CALL AMDCMD ;XI*XI
0039	C1	52	POP R
003A	CD0000	53	CALL AMDLDD ;YI ↑
003D	3E17	54	MVI A,PUSHT
003F	CD0000	55	CALL AMDCMD ;YI:YI:XI*XI:--
0042	3E12	56	MVI A,MULT
0044	CD0000	57	CALL AMDCMD ;YI*YI:XI*XI:--
0047	44	58	MOV B,H
0048	4D	59	MOV C,L
0049	CD0000	60	CALL AMDLDD ;ALT ↑
004C	3E17	61	MVI A,PUSHT
004E	CD0000	62	CALL AMDCMD ;ALT:ALT:--

LOC	OBJ	LINE	SOURCE STATEMENT
0051	3E12	63	MVI A,MULT
0053	CD0000	E 64	CALL AMDCMD ;ALT*ALT
0056	3E10	65	MVI A,FADD
0058	CD0000	E 66	CALL AMDCMD ;(ALT**2)+(YI**2);
005B	3E13	67	MVI A,FDIV
005D	CD0000	E 68	CALL AMDCMD ;(XI**2)/(ALT**2+YI**2)
0060	3E01	69	MVI A,SQRT
0062	CD0000	E 70	CALL AMDCMD ;SQRT(")
0065	3E07	71	MVI A,ATAN
0067	CD0000	E 72	CALL AMDCMD ;ATAN(SQRT(...))
006A	3E17	73	MVI A,PUSHT
006C	CD0000	E 74	CALL AMDCMD ;ATAN(...);ATAN(...)
006F	C1	75	POP B
0070	CD0000	E 76	CALL AMDSTR ;ANGLD= "
0073	3E17	77	MVI A,PUSHT
0075	CD0000	E 78	CALL AMDCMD ;ANGLD:ANGLD
007B	110600	D 79	LXI D,ISAVE+6
007B	CD0000	E 80	CALL GET ;SAVE CY OF ANGLD
007E	3E02	81	MVI A,SIN
0080	CD0000	E 82	CALL AMDCMD ;SIN(ANGLD)
0083	110200	D 83	LXI D,ISAVE+2
0086	CD0000	E 84	CALL GIVE ;XT:SIN(A..)
0089	3E12	85	MVI A,MULT
008B	CD0000	E 86	CALL AMDCMD ;XT*SIN(A..)
008E	C1	87	POP B
008F	CD0000	E 88	CALL AMDSTR ;FDOP = "
0092	110600	D 89	LXI D,ISAVE+6
0095	CD0000	E 90	CALL GIVE ;ANGLD ↑
0098	3E03	91	MVI A,COS
009A	CD0000	E 92	CALL AMDCMD ;COS(A..)
009D	3E17	93	MVI A,PUSHT
009F	CD0000	E 94	CALL AMDCMD ;COS(A..):COS(A..):-!-
00A2	3E17	95	MVI A,PUSHT
00A4	CD0000	E 96	CALL AMDCMD ;COSA:COSA:COSA:-
00A7	3E12	97	MVI A,MULT
00A9	CD0000	E 98	CALL AMDCMD ;COSA*COSA:COSA
00AC	3E12	99	MVI A,MULT
00AE	CD0000	E 100	CALL AMDCMD ;COSA**3
00B1	110200	D 101	LXI D,ISAVE+2
00B4	CD0000	E 102	CALL GIVE ;XT↑
00B7	3E12	103	MVI A,MULT
00B9	CD0000	E 104	CALL AMDCMD ;XT*COSA**3
00BC	44	105	MOV B,H
00BD	44	106	MOV C,L
00BE	CD0000	E 107	CALL AMDLDD ;ALT:XT*COSA**3
00C1	3E13	108	MVI A,FDIV
00C3	CD0000	E 109	CALL AMDCMD ;XT*COSA**3/ALT
00C6	C1	110	POP B
00C7	CD0000	E 111	CALL AMDLDD ;CELL ↑
00CA	3E12	112	MVI A,MULT
00CC	CD0000	E 113	CALL AMDCMD ;CELL*XT*COSA.....
00CF	3E17	114	MVI A,PUSHT
00D1	CD0000	E 115	CALL AMDCMD ;BNDW:BNDW:-!-
00D4	C1	116	POP B
00D5	CD0000	E 117	CALL AMDSTR ;SAVE VALUE OF BNDW
00D8	C1	118	POP B
00D9	CD0000	E 119	CALL AMDLDD ;DEL ↑
00DC	3E13	120	MVI A,FDIV
00DE	CD0000	E 121	CALL AMDCMD ;BNDW/DEL
00E1	11F900	C 122	LXI D,HALF
00E4	CD0000	E 123	CALL GIVE ;0.5 ↑
00E7	3E10	124	MVI A,FADD
00E9	CD0000	E 125	CALL AMDCMD ;(BNDW/DEL+0.5)
00EC	3E1F	126	MVI A,FIX

LOC	OBJ	LINE	SOURCE STATEMENT
00EE	CD0000	E 127	CALL AMDCHD ;IFIX(BNDW...)
00F1	C1	128	POP R
00F2	CD0000	E 129	CALL INTSTR ;NFEL= "
00F5	2A0000	D 130	LDI ISAVE ;GET RTN ADDR
00F8	E9	131	PCHL ;RETURN
		132	;
00F9	00	133	HALF: DB 00H,00H,80H,00H
00FA	00		
00FB	80		
00FC	00		
		134	;
		135	DSEG
		136	;
0000		137	ISAVE: DS 10
		138	;
		139	END

PUBLIC SYMBOLS
CNFILT C 0000

EXTERNAL SYMBOLS

AMDCHD E 0000	ANDLDD E 0000	AMDSTR E 0000	GET E 0000	GIVE E 0000
INTSTR E 0000				

USER SYMBOLS

AMDCHD E 0000	ANDLDD E 0000	AMDSTR E 0000	ATAN A 0007	CNFILT C 0000
COS A 0003	FADD A 0010	FDIV A 0013	FIX A 001F	GET E 0000
GIVE E 0000	HALF C 00F9	INTSTR E 0000	ISAVE D 0000	MULT A 0012
PUSHT A 0017	SIN A 0002	SQRT A 0001		

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME CRG4
		2	;
		3	;
		4	SUBROUTINE CRG4(RG4,FRTY,ALT,ANGTL)
		5	;
		6	:: RG4=40,*ALOG10(ALT/COS(ANGTL)), WHERE FRTY=40.
0003		7	COS EQU 03H
0013		8	FDIV EQU 13H
0008		9	ALOG EQU 08H
0012		10	MULT EQU 12H
		11	;
		12	EXTRN AMDLDD,AMDCMD,AMDSTR
		13	;
		14	PUBLIC CRG4
		15	;
		16	CSEG
		17	;
		18	CRG4:
0000	E1	18	POP H ;SAVE RTN ADDR
0001	CD0000	19	CALL AMDLDD ;ALT ↑
0004	42	20	MOV B,D
0005	4B	21	MOV C,E
0006	CD0000	22	CALL AMDLDD ;ANGTL ↑
0009	3E03	23	MVI A,COS
000B	CD0000	24	CALL AMDCMD ;COS(ANGTL):ALT
000E	3E13	25	MVI A,FDIV
0010	CD0000	26	CALL AMDCMD ;ALT/COS(ANGTL)
0013	3E08	27	MVI A,ALOG
0015	CD0000	28	CALL AMDCMD ;ALOG10(ALT/COS(...))
0018	C1	29	POP B
0019	CD0000	30	CALL AMDLDD ;40. ↑
001C	3E12	31	MVI A,MULT
001E	CD0000	32	CALL AMDCMD ;40.*ALOG10(ALT/COS(...))
0021	C1	33	POP B
0022	CD0000	34	CALL AMDSTR ;RG4= "
0025	E9	35	PCHL ;RETURN
		36	;
		37	END

PUBLIC SYMBOLS
CRG4 C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000

USER SYMBOLS
ALOG A 0008 AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000 COS A 0003
CRG4 C 0000 FDIV A 0013 MULT A 0012

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME CZTINT
		2	;
		3	FUNCTION: CZT-BOARD INTERRUPT HANDLER
		4	;
		5	PUBLIC CZTINT
		6	EXTRN IENDI
		7	;
		8	CSEG
		9	;
0000	F5	10	CZTINT: PUSH PSW ;SAVE
0001	C5	11	PUSH B ;REGISTERS
0002	D5	12	PUSH D ;IN THE CURRENT
0003	E5	13	PUSH H ;STATE OF THE MACHINE
0004	CD0000	14	CALL IENDI
0007	E1	15	POP H ;RESTORE
0008	D1	16	POP D ;REGISTERS
0009	C1	17	POP B ;TO THEIR ORIG STATE
000A	3E20	18	MVI A,20H ;RESTORE
000C	D3D8	19	OUT 0D8H ;MACHINE TO OPER LEVEL
000E	3EFF	20	MVI A,0FFH ;RESET INTERRUPT MASK
0010	D3D9	21	OUT 0D9H ;FOR NO INTERRUPTS
0012	F1	22	POP PSW ;RESTORE ACCUM & FLAGS
0013	FB	23	EI ;ENABLE INTERRUPTS
0014	C9	24	RET ;RETURN
		25	END

PUBLIC SYMBOLS
CZTINT C 0000

EXTERNAL SYMBOLS
IENDI E 0000

USER SYMBOLS
CZTINT C 0000 IENDI E 0000

ASSEMBLY COMPLETE, NO ERRORS

ORIGINAL PAGE
OF FOUR QUALITY

C-3

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME DAREA
		2	
		3	
		4	SUBROUTINE DAREA(AL, TEN, TRM1, TRM2, TRM3)
		5	:: AL=10.*ALOG10(TRM1*(TRM2-TRM3))
0012		6	
0008		7	MULT EQU 12H
0011		8	LOG EQU 08H
		9	FSUB EQU 11H
		10	
		11	EXTRN AMDL0D,AMDCMD,AMDSTR
		12	
		13	PUBLIC DAREA
		14	
		15	CSEG
		16	
		17	DAREA: POP H ;HOLD RTN ADDR
0000	E1	18	CALL AMDL0D ;TRM2 ↑
0004	42	19	MOV B,D
0005	4B	20	MOV C,E
0006	CD0000	21	CALL AMDL0D ;TRM3 ↑
0009	3E11	22	MVI A,FSUB
000B	CD0000	23	CALL AMDCMD ;TRM2-TRM3
000E	C1	24	POP B
000F	CD0000	25	CALL AMDL0B ;TRM1 ↑
0012	3E12	26	MVI A,MULT
0014	CD0000	27	CALL AMDCMD ;TRM1*(TRM2-TRM3)
0017	3E08	28	MVI A,LOG
0019	CD0000	29	CALL AMDCMD ;ALOG10(")
001C	C1	30	POP B
001D	CD0000	31	CALL AMDL0D ;10. ↑
0020	3E12	32	MVI A,MULT
0022	CD0000	33	CALL AMDCMD ;10.*ALOG10(...)
0025	C1	34	POP B
0026	CD0000	35	CALL AMDSTR ;AL= "
0029	E9	36	PCHL ;RETURN
		37	
		38	END

PUBLIC SYMBOLS
DAREA C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDL0D E 0000 AMDSTR E 0000

USER SYMBOLS
AMDCMD E 0000 AMDL0D E 0000 AMDSTR E 0000 DAREA C 0000 FSUB A 0011 LOG A 0
MULT A 0012

ASSEMBLY COMPLETE, NO ERRORS

ORIGINAL PAGE IS
OF EQUAL QUALITY

ISIS-II FORTRAN-80 V2.0 COMPILATION OF PROGRAM UNIT DECODA
 OBJECT MODULE PLACED IN IF1:DECODA.OBJ
 COMPILER INVOKED BY: FORTR0 IF1:DECODA.SRC DEBUG PAGEDLENGTH(77) PAGEDWIDTH(90)

```

1      SUBROUTINE DECODA(IOV,NERZ,T1,PARK)
C
C   :: DECODES NERDAS FRAME USING AND SUBROUTINES
C
2      DIMENSION PARK(5),II(5)
3      INTEGER*1 IOV(5),NERZ(60)
C
4      COMMON/GS/CL(6),CD(7,2),CST(9)
C
C   ===== DECODE ALT UP TO 9999. FEET =====
5      10  IF(IOV(1).NE.0) GO TO 30
6          DO 20 K=1,4
7              II(K)=NERZ(27+K)
8          CONTINUE
9          20  CALL ALTFF(PARK(1),CST(8),II(2),II(3),II(4),II(1))
C
C   ===== DECODE DRIFT,ROLL,PITCH UP TO 99.9 DEG
C
10     30  DO 50 J=2,4
11         IF(IOV(J).NE.0) GO TO 50
12         II(4)=1
13         DO 40 K=1,3
14             L=39+K+4*(J-2)
15             II(K)=NERZ(L)
16         CONTINUE
17         40  IF(NERZ(L+1).EQ.14)II(4)=1
18             CALL DRPF(PARK(J),CST(7),II(4),II(2),II(3),II(1))
19         CONTINUE
C
C   ===== DECODE VEL UP TO 999. KNOTS =====
C
20     60  IF(IOV(5).NE.0) GO TO 80
21         DO 70 K=1,3
22             II(K)=NERZ(51+K)
23         CONTINUE
24         70  CALL VELFF(PARK(5),CST(9),II(2),II(3),II(1))
C
C   ===== DECODE TIME UP TO 24*3600. SECONDS =====
C
25     80  DO 85 K=2,4
26         II(K)=NERZ(K)
27         CONTINUE
28         85  CALL TSECS(II(1),II(3),II(4),II(2))
29             IF(II(1).GT.599)II(1)=599
30             II(3)=NERZ(5)
31             II(4)=NERZ(6)
32             CALL MINHR(II(2),II(3),II(4))
33             IF(II(2).GT.59)II(2)=59
34             II(4)=NERZ(7)
35             II(5)=NERZ(8)
36             CALL MINHR(II(3),II(4),II(5))
37             IF(II(3).GT.23)II(3)=23
38             CALL TIMEFF(T1,II(1),II(2),II(3))
C
39     RETURN
40     END

```

MODULE INFORMATION:

CODE AREA SIZE	= 0271H	6570
VARIABLE AREA SIZE	= 0018H	240
MAXIMUM STACK SIZE	= 000AH	104
57 LINES READ		

0 PROGRAM ERROR(S) IN PROGRAM UNIT DECODA

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

```

LOC  OBJ      LINE      SOURCE STATEMENT
      1 ;      SUBROUTINE DRPFP(DRP,CRAD,ISGN,ITN,IHD,IUN)
      2 ;      NAME DRPFP
      3 ;      EVALUATES DRIFT,ROLL,OR PITCH IN RADIANS
      4 ;      DRP=FLOAT((IUN+10*(ITN+10*IHD))*ISGN)/CRAD
      5 ;      WHERE CRAD=573.
      6 ;
0013  11      DIV      EQU      13H
0012  12      MULT     EQU      12H
006C  13      IAD      EQU      6CH
006E  14      IMULT    EQU      6EH
001D  15      FLOAT     EQU      1DH
      16 ;
      17      EXTRN   INTLOD,AMDCMD,AMDSTR,AMDLOD
      18      PUBLIC  DRPFP
      19 ;
      20      CSEG
      21 ;
0000  E1      22 DRPFP:  POP      H          ;SAVE RETURN ADDR
0001  CD0000  E      23      CALL   INTLOD   ;IHD ↑
0004  014900  C      24      LXI   B,TEN
0007  CD0000  E      25      CALL   INTLOD   ;10 ↑
000A  3E6E    E      26      MVI   A,IMULT
000C  CD0000  E      27      CALL   AMDCMD   ;X
000F  C1      28      POP   B
0010  CD0000  E      29      CALL   INTLOD   ;ITN
0013  3E6C    E      30      MVI   A,IAD
0015  CD0000  E      31      CALL   AMDCMD   ;+
0018  014900  C      32      LXI   B,TEN
001B  CD0000  E      33      CALL   INTLOD   ;10 ↑
001E  3E6E    E      34      MVI   A,IMULT
0020  CD0000  E      35      CALL   AMDCMD   ;X
0023  42      36      MOV   B,D
0024  4B      37      MOV   C,E
0025  CD0000  E      38      CALL   INTLOD   ;IUN ↑
0028  3E6C    E      39      MVI   A,IAD
002A  CD0000  E      40      CALL   AMDCMD   ;+
002D  C1      41      POP   B
002E  CD0000  E      42      CALL   INTLOD   ;ISGN ↑
0031  3E6E    E      43      MVI   A,IMULT
0033  CD0000  E      44      CALL   AMDCMD   ;X
0036  3E1D    E      45      MVI   A,FLOAT
0038  CD0000  E      46      CALL   AMDCMD   ;FLOAT
003B  C1      47      POP   B
003C  CD0000  E      48      CALL   AMDLOD   ;CRAD
003F  3E13    E      49      MVI   A,DIV
0041  CD0000  E      50      CALL   AMDCMD   ;/
0044  C1      51      POP   B
0045  CD0000  E      52      CALL   AMDSTR   ;SAVE RESULT IN DRP
0048  E9      53      PCHL  ;RETURN
0049  0A00    54      DW    10
      55      END

```

PUBLIC SYMBOLS
DRPFP C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLOD E 0000 AMDSTR E 0000 INTLOD E 0000

USER SYMBOLS

ANDCMD E 0000	ANDL0D E 0000	ANDSTR E 0000	DIV A 0013	DRPFP C 0000
FLOAT A 001D	IAD A 006C	IMULT A 003E	INTL0D E 0000	MULT A 0012
TEN C 0049				

ASSEMBLY COMPLETE, NO ERRORS

```

LOC OBJ          LINE          SOURCE STATEMENT
003F C31000      C           55          JMP      STP0      ; ELSE REPLACE THE CHAR WITH '0'
                    56 ;
0042 OC          57 CP2:      INR      C          ;ADD 2 TO THE
0043 OC          58          INR      C          ; REGISTER C TO SET
0044 C34800      C           59          JMP      CP1+1     ; THE UNARY FLAG
                    60 ;
0047 OC          61 CP1:      INR      C          ;SET DECIMAL FLAG BY ADDING 1 TO C(C)
0048 FE2B        62          CPI      02BH     ;CHECK FOR PLUS SIGN
004A CA1000      C           63          JZ       STP0
004D C31200      C           64          JMP      STP       ;CONTINUE
                    65 ;
0050 79          66 CO:      MOV      A,C       ;CHECK DECIMAL FLAG
0051 FE00        67          CPI      00H     ; FOR ZERO, IF NOT
0053 CA4200      C           68          JZ       CP2      ; SET, THEN DO IT
0056 C31200      C           69          JMP      STP       ;ELSE GO TO GET THE NEXT BYTE
                    70 ;
                    71          END
    
```

PUBLIC SYMBOLS
ASCDV C 0000

EXTERNAL SYMBOLS

USER SYMBOLS

```

ASCDV C 0000      CO      C 0050      CC0      C 002A      CP0      C 0039      CP1      C 0047      CP2      C 0042
DC      C 0018
GC      C 0005      STP      C 0012      STP0     C 0010
    
```

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME DVERIF
		2	
		3	CSEG
		4	
		5	PUBLIC ASCDV
		6	
		7	FUNCTION:
		8	EDITS A BUFFER FOR 'F' FORMAT INPUT DATA. LEADING
		9	'+' OR '-' UNARIES ARE ACCEPTED, BUT OTHERS ARE
		10	CONVERTED TO BLANK. ONLY THE FIRST '.' IS ACCEPTED,
		11	ALL OTHER NON-DIGIT CHARACTERS ARE CONVERTED TO ZERO.
		12	LEADING '+' IS REPLACED WITH BLANK, LEADING '-' UNCHANGED.
		13	
		14	REGISTERS UPON ENTRY:
		15	C(BC)=ADDR OF CHAR COUNT IN BUFFER
		16	C(DE)=ADDR OF BUFFER CONTAINING THE CHAR STRING
		17	
		18	REGISTERS ALTERED:
		19	C(HL), C(DE), C(A), C(B), C(C)
		20	
0000	EB	21	ASC DV: XCHG ;SET BUFFER POINTER
0001	0A	22	LDAX B ;GET BYTE COUNT
0002	0E00	23	MVI C,00H ;SET C(C)=0
0004	47	24	MOV B,A ;SET BYTE COUNT REGISTER
0005	7E	25	GC: MOV A,M ;GET A BYTE FROM MEMORY
0006	FE30	26	CPI 030H ;CHECK FOR CHAR '<'
0008	DA1800	27	JC DC ;IF YES, CHECK FOR A ''
000B	FE3A	28	CPI 03AH ;ELSE CHECK FOR '<', THEN VERIFY
000D	DA5000	29	JC CO ; THAT THE UNARY FLAG IS SET
0010	3620	30	STP0: MVI M,020H ;ELSE REPLACE CHAR WITH BLANK
0012	23	31	STP: INX H ;STEP BUFFER POINTER
0013	05	32	DCR B ; AND DECREMENT BYTE COUNT
0014	C20500	33	JNZ GC ;IF NOT LAST BYTE, GO GET NEXT
0017	C9	34	RET ; RETURN WHEN DONE....
		35	
0018	FE2E	36	DC: CPI 02EH ;CHECK FOR ''
001A	CA2A00	37	JZ CCO ;IF CHAR IS '', CHECK FOR DECIMAL FLAG SET
001D	FE2D	38	CPI 02DH ; CHECK FOR '-' FIRST
001F	CA3900	39	JZ CPO ;IF CHAR IS '-', CHECK DECIMAL FLAG
0022	FE2B	40	CPI 02BH ; FOR A '+' CAROT
0024	CA3900	41	JZ CPO ;IF '+', CHECK DECIMAL FLAG SET
0027	C31000	42	JMP STP0 ;IF NOT SET, REPLACE THE CHAR WITH '0'
		43	
002A	79	44	CCO: MOV A,C ;CHECK THE DECIMAL FLAG
002B	FE00	45	CPI 00H ; TO SEE IF IT IS SET
002D	CA4700	46	JZ CPO ;IF NOT SET, SET IT
0030	79	47	MOV A,C ;IF SET, CHECK THE UNARY FLAG
0031	FE02	48	CPI 02H ;IF UNARY FLAG IS SET, THEN
0033	CA4700	49	JZ CPO ; SET THE DECIMAL FLAG
0036	C31000	50	JMP STP0 ; ELSE REPLACE THE CHAR WITH '0'
		51	
0039	79	52	CPO: MOV A,C ;CHECK DECIMAL FLAG
003A	FE00	53	CPI 00H ;IF FLAG IS NOT SET,
003C	CA4200	54	JZ CP2 ; THEN SET THE UNARY FLAG

```

1      NAME      FDLEV
2      ;
3      ;
4      ;
5      ;
6      ;
7      FDIV      EQU      13H
8      FIX      EQU      1FH
9      IAD      EQU      6CH
10     IDIV     EQU      6FH
11     ISUB    EQU      6DH
12     ;
13     EXTRN   AMDLDB,AMDCHD,INTLDB,INTSTR
14     ;
15     PUBLIC  FDLEV
16     ;
17     CSEG
18     ;
19     FDLEV:   POP      R      ;PICK OF RTN ADDR
20     CALL    AMDLDB ;FROP ↑
21     MOV     R,D
22     MOV     C,E
23     CALL    AMDLDB ;BELF ↑
24     MVI    A,FDIV
25     CALL    AMDCHD ;(FROP/BELF)
26     MVI    A,FIX
27     CALL    AMDCHD ;IFIX(...)
28     POP    R
29     CALL    INTLDB ;NEV ↑
30     MVI    A,IAD
31     CALL    AMDCHD ;IFIX(...)+NEV
32     POP    R
33     CALL    INTLDB ;NFEL ↑
34     LXI    R,TWO
35     CALL    INTLDB ;2 ↑
36     MVI    A,IDIV
37     CALL    AMDCHD ;(NFEL/2)
38     MVI    A,ISUB
39     CALL    AMDCHD ;(IFIX(...)+NEV-(NFEL/2)
40     POP    R
41     CALL    INTSTR ;IFDL=
42     PCHL   ;RETURN
43     ;
44     TWO:   DW      2
45     ;
46     END

```

PUBLIC SYMBOLS
FDLEV C 0000

EXTERNAL SYMBOLS
AMDCHD E 0000 AMDLDB E 0000 INTLDB E 0000 INTSTR E 0000

USER SYMBOLS
AMDCHD E 0000 AMDLDB E 0000 FDIV A 0013 FDLEV C 0000 FIX A 001F
IAD A 006C IDIV A 006F INTLDB E 0000 INTSTR E 0000 ISUB A 006D
TWO C 0035

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME FDL0D
		2	;
		3	;
		4	SUBROUTINE FDL0D(IFDL,NFEL,NOD,FDCP,DELF)
		5	;
		6	::: IFDL=IFIX(FDOP/DELF+0.5)+NOD-(NFEL-1)/2
0013		7	FDIV EQU 13H
0010		8	FADD EQU 10H
001F		9	FIX EQU 1FH
006C		10	IAD EQU 6CH
006D		11	ISUR EQU 6DH
006F		12	IDIV EQU 6FH
		13	;
		14	EXTRN AMDL0D,AMDCMD,INTL0D,INTSTR,GIVE
		15	;
		16	PUBLIC FDL0D
		17	;
		18	CSEG
		19	;
0000	E1	20	FDL0D: FOP H ;PULL RTN ADDR OFF STK
0001	CD0000	21	CALL AMDL0D ;FDOP ↑
0004	42	22	MOV B,D
0005	4B	23	MOV C,E
0006	CD0000	24	CALL AMDL0D ;DELF ↑
0009	3E13	25	MVI A,FDIV
000B	CD0000	26	CALL AMDCMD ;FDOP/DELF
000E	114B00	27	LXI D,HALF
0011	CD0000	28	CALL GIVE ;0.5 ↑
0014	3E10	29	MVI A,FADD
0016	CD0000	30	CALL AMDCMD ;(FDOP/DELF)+.5
0019	3E1F	31	MVI A,FIX
001B	CD0000	32	CALL AMDCMD ;IFIX(F/D+.5)
001E	C1	33	POP B
001F	CD0000	34	CALL INTL0D ;NOD ↑
0022	3E6C	35	MVI A,IAD
0024	CD0000	36	CALL AMDCMD ;IFIX(...)+NOD
0027	C1	37	POP B
0028	CD0000	38	CALL INTL0D ;NFEL ↑
002B	014F00	39	LXI B,ONE
002E	CD0000	40	CALL INTL0D ;ONE ↑
0031	3E6D	41	MVI A,ISUR
0033	CD0000	42	CALL AMDCMD ;(NFEL-1)
0036	015100	43	LXI B,TWO
0039	CD0000	44	CALL INTL0D ;2 ↑
003C	3E6F	45	MVI A,IDIV
003E	CD0000	46	CALL AMDCMD ;(...)/2
0041	3E6D	47	MVI A,ISUR
0043	CD0000	48	CALL AMDCMD ;IFIX(...)-(...)/2
0046	C1	49	FOP B
0047	CD0000	50	CALL INTSTR ;IFDL= "
004A	E9	51	FCHL ;RETURN
		52	;
004B	00	53	HALF: DB 00H,00H,80H,00H
004C	00		

LOC	OBJ	LINE	SOURCE STATEMENT
004D	80		
004E	00		
004F	0100	54 ONE:	DW 1
0051	0200	55 TWO:	DW 2
		56	;
		57	END

PUBLIC SYMBOLS
FDLDD C 0000

EXTERNAL SYMBOLS
AMDCHD E 0000 AMDLDD E 0000 GIVE E 0000 INTLDD E 0000 INTSTR E 0000

USER SYMBOLS
AMDCHD E 0000 AMDLDD E 0000 FADD A 0010 FDIV A 0013 FPLDD C 0000 FIX A 001F
GIVE E 0000
HALF C 004B IAD A 006C IBIV A 006F INTLDD E 0000 INTSTR E 0000 ISUB A 005E
ONE C 004F
TWO C 0051

ASSEMBLY COMPLETE, NO ERRORS

ISIS-II FORTRAN-80 V2.0 COMPILE OF PROGRAM UNIT GAIN
OBJECT MODULE PLACED IN IF1:GAIN.OBJ
COMPILER INVOKED BY: FORT80 IF1:GAIN.SRC DEBUG PAGELNGTH(77)

```

1      SUBROUTINE GAIN(ANG,PCH,IGTBL,IGTBC,NPZN,IBND,GSQ)
2      INTEGER*2 IGTBL(80,4),IGTBC(31,4)
      C
      C      *** L-BAND TABLES ***
      C      TABLE FOR EACH FREQ/POLARIZ COMBO, 80 VALUES IN EACH,
      C      ORDERED FOR ANGLES FROM -70 TO +10 DEGREES
      C      INPUT ANG & PCH IN RADIAN
3      COMMON/CG/CL(6),CC(7,2),CST(9)
      C
4      CALL AMDSUB(AGL,ANG,PCH)
5      GO TO (5,10),IBND
6      IN=80
7      CALL AINDX(INDX,IN,CST(4),AGL,CST(3))
8      IF(INDX.LE.0)INDX=1
9      IF(INDX.GE.81)INDX=IN
10     CALL AFLOAT(IGTBL(INDX,NPZN),GSQ)
11     CALL AMDIV(GSQ,GSQ,CST(4))
12     RETURN
      C
      C*** C-BAND PATTERN ***
      C      TABLE FOR EACH FREQ/POLZ COMBO, 31 VALUES IN EACH,
      C      ORDERED FOR ANGLES FROM 0 TO -60 DEGREES
13     10 CALL AMDMUL(D,AGL,CST(3))
14     D=ABS(D)
15     IN=1
16     CALL CINDX(INDX,IN,CST(2),CST(6),D)
17     IF(INDX.GE.31)GO TO 15
18     IF(INDX.LE.0)INDX=1
19     CALL AFLOAT(INDX,DG)
20     CALL AMDMUL(DG,DG,CST(2))
21     CALL AMDSUB(DG,DG,CST(2))
22     CALL AMDSUB(DG,D,DG)
23     CALL AMDGN(DGN,CST(2),DG,CST(4),IGTBC(INDX+1,NPZN),IGTBC(INDX,NPZN))
24     CALL AMDGSO(GSQ,DGN,IGTBC(INDX,NPZN),CST(4))
25     RETURN
26     15 CALL AFLOAT(IGTBC(31,NPZN),GSQ)
27     CALL AMDIV(GSQ,GSQ,CST(4))
28     RETURN
      C
29     END

```

MODULE INFORMATION:

CODE AREA SIZE = 0110CH 476D
VARIABLE AREA SIZE = 0022H 34D
MAXIMUM STACK SIZE = 000CH 12D
43 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT GAIN

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME GANC
		2	SUBROUTINE GANC(Z,C1,C2,C3,C4,C5,C6,C7,F)
		3	;
		4	;
		5	;; Z=C1+F*(C2+F*(C3+F*(C4+C5*F)))*(C6+C7/F)/F
		6	Z=-Z
0010		8	FADD EQU 10H
0011		9	SURT EQU 11H
0012		10	MULT EQU 12H
0013		11	FDIV EQU 13H
0015		12	FCHS EQU 15H
		13	;
		14	EXTRN AMDLDD,AMDCHD,AMSTR
		15	;
		16	PUBLIC GANC
		17	;
		18	CSEG
		19	;
0000	E1	20	GANC: POP H ;PICK OFF RETURN ADDR
0001	220000	21	SHLD ISAVE ;SAVE IT
0004	62	22	MOV H,D
0005	6E	23	MOV L,E ;SAVE F ADDR IN R(C,HL)
0006	CD0000	24	CALL AMDLDD ;C7 ↑
0009	42	25	MOV E,D
000A	4E	26	MOV C,E
000B	CD0000	27	CALL AMDLDD ;F ↑
000E	3E13	28	MVI A,FDIV
0010	CD0000	29	CALL AMDCHD ;C7/F
0013	C1	30	POP E
0014	CD0000	31	CALL AMDLDD ;C6 ↑
0017	3E10	32	MVI A,FADD
0019	CD0000	33	CALL AMDCHD ;(C6+C7/F)
001C	44	34	MOV B,H
001D	4D	35	MOV C,L
001E	CD0000	36	CALL AMDLDD ;F ↑
0021	3E13	37	MVI A,FDIV
0023	CD0000	38	CALL AMDCHD ;(...)/F
0026	C1	39	POP E
0027	CD0000	40	CALL AMDLDD ;C5 ↑
002A	44	41	MOV B,H
002B	4D	42	MOV C,L
002C	CD0000	43	CALL AMDLDD ;F ↑
002F	3E12	44	MVI A,MULT
0031	CD0000	45	CALL AMDCHD ;C5*F
0034	C1	46	POP E
0035	CD0000	47	CALL AMDLDD ;C4 ↑
0038	3E10	48	MVI A,FADD
003A	CD0000	49	CALL AMDCHD ;(C4+C5*F)
003D	44	50	MOV B,H
003E	4D	51	MOV C,L
003F	CD0000	52	CALL AMDLDD ;F ↑
0042	3E12	53	MVI A,MULT
0044	CD0000	54	CALL AMDCHD ;(...)*F
0047	C1	55	POP E
0048	CD0000	56	CALL AMDLDD ;C3 ↑
004B	3E10	57	MVI A,FADD
004D	CD0000	58	CALL AMDCHD ;C3+(...)*F
0050	44	59	MOV B,H
0051	4D	60	MOV C,L
0052	CD0000	61	CALL AMDLDD ;F ↑
0055	3E12	62	MVI A,MULT
0057	CD0000	63	CALL AMDCHD ;F*(C3+F*(...))
005A	C1	64	POP E
005B	CD0000	65	CALL AMDLDD ;C2 ↑

LOC	OBJ	LINE	SOURCE STATEMENT
005E	3E10	66	MVI A,FADD
0060	CD0000	E 67	CALL AMDCMD #C2+F*(C3+F*(...))
0063	44	68	MOV B,H
0064	4D	69	MOV C,L
0065	CD0000	E 70	CALL AMDLDD #F ↑
006C	3E12	71	MVI A,MULT
006A	CD0000	E 72	CALL AMDCMD #F*(C2+F*(...))
006D	C1	73	POP B
006E	CD0000	E 74	CALL AMDLDD #C1 ↑
0071	3E10	75	MVI A,FADD
0073	CD0000	E 76	CALL AMDCMD #C1+F*(C2+F*(...))
0076	3E10	77	MVI A,FADD
0078	CD0000	E 78	CALL AMDCMD #C1+F*(C2+F*(...))+(C6+...)/F
007B	3E15	79	MVI A,FCHS
007D	CD0000	E 80	CALL AMDCMD #Z=-Z
0080	C1	81	POP B
0081	CD0000	E 82	CALL AMDSTR #SAVE RESULT AT Z
0084	2A000C	D 83	LHLD ISAVE
0087	E9	84	PCHL #RETURN
		85	;
		86	DSEG
		87	;
0000		88	ISAVE: DS 2
		89	;
		90	END

PUBLIC SYMBOLS
GAMC C 0000

EXTERNAL SYMBOLS
AMDCHD E 0000 AMDLDD E 0000 AMDSTR E 0000

USER SYMBOLS
AMDCHD E 0000 AMDLDD E 0000 AMDSTR E 0000 FADD A 0010 FCHS A 0015
FDIV A 0013 GAMC C 0000 ISAVE D 0000 MULT A 0012 SUBT A 0011

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME GAML
		2	SUBROUTINE GAML(Z,C1,C4,C5,C2,F,C3)
		3	;
		4	;;; Z=-(C1+F*(C2+C3*F)(C4+C5/F)/F)
		5	;
0012		6	MULT EQU 12H
0011		7	SUBT EQU 11H
0010		8	FADD EQU 10H
0015		9	FCHS EQU 15H
0013		10	FDIV EQU 13H
0017		11	PUSHT EQU 17H
		12	;
		13	EXTRN AMDCMD,AMDSTR,AMDLOD
		14	;
		15	PUBLIC GAML
		16	;
		17	CSEG
		18	;
		19	GAML:
0000	E1	20	POP H ;SAVE RETURN ADDR
0001	220000	21	SHLD H,ISAVE ;AT INTERNAL BUFFER
0004	60	22	MOV H,R
0005	67	23	MOV L,C ;SAVE ADDR OF F IN R(C,HL)
0006	CD0000	24	CALL AMDLOD ;F ↑
0009	3E17	25	MVI A,FUSHT
000B	CD0000	26	CALL AMDCMD ;F ↑
000E	42	27	MOV B,D
000F	4B	28	MOV C,E
0010	CD0000	29	CALL AMDLOD ;C3 ↑
0013	3E12	30	MVI A,MULT
0015	CD0000	31	CALL AMDCMD ;C3*F ;F
0018	C1	32	POP B
0019	CD0000	33	CALL AMDLOD ;C2 ↑
001C	3E10	34	MVI A,FADD
001E	CD0000	35	CALL AMDCMD ;(C2+C3*F) ;F
0021	3E12	36	MVI A,MULT
0023	CD0000	37	CALL AMDCMD ;(C2+C3*F)*F
0026	010200	38	LXI B,ISAVE+2
0029	CD0000	39	CALL AMDSTR ;SAVE RESULT AT INTERNAL BUFFER
002C	C1	40	POP B
002D	CD0000	41	CALL AMDLOD ;C5 ↑
0030	44	42	MOV B,H
0031	4D	43	MOV C,L
0032	CD0000	44	CALL AMDLOD ;F ↑
0035	3E13	45	MVI A,FDIV
0037	CD0000	46	CALL AMDCMD ;C5/F
003A	C1	47	POP B
003B	CD0000	48	CALL AMDLOD ;C4 ↑
003E	3E10	49	MVI A,FADD
0040	CD0000	50	CALL AMDCMD ;(C4+C5/F)
0043	44	51	MOV B,H
0044	4D	52	MOV C,L
0045	CD0000	53	CALL AMDLOD ;F ↑
0048	3E13	54	MVI A,FDIV
004A	CD0000	55	CALL AMDCMD ;(...)/F
004D	010200	56	LXI B,ISAVE+2
0050	CD0000	57	CALL AMDLOD ;(C2+C3*F)*F ↑
0053	3E10	58	MVI A,FADD
0055	CD0000	59	CALL AMDCMD ;(...)/F + (...)*F
0058	C1	60	POP B
0059	CD0000	61	CALL AMDLOD ;C1 ↑
005C	3E10	62	MVI A,FADD
005E	CD0000	63	CALL AMDCMD ; ↑
0061	3E15	64	MVI A,FCHS
0063	CD0000	65	CALL AMDCMD ;Z=-Z
0066	C1	66	POP B

LOC	OBJ		LINE	SOURCE STATEMENT
0067	C00000	E	66	CALL AMDSTR ;SAVE RESULT AT Z
006A	2A0000	D	67	LHLD ISAVE
006D	E9		68	PCHL ;RETURN
			69	;
			70	DSEG
			71	;
0000			72	ISAVE: DS 6
			73	END

PUBLIC SYMBOLS
GAML C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000

USER SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000 FADD A 0010 FCHS A 0015
FDIV A 0013 GAML C 0000 ISAVE D 0000 MULT A 0012 PUSH A 0017
SUBT A 0011

ASSEMBLY COMPLETE, NO ERRORS

ISIS-II FORTRAN-80 V2.0 COMPILATION OF PROGRAM UNIT GAMMA
OBJECT MODULE PLACED IN IF1:GAMMA.OBJ
COMPILER INVOKED BY: FORT80 IF1:GAMMA.SRC DEBUG

```
1      SUBROUTINE GAMMA(FRQ,IB,NPZ,Z)
      C
      C      ::: EVALUATES ROLL-OFF FOR FREQUENCY,BAND,POLARIZATION COMBOS
      C
2      C      COMMON/GG/CL(6),CC(7,2),CST(9)
      C
3      CALL AHDIV(F,FRQ,CL(6))
4      GO TO(10,15),IP
      C
5      C      ::: L-BAND DATA :::
6      10 CALL GAHL(Z,CL(1),CL(4),CL(5),CL(2),F,CL(3))
      RETURN
      C
7      C      ::: C-BAND DATA :::
      C
8      15 GO TO (17,17,16,16),NPZ
9      C      ::: NPZ: 1=VV,2=VH,3=HH,4=HV
10     C
11     16 CALL GAMC(Z,CC(1,2),CC(2,2),CC(3,2),CC(4,2),CC(5,2),CC(6,2),CC(7,2),F)
12     RETURN
17     CALL GAMC(Z,CC(1,1),CC(2,1),CC(3,1),CC(4,1),CC(5,1),CC(6,1),CC(7,1),F)
      RETURN
      END
```

MODULE INFORMATION:

CODE AREA SIZE = 00B1H 177D
VARIABLE AREA SIZE = 000CH 12D
MAXIMUM STACK SIZE = 0010H 16D
22 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT GAMMA

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

ISIS-II FORTRAN-80 V2.0 COMPILATION OF PROGRAM UNIT GETVLU
OBJECT MODULE PLACED IN :F1:GETVLU.OBJ
COMPILER INVOKED BY: FORT80 :F1:GETVLU.SRC DEBUG SYMBOLS PAGELNGTH(53)

```
1      SUBROUTINE GETVLU(CIOBUF,IOBUFF,N10,N20,ISTAT,IERR,XNUM)
2      EXTERNAL MSGOUT,KEYBRD,ASCDV,CRLF
3      INTEGER*1 IOBUFF(30),N10,N20
4      CHARACTER*30 CIOBUF
5      C
6      CALL CRLF
7      IERR=0
8      5  WRITE(CIOBUF,10,IOSTAT=ISTAT,ERR=50)
9      10  FORMAT(5X,'ENTER VALUE:',13X)
10     15  CALL MSGOUT(N20,IOBUFF)
11     20  CALL KEYBRD(N10,IOBUFF)
12     25  CALL ASCDV(N10,IOBUFF)
13     30  READ(CIOBUF,35,IOSTAT=ISTAT,ERR=51,END=52) XNUM
14     35  FORMAT(F5.0)
15     40  RETURN
16     C
17     50  IERR=1
18     GO TO 40
19     51  IERR=2
20     GO TO 40
21     52  IERR=3
22     GO TO 40
23     C
24     END
```

SYMBOL LISTING

<u>DEFN</u>	<u>ADDR</u>	<u>SIZE</u>	<u>NAME, ATTRIBUTES, AND REFERENCES</u>
8	0000H	10	LABEL
9	008CH	15	LABEL
10	009BH	20	LABEL
11	00A4H	25	LABEL
12	00B0H	30	LABEL
13	0017H	35	LABEL
14	00FAH	40	LABEL
7	0053H	5	LABEL
15	00FBH	50	LABEL
17	0107H	51	LABEL
19	0113H	52	LABEL
	0010H	18	@IOPB INTEGER*2 DIMENSIONED
			ASCDV EXTERNAL SUBROUTINE
	0000H	2	CI0BUF CHARACTER*30 PARAMETER
	0002H	2	CI0BUF@ INTEGER*2
			CRLF EXTERNAL SUBROUTINE
	000CH	2	IERR INTEGER*2 PARAMETER
	0004H	2	IOBUFF INTEGER*1 PARAMETER DIMENSIONED
	000AH	2	ISTAT INTEGER*2 PARAMETER
			KEYBRD EXTERNAL SUBROUTINE
			MSGOUT EXTERNAL SUBROUTINE

FORTRAN COMPILER
PAGE 3

0006H	2	N10	INTEGER*1 PARAMETER
0008H	2	N20	INTEGER*1 PARAMETER
000EH	2	XNUM	REAL*4 PARAMETER

MODULE INFORMATION:

CODE AREA SIZE	=	011FH	287D
VARIABLE AREA SIZE	=	0022H	34D
MAXIMUM STACK SIZE	=	000CH	12D
24 LINES READ			

0 PROGRAM ERROR(S) IN PROGRAM UNIT GETVLU

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

```

LOC OBJ      LINE      SOURCE STATEMENT
      1          NAME      GXI
      2          ;
      3          ;
      4          ;
      5          ;
      6          ;
      7          ;
      8          ;
      9          ;
     10          ;
     11          ;
     12          ;
     13          ;
     14          ;
     15          ;
     16          ;
     17          ;
     18          ;
     19          ;
     20          ;
     21          ;
     22          ;
     23          ;
     24          ;
     25          ;
     26          ;
     27          ;
     28          ;
     29          ;
     30          ;
     31          ;
     32          ;
     33          ;
     34          ;
     35          ;
     36          ;
     37          ;
     38          ;
     39          ;
     40          ;
     41          ;
     42          ;
     43          ;
     44          ;
     45          ;
     46          ;
     47          ;
     48          ;
     49          ;

```

0003 EQU 03H
0002 EQU 02H
0012 EQU 12H
0004 EQU 04H
0010 EQU 10H
0019 EQU 19H

```

EXTRN AMDL0D,AMDCHD,AMDSTR
PUBLIC GXI
CSEG
GXI: POP H ;SAVE RTN ADDR
CALL AMDL0D ;RL ↑
MVI A,TAN
CALL AMDCHD ;TAN(RL)
MOV R,D
MOV C,E
CALL AMDL0D ;DR ↑
MVI A,SIN
CALL AMDCHD ;SIN(DR)
MVI A,MULT
CALL AMDCHD ;SIN*DANR
POP R
CALL AMDL0D ;ALT;SIN*DANR
MVI A,MULT
CALL AMDCHD ;ALT*SIN*DANR
MOV R,D
MOV C,E
CALL AMDL0D ;DR ↑
MVI A,COS
CALL AMDCHD ;COS(DR)
POP R
CALL AMDL0D ;XT ↑
MVI A,MULT
CALL AMDCHD ;XT*COS(DR);ALT*...!-!-!-!-
MVI A,FADD
CALL AMDCHD ;(XT...)+(ALT*SD*TR)
POP R
CALL AMDSTR ;XI = "
PCHL ;RETURN
END

```

PUBLIC SYMBOLS

GXI C 0000

EXTERNAL SYMBOLS

AMDCHD E 0000 AMDL0D E 0000 AMDSTR E 0000

USER SYMBOLS

AMDCHD E 0000 AMDL0D E 0000 AMDSTR E 0000 COS A 0003 EXCHG A 0019
FADD A 0010 GXI C 0000 MULT A 0012 SIN A 0002 TAN A 0004

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME GXT
		2	;
		3	;
		4	;
		5	;
		6	TAN EQU 04H
0004		7	MULT EQU 12H
0012		8	PUSHT EQU 17H
0017		9	FSUB EQU 11H
0011		10	SQRT EQU 01H
0001		11	;
		12	EXTRN AMDLOD,AMDCMD,AMDSTR
		13	;
		14	PUBLIC GXT
		15	;
		16	CSEG
		17	;
		18	GXT: POP H ;PICK OF RTN ADDR
0000	E1	19	CALL AMDLOD ;ANGL ↑
0001	CD0000	E 19	
0004	3E04	20	MVI A,TAN
0006	CD0000	E 21	CALL AMDCMD ;TAN(ANGL)
0009	42	22	MOV B,D
000A	4B	23	MOV C,E
000B	CD0000	E 24	CALL AMDLOD ;ALT ↑
000E	3E12	25	MVI A,MULT
0010	CD0000	E 26	CALL AMDCMD ;ALT*TAN(ANGL)
0013	3E17	27	MVI A,PUSHT
0015	CD0000	E 28	CALL AMDCMD ;(ATA):(ATA)
0018	3E12	29	MVI A,MULT
001A	CD0000	E 30	CALL AMDCMD ;(ATA)**2
001D	C1	31	POP B
001E	CD0000	E 32	CALL AMDLOD ;RL ↑
0021	3E04	33	MVI A,TAN
0023	CD0000	E 34	CALL AMDCMD ;TAN(RL):(.**2)
0026	42	35	MOV B,D
0027	4B	36	MOV C,E
0028	CD0000	E 37	CALL AMDLOD ;ALT ↑
002B	3E12	38	MVI A,MULT
002D	CD0000	E 39	CALL AMDCMD ;ALT*TANR:(.**2)
0030	3E17	40	MVI A,PUSHT
0032	CD0000	E 41	CALL AMDCMD ;ATR:ATR:(.**2)
0035	3E12	42	MVI A,MULT
0037	CD0000	E 43	CALL AMDCMD ;ATR**2:(.**2)
003A	3E11	44	MVI A,FSUB
003C	CD0000	E 45	CALL AMDCMD ;ATA**2 - ATR**2
003F	3E01	46	MVI A,SQRT
0041	CD0000	E 47	CALL AMDCMD ;SQRT(")
0044	C1	48	POP B
0045	CD0000	E 49	CALL AMDSTR ;XT = "
0048	E9	50	PCHL ;RETURN
		51	;
		52	END

PUBLIC SYMBOLS
GXT C 0000

EXTERNAL SYMBOLS
AMDCHD E 0000 AMDLUD E 0000 AMDSTR E 0000

USER SYMBOLS
AMDCHD E 0000 AMDLUD E 0000 AMDSTR E 0000 FSUB A 0011 GXT C 0000 MULT A 0012
PUSHT A 0017
SQRT A 0001 TAN A 0004

ASSEMBLY COMPLETE, NO ERRORS

ORIGINAL PRICE \$
OF EQUAL QUALITY

```

LOC OBJ      LINE      SOURCE STATEMENT
1           ;          NAME      GYI
2           ;
3           ;          SUBROUTINE GYI(YI,XT,ALT,RL,DR)
4           ;          !!! YI=XT*SIN(DR)-ALT*TAN(RL)*COS(DR)
5           ;
0003        6      COS      EQU      03H
0002        7      SIN      EQU      02H
0012        8      MULT     EQU      12H
0004        9      TAN      EQU      04H
0011       10      FSUB     EQU      11H
0019       11      EXCHG    EQU      19H
12         ;
13         EXTRN  AMDLDD,AMDCHD,AMDSTR
14         ;
15         PUBLIC GYI
16         ;
17         CSEG
18         ;
0000 E1       19      GYI:    POP      H          ;SAVE RTN ADDR
0001 CD0000  E 20      CALL    AMDLDD     ;RL ↑
0004 3E04    21      MVI     A,TAN
0006 CD0000  E 22      CALL    AMDCHD     ;TAN(RL)
0009 42     23      MOV     R,D
000A 4B     24      MOV     C,E
000B CD0000  E 25      CALL    AMDLDD     ;DR ↑
000E 3E03    26      MVI     A,COS
0010 CD0000  E 27      CALL    AMDCHD     ;COS(DR)
0013 3E12    28      MVI     A,MULT
0015 CD0000  E 29      CALL    AMDCHD     ;COSD*TANR
0018 C1     30      POP     B
0019 CD0000  E 31      CALL    AMDLDD     ;ALT: COSD*TANR
001C 3E12    32      MVI     A,MULT
001E CD0000  E 33      CALL    AMDCHD     ;ALT: COSD*TANR
0021 42     34      MOV     R,D
0022 4B     35      MOV     C,E
0023 CD0000  E 36      CALL    AMDLDD     ;DR ↑
0026 3E02    37      MVI     A,SIN
0028 CD0000  E 38      CALL    AMDCHD     ;SIN(DR)
002B C1     39      POP     B
002C CD0000  E 40      CALL    AMDLDD     ;XT ↑
002F 3E12    41      MVI     A,MULT
0031 CD0000  E 42      CALL    AMDCHD     ;XT*SIN(DR)
0034 3E19    43      MVI     A,EXCHG
0036 CD0000  E 44      CALL    AMDCHD     ;ALT*SD*TR:XT*SIN(DR):---
0039 3E11    45      MVI     A,FSUB
003B CD0000  E 46      CALL    AMDCHD     ;(XT,...)-(ALT*SD*TR)
003E C1     47      POP     B
003F CD0000  E 48      CALL    AMDSTR     ;YI = "
0042 E9     49      PCHL
50         ;
51         END

```

PUBLIC SYMBOLS
GYI C 0000

EXTERNAL SYMBOLS
AMDCHD E 0000 AMDLDD E 0000 AMDSTR E 0000

USER SYMBOLS
AMDCHD E 0000 AMDLDD E 0000 AMDSTR E 0000 COS A 0003 EXCHG A 0019
FSUB A 0011 GYI C 0000 MULT A 0012 SIN A 0002 TAN A 0004

ASSEMBLY COMPLETE, NO ERRORS

ASM80 (F1)IBELL.SRC DEBUG PAGELNGTH(75) PAGEWIDTH(90)

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0 IBELL PAGE 1

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME IBELL
		2	;
		3	;
		4	FUNCTION: SENDS A BELL CHARACTER TO THE CRT
		5	;
		6	EXTRN ECHO
		7	;
		8	CSEG
		9	;
		10	PUBLIC IBELL
0000	0E07	11	IBELL: MVI C,07H ;GET BELL CHARACTER
0002	CD0000	12	CALL ECHO ;SEND IT TO CRT
0005	C9	13	RET
		14	END

PUBLIC SYMBOLS
IBELL C 0000

EXTERNAL SYMBOLS
ECHO E 0000

USER SYMBOLS
ECHO E 0000 IBELL C 0000

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME IBFILL
		2	
		3	
		4	FUNCTION:
		5	FILL IOUT BUFFER WITH 099H.
		6	EACH LINE IS 128 BYTES.
		7	LINE COUNT PASSED IN R0(BC).
		8	LINE COUNT IS LESS THAN 256.
		9	
		10	BUFFER ADDR PASSED IN R0(DE).
		11	
		12	CSEG
		13	
		14	PUBLIC IBFILL
0000	EB	14	IBFILL: XCHG
0001	0A	15	LDAX
0002	47	15	MOV
0003	0E80	17	LL1: MVI
0005	3699	18	LL2: MVI
0007	23	19	INX
0008	0B	20	DCR
0009	C20500	21	JNZ
000C	05	22	DCR
000E	C20300	23	JNZ
0010	C9	24	RET
		25	
		26	END

PUBLIC SYMBOLS
IBFILL C 0000

EXTERNAL SYMBOLS

USER SYMBOLS
IBFILL C 0000 LL1 C 0003 LL2 C 0005

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME ICRLF
		2	
		3	FUNCTION IS TO OUTPUT A CARRIAGE-RETURN AND LINE FEED
		4	
		5	REGISTERS ALTERED: C(C),C(B),C(A)
		6	
		7	CSEG
		8	PUBLIC CRLF,ECHO,CO,DELAY
		9	
00E4		10	PORTA EQU 0E4H
00E6		11	PORTC EQU 0E6H
001B		12	FSC EQU 01BH ;ESCAPE CHARACTER
000D		13	CR EQU 0DH ;CARRIAGE RETURN CHARACTER
000A		14	LF EQU 0AH ;LINE FEED CHARACTER
		15	
0000	0E0D	16	CRLF: MVI C,0DH ;PUT C/R RETURN CHAR INTO REG C
0002	CD0600	17	CALL ECHO ;SEND TO OUTPUT UNIT
0005	C9	18	RET
		19	
0006	41	20	ECHO: MOV B,C ;SAVE ARG
0007	3E1B	21	MVI A,ESC ;
0009	B8	22	CMP B ;ARE WE ECHOING AN ESCAPE?
000A	C20F00	23	JNZ ECH05 ;IF NO, BRANCH
000D	0E24	24	MVI C,'\$' ;IF YES, SEND OUT A '\$'
000F	CD2800	25	CALL CO ;DO OUTPUT THROUGH CO ROUTINE
0012	CD4100	26	CALL BUSY ;OUT TO PRINTER
0015	3E0D	27	MVI A,CR ;
0017	B8	28	CMP B ;WAS CHAR = CR
0018	C22600	29	JNZ ECH10 ;IF NO, CONTINUE
001B	0E0A	30	MVI C,LF ;IF YES, SEND OUT A LINE FEED ALSO
001D	CD2800	31	CALL CO ;
0020	CD4100	32	CALL BUSY ;OUT TO PRINTER
0023	CD3300	33	CALL DELAY ;WAIT FOR CARRIAGE TO GET HOME
0026	48	34	MOV C,B ;RESTORE ARG
0027	C9	35	RET
		36	
0028	DBCD	37	CO: IN OCDH ;GET STATUS OF CONSOLE
002A	E801	38	ANI 01H ;SEE IF XMTR READY
002C	CA2800	39	JZ CO ;NO - TRY AGAIN
002F	79	40	MOV A,C ;ELSE MOVE CHAR TO A FOR OUTPUT
0030	D3CC	41	OUT OCCH ;SEND TO CONSOLE
0032	C9	42	RET
		43	
		44	
0033	3E16	45	DELAY: MVI A,16H ;SET REG A TO 16H
0035	0E16	46	MVI C,16H ; " " C " "
0037	3D	47	LOOP: DCR A
0038	00	48	NOP
0039	C23700	49	JNZ LOOP
003C	0D	50	DCR C
003D	C23700	51	JNZ LOOP
0040	C9	52	RET
		53	
0041	DBE6	54	BUSY: IN PORTC ;GET STATUS

LOC	OBJ	LINE	SOURCE STATEMENT
0043	E630	55	ANI 30H ;MASK LOWER NIB,UNUSED UPPER BITS
0045	320000	56	STA LOC ;SAVE STATUS
0048	FE00	57	CPI 00H ;CHECK FOR SWITCH +BUSY,LOW TRUE
004A	CA5E00	58	JZ PRINT ;IF TRUE PRINT
004D	E620	59	ANI 20H ;MASK OFF BUSY
004F	FE00	60	CPI 00H ;CHECK FOR TRUE
0051	C26D00	61	JNZ INCRMT
0054	3A0000	62	LDA LOC ;RECALL STATUS BYTE
0057	E610	63	ANI 10H ;MASK OFF SWITCH STATUS
0059	FE00	64	CPI 00H ;CHECK FOR BUSY
005B	C24100	65	JNZ BUSY
005E	7E	66	PRINT: MOV A,M
005F	2F	67	CMA ;GET BYTE AND COMPLIMENT
0060	D3E4	68	OUT PORTA ;SEND OUT
0062	3E00	69	MVI A,00H ;SEND
0064	D3E6	70	OUT PORTC ; STROBE
0066	00	71	NOP ;THEN
0067	00	72	NOP ; DELAY
0068	00	73	NOP ; A LITTLE
0069	3E01	74	MVI A,01H ;SEND HIGH TO PRINTER
006B	D3E6	75	OUT PORTC
006D	C9	76	INCRMT: RET
		77	;
		78	DSEG
		79	;
0000		80	LOC: DS 1
		81	END

PUBLIC SYMBOLS

CO C 0028 CRLF C 0000 DELAY C 0033 ECHO C 0006

EXTERNAL SYMBOLS

USER SYMBOLS

BUSY C 0041 CO C 0028 CR A 000D CRLF C 0000 DELAY C 0033 ECHO5 C 00
 ECHO C 0006 ESC A 001B INCRMT C 006D LF A 000A LOC D 0000 LOOP C 00
 PORTA A 00E4 PRINT C 005E

ASSEMBLY COMPLETE, NO ERRORS

ORIGINAL PAGE IS
 OF POOR QUALITY

ISIS-II FORTRAN-80 V2.0 COMPILATION OF PROGRAM UNIT IKEYI
OBJECT MODULE PLACED IN :F1:IKEYI.OBJ
COMPILER INVOKED BY: FORT80 :F1:IKEYI.SRC DEBUG PAGELNGTH(77) PAGESWIDTH(90)

```
1          SUBROUTINE IKEYI
2          INTEGER*1 KESC
3          C
4          C : : : SETS FLAG FOR KEYBOARD 'ESC' INTERRUPT
5          C
6          COMMON/A/IEOC,KESC
7          C
8          SAVE/A/
9          C
10         KESC=2
11         RETURN
12         END
```

MODULE INFORMATION:

CODE AREA SIZE = 0006H 6D
VARIABLE AREA SIZE = 0000H 0D
MAXIMUM STACK SIZE = 0000H 0D
12 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT IKEYI

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

```

LOC 03J      LINE      SOURCE STATEMENT
1
2 *****
3 *****
4 *****
5 *****
6 *****
7 *****
8 *****
9 *****
10 *****
11 *****
12 *****
13 *****
14 *****
15 *****
16 *****
17 *****
18 *****
19 *****
20 *****
21 *****
22 *****
23 *****
24 ICW1      EQU      01FH
25 ICW2      EQU      40H
26 OCW2      EQU      20H
27 IMA       EQU      07H
28 CB255     EQU      0A5H
29 CB255     EQU      0E9H
30 INTEA     EQU      05H
31 INTEB     EQU      05H
32 FILL      EQU      07H
33 FILL      EQU      05CH
34 DB255     EQU      0E9H
35 DATA     EQU      0E7H
36 RATE      EQU      08H
37 LSE       EQU      0AH
38 MSE       EQU      05H
39 UMASK     EQU      0F0H
40 MASK      EQU      0FFH
41 DB251     EQU      0ECH
42 EOI       EQU      20H
43 LOADN     EQU      08H
44 UNSYN     EQU      0EFH
45 UNSTR     EQU      0E7H
46
47
48 ***** INITIALIZE 8255 PIO FOR BI-PHASE-L OUTPUT, NERDAS INPUT ***
49 *****
50 INTPIO:
51 MVI A, CB255 ;CONTROL WORD FOR: PORT A MODE 1 OUTPUT
   OUT CB255 ;PORT B MODE 1 INPUT, GROUP 2 SBC 80/20 E
52 OADR
53 MVI A, INTEA ;ENABLE INTERRUPT UPON ACKNOWLEDGE
   OUT CB255 ;FOR PORT A.
54 MVI A, INTEB ;ENABLE INTERRUPT UPON STROBE
   OUT CB255 ;FOR PORT I
55 MVI A, FILL ;SET BIT 4 OF PORT C
56 OUT CB255 ;TO INDICATE FILL DATA
57 MVI A, FILL ;WRITE FILL DATA TO PORT A
58 OUT CB255 ;FILL DATA NOW LOADED
59 MVI A, DATA ;RESET BIT 4 BI PORT C. ALL DATA
60
001F
0040
0020
0009
00A6
00EB
000E
0005
0007
00E3
00E8
00E9
000B
0003
00FB
00FF
00EC
0020
0008
00EF
00E7

```

```

LOC 000      LINE      SOURCE STATEMENT
0016 D3E8    61      OUT C0255      ;NOW WRITTEN TO PORT A IS INTERPRETED AS R
0018 C9      62      EAL IATA      RET           ;RETURN FROM INI610
63 ;
64 ;
65 ;
66 ;***** INITIALIZE C7T LOADR *****
67 INIC7T:
0019 79      68      MOV A,C      ;PUT IP55(L55) INTO ACCUM
001A E30A    69      OUT L55     ;OUTPUT L55 OF IMA ADDRESS
001C 73      70      MOV A,E      ;MOVE IP55(H55) TO ACCUM
001E D30E    71      OUT M5E     ;OUTPUT M5E OF IMA ADDRESS
001F C9      72      RET           ;RETURN FROM INI027
73 ;
74 ;
75 ;
76 ;***** INITIALIZE INTERRUPT CONTROLLER *****
77 INI257:
0020 F3      78      EI           ;ENABLE 8080 INTERRUPTS
0021 3E1F    79      MVI A,ICW1  ;SET NESTED MODE INTERRUPTS
0023 E3E8    80      OUT 010H   ;CALL TABLE 4 BYTES ADDR.
0025 3E40    81      MVI A,ICW2  ;SET CALL TABLE AT 4000H
0027 E3E9    82      OUT 017H   ;
0029 3EFF    83      MVI A,MASK  ;MASK ALL INTERRUPTS
002B E3L9    84      OUT 017H   ;BEFORE LEAVING INI257
002E FE      85      EI           ;REENABLE 8080 INTERRUPTS
002E C9      86      RET           ;RETURN FROM INI257
87 ;
88 ;
89 ;
90 ;***** MASK SET ROUTINE *****
91 HSKSET:
002F 0A      92      LDAX B     ;MOVE MASK TO ACCUM
0030 D3D9    93      OUT 0E9H   ;OUTPUT TO 0E9H
0032 C9      94      RET           ;RETURN FROM HSKSET
95 ;
96 ;
97 ;
98 ;***** MASK ALL INTERRUPTS ROUTINE *****
99 INTSET:
0033 0A      100     LDAX B     ;PUT INCI INTO ACCUM
0034 FE01    101     CPI 01H   ;SEE IF INCI IS A 1
0036 C13B00 102     JNZ DSABLE ;IF NOT JUMP
0039 FB      103     EI           ;IF A 1 ENABLE INTERRUPT
003A C9      104     RET           ;AND RETURN FROM INTSET
003B F3      105     DSABLE: EI ;IF NOT A 1 ENABLE INTERRUPTS
003C C9      106     RET           ;AND RETURN FROM INTSET
107 ;
108 ;
109 ;
110 ;***** BEGIN INTEGRATION ROUTINE *****
111 CZT:
003E 0A      112     LDAX B     ;MOVE N INTO ACCUM
003E D308    113     OUT LOADR  ;OUTPUT NUMBER OF RECORDS AND BEGIN INTEGRATION
0040 C9      114     RET           ;RETURN FROM CZT
115 ;
116 ;
117 ;
118 ;***** DMA TRANSFER ROUTINE *****
119 CZTR:
0041 E309    120     OUT DMA    ;START DMA TRANSFER
0043 C9      121     RET           ;RETURN FROM CZTR
122 ;
123 ;

```

```

LOC  OBJ      LINE      SOURCE STATEMENT
124 ;
125 ;***** BI-PHASE-L OUTPUT ROUTINE *****;
126 IBIPHLL:
0044 1680      127          MVI D, 80H          ;SET BYTE COUNT TO 120
0045 0A        128 LOAD:        LDAX B          ;GET BYTE FROM DATA BUFFER
0046 0F        129          RRC              ;SWAP NIBBLES
0047 0F        130          RRC
0048 0F        131          RRC
0049 0F        132          RRC
004A 0F        133          OUT 0255        ;WRITE DATA TO 8255
004B E3E9     134          HLT              ;WAIT FOR ACKNOWLEDGE
004C 76        135 BPLISR:    INX B          ;INCREMENT DATA BUFFER POINTER
004D 03        136          MVI A, EOI          ;SEND NON-SPECIFIC END OF INTERRUPT
004E 3E20     137          OUT 025H        ;TO 8259 INTERRUPT CONTROLLER
004F 3E20     138          EI              ;REENABLE 8080 INTERRUPT
0050 E3E9     139          ICR D          ;DECREMENT BYTE COUNT
0051 E3E9     140          JNZ LOAD        ;OUTPUT MORE DATA IF NOT DONE
0052 FE        141          RET              ;RETURN FROM INTERRUPT SERVICE ROUTINE
0053 FE
0054 15
0055 C24600   C 142 ;
0056 C9       143 ;
0057 C9       144 ;
145 ;***** NERIAS INPUT ROUTINE *****;
146 NERI:
0059 0A        147          LIAX B          ;GET ICNT
005A 0F        148          RRC              ;DIVIDE ICNT BY 2 TO CHANGE TO BYTE COUNT
005B E67F     149          ANI 7FH          ;STRIP OFF MSB
005C 47        150          MOV B,A          ;PUT ICNT IN B
005D 3EEF     151          MVI A, UNSYN      ;UNMASK SYNC INTERRUPT
005E E3E9     152          OUT 027H        ;
005F 76        153          HLT              ;WAIT FOR SYNC
0060 FE        154          EI              ;REENABLE 8080 INTERRUPTS
0061 3EE7     155          MVI A, UNSTB      ;UNMASK STROBE INTERRUPT
0062 E3E9     156          OUT 027H        ;
0063 76        157 STB:      HLT              ;WAIT FOR STROBE INTERRUPT
0064 FE        158          EI              ;REENABLE 8080 INTERRUPT UPON RETURN
0065 E3E9     159          IN DATA        ;GET A BYTE OF DATA
0066 0F        160          RRC              ;SWAP
0067 0F        161          RRC              ;
0068 0F        162          RRC              ;
0069 0F        163          RRC              ;
0070 12        164          STAX D          ;STORE IN IOBUF
0071 13        165          INX D          ;INCREMENT IOBUF POINTER
0072 05        166          INR B          ;DECREMENT ICNT
0073 CA7D00   C 167          JZ FN5H          ;JUMP IF ICNT = 0
0074 3E20     168          MVI A, EOI          ;SEND END OF INTERRUPT
0075 E3E9     169          OUT 025H        ;(LEVEL 3)
0076 C36800   C 170          JMP STR          ;GET MORE DATA
0077 3E20     171 FN5H:    MVI A, EOI          ;SEND END OF INTERRUPT
0078 E3E9     172          OUT 025H        ;(LEVEL 3) STROBE INTERRUPT
0079 E3E9     173          OUT 025H        ;(LEVEL 4) SYNC INTERRUPT
0080 3EFF     174          MVI A, MASK        ;MASK ALL INTERRUPTS BEFORE RETURNING
0081 E3E9     175          OUT 027H        ;
0082 C9       176          RET              ;RETURN FROM NERI
0083 C9
0084 C9
0085 C9
0086 C9
0087 C9
177 ;
178 ;
179 ;
180 ;*****
181 ;
182 ;***** END OF UTILITY AND INITIALIZATION ROUTINES *****;
183 ;
184 ;
185 ;
186 ;
187 ;
188 ;
189 ;
190 ;
191 ;
192 ;
193 ;

```

PUBLIC SYMBOLES
 BPLISR C 004E CZT C 003I CZTR C 004I IBIPHLL C 0044 INT259 C 0020

INICZT C 0019 INIPID C 0000 INTSET C 0033 MASKET C 002F HERE C 0057

EXTERNAL SYMBOLS

USER SYMBOLS

EPLISR C 004E	CB255 A 00E7	CB0255 A 00A5	CZT C 003E	CZTR C 0041
EB251 A 00EC	EB255 A 00E3	LATA A 00E9	LATE A 00E5	LHA A 0007
ESABLC C 003E	EOI A 0020	FILE A 0007	FILL A 0017	FRSH C 0071
IBIPHL C 0044	ICU1 A 001F	ICU2 A 0040	INT257 C 00E0	INTSET C 0019
INIPID C 0000	INTEA A 000C	INTER A 000E	INTSET C 0033	LOAD C 0046
LOADR A 0003	LSB A 000A	MASK A 00FF	MSE A 000E	MASKET C 002F
HERE C 0057	OCU2 A 0070	STE C 0033	UMASK A 00FF	URJIT A 00E7
URSYN A 00EF				

ASSEMBLY COMPLETE, NO ERRORS

```

LOC  OBJ          LINE      SOURCE STATEMENT
                                1      NAME      INTGT
                                2      ;
                                3      SUBROUTINE INTGT(ICNT, TI, DELF)
                                4      ;
                                5      ;;; ICNT=IFIX(TI*DELF+.5)
0010          7      FADD    EQU      10H
0012          8      MULT   EQU      12H
001F          9      FIX    EQU      1FH
                                10     ;
                                11     EXTRN AMDLDD,AMDCHD,INTSTR,GIVE
                                12     ;
                                13     PUBLIC INTGT
                                14     ;
                                15     CSEG
                                16     ;
0000  E1          17     INTGT:  POP     H           ;SAVE RTN ADDR
0001  CD0000     E      18     CALL   AMDLDD    ;TI ↑
0004  42          19     MOV    R,D
0005  4B          20     MOV    C,E
0006  CD0000     E      21     CALL   AMDLDD    ;DELF ↑
0009  3E12       22     MVI   A,MULT
000B  CD0000     E      23     CALL   AMDCHD    ;TI*DELF
000E  112300     C      24     LXI   D,HALF
0011  CD0000     E      25     CALL   GIVE      ;0.5 ↑
0014  3E10       26     MVI   A,FADD
0016  CD0000     E      27     CALL   AMDCHD    ;TI*DELF + .5
0019  3E1F       28     MVI   A,FIX
001B  CD0000     E      29     CALL   AMDCHD    ;IFIX(TI*DELF+.5)
001E  C1         30     POP     B
001F  CD0000     E      31     CALL   INTSTR    ;ICNT=
0022  E9         32     FCHL   ;RETURN
                                33     ;
0023  00          34     HALF:  DB      00H,00H,80H,00H
                                35     ;
                                36     END

```

PUBLIC SYMBOLS
INTGT C 0000

EXTERNAL SYMBOLS
AMDCHD E 0000 AMDLDD E 0000 GIVE E 0000 INTSTR E 0000

USER SYMBOLS
AMDCHD E 0000 AMDLDD E 0000 FADD A 0010 FIX A 001F GIVE E 0000 HALF C 0023
INTGT C 0000
INTSTR E 0000 MULT A 0012

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME IUSART
		2	;
		3	FUNCTION IS TO INITIALIZE THE PROTOTYPE USART
		4	;
		5	REGISTERS ALTERED: C(A)
		6	;
		7	CSEG
		8	PUBLIC IUSRT
		9	;
00CE		10	MODE EQU OCEH
00CD		11	CNCTL EQU OCDH
0027		12	CMD EQU 027H
		13	;
0000	3ECE	14	IUSRT: MVI A,MODE ;GET MODE SETTING
0002	D3CD	15	OUT CNCTL ;OUT-PUT IT TO THE USART
0004	3E27	16	MVI A,CMD ;GET CMD
0006	D3CD	17	OUT CNCTL ; AND SEND IT OUT
0008	C9	18	RET
		19	;
		20	END

PUBLIC SYMBOLS
IUSRT C 0000

EXTERNAL SYMBOLS

USER SYMBOLS
CMD A 0027 CNCTL A 00CD IUSRT C 0000 MODE A 00CE

ASSEMBLY COMPLETE, NO ERRORS

```

LOC OBJ          LINE          SOURCE STATEMENT
      1          NAME          I32SUM
      2          ;
      3          ;
      4          ;
      5          ;
      6          ;
      7          ;
0008      8 LOG          EQU          08H
0012      9 MULT         EQU          12H
001C     10 FLOT32      EQU          1CH
002C     11 IADD32      EQU          2CH
      12          ;
      13          ;
      14          ;
      15          ;
      16          ;
      17          ;
      18          ;
0000 E1     19 I32SUM:  POP          H          ;SAVE RETURN ADDRESS
0001 IA     20          LDAX         D          ;GET COUNT
0002 57     21          MOV          D,A       ;PUT IN D
0003 CD0000 E 22          CALL        I32LOD      ;LOAD FIRST ENTRY
0006 15     23 LOOP:    DCR          D          ;IF LAST ENTRY GO TO FLOAT
0007 CA1500 C 24          JZ          FLOAT
000A CD0000 E 25          CALL        I32LOD      ;SUM IN NEXT ENTRY
000D 3E2C   26          MVI         A,IADD32
000F CD0000 E 27          CALL        AMDCMD
0012 C30600 C 28          JMP          LOOP
0015 3E1C   29 FLOAT:   MVI         A,FLOT32
0017 CD0000 E 30          CALL        AMDCMD      ;FLOAT RESULT
001A 3E08   31          MVI         A,LOG
001C CD0000 E 32          CALL        AMDCMD      ;ALOG10(SUM)
001F 112F00 C 33          LXI         D,TEN
0022 CD0000 E 34          CALL        GIVE          ;10. ↑
0025 3E12   35          MVI         A,MULT
0027 CD0000 E 36          CALL        AMDCMD      ;10.*ALOG10(SUM)
002A C1     37          POP          B
002B CD0000 E 38          CALL        AMDSTR      ;SAVE IN PX
002E E9     39          PCHL
      40          ;
002F 00     41 TEN:    DB          00H,00H,0A0H,04H
0030 00
0031 A0
0032 04
      42          ;
      43          ;
      44          ;
      45          ;
      46          ;
      47          ;
      48          ;
      49          ;
      50          ;
      51          ;
      52          ;
      53          ;
      54          ;
      55          ;
      56          ;
      57          ;
      58          ;
      59          ;
      60          ;
      61          ;
      62          ;
      63          ;
      64          ;
      65          ;
      66          ;
      67          ;
      68          ;
      69          ;
      70          ;
      71          ;
      72          ;
      73          ;
      74          ;
      75          ;
      76          ;
      77          ;
      78          ;
      79          ;
      80          ;
      81          ;
      82          ;
      83          ;
      84          ;
      85          ;
      86          ;
      87          ;
      88          ;
      89          ;
      90          ;
      91          ;
      92          ;
      93          ;
      94          ;
      95          ;
      96          ;
      97          ;
      98          ;
      99          ;
     100         ;
     101         ;
     102         ;
     103         ;
     104         ;
     105         ;
     106         ;
     107         ;
     108         ;
     109         ;
     110         ;
     111         ;
     112         ;
     113         ;
     114         ;
     115         ;
     116         ;
     117         ;
     118         ;
     119         ;
     120         ;
     121         ;
     122         ;
     123         ;
     124         ;
     125         ;
     126         ;
     127         ;
     128         ;
     129         ;
     130         ;
     131         ;
     132         ;
     133         ;
     134         ;
     135         ;
     136         ;
     137         ;
     138         ;
     139         ;
     140         ;
     141         ;
     142         ;
     143         ;
     144         ;
     145         ;
     146         ;
     147         ;
     148         ;
     149         ;
     150         ;
     151         ;
     152         ;
     153         ;
     154         ;
     155         ;
     156         ;
     157         ;
     158         ;
     159         ;
     160         ;
     161         ;
     162         ;
     163         ;
     164         ;
     165         ;
     166         ;
     167         ;
     168         ;
     169         ;
     170         ;
     171         ;
     172         ;
     173         ;
     174         ;
     175         ;
     176         ;
     177         ;
     178         ;
     179         ;
     180         ;
     181         ;
     182         ;
     183         ;
     184         ;
     185         ;
     186         ;
     187         ;
     188         ;
     189         ;
     190         ;
     191         ;
     192         ;
     193         ;
     194         ;
     195         ;
     196         ;
     197         ;
     198         ;
     199         ;
     200         ;
     201         ;
     202         ;
     203         ;
     204         ;
     205         ;
     206         ;
     207         ;
     208         ;
     209         ;
     210         ;
     211         ;
     212         ;
     213         ;
     214         ;
     215         ;
     216         ;
     217         ;
     218         ;
     219         ;
     220         ;
     221         ;
     222         ;
     223         ;
     224         ;
     225         ;
     226         ;
     227         ;
     228         ;
     229         ;
     230         ;
     231         ;
     232         ;
     233         ;
     234         ;
     235         ;
     236         ;
     237         ;
     238         ;
     239         ;
     240         ;
     241         ;
     242         ;
     243         ;
     244         ;
     245         ;
     246         ;
     247         ;
     248         ;
     249         ;
     250         ;
     251         ;
     252         ;
     253         ;
     254         ;
     255         ;
     256         ;
     257         ;
     258         ;
     259         ;
     260         ;
     261         ;
     262         ;
     263         ;
     264         ;
     265         ;
     266         ;
     267         ;
     268         ;
     269         ;
     270         ;
     271         ;
     272         ;
     273         ;
     274         ;
     275         ;
     276         ;
     277         ;
     278         ;
     279         ;
     280         ;
     281         ;
     282         ;
     283         ;
     284         ;
     285         ;
     286         ;
     287         ;
     288         ;
     289         ;
     290         ;
     291         ;
     292         ;
     293         ;
     294         ;
     295         ;
     296         ;
     297         ;
     298         ;
     299         ;
     300         ;
     301         ;
     302         ;
     303         ;
     304         ;
     305         ;
     306         ;
     307         ;
     308         ;
     309         ;
     310         ;
     311         ;
     312         ;
     313         ;
     314         ;
     315         ;
     316         ;
     317         ;
     318         ;
     319         ;
     320         ;
     321         ;
     322         ;
     323         ;
     324         ;
     325         ;
     326         ;
     327         ;
     328         ;
     329         ;
     330         ;
     331         ;
     332         ;
     333         ;
     334         ;
     335         ;
     336         ;
     337         ;
     338         ;
     339         ;
     340         ;
     341         ;
     342         ;
     343         ;
     344         ;
     345         ;
     346         ;
     347         ;
     348         ;
     349         ;
     350         ;
     351         ;
     352         ;
     353         ;
     354         ;
     355         ;
     356         ;
     357         ;
     358         ;
     359         ;
     360         ;
     361         ;
     362         ;
     363         ;
     364         ;
     365         ;
     366         ;
     367         ;
     368         ;
     369         ;
     370         ;
     371         ;
     372         ;
     373         ;
     374         ;
     375         ;
     376         ;
     377         ;
     378         ;
     379         ;
     380         ;
     381         ;
     382         ;
     383         ;
     384         ;
     385         ;
     386         ;
     387         ;
     388         ;
     389         ;
     390         ;
     391         ;
     392         ;
     393         ;
     394         ;
     395         ;
     396         ;
     397         ;
     398         ;
     399         ;
     400         ;
     401         ;
     402         ;
     403         ;
     404         ;
     405         ;
     406         ;
     407         ;
     408         ;
     409         ;
     410         ;
     411         ;
     412         ;
     413         ;
     414         ;
     415         ;
     416         ;
     417         ;
     418         ;
     419         ;
     420         ;
     421         ;
     422         ;
     423         ;
     424         ;
     425         ;
     426         ;
     427         ;
     428         ;
     429         ;
     430         ;
     431         ;
     432         ;
     433         ;
     434         ;
     435         ;
     436         ;
     437         ;
     438         ;
     439         ;
     440         ;
     441         ;
     442         ;
     443         ;
     444         ;
     445         ;
     446         ;
     447         ;
     448         ;
     449         ;
     450         ;
     451         ;
     452         ;
     453         ;
     454         ;
     455         ;
     456         ;
     457         ;
     458         ;
     459         ;
     460         ;
     461         ;
     462         ;
     463         ;
     464         ;
     465         ;
     466         ;
     467         ;
     468         ;
     469         ;
     470         ;
     471         ;
     472         ;
     473         ;
     474         ;
     475         ;
     476         ;
     477         ;
     478         ;
     479         ;
     480         ;
     481         ;
     482         ;
     483         ;
     484         ;
     485         ;
     486         ;
     487         ;
     488         ;
     489         ;
     490         ;
     491         ;
     492         ;
     493         ;
     494         ;
     495         ;
     496         ;
     497         ;
     498         ;
     499         ;
     500         ;
     501         ;
     502         ;
     503         ;
     504         ;
     505         ;
     506         ;
     507         ;
     508         ;
     509         ;
     510         ;
     511         ;
     512         ;
     513         ;
     514         ;
     515         ;
     516         ;
     517         ;
     518         ;
     519         ;
     520         ;
     521         ;
     522         ;
     523         ;
     524         ;
     525         ;
     526         ;
     527         ;
     528         ;
     529         ;
     530         ;
     531         ;
     532         ;
     533         ;
     534         ;
     535         ;
     536         ;
     537         ;
     538         ;
     539         ;
     540         ;
     541         ;
     542         ;
     543         ;
     544         ;
     545         ;
     546         ;
     547         ;
     548         ;
     549         ;
     550         ;
     551         ;
     552         ;
     553         ;
     554         ;
     555         ;
     556         ;
     557         ;
     558         ;
     559         ;
     560         ;
     561         ;
     562         ;
     563         ;
     564         ;
     565         ;
     566         ;
     567         ;
     568         ;
     569         ;
     570         ;
     571         ;
     572         ;
     573         ;
     574         ;
     575         ;
     576         ;
     577         ;
     578         ;
     579         ;
     580         ;
     581         ;
     582         ;
     583         ;
     584         ;
     585         ;
     586         ;
     587         ;
     588         ;
     589         ;
     590         ;
     591         ;
     592         ;
     593         ;
     594         ;
     595         ;
     596         ;
     597         ;
     598         ;
     599         ;
     600         ;
     601         ;
     602         ;
     603         ;
     604         ;
     605         ;
     606         ;
     607         ;
     608         ;
     609         ;
     610         ;
     611         ;
     612         ;
     613         ;
     614         ;
     615         ;
     616         ;
     617         ;
     618         ;
     619         ;
     620         ;
     621         ;
     622         ;
     623         ;
     624         ;
     625         ;
     626         ;
     627         ;
     628         ;
     629         ;
     630         ;
     631         ;
     632         ;
     633         ;
     634         ;
     635         ;
     636         ;
     637         ;
     638         ;
     639         ;
     640         ;
     641         ;
     642         ;
     643         ;
     644         ;
     645         ;
     646         ;
     647         ;
     648         ;
     649         ;
     650         ;
     651         ;
     652         ;
     653         ;
     654         ;
     655         ;
     656         ;
     657         ;
     658         ;
     659         ;
     660         ;
     661         ;
     662         ;
     663         ;
     664         ;
     665         ;
     666         ;
     667         ;
     668         ;
     669         ;
     670         ;
     671         ;
     672         ;
     673         ;
     674         ;
     675         ;
     676         ;
     677         ;
     678         ;
     679         ;
     680         ;
     681         ;
     682         ;
     683         ;
     684         ;
     685         ;
     686         ;
     687         ;
     688         ;
     689         ;
     690         ;
     691         ;
     692         ;
     693         ;
     694         ;
     695         ;
     696         ;
     697         ;
     698         ;
     699         ;
     700         ;
     701         ;
     702         ;
     703         ;
     704         ;
     705         ;
     706         ;
     707         ;
     708         ;
     709         ;
     710         ;
     711         ;
     712         ;
     713         ;
     714         ;
     715         ;
     716         ;
     717         ;
     718         ;
     719         ;
     720         ;
     721         ;
     722         ;
     723         ;
     724         ;
     725         ;
     726         ;
     727         ;
     728         ;
     729         ;
     730         ;
     731         ;
     732         ;
     733         ;
     734         ;
     735         ;
     736         ;
     737         ;
     738         ;
     739         ;
     740         ;
     741         ;
     742         ;
     743         ;
     744         ;
     745         ;
     746         ;
     747         ;
     748         ;
     749         ;
     750         ;
     751         ;
     752         ;
     753         ;
     754         ;
     755         ;
     756         ;
     757         ;
     758         ;
     759         ;
     760         ;
     761         ;
     762         ;
     763         ;
     764         ;
     765         ;
     766         ;
     767         ;
     768         ;
     769         ;
     770         ;
     771         ;
     772         ;
     773         ;
     774         ;
     775         ;
     776         ;
     777         ;
     778         ;
     779         ;
     780         ;
     781         ;
     782         ;
     783         ;
     784         ;
     785         ;
     786         ;
     787         ;
     788         ;
     789         ;
     790         ;
     791         ;
     792         ;
     793         ;
     794         ;
     795         ;
     796         ;
     797         ;
     798         ;
     799         ;
     800         ;
     801         ;
     802         ;
     803         ;
     804         ;
     805         ;
     806         ;
     807         ;
     808         ;
     809         ;
     810         ;
     811         ;
     812         ;
     813         ;
     814         ;
     815         ;
     816         ;
     817         ;
     818         ;
     819         ;
     820         ;
     821         ;
     822         ;
     823         ;
     824         ;
     825         ;
     826         ;
     827         ;
     828         ;
     829         ;
     830         ;
     831         ;
     832         ;
     833         ;
     834         ;
     835         ;
     836         ;
     837         ;
     838         ;
     839         ;
     840         ;
     841         ;
     842         ;
     843         ;
     844         ;
     845         ;
     846         ;
     847         ;
     848         ;
     849         ;
     850         ;
     851         ;
     852         ;
     853         ;
     854         ;
     855         ;
     856         ;
     857         ;
     858         ;
     859         ;
     860         ;
     861         ;
     862         ;
     863         ;
     864         ;
     865         ;
     866         ;
     867         ;
     868         ;
     869         ;
     870         ;
     871         ;
     872         ;
     873         ;
     874         ;
     875         ;
     876         ;
     877         ;
     878         ;
     879         ;
     880         ;
     881         ;
     882         ;
     883         ;
     884         ;
     885         ;
     886         ;
     887         ;
     888         ;
     889         ;
     890         ;
     891         ;
     892         ;
     893         ;
     894         ;
     895         ;
     896         ;
     897         ;
     898         ;
     899         ;
     900         ;
     901         ;
     902         ;
     903         ;
     904         ;
     905         ;
     906         ;
     907         ;
     908         ;
     909         ;
     910         ;
     911         ;
     912         ;
     913         ;
     914         ;
     915         ;
     916         ;
     917         ;
     918         ;
     919         ;
     920         ;
     921         ;
     922         ;
     923         ;
     924         ;
     925         ;
     926         ;
     927         ;
     928         ;
     929         ;
     930         ;
     931         ;
     932         ;
     933         ;
     934         ;
     935         ;
     936         ;
     937         ;
     938         ;
     939         ;
     940         ;
     941         ;
     942         ;
     943         ;
     944         ;
     945         ;
     946         ;
     947         ;
     948         ;
     949         ;
     950         ;
     951         ;
     952         ;
     953         ;
     954         ;
     955         ;
     956         ;
     957         ;
     958         ;
     959         ;
     960         ;
     961         ;
     962         ;
     963         ;
     964         ;
     965         ;
     966         ;
     967         ;
     968         ;
     969         ;
     970         ;
     971         ;
     972         ;
     973         ;
     974         ;
     975         ;
     976         ;
     977         ;
     978         ;
     979         ;
     980         ;
     981         ;
     982         ;
     983         ;
     984         ;
     985         ;
     986         ;
     987         ;
     988         ;
     989         ;
     990         ;
     991         ;
     992         ;
     993         ;
     994         ;
     995         ;
     996         ;
     997         ;
     998         ;
     999         ;
    1000        ;
    1001        ;
    1002        ;
    1003        ;
    1004        ;
    1005        ;
    1006        ;
    1007        ;
    1008        ;
    1009        ;
    1010        ;
    1011        ;
    1012        ;
    1013        ;
    1014        ;
    1015        ;
    1016        ;
    1017        ;
    1018        ;
    1019        ;
    1020        ;
    1021        ;
    1022        ;
    1023        ;
    1024        ;
    1025        ;
    1026        ;
    1027        ;
    1028        ;
    1029        ;
    1030        ;
    1031        ;
    1032        ;
    1033        ;
    1034        ;
    1035        ;
    1036        ;
    1037        ;
    1038        ;
    1039        ;
    1040        ;
    1041        ;
    1042        ;
    1043        ;
    1044        ;
    1045        ;
    1046        ;
    1047        ;
    1048        ;
    1049        ;
    1050        ;
    1051        ;
    1052        ;
    1053        ;
    1054        ;
    1055        ;
    1056        ;
    1057        ;
    1058        ;
    1059        ;
    1060        ;
    1061        ;
    1062        ;
    1063        ;
    1064        ;
    1065        ;
    1066        ;
    1067        ;
    1068        ;
    1069        ;
    1070        ;
    1071        ;
    1072        ;
    1073        ;
    1074        ;
    1075        ;
    1076        ;
    1077        ;
    1078        ;
    1079        ;
    1080        ;
    1081        ;
    1082        ;
    1083        ;
    1084        ;
    1085        ;
    1086        ;
    1087        ;
    1088        ;
    1089        ;
    1090        ;
    1091        ;
    1092        ;
    1093        ;
    1094        ;
    1095        ;
    1096        ;
    1097        ;
    1098        ;
    1099        ;
    1100        ;
    1101        ;
    1102        ;
    1103        ;
    1104        ;
    1105        ;
    1106        ;
    1107        ;
    1108        ;
    1109        ;
    1110        ;
    1111        ;
    1112        ;
    1113        ;
    1114        ;
    1115        ;
    1116        ;
    1117        ;
    1118        ;
    1119        ;
    1120        ;
    1121        ;
    1122        ;
    1123        ;
    1124        ;
    1125        ;
    1126        ;
    1127        ;
    1128        ;
    1129        ;
    1130        ;
    1131        ;
    1132        ;
    1133        ;
    1134        ;
    1135        ;
    1136        ;
    1137        ;
    1138        ;
    1139        ;
    1140        ;
    1141        ;
    1142        ;
    1143        ;
    1144        ;
    1145        ;
    1146        ;
    1147        ;
    1148        ;
    1149        ;
    1150        ;
    1151        ;
    1152        ;
    1153        ;
    1154        ;
    1155        ;
    1156        ;
    1157        ;
    1158        ;
    1159        ;
    1160        ;
    1161        ;
    1162        ;
    1163        ;
    1164        ;
    1165        ;
    1166        ;
    1167        ;
    1168        ;
    1169        ;
    1170        ;
    1171        ;
    1172        ;
    1173        ;
    1174        ;
    1175        ;
    1176        ;
    1177        ;
    1178        ;
    1179        ;
    1180        ;
    1181        ;
    1182        ;
    1183        ;
    1184        ;
    1185        ;
    1186        ;
    1187        ;
    1188        ;
    1189        ;
    1190        ;
    1191        ;
    1192        ;
    1193        ;
    1194        ;
    1195        ;
    1196        ;
    1197        ;
    1198        ;
    1199        ;
    1200        ;
    1201        ;
    1202        ;
    1203        ;
    1204        ;
    1205        ;
    1206        ;
    1207        ;
    1208        ;
    1209        ;
    1210        ;
    1211        ;
    1212        ;
    1213        ;
    1214        ;
    1215        ;
    1216        ;
    1217        ;
    1218        ;
    1219        ;
    1220        ;
    1221        ;
    1222        ;
    1223        ;
    1224        ;
    1225        ;
    1226        ;
    1227        ;
    1228        ;
    1229        ;
    1230        ;
    1231        ;
    1232        ;
    1233        ;
    1234        ;
    1235        ;
    1236        ;
    1237        ;
    1238        ;
    1239        ;
    1240        ;
    1241        ;
    1242        ;
    1243        ;
    1244        ;
    1245        ;
    1246        ;
    1247        ;
    1248        ;
    1249        ;
    1250        ;
    1251        ;
    1252        ;
    1253        ;
    1254        ;
    1255        ;
    1256        ;
    1257        ;
    1258        ;
    1259        ;
    1260        ;
    1261        ;
    1262        ;
    1263        ;
    1264        ;
    1265        ;
    1266        ;
    1267        ;
    1268        ;
    1269        ;
    1270        ;
    1271        ;
    1272        ;
    1273        ;
    1274        ;
    1275        ;
    1276        ;
    1277        ;
    1278        ;
    1279        ;
    1280        ;
    1281        ;
    1282        ;
    1283        ;
    1284        ;
    1285        ;
    1286        ;
    1287        ;
    1288        ;
    1289        ;
    1290        ;
    1291        ;
    1292        ;
    1293        ;
    1294        ;
    1295        ;
    1296        ;
    1297        ;
    1298        ;
    1299        ;
    1300        ;
    1301        ;
    1302        ;
    1303        ;
    1304        ;
    1305        ;
    1306        ;
    1307        ;
    1308        ;
    1309        ;
    1310        ;
    1311        ;
    1312        ;
    1313        ;
    1314        ;
    1315        ;
    1316        ;
    1317        ;
    1318        ;
    1319        ;
    1320        ;
    1321        ;
    1322        ;
    1323        ;
    1324        ;
    1325        ;
    1326        ;
    1327        ;
    1328        ;
    1329        ;
    1330        ;
    1331        ;
    1332        ;
    1333        ;
    1334        ;
    1335        ;
    1336        ;
    1337        ;
    1338        ;
    1339        ;
    1340        ;
    1341        ;
    1342        ;
    1343        ;
    1344        ;
    1345        ;
    1346        ;
    1347        ;
    1348        ;
    1349        ;
    1350        ;
    1351        ;
    1352        ;
    1353        ;
    1354        ;
    1355        ;
    1356        ;
    1357        ;
    1358        ;
    1359        ;
    1360        ;
    1361        ;
    1362        ;
    1363        ;
    1364        ;
    1365        ;
    1366        ;
    1367        ;
    1368        ;
    1369        ;
    1370        ;
    1371        ;
    1372        ;
    1373        ;
    1374        ;
    1375        ;
    1376        ;
    1377        ;
    1378        ;
    1379        ;
    1380        ;
    1381        ;
    1382        ;
    1383        ;
    1384        ;
    1385        ;
    1386        ;
    1387        ;
    1388        ;
    1389        ;
    1390        ;
    1391        ;
    1392        ;
    1393        ;
    1394        ;
    1395        ;
    1396        ;
    1397        ;
    1398        ;
    1399        ;
    1400        ;
    1401        ;
    1402        ;
    1403        ;
    1404        ;
    1405        ;
    1406        ;
    1407        ;
    1408        ;
    1409        ;
    1410        ;
    1411        ;
    1412        ;
    1413        ;
    1414        ;
    1415        ;
    1416        ;
    1417        ;
    1418        ;
    1419        ;
    1420        ;
    1421        ;
    1422        ;
    1423        ;
    1424        ;
    1425        ;
    1426        ;
    1427        ;
    1428        ;
    1429        ;
    1430        ;
    1431        ;
    1432        ;
    1433        ;
   
```

ISIS-II FORTRAN-80 V2.0 COMPILATION OF PROGRAM UNIT KBPHAL
 OBJECT MODULE PLACED IN :F1:KBPHAL.OBJ
 COMPILER INVOKED BY: FORT80 :F1:KBPHAL.SRC DEBUG PAGELNGTH(77) PAGEWIDTH(90)

```

1      SUBROUTINE KBPHAL(IOUT)
      C
      C ***** HANDLES BI-PHASE-L OUTPUTS *****
      C
      C INTEGER*1 IOUT(128)
      C :: SET MASK TO ALLOW LEVEL 1 INTERRUPTS ::
      C   J=#0FDH
      C   CALL MSKSET(J)
      C :: WRITE LINE TO BI-PHASE-L INTERFACE ::
      C   CALL IBIPHL(IOUT)
      C :: SET MASK TO MASK ALL INTERRUPTS ::
      C   J=#OFFH
      C   CALL MSKSET(J)
      C :: BACK-FILL BUFFER LINE JUST WRITTEN WITH #099H ::
      C   DO 10 J=1,128
      C     IOUT(J)=#099H
      C   10 CONTINUE
      C   RETURN
      C   END
  
```

MODULE INFORMATION:

```

CODE AREA SIZE = 004EH      78D
VARIABLE AREA SIZE = 0004H   4D
MAXIMUM STACK SIZE = 0002H   2D
20 LINES READ
  
```

0 PROGRAM ERROR(S) IN PROGRAM UNIT KBPHAL

0 TOTAL PROGRAM ERROR(S)
 END OF FORTRAN COMPILATION

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME KEYCHK
		2	
		3	
		4	FUNCTION: SENSES KEYBOARD FOR AN ESCAPE CHARACTER. IF NO
		5	CHARACTER IS PENDING OR IF PENDING CHARACTER IS NOT
		6	AN 'ESC', THEN NO ACTION IS TAKEN. IF 'ESC' IS FOUND
		7	A BRANCH TO 'IKEYI' IS TAKEN TO SET THE MAIN PROGRAM
		8	FLAG 'KESC'.
		9	
		10	PUBLIC KEYCHK
		11	EXTRN ECHO,IKEYI
		12	
		13	CSEG
		14	
0000	DBCD	15	KEYCHK: IN 0CDH ;GET STATUS OF CONSOLE
0002	E602	16	ANI 02H ;RCVR READY?
0004	CA1C00	17	JZ XIT ;NO, THEN EXIT
0007	DBCC	18	IN 0CCH ;YES, GET CHARACTER
0009	E67F	19	ANI 07FH ;STRIP PARITY
000B	FE1B	20	CPI 01BH ;IS IT 'ESC' ?
000D	C21C00	21	JNZ XIT ;NO, THEN IGNORE
0010	4F	22	MOV C,A ;YES, SAVE VALUE
0011	CD0000	23	CALL ECHO ;AND ECHO TO SCREEN
0014	CD0000	24	CALL IKEYI ;SET KEY INTERRUPT FLAG
0017	0E07	25	MVI C,07H ;GET BELL CHARACTER
0019	CD0000	26	CALL ECHO ;SEND IT OUT
001C	C9	27	XIT: RET
		28	
		29	END

PUBLIC SYMBOLS
KEYCHK C 0000

EXTERNAL SYMBOLS
ECHO E 0000 IKEYI E 0000

USER SYMBOLS
ECHO E 0000 IKEYI E 0000 KEYCHK C 0000 XIT C 001C

ASSEMBLY COMPLETE, NO ERRORS

```

LOC 07-J      LINE      SOURCE STATEMENT
1 ;          NAME KEYIN
2 ;
3 ;          PUBLIC KEYBRD,GETCH,CI
4 ;
5 ;          CSEG
6 ;
7 ;          EXTRN  ECHO
0003 B BACK EQU 08H ;HEX VALUE OF 'BS'
0004 9 CR EQU 0DH ; 'CR'
0007 10 BELL EQU 07H ; BELL CHARACTER
11 ;
12 ;          KEYBOARD INPUT ROUTINE FOR SCAT PROCESSOR
13 ;
14 ;          REGISTERS UPON INPUT:
15 ;          (BC)=ADDR OF CHARACTER COUNT TO BE ENTERED
16 ;          (DE)=BUFFER ADDR INTO WHICH THE CHARACTERS GO
17 ;
0000 EB 18 KEYBRD: XCHG ;BUFFER POINTER TO HL
0001 0A 19 LDAX B ;GET CHARACTER COUNT
0002 57 20 MOV D,A ;CHARACTER COUNT TO D
0003 5F 21 MOV E,A ;LENGTH OF STRING TO E ALSO
0004 C83C00 C 22 NEXT: CALL GETCH ;C(C)=C(A)='CHAR'
0007 FE08 C 23 CPI BACK ;BACK SPACE?
0009 C22100 C 24 JNZ SKIP1 ;IF NOT SKIP
000C 7A 25 MOV A,E ;START OF LINE?
000E BB 26 CMP E ;IF SO, THEN 'BLEEP' AND IGNORE
000E C21900 C 27 JNZ BKSPC ;IF NOT BACK SPACE POINTERS AND CURSOR
0011 0E07 28 MVI C,BELL ;GET BELL CHARACTER
0013 CD0000 E 29 CALL ECHO
0016 C30400 C 30 JMP NEXT
0019 14 31 BKSPC: INR D ;BACK SPACE
001A 2B 32 DCX H ;POINTER
001B CD0000 E 33 CALL ECHO ;ALSO ECHO ASCII BACKSPACE TO CONSOLE
001E C30400 C 34 JMP NEXT
0021 FE08 C 35 SKIP1: CPI CR ;CARRIAGE RETURN?
0023 C23200 C 36 JNZ SKIP2 ;IF NOT SKIP
0024 CD0000 E 37 CALL ECHO ;IF SO, THEN ECHO CR & LF
0029 3E20 38 MVI A,' ' ;ALSO, FILL OTHER BUFFER SPACE WITH BLANKS
002B 77 39 FILL: MOV M,A
002C 23 40 INX H ;STEP POINTER
002D 15 41 DCR D
002E C22B00 C 42 JNZ FILL
0031 C9 43 RET ;RETURN WHEN FINISHED
0032 CD0000 E 44 SKIP2: CALL ECHO ;ECHO CHARACTER
0035 71 45 MOV M,C ;PUT IT INTO BUFFER
0036 23 46 INX H
0037 15 47 DCR D ;DECREMENT COUNTER
0038 C20400 C 48 JNZ NEXT ;LOOP UNTIL END OF BUFFER
003B C9 49 RET
50 ;
003C CD4300 C 51 GETCH: CALL CI ;GET CHAR FROM TERMINAL
003F E67F 52 ANI 07FH ;STRIP OF PARITY
0041 4F 53 MOV C,A ;PUT VALUE IN C REG
0042 C9 54 RET
55 ;
0043 BECF 56 CI: IN ODDH ;GET STATUS OF CONSOLE
0045 E602 57 ANI 02H ;RCUR BUFFER READY?
0047 CA4300 C 58 JZ CI ; IF NOT , TRY AGAIN
004A BECF 59 IN ODDH ; IF YES , GET CHARACTER
004C C9 60 RET
61 ;
62 ;          END

```

PUBLIC SYMBOLS

CI C 0043 GETCH C 003C KEYERR C 0000

EXTERNAL SYMBOLS
ECHO E 0000

USER SYMBOLS

BACH	A 0008	BELL	A 0007	FKSPC	C 0019	CI	C 0043	CR	A 000E	ECHO	E 0000
	FILL	C 002B									
GETCH	C 003C	KEYERR	C 0000	NEXT	C 0004	SKIP1	C 0021	SKIP2	C 0032		

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME LDJMPS
		2	;
		3	PUBLIC LDJMPS
		4	;
		5	ASEG
		6	EXTRN CZTINT
4000		7	ORG 4000H
4000	C30000	8	JMP CZTINT ;LEVEL 0 INTERRUPT, CZT INTEGRATION TIMER
4003	00	9	NOP
4004	C9	10	RET ;LEVEL 1 INTERRUPT, BI-PHASE-L, PARALLEL P
			ORT A2
4005	00	11	NOP
4006	00	12	NOP
4007	00	13	NOP
4008	C9	14	RET ;LEVEL 2 INTERRUPT, BUSS INT 2
4009	00	15	NOP
400A	00	16	NOP
400B	00	17	NOP
400C	C9	18	RET ;LEVEL 3 INTERRUPT, STROBE, PARALLEL PORT B
		2	
400D	00	19	NOP
400E	00	20	NOP
400F	00	21	NOP
4010	C9	22	RET ;LEVEL 4 INTERRUPT, SYNC, BUSS INT 1
4011	00	23	NOP
4012	00	24	NOP
4013	00	25	NOP
4014	C9	26	RET ;LEVEL 5 INTERRUPT, GND
4015	00	27	NOP
4016	00	28	NOP
4017	00	29	NOP
4018	C9	30	RET ;LEVEL 6 INTERRUPT, GND
4019	00	31	NOP
401A	00	32	NOP
401B	00	33	NOP
401C	C9	34	RET ;LEVEL 7 INTERRUPT, GND
401D	00	35	NOP
401E	00	36	NOP
401F	00	37	NOP
4020	C9	38	LDJMPS: RET
		39	END

PUBLIC SYMBOLS
LDJMPS A 4020

EXTERNAL SYMBOLS
CZTINT E 0000

USER SYMBOLS
CZTINT E 0000 LDJMPS A 4020

ASSEMBLY COMPLETE, NO ERRORS

ISIS-II FORTRAN-80 V2.0 COMPILATION OF PROGRAM UNIT MFLAG
OBJECT MODULE PLACED IN :F1:MFLAG.OBJ
COMPILER INVOKED BY: FORT80 :F1:MFLAG.SRC DEBUG

```
1      SUBROUTINE MFLAG(IC,IOF,IOUT,IB,NZ)
      C
      C ***** MOVES FLAG DATA WORDS TO OUTPUT LINE *****
      C
2      INTEGER*1 IOF(5),IOUT(128)
3      INTEGER*2 IB,NZ,IC
      C
      C **** IB=BAND IDENT, NZ=POLARIZ IDENT
      C **** IC=COL COUNTER
      C **** IOF=FLAG WORD BUFFER, IOUT=OUTPUT BUFFER
      C
4      DO 10 I=1,5
5          IOUT(IC-1+I)=IOF(I)
6      CONTINUE
7      IOUT(IC+5)=IB
8      IOUT(IC+6)=NZ
      C
9      DO 20 K=IC+7,128
10         IOUT(K)=#0DDH
11     CONTINUE
      C
12     RETURN
13     END
```

MODULE INFORMATION:

CODE AREA SIZE = 00AEH 174D
VARIABLE AREA SIZE = 000EH 14D
MAXIMUM STACK SIZE = 0006H 6D
23 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT MFLAG

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

ISIS-II FORTRAN-80 V2.0 COMPILATION OF PROGRAM UNIT MFPNUM
OBJECT MODULE PLACED IN :F1:MFPNUM.OBJ
COMPILER INVOKED BY: FORT80 :F1:MFPNUM.SRC DEBUG

```

1          SUBROUTINE MFPNUM(FPNBR,ICOL,IBPT,IDOUT,IBCD,NK)
2          DIMENSION FPNBR(8),IBPT(8)
3          INTEGER*1 IBCD(4),IDOUT(128,70)
C*** NK SPECIFIES HOW MANY NBR'S TO CONVERT & MOVE
C
C          UP TO 8 FLTG POINT NBR'S CAN BE HANDLED
C
4          DO 30 K=1,NK
5             FP=FPNBR(K)
6             IF(FP.LT.-99.9)FP=-99.9
7             IF(FP.GT.99.9)FP=99.9
8             INUM=IFIX(FP*10.)
C      :::      :::      SET SIGN OF NBR      :::      :::
9             IBCD(4)=10
10            IF(INUM.GE.0)GO TO 5
11            IBCD(4)=14
12            INUM=-INUM
C
C      :::      UNPACK INTO BCD DIGITS      :::
C
13           DO 10 J=1,2
14              IH=(INUM/10)*10
15              IBCD(J)=INUM-IH
16              INUM=IH/10
17           10  CONTINUE
18              IBCD(3)=INUM
C
C      :::      PACK INTO BI-PHASE-L OUTPUT      :::
C
19           DO 20 L=1,2
20              INDX=ICOL-1+(K-1)*2+L
21              J=1+(L-1)*2
22              IBCD(J)=IBCD(J)*16+IBCD(J+1)
23              IDOUT(INDX,IBPT(K))=IBCD(J)
24           20  CONTINUE
25           30  CONTINUE
26              RETURN
27              END

```

MODULE INFORMATION:

CODE AREA SIZE = 01D2H 466D
VARIABLE AREA SIZE = 001EH 30D
MAXIMUM STACK SIZE = 0008H 8D
38 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT MFPNUM
0 TOTAL PROGRAM ERROR(S)

FORTRAN COMPILER
PAGE 2

END OF FORTRAN COMPILATION

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;
		2	;
		3	;
		4	NAME MINHR
		5	;
		6	EVALUATES MINUTES OR HOURS BY:
		7	;
		8	IMH=NT*I10+NU
		9	WHERE I10=10
		10	;
006C		12	IAD EQU 6CH
006E		13	IMULT EQU 6EH
		14	;
		15	EXTRN INTLDD,AMDCMD,AMDSTR,AMDLOD,INTSTR
		16	PUBLIC MINHR
		17	;
		18	CSEG
		19	;
		20	MINHR: POP H ;SAVE RETURN ADDRESS
0001	CD0000	E 21	CALL INTLDD ;INT ↑
0004	42	22	MOV B,D
0005	4B	23	MOV C,E
0006	CD0000	E 24	CALL INTLDD ;I10 ↑
0009	3E4E	25	MVI A,IMULT
000B	CD0000	E 26	CALL AMDCMD ;X
000E	C1	27	POP B
000F	CD0000	E 28	CALL INTLDD ;NU ↑
0012	3E4C	29	MVI A,IAD
0014	CD0000	E 30	CALL AMDCMD ;+
0017	C1	31	POP B
0018	CD0000	E 32	CALL INTSTR ;SAVE IN IMH
001B	E9	33	PCHL ;RETURN
		34	;
		35	END

PUBLIC SYMBOLS
MINHR C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLOD E 0000 AMDSTR E 0000 INTLDD E 0000 INTSTR E 0000

USER SYMBOLS
AMDCMD E 0000 AMDLOD E 0000 AMDSTR E 0000 IAD A 006C IMULT A 006E
INTLDD E 0000 INTSTR E 0000 MINHR C 0000

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME OUTPUT
		2	;
		3	CSEG
		4	;
		5	PUBLIC MSGOUT
		6	;
		7	EXTRN CO
		8	;
		9	FUNCTION:
		10	OUTPUTS A CHARACTER STRING TO THE CRT OR CONSOLE
		11	LOGGING DEVICE AT PORT LOCATION OCCH.
		12	;
		13	REGISTERS:
		14	C(BC)=ADDRESS OF CHARACTER COUNT TO SEND
		15	C(DE)=ADDRESS OF BUFFER CONTAINING THE STRING
		16	;
		17	REGISTERS ALTERED:
		18	C(HL), C(BC), C(A)
		19	;
0000	EB	20	MSGOUT: XCHG ;SET POINTER TO BYTE BUFFER
0001	0A	21	LDAX B ;GET COUNT FROM ADDR IN (BC)
0002	47	22	MOV B,A ;SET COUNTER OF BYTES
0003	4E	23	GBYT: MOV C,M ;GET A CHARACTER
0004	CD0000	24	E CALL CO ;SEND IT OUT
0007	23	25	INX H ;STEP BUFFER POINTER
0008	05	26	DCR B ;DROP ONE FROM COUNTER
0009	C20300	27	C JNZ GBYT ;IF NOT DONE , GET NEXT BYTE
		28	;
		29	CLEAR KEY BUFFER
000C	DBCD	29	JN OCDH ;GET STATUS
000E	E602	30	ANI 02H ;CHECK FOR PENDING CHARACTER
0010	CA1500	31	C JZ XIT ;IF NOT CONTINUE
0013	DBCC	32	IN OCCH ;GET CHARACTER
0015	C9	33	XIT: RET ;RETURN TO CALLING ROUTINE
		34	;
		35	END

PUBLIC SYMBOLS
MSGOUT C 0000

EXTERNAL SYMBOLS

ORIGINAL PAGE IS
OF POOR QUALITY

CO E 0000

USER SYMBOLS

CO E 0000 GBYT C 0003 MSGOUT C 0000 XIT C 0015

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1 ;	SUBROUTINE PCPN(PN, TI, PC, TC, ALT, VEL)
		2 ;	
		3 ;	NAME PCPN
		4 ;	
		5 ;	
00F0		6 PORT	EQU 00F0H
0012		7 MULT	EQU 12H
0013		8 DIV	EQU 13H
0017		9 ENTR	EQU 17H
0019		10 EXCHG	EQU 19H
		11 ;	
		12	EXTRN AMDL0D
		13	EXTRN AMDSTR
		14	EXTRN AMDCMD
		15	EXTRN GIVE
		16 ;	
		17	PUBLIC PCPN
		18 ;	
		19	CSEG
		20 ;	
0000	E1	21 PCPN:	POP H ;SAVE RETURN ADDRESS
0001	CD0000	E 22	CALL AMDL0D ;ALT ↑
0004	42	23	MOV B, D
0005	4B	24	MOV C, E
0006	CD0000	E 25	CALL AMDL0D ;VEL ↑
0009	3E13	26	MVI A, DIV
000B	CD0000	E 27	CALL AMDCMD ;/
000E	3E17	28	MVI A, ENTR
0010	CD0000	E 29	CALL AMDCMD ;↑
0013	C1	30	POP B
0014	CD0000	E 31	CALL AMDL0D ;TC ↑
0017	3E13	32	MVI A, DIV
0019	CD0000	E 33	CALL AMDCMD ;/
001C	C1	34	POP B
001D	CD0000	E 35	CALL AMDSTR ;SAVE IT IN PC
0020	C1	36	POP B
0021	CD0000	E 37	CALL AMDL0D ;TI ↑
0024	3E19	38	MVI A, EXCHG
0026	CD0000	E 39	CALL AMDCMD ;EXCHANGE X, Y
0029	3E13	40	MVI A, DIV
002B	CD0000	E 41	CALL AMDCMD ;/
002E	113E00	42	LXI D, HALF
0031	CD0000	E 43	CALL GIVE ;.5 ↑
0034	3E12	44	MVI A, MULT
0036	CD0000	E 45	CALL AMDCMD ;X
0039	C1	46	POP B
003A	CD0000	E 47	CALL AMDSTR ;SAVE IT IN PN
003D	E9	48	PCHL ;RETURN
		49 ;	
003E	00	50 HALF:	DB 00H, 00H, 80H, 00H
003F	00		
0040	80		
0041	00		
		51	END

PUBLIC SYMBOLS
PCPN C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDL0D E 0000 AMDSTR E 0000 GIVE E 0000

USER SYMBOLS

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0

PCPN PAGE 2

ANDCMD	E	0000	ANDLOD	E	0000	AMDSTR	E	0000	DIV	A	0013	ENTR	A	0017
EXCHG	A	0019	GIVE	E	0000	HALF	C	003E	MULT	A	0012	PCPN	C	0000
PORT	A	00F0												

ASSEMBLY COMPLETE, NO ERRORS

ISIS-II FORTRAN-80 V2.0 COMPIATION OF PROGRAM UNIT PRIVLU
OBJECT MODULE PLACED IN :F1:PRIVLU.OBJ
COMPILER INVOKED BY: FORT80 :F1:PRIVLU.SRC DEBUG

```
1          SUBROUTINE PRIVLU(CR,IR,N,IERR,X)
C
C      ::: PRINTS MESSAGE OF N CHAKACTERS ALREADY IN
C      BUFFER IR.  NEXT, PRINTS VALUE OF A FLOATING
C      POINT NUMBER IN F10.3 FORMAT LOCATED IN X.
C
C      NOTE: *** N.LE.30 ***
2          INTEGER*1 IR(30),N
3          CHARACTER*30 CR
C
4          CALL CRLF
5          CALL MSGOUT(N,IR)
6          WRITE(CR,10,ERR=20)X
7          10  FORMAT(F10.3,20X)
8          CALL MSGOUT(N,IR)
9          15  RETURN
C
10         20  IERR=1
11         GO TO 15
C
12        END
```

MODULE INFORMATION:

CODE AREA SIZE = 0091H 145D
VARIABLE AREA SIZE = 001EH 30D
MAXIMUM STACK SIZE = 0008H 8D
22 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT PRIVLU

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME DWAIT
		2	;
		3	CSEG
		4	EXTRN CO,CRLF,DELAY
		5	;
		6	PUBLIC DWAIT
		7	;
		8	FUNCTION:
		9	LINKAGE ROUTINE FOR FORTRAN ERROR RECOVERY.
		10	CONTROL REMAINS HERE UNTIL THE MACHINE IS RESET.
		11	;
		12	;
0000	CD0000	E 13	DWAIT: CALL CRLF
0003	060F	14	MVI B,0FH ;SET COUNT
0005	0E2A	15	LOOP: MVI C,'*' ;GET '*'
0007	CD0000	E 16	CALL CO ;OUTPUT '*'
000A	0E07	17	MVI C,07H ;INSERT BELL CHAR
000C	CD0000	E 18	CALL CO ;SEND IT TO CRT
000F	CD0000	E 19	CALL DELAY ;WAIT AWHILE , ALTERS C(A), C(C)
0012	05	20	DCR B ;COUNT
0013	C20500	C 21	JNZ LOOP ;DOWN
0016	CD0000	E 22	CALL DELAY ;DELAY AND
0019	C31600	C 23	JMP LP2 ; WAIT HERE FOR RESET OR INTERRUPT
		24	;
		25	END

PUBLIC SYMBOLS
DWAIT C 0000

EXTERNAL SYMBOLS
CO E 0000 CRLF E 0000 DELAY E 0000

USER SYMBOLS
CO E 0000 CRLF E 0000 DELAY E 0000 DWAIT C 0000 LOOP C 0005 LP2 C 00

ASSEMBLY COMPLETE, NO ERRORS

ISIS-II FORTRAN-80 V2.0 COMPILATION OF PROGRAM UNIT RUNLMT
OBJECT MODULE PLACED IN :F1:RUNLMT.OBJ
COMPILER INVOKED BY: FORT80 :F1:RUNLMT.SRC DEBUG

```

1      SUBROUTINE RUNLMT(PREV,IOVER)
2      DIMENSION PREV(5),PARM(5)
3      INTEGER*1 IOVER(5)
4      LOGICAL LZ,UZ
5      COMMON T1,PARM,BMW
C
C *** ROUTINE PERFORMS RUNNING LIMIT CHECK ON A/C DATA ***
C
6      IF(IOVER(1),NE,0)GO TO 10
7      AL=PREV(1)-15.24
8      AU=PREV(1)+15.24
9      LZ=(PARM(1),LT,AL)
10     UZ=(PARM(1),GT,AU)
11     IF(LZ.OR,UZ)PARM(1)=PREV(1)
12     10 CONTINUE
13     DO 20 L=2,4
14     IF(IOVER(L),NE,0) GO TO 20
15     AL=PREV(L)-0.0873
16     AU=PREV(L)+0.0873
17     LZ=(PARM(L),LT,AL)
18     UZ=(PARM(L),GT,AU)
19     IF(LZ.OR,UZ)PARM(L)=PREV(L)
20     20 CONTINUE
21     IF(IOVER(5),NE,0)GO TO 30
22     AL=PREV(5)-2.57
23     AU=PREV(5)+2.57
24     LZ=(PARM(5),LT,AL)
25     UZ=(PARM(5),GT,AU)
26     IF(LZ.OR,UZ)PARM(5)=PREV(5)
27     30 RETURN
28     END

```

MODULE INFORMATION:

CODE AREA SIZE = 01AEH 430D
VARIABLE AREA SIZE = 0010H 16D
MAXIMUM STACK SIZE = 0004H 4D
31 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT RUNLMT

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;
		2	;
		3	;
		4	NAME TIMEFP
		5	;
		6	COMPUTES FLOATING POINT VALUE OF TIME IN SECONDS BY:
		7	;
		8	;
		9	T1=FLOAT(IH*I60+IM)*F60+FLOAT(ITS)/F10
		10	;
		11	WHERE I60=60, F60=60., AND F10=10.
		12	;
0012		13	MULT EQU 12H
0013		14	DIV EQU 13H
0010		15	AD EQU 10H
006C		16	IAD EQU 6CH
006E		17	IMULT EQU 6EH
001D		18	FLOAT EQU 1DH
		19	;
		20	EXTRN INTLOD,AMDCMD,AMDSTR,AMDLOD
		21	;
		22	PUBLIC TIMEFP
		23	;
		24	CSEG
		25	;
0000	E1	26	TIMEFP: POP H ;SAVE RETURN ADDR
0001	CD0000	27	CALL INTLOD ;IH ↑
0004	42	28	MOV B,D
0005	4B	29	MOV C,E
0006	CD0000	30	CALL INTLOD ;I60 ↑
0009	3E6E	31	MVI A,IMULT
000B	CD0000	32	CALL AMDCMD ;X
000E	C1	33	POP B
000F	CD0000	34	CALL INTLOD ;IM ↑
0012	3E6C	35	MVI A,IAD
0014	CD0000	36	CALL AMDCMD ;+
0017	3E1D	37	MVI A,FLOAT
0019	CD0000	38	CALL AMDCMD ;FLOAT
001C	C1	39	POP B
001D	CD0000	40	CALL AMDLOD ;F60 ↑
0020	3E12	41	MVI A,MULT
0022	CD0000	42	CALL AMDCMD ;X
0025	C1	43	POP B
0026	CD0000	44	CALL INTLOD ;ITS ↑
0029	3E1D	45	MVI A,FLOAT
002B	CD0000	46	CALL AMDCMD ;FLOAT
002E	C1	47	POP B
002F	CD0000	48	CALL AMDLOD ;F10 ↑
0032	3E13	49	MVI A,DIV
0034	CD0000	50	CALL AMDCMD ;/
0037	3E10	51	MVI A,AD
0039	CD0000	52	CALL AMDCMD ;+
003C	C1	53	POP B
003D	CD0000	54	CALL AMDSTR ;SAVE AT T1
0040	E9	55	PCHL ;RETURN
		56	;
		57	END

PUBLIC SYMBOLS
 TIMEFP C 0000
 EXTERNAL SYMBOLS

AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000 INTLDD E 0000

USER SYMBOLS

AD	A 0010	AMDCMD E 0000	AMDLDD E 0000	AMDSTR E 0000	DIV	A 0013
FLOAT	A 001D	IAD A 006C	IMULT A 006E	INTLDD E 0000	MULT	A 0012
TIMEFP	C 0000					

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME TMP12
		2	
		3	
		4	SUBROUTINE TMP12(TMP1,VEL,X12)
		5	
		6	::: TMP1=X12*(VEL**2), WHERE X12=XK1 OR XK2
0017		7	PUSHT EQU 17H
0012		8	MULT EQU 12H
		9	
		10	EXTRN AMDL0D,AMDCMD,AMDSTR
		11	
		12	PUBLIC TMP12
		13	
		14	CSEG
		15	
0000	E1	16	TMP12: POP H ;SAVE RTN ADDR OFF STACK
0001	CD0000	17	CALL AMDL0D ;VEL ↑
0004	3E17	18	MVI A,PUSHT
0006	CD0000	19	CALL AMDCMD ;VEL ↑
0009	3E12	20	MVI A,MULT
000B	CD0000	21	CALL AMDCMD ;VEL*VEL
000E	42	22	MOV B,D
000F	4B	23	MOV C,E
0010	CD0000	24	CALL AMDL0D ;XK1 OR XK2 ↑
0013	3E12	25	MVI A,MULT
0015	CD0000	26	CALL AMDCMD ;X12*(VEL**2)
0018	C1	27	POP B
0019	CD0000	28	CALL AMDSTR ;TMP1 = "
001C	E9	29	PCHL ;RETURN
		30	
		31	END

PUBLIC SYMBOLS
TMP12 C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDL0D E 0000 AMDSTR E 0000

USER SYMBOLS
AMDCMD E 0000 AMDL0D E 0000 AMDSTR E 0000 MULT A 0012 PUSHT A 0017 TMP12 C 00

ASSEMBLY COMPLETE, NO ERRORS

ORIGINAL PAGE 2
OF POOR QUALITY

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME TRM1
		2	;
		3	SUBROUTINE TRM1(T1,BMW,ANG,ALT,RL)
		4	;
		5	::: T1=2.*TAN(BMW/2.)*(ALT**2)/COS(RL)/COS(ANG)
		6	;
0017		7	PUSHT EQU 17H
0003		8	COS EQU 03H
0012		9	MULT EQU 12H
0010		10	FADD EQU 10H
0013		11	FDIV EQU 13H
0004		12	TAN EQU 04H
		13	;
		14	EXTRN AMDL0D,AMDCMD,AMDSTR,GIVE
		15	;
		16	PUBLIC TRM1
		17	;
		18	CSEG
		19	;
0000	E1	20	TRM1: POP H ;HOLD RTN ADDR
0001	CD0000	21	CALL AMDL0D ;ALT ↑
0004	3E17	22	MVI A,PUSHT
0006	CD0000	23	CALL AMDCMD ;ALT ↑
0009	3E12	24	MVI A,MULT
000B	CD0000	25	CALL AMDCMD ;ALT*ALT
000E	42	26	MOV B,D
000F	4B	27	MOV C,E
0010	CD0000	28	CALL AMDL0D ;RL ↑
0013	3E03	29	MVI A,COS
0015	CD0000	30	CALL AMDCMD ;COS(RL):ALT**2:-:-
0018	3E13	31	MVI A,FDIV
001A	CD0000	32	CALL AMDCMD ;(ALT**2)/COS(RL):-:-:-
001D	C1	33	POP B
001E	CD0000	34	CALL AMDL0D ;ANG ↑
0021	3E03	35	MVI A,COS
0023	CD0000	36	CALL AMDCMD ;COS(ANG): (ALT...):-:-
0026	3E13	37	MVI A,FDIV
0028	CD0000	38	CALL AMDCMD ; ((ALT...) / COS(ANG)):-:-:-
002B	C1	39	POP B
002C	CD0000	40	CALL AMDL0D ;BMW ↑
002F	115300	41	LXI B,TWO
0032	CD0000	42	CALL GIVE ;2. ↑
0035	3E13	43	MVI A,FDIV
0037	CD0000	44	CALL AMDCMD ;BMW/2.:(ALT...ANG)
003A	3E04	45	MVI A,TAN
003C	CD0000	46	CALL AMDCMD ;TAN(BMW/2.): (ALT...):-:-
003F	3E12	47	MVI A,MULT
0041	CD0000	48	CALL AMDCMD ;TAN(..)*(ALT...)
0044	3E17	49	MVI A,PUSHT
0046	CD0000	50	CALL AMDCMD ;TAN(..)*(ALT...);TAN(..)*(ALT...)
0049	3E10	51	MVI A,FADD
004B	CD0000	52	CALL AMDCMD ;2.*TAN(..)*(ALT..)
004E	C1	53	POP B
004F	CD0000	54	CALL AMDSTR ;T1= "
0052	E9	55	PCHL ;RETURN
		56	;
0053	00	57	TWO: DB 00H,00H,80H,02H
0054	00		
0055	80		
0056	02		
		58	;
		59	END

PUBLIC SYMBOLS
TRM1 C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000 GIVE E 0000

USER SYMBOLS
AMDCMD E 0000 AMDLDD E 0000 AMDSTR E 0000 COS A 0003 FADD A 0010 FDIV A 00
GIVE E 0000
MULT A 0012 PUSHT A 0017 TRM1 A 0004 TRM1 C 0000 TWO C 0053

ASSEMBLY COMPLETE, NO ERRORS

```

LOC  OBJ      LINE      SOURCE STATEMENT
      1          NAME      TRM23
      2          ;
      3          ;
      4          ;
      5          ;
1.))/;
      6          ;
      7          ;
0017  8  PUSHT  EQU      17H
0012  9  MULT  EQU      12H
0011 10  FSUB  EQU      11H
0010 11  FADD  EQU      10H
0001 12  SQRT  EQU      01H
0004 13  TAN   EQU      04H
0003 14  COS   EQU      03H
0019 15  XCHF  EQU      19H
0013 16  FDIV  EQU      13H
      17          ;
      18  EXTRN  AMDLOD,AMDCMD,AMDSTR,GET,GIVE
      19          ;
      20  PUBLIC TRM23
      21          ;
      22  CSEG
      23          ;
0000  E1      24  TRM23:  POP     H           ;SAVE RTN ADDR
0001  220000  25          SHLD   ISAVE  ;IN INTERNAL BUFFER
0004  EB      26          XCHG  ;HOLD CY OF DR ADDR IN RP(HL)
0005  CD0000  27          CALL  AMDLOD ;RL ↑
0008  3E04    28          MVI   A,TAN   ;TAN(RL)
000A  CD0000  29          CALL  AMDCMD ;TAN(RL)
000D  3E17    30          MVI   A,PUSHT
000F  CD0000  31          CALL  AMDCMD ;TAN(RL);TAN(RL):-:-
0012  CD0000  32          CALL  AMDCMD ;" " " " ;TAN(RL):-
0015  110200  33          LXI   D,ISAVE+2
0018  CD0000  34          CALL  GET     ;SAVE CY OF TAN(RL)
001B  3E12    35          MVI   A,MULT
001D  CD0000  36          CALL  AMDCMD ;TAN(RL)**2
0020  C1      37          POP     B
0021  CD0000  38          CALL  AMDLOD ;TMP1 ↑
0024  3E17    39          MVI   A,PUSHT
0026  CD0000  40          CALL  AMDCMD ;TMP1;TMP1;TAN(RL)**2:-
0029  110600  41          LXI   D,ISAVE+6
002C  CD0000  42          CALL  GET     ;SAVE CY OF TMP1
002F  11C600  43          LXI   D,ONE
0032  CD0000  44          CALL  GIVE    ;1. ↑
0035  3E11    45          MVI   A,FSUB
0037  CD0000  46          CALL  AMDCMD ;(TMP1-1.);TAN(RL)**2:-:-
003A  3E12    47          MVI   A,MULT
003C  CD0000  48          CALL  AMDCMD ;(TMP1-1.)*TAN(RL)**2:-:-:-
003F  11C600  49          LXI   D,ONE
0042  CD0000  50          CALL  GIVE    ;1. ↑
0045  3E11    51          MVI   A,FSUB
0047  CD0000  52          CALL  AMDCMD ;(TMP1.....**2)-1.:-:-:-
004A  44      53          MOV    B,H
004B  4D      54          MOV    C,L
004C  CD0000  55          CALL  AMDLOD ;DR ↑
004F  3E03    56          MVI   A,COS
0051  CD0000  57          CALL  AMDCMD ;COS(DR)
0054  3E17    58          MVI   A,PUSHT
0056  CD0000  59          CALL  AMDCMD ;COSD;COSD:(TMP1...):-:-
0059  3E12    60          MVI   A,MULT
005B  CD0000  61          CALL  AMDCMD ;(COS(DR)**2):(TMP1...):-:-
005E  3E17    62          MVI   A,PUSHT

```

LOC	OBJ	LINE	SOURCE STATEMENT
0060	CD0000	E 63	CALL AMDCMD ;COSD**2;COSD**2;(TMP1...):-
0063	110A00	D 64	LXI D,ISAVE+10
0066	CD0000	E 65	CALL GET ;SAVE CY OF COS(DR)**2
0069	110600	D 66	LXI D,ISAVE+6
006C	CD0000	E 67	CALL GIVE ;TMP1 ↑
006F	3E12	E 68	MVI A,MULT
0071	CD0000	E 69	CALL AMDCMD ;TMP1*COSD**2:(...)
0074	3E10	E 70	MVI A,FADD
0076	CD0000	E 71	CALL AMDCMD ;((TAN**2 ...-1.):-!:-!
0079	3E01	E 72	MVI A,SQRT
007B	CD0000	E 73	CALL AMDCMD ;SQRT(...)
007E	42	E 74	MOV B,D
007F	4B	E 75	MOV C,E
0080	CD0000	E 76	CALL AMDLOD ;DR ↑
0083	3E04	E 77	MVI A,TAN
0085	CD0000	E 78	CALL AMDCMD ;TAN(DR)
0088	110200	D 79	LXI D,ISAVE+2
008B	CD0000	E 80	CALL GIVE ;TAN(RL) ↑
008E	3E12	E 81	MVI A,MULT
0090	CD0000	E 82	CALL AMDCMD ;TANR*TAND;SQRT(...)
0093	3E11	E 83	MVI A,FSUB
0095	CD0000	E 84	CALL AMDCMD ;SQRT(..)-TANR*TAND
0098	110600	D 85	LXI D,ISAVE+6
009B	CD0000	E 86	CALL GIVE ;TMP1 ↑
009E	110A00	D 87	LXI D,ISAVE+10
00A1	CD0000	E 88	CALL GIVE ;COS(DR)**2 ↑
00A4	3E12	E 89	MVI A,MULT
00A6	CD0000	E 90	CALL AMDCMD ;(TMP1*COSD**2);SQRT(...):-!
00A9	11C600	C 91	LXI D,ONE
00AC	CD0000	E 92	CALL GIVE ;1. ↑
00AF	3E19	E 93	MVI A,XCHF
00B1	CD0000	E 94	CALL AMDCMD ;(TMP1...):1.;SQRT(...):-
00B4	3E11	E 95	MVI A,FSUB
00B6	CD0000	E 96	CALL AMDCMD ;1.-(TMP1...);SQRT(...)
00B9	3E13	E 97	MVI A,FDIV
00BB	CD0000	E 98	CALL AMDCMD ;(SQRT(..)-TAN(...))/(1. ...)
00BE	C1	E 99	POP B
00BF	CD0000	E 100	CALL AMDSTR ;TRM1 = "
00C2	2A0000	D 101	LHLD ISAVE
00C5	E9	E 102	PCHL ;RETURN
		E 103	;
00C6	00	E 104	DB 00H,00H,80H,01H
00C7	00		
00C8	80		
00C9	01		
		E 105	;
		E 106	DSEG ;
		E 107	;
0000		E 108	ISAVE: DS 14
		E 109	;
		E 110	END

PUBLIC SYMBOLS
TRM23 C 0000

EXTERNAL SYMBOLS

AMDCMD	E 0000	AMDLOD	E 0000	AMDSTR	E 0000	GET	E 0000	GIVE	E 0000
AMDCMD	E 0000	AMDLOD	E 0000	AMDSTR	E 0000	COS	A 0003	FADD	A 0010
FSUB	A 0011					FDIV	A 001		
GET	E 0000	GIVE	E 0000	ISAVE	D 0000	MULT	A 0012	ONE	C 00C6
SQRT	A 0001					PUSHT	A 001		
TAN	A 0004	TRM23	C 0000	XCHF	A 0019				

LOC	OBJ	LINE	SOURCE STATEMENT
ASSEMBLY COMPLETE, NO ERRORS			

LOC	OBJ	LINE	SOURCE STATEMENT
		1 ;	SUBROUTINE TSECS(ITS,ITN,IHD,IUN)
		2 ;	
		3 ;	
		4 ;	NAME TSECS
		5 ;	
		6 ;	EVALUATES SECONDS BY:
		7 ;	
		8 ;	ITS=10*(ITN+10*IHD)+IUN
		9 ;	
006C		10 IAD	EQU 6CH
006E		11 IMULT	EQU 6EH
		12 ;	
		13 ;	EXTRN INTLOD,AMDCMD,AMDSTR,AMDLOD,INTSTR
		14 ;	PUBLIC TSECS
		15 ;	
		16 ;	CSEG
		17 ;	
		18 TSECS:	POP H ;SAVE RETURN ADDR
0001	E1	19	CALL INTLOD ;IHD ↑
0004	CD0000	E 20	LXI B,TEN
0007	013200	E 21	CALL INTLOD ;10 ↑
000A	CD0000	E 22	MVI A,IMULT
000C	3E6E	E 23	CALL AMDCMD ;X
000F	CD0000	E 24	POP B
0010	C1	E 25	CALL INTLOD ;ITN
0013	CD0000	E 26	MVI A,IAD
0015	3E6C	E 27	CALL AMDCMD ;+
0018	CD0000	E 28	LXI B,TEN
001B	013200	E 29	CALL INTLOD ;10 ↑
001E	CD0000	E 30	MVI A,IMULT
0020	3E6E	E 31	CALL AMDCMD ;X
0023	42	32	MOV B,D
0024	4B	33	MOV C,E
0025	CD0000	E 34	CALL INTLOD ;IUN ↑
0028	3E6C	35	MVI A,IAD
002A	CD0000	E 36	CALL AMDCMD ;+
002D	C1	37	POP B
002E	CD0000	E 38	CALL INTSTR ;SAVE RESULT IN ITS
0031	E9	39	PCHL ;RETURN
		40 ;	
0032	0A00	41 TEN:	DW 10
		42	END

PUBLIC SYMBOLS
TSECS C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLOD E 0000 AMDSTR E 0000 INTLOD E 0000 INTSTR E 0000

USER SYMBOLS
AMDCMD E 0000 AMDLOD E 0000 AMDSTR E 0000 IAD A 006C IMULT A 006E
INTLOD E 0000 INTSTR E 0000 TEN C 0032 TSECS C 0000

ASSEMBLY COMPLETE, NO ERRORS

```

LOC OBJ          LINE      SOURCE STATEMENT
1              NAME UNPACK
2              ;
3              ;
4              ;
5              ;
6              ;
7 PUBLIC UNPACK,RR4B
8              ;
9 CSEG
10             ;
0000 03         11 UNPACK: INX      B      ;SKIP FIRST 2 NIBS, FRAME SYNC PART 1
0001 261B      12         MVI      H,01BH ;SET BYTE COUNTER = 255
0003 0A        13         LDAX    B      ;GET SYNC NIB 3 AND FRAME NBR
0004 12        14         STAX    D      ;STORE AT BYTE 1 LOCATION
15             ;
0005 03        16 STEP:  INX      B      ;SET POINTERS TO NEXT
0006 13        17         INX      D      ;BYTE, STARTING AT NIB 5
0007 0A        18         LDAX    B      ;GET BYTE IN ACCUM
0008 6F        19         MOV     L,A    ;SAVE A CY IN R1E)
0009 E6F0      20         ANI     OF0H ;STRIP OUT UPPER NIB
000B CD1900    C 21         CALL   RR4B  ;ROTATE ACCUM 4 BITS RIGHT
000E 12        22         STAX    D      ;SAVE UPPER NIB
000F 13        23         INX      D      ;ADVANCE ADDR POINTER
0010 7D        24         MOV     A,L    ;GET NEW CY OF BYTE
0011 E60F      25         ANI     OFH  ;STRIP OUT LOWER NIB
0013 12        26         STAX    D      ;SAVE AT NEXT BYTE LOC
0014 25        27         DCR     H      ;SUB 1 FROM BYTE COUNT
0015 C20500    C 28         JNZ    STEP  ;LOOP TILL DONE
0018 C9        29         RET
30             ;
0019 0F        31 RR4B:  RRC      ;ROTATE
001A 0F        32         RRC      ;ACCUM
001B 0F        33         RRC      ;RIGHT
001C 0F        34         RRC      ;4 POSITIONS
001D C9        35         RET
36             END

```

PUBLIC SYMBOLS
RR4B C 0019 UNPACK C 0000

EXTERNAL SYMBOLS

USER SYMBOLS
RR4B C 0019 STEP C 0005 UNPACK C 0000

ASSEMBLY COMPLETE, NO ERRORS

ISIS-II FORTRAN-80 V2.0 COMPILATION OF PROGRAM UNIT VALIE
OBJECT MODULE PLACED IN IF1:VALID.OBJ
COMPILER INVOKED BY: FORT80 IF1:VALID.SRC DEBUG

```

1      SUBROUTINE VALIE(DL,DF,IOV,IVAL,PARM)
      C
      C ***** VALIDATES NERDAS DATA VALUES UNLESS THE OVER-RIDE FLAG IS
      C SET. IF VALUE IS OUTSIDE PRESET LIMITS, THE DEFAULT VALUE IS USED.
      C WHENEVER DEFAULT VALUES ARE USED AN APPROPRIATE FLAG IS SET.
2      INTEGER*1 IOV(5),IVAL
3      DIMENSION DL(5,2),DF(5),PARM(5)
4      LOGICAL LZ,UZ
      C
      C *** PARM IS AN ORDERED SET: ALT,DRF,ROL,FCH,VEL
5      IVAL=0
6      KK=16
      C
7      DO 20 K=1,5
8          IDFG=0
9          IF(IOV(K).NE.0) GO TO 10
10         LZ=(PARM(K).GE.DL(K,1))
11         UZ=(PARM(K).LE.DL(K,2))
12         IF(LZ.AND.UZ)GO TO 10
13         PARM(K)=DF(K)
14         IDFG=1
15     10    CONTINUE
16         IF(IDFG.NE.0)IVAL=IVAL+KK
17         KK=KK/2
18     20    CONTINUE
19         RETURN
20         END

```

MODULE INFORMATION:

CODE AREA SIZE = 010AH 266D
VARIABLE AREA SIZE = 0012H 18D
MAXIMUM STACK SIZE = 0006H 6D
29 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT VALIE

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;
		2	SUBROUTINE VELFP(VEL,CVEL,ITN,IHD,IUN)
		3	;
		4	NAME VELFP
		5	;
		6	COMPUTES VELOCITY IN METERS/SECOND;
		7	;
		8	VEL=FLOAT(IUN+10*(ITN+10*IHD))*CVEL
		9	WHERE CVEL=0.514
		10	;
0013		12	DIV EQU 13H
0012		13	MULT EQU 12H
006C		14	IAD EQU 6CH
006E		15	IMULT EQU 6EH
001D		16	FLOAT EQU 1DH
		17	;
		18	EXTRN INTLOD,AMDCMD,AMDSTR,AMDLOD
		19	PUBLIC VELFP
		20	;
		21	CSEG
		22	;
0000	E1	23	VELFP: POP H ;SAVE RETURN ADDR
0001	CD0000	24	CALL INTLOD ;IHD ↑
0004	014000	25	LXI B,TEN
0007	CD0000	26	CALL INTLOD ;10 ↑
000A	3E6E	27	MVI A,IMULT
000C	CD0000	28	CALL AMDCMD ;X
000F	C1	29	POP B
0010	CD0000	30	CALL INTLOD ;ITN
0013	3E6C	31	MVI A,IAD
0015	CD0000	32	CALL AMDCMD ;+
0018	014000	33	LXI B,TEN
001B	CD0000	34	CALL INTLOD ;10 ↑
001E	3E6E	35	MVI A,IMULT
0020	CD0000	36	CALL AMDCMD ;X
0023	42	37	MOV B,D
0024	4B	38	MOV C,E
0025	CD0000	39	CALL INTLOD ;IUN ↑
0028	3E6C	40	MVI A,IAD
002A	CD0000	41	CALL AMDCMD ;+
002D	3E1D	42	MVI A,FLOAT
002F	CD0000	43	CALL AMDCMD ;FLOAT
0032	C1	44	POP B
0033	CD0000	45	CALL AMDLOD ;CVEL ↑
0036	3E12	46	MVI A,MULT
0038	CD0000	47	CALL AMDCMD ;X
003B	C1	48	POP B
003C	CD0000	49	CALL AMDSTR ;SAVE RESULT IN VEL
003F	E9	50	PCHL ;RETURN
0040	0A00	51	TEN: DW 10
		52	END

PUBLIC SYMBOLS
VELFP C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000 AMDLOD E 0000 AMDSTR E 0000 INTLOD E 0000

USER SYMBOLS
AMDCMD E 0000 AMDLOD E 0000 AMDSTR E 0000 DIV A 0013 FLOAT A 001D
IAD A 006C IMULT A 006E INTLOD E 0000 MULT A 0012 TEN C 0040

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0

VELFP PAGE 2

VELFP C 0000

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME XK12
		2	;
		3	;
		4	SUBROUTINE XK12(XK2,XK1,SLMDA,BNDW,FDOP)
		5	;
		6	::: XK1 = 4/(SLMDA*(FDOP-BNDW/2))*2
		7	XK2 = 4/(SLMDA*(FDOP+BNDW/2))*2
0017		8	PUSHT EQU 17H
0012		9	FMUL EQU 12H
0019		10	EXCHG EQU 19H
0013		11	FDIV EQU 13H
0010		12	FADD EQU 10H
0011		13	FSUB EQU 11H
0018		14	ROL EQU 18H
		15	;
		16	EXTRN AMDLDD,AMDSTR,AMDCMD,INTLDD,INTSTR,GIVE,GET
		17	;
		18	PUBLIC XK12
		19	;
		20	CSEG
		21	;
0000	E1	22	XK12: POP H ;PULL RETURN ADDR OFF STACK
0001	CD0000	23	CALL AMDLDD ;BNDW ↑
0004	3E17	24	MVI A,PUSHT
0006	CD0000	25	CALL AMDCMD ;BNDW:BNDW!--!--
0009	D5	26	PUSH D
000A	117B00	27	LXI D,TWO
000D	CD0000	28	CALL GIVE
0010	3E13	29	MVI A,FDIV
0012	CD0000	30	CALL AMDCMD ;BNDW/2:BNDW!--!--
0015	C1	31	POP B
0016	CD0000	32	CALL AMDLDD ;FDOP ↑
0019	3E10	33	MVI A,FADD
001B	CD0000	34	CALL AMDCMD ;BNDW/2+FDOP:BNDW!--!--
001E	3E17	35	MVI A,PUSHT
0020	CD0000	36	CALL AMDCMD
0023	CD0000	37	CALL AMDCMD ;BNDW/2+FDOP:BNDW/2+FDOP:BNDW/2+FDOP:BNDW
0026	3E18	38	MVI A,ROL
0028	CD0000	39	CALL AMDCMD
002B	CD0000	40	CALL AMDCMD
002E	CD0000	41	CALL AMDCMD ;BNDW:BNDW/2+FDOP:BNDW/2+FDOP:BNDW/2+FDOP
0031	3E11	42	MVI A,FSUB
0033	CD0000	43	CALL AMDCMD ;FDOP-BNDW/2:FDOP+BNDW/2:FDOP+BNDW/2:--
0036	117B00	44	LXI D,TWO
0039	CD0000	45	CALL GIVE ;2 ↑
003C	C1	46	POP B
003D	CD0000	47	CALL AMDLDD ;SLMDA ↑
0040	3E13	48	MVI A,FDIV
0042	CD0000	49	CALL AMDCMD ;2/SLMDA:FDOP-BNDW/2:FDOP+BNDW/2:--
0045	3E17	50	MVI A,PUSHT
0047	CD0000	51	CALL AMDCMD ;2/SLMDA:2/SLMDA:FDOP-BNDW/2:FDOP+BNDW/2
004A	3E18	52	MVI A,ROL
004C	CD0000	53	CALL AMDCMD
004F	3E19	54	MVI A,EXCHG
0051	CD0000	55	CALL AMDCMD ;FDOP-BNDW/2:2/SLMDA:FDOP+BNDW/2:2/SLMDA
0054	3E13	56	MVI A,FDIV
0056	CD0000	57	CALL AMDCMD ;2/(SLMDA*(FDOP-BNDW/2)):FDOP+BNDW/2:2/SLM
		58	DA:-- MVI A,PUSHT
0059	3E17	58	MVI A,PUSHT
005B	CD0000	59	CALL AMDCMD
005E	3E12	60	MVI A,FMUL
0060	CD0000	61	CALL AMDCMD ;4/(SLMDA*(FDOP-BNDW/2))*2:FDOP+BNDW/2:2/SLMDA/2:--!--

LOC	OBJ	LINE	SOURCE STATEMENT
0063	C1	62	POP B
0064	CD0000	E 63	CALL AMDSTR ;SAVE RESULT IN XK1
0067	3E13	64	MVI A,FDIV
0069	CD0000	E 65	CALL AMDCMD ;2/(SLMDA*(FDDP+BNBW/2))!-!-!-!
006C	3E17	66	MVI A,PUSHT
006E	CD0000	E 67	CALL AMDCMD
0071	3E12	68	MVI A,FMUL
0073	CD0000	E 69	CALL AMDCMD ;4/(SLMDA*(FDDP+BNBW/2))*2!-!-!-!
0076	C1	70	POP B
0077	CD0000	E 71	CALL AMDSTR ;SAVE RESULT IN XK2
007A	E9	72	FCHL
		73	;
007B	00	74	TWO: DB 00H,00H,80H,02H
007C	00		
007D	80		
007E	02		
		75	END

PUBLIC SYMBOLS
XK12 C 0000

EXTERNAL SYMBOLS
AMDCMD E 0000
INTLOD E 0000

AMDLOD E 0000
INTSTR E 0000

AMDSTR E 0000

GET E 0000

GIVE E 0000

USER SYMBOLS

AMDCMD E 0000
FDIV A 0013
INTLOD E 0000
XK12 C 0000

AMDLOD E 0000
FMUL A 0012
INTSTR E 0000

AMDSTR E 0000
FSUB A 0011
PUSHT A 0017

EXCHG A 0019
GET E 0000
ROL A 0018

FADD A 0010
GIVE E 0000
TWO C 007B

ASSEMBLY COMPLETE, NO ERRORS

ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1

PAGE 1

CMD	12*	16	
CNCTL	11*	15	17
IUSART	1		
IUSRT	8	14*	
MODE	10*	14	

CROSS REFERENCE COMPLETE

ASCDV	5	21*			
CO	29	66*			
CCO	37	44*			
CP0	39	41	52*		
CP1	46	49	39	61*	
CP2	54	57*	68		
DC	27	36*			
DVERIF	1				
GC	25*	33			
STP	31*	64	69		
STP0	30*	42	50	55	63

CROSS REFERENCE COMPLETE

CO	7	24
CRYPT	23*	27
MSGOUT	5	20*
OUTPUT	1	
XIT	31	33*

CROSS REFERENCE COMPLETE

APPENDIX E

AMC 95/6511 Device Routine Listings

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;ROUTINE TO SEND A COMMAND IN THE A REGISTER TO THE AMC 95/6011
		2	;
		3	;
		4	ERRORS ARE TREATED IN THE FOLLOWING MANNER:
		5	1) UNDERFLOW RESULT = 0.0
		6	2) OVERFLOW RESULT * +/- 2**63
		7	3) DIVIDE BY ZERO RESULT * +/- 2**63
		8	4) SQRT OR LOG OF A
		9	NEGATIVE NUMBER RESULT = 0.0
		10	5) ARGUMENT OF INVERSE
		11	TRIG. OR EXP OUT
		12	OF RANGE RESULT * +/- 2**63
		13	NAME AMDCMD
		14	;
00F0		15	PORT EQU 00F0H
0018		17	PULL EQU 18H
		18	;
		19	PUBLIC AMDCMD
		20	;
		21	CSEG
		22	;
0000	F5	23	AMDCMD: PUSH PSW ;SAVE COMMAND
0001	DBF1	24	BZY: IN PORT+1 ;WAIT TILL NOT BUSY
0003	B7	25	ORA A
0004	FA0100	26	JM BZY
0007	F1	27	POP PSW ;GET COMMAND
0008	D3F1	28	OUT PORT+1 ;SEND COMMAND
000A	F5	29	PUSH PSW ;SAVE COMMAND
000B	DBF1	30	BUSY: IN PORT+1 ;WAIT TILL NOT BUSY (COMMAND DONE)
000D	B7	31	ORA A
000E	FA0B00	32	JM BUSY
0011	D5	33	PUSH D ;SAVE DE REGISTER PAIR
0012	5F	34	MOV E,A
0013	E61E	35	ANI 1EH ;STRIP OUT ERROR FLAGS
0015	CA4800	36	JZ NOERR ;JUMP ON NO ERROR
0018	3E18	37	MVI A,PULL
001A	E3F1	38	OUT PORT+1 ;PULL BAD VALUE OFF AMD STACK
001C	7B	39	MOV A,E ;GET STATUS BACK
001D	E61E	40	ANI 1EH ;STRIP OUT ERROR FLAGS
001F	FE08	41	CPI 08H ;SQRT OR LOG OF NEG NUMBER = UNDFLO
0021	CA3E00	42	JZ UNDFLO
0024	E604	43	ANI 04H ;UNDERFLOW
0026	C23E00	44	JNZ UNDFLO
0029	3EFF	45	MVI A,OFFH ;TREAT ALL OTHER ERRORS AS OVERFLOW
002B	1603	46	MVI D,03
002D	D3F0	47	LOOP: OUT PORT
002F	15	48	DCR D
0030	C22D00	49	JNZ LOOP
0033	7B	50	MOV A,E ;GET ERROR FLAGS
0034	17	51	RAL ;GET SIGN
0035	E680	52	ANI 80H
0037	F63F	53	ORI 3FH ;MAKE MAX EXP WITH PROPER SIGN
0039	D3F0	54	OUT PORT ;SEND TO BOARD

LOC	OBJ	LINE	SOURCE STATEMENT
003B	C34800	C 55	JMP NOERR
003E	3E00	56	UNDFLO: MVI A,00H ;UNDERFLOW, THEN SEND ZERO TO THE BOARD
0040	1604	57	MVI 04
0042	D3F0	58	LOOP2: OUT PORT
0044	15	59	DCR D
0045	C24200	C 60	JNZ LOOP2
0048	D1	61	NOERR: POP D
0049	F1	62	POP PSW
004A	C9	63	RET
		64	END

PUBLIC SYMBOLS
AMDCMD C 0000

EXTERNAL SYMBOLS

USER SYMBOLS
 AMDCMD C 0000 BUSY C 000B BZY C 0001 LOOP C 002B LOOP2 C 0042 NOERR C 0048
 PORT A 00F0
 PULL A 0018 UNDFLO C 003E
 ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	ROUTINE TO STORE A FLOATING POINT NUMBER (IN INTEL FORMAT)
		2	FROM THE AMC 95/6011 INTERNAL REGISTER AT THE ADDRESS IN BC
		3	;
		4	;
		5	ERRORS ARE TREATED IN THE FOLLOWING MANNER:
		6	1) UNDERFLOW RESULT = 0.0
		7	2) OVERFLOW RESULT ~ +/- 2**127
		8	3) DIVIDE BY ZERO RESULT ~ +/- 2**127
		9	4) SQRT OR LOG OF A
		10	NEGATIVE NUMBER RESULT = 0.0
		11	6) ARGUMENT OF INVERSE
		12	TRIG. OR EXP OUT
		13	OF RANGE RESULT ~ +/- 2**127
		14	NAME AMDSTR
		15	;
00F0		16	PORT EQU 00F0H
0018		17	PULL EQU 18H
		18	;
		19	;
		20	PUBLIC AMDSTR
		21	;
		22	CSEG
		23	;
0000	D5	24	AMDSTR: PUSH D ;SAVE DE REGISTER PAIR
0001	03	25	INX B ;MOVE POINTER TO BEGINNING
0002	03	26	INX B
0003	03	27	INX B
0004	DBF1	28	BUSY: IN PORT+1 ;WAIT TILL NOT BUSY
0006	B7	29	ORA A
0007	FA0400	30	JM BUSY
000A	5F	31	MOV E,A
000B	E61E	32	ANI 1EH ;CHECK FOR ERROR
000D	CA3800	33	JZ NOERR
0010	3E18	34	MVI A,PULL ;PULL BAD ENTRY OFF AMD STACK
0012	D3F1	35	OUT PORT+1
0014	7B	36	MOV A,E ;GET STATUS BACK
0015	E61E	37	ANI 1EH ;STRIP OUT ERROR FLAGS
0017	FE08	38	CPI 08H ;SQRT OR LOG OF A NEGATIVE NUMBER = UNDFLO
0019	CA3200	39	JZ UNDFLO
001C	E604	40	ANI 04H ;UNDERFLO
001E	C23200	41	JNZ UNDFLO
0021	7B	42	MOV A,E ;GET ERROR FLAGS
0022	17	43	RAL
0023	E680	44	ANI 80H
0025	F67F	45	ORI 7FH ;MAKE MAX EXPONENT WITH PROPER SIGN
0027	02	46	STAX B
0028	3EFF	47	MVI A,OFFH ;SAVE MANTISSA
002A	0B	48	BACK: DCX B
002B	02	49	STAX B
002C	0B	50	DCX B
002D	02	51	STAX B
002E	0B	52	DCX B
002F	02	53	STAX B
0030	D1	54	POP D ;RESTORE DE REGISTER PAIR

LOC	OBJ	LINE	SOURCE STATEMENT
0031	C9	55	RET
0032	3E00	56	UNDFLO: MVI A,00H ;UNDERFLOW, THEN RESULT = 0.0
0034	02	57	STAX B
0035	C32A00	58	JMP BACK
0038	DBF0	59	NOERR: IN PORT ;GET BYTE 3 (EXPONENT)
003A	07	60	RLC ;CONVERT EXPONENT TO INTEL FORMAT
003B	B7	61	ORA A
003C	F24500	62	JP SKIP1
003F	C6FC	63	ADI OFCH
0041	3F	64	CMC
0042	C34700	65	JMP SKIP2
0045	C6FC	66	ADI OFCH
0047	1F	67	SKIP1: RAR
0048	1F	68	RAR
0049	02	69	STAX B ;SAVE BYTE 3
004A	DBF0	70	IN PORT ;GET BYTE 2
004C	17	71	RAL ;REPLACE MSB WITH EXPONENT LSB
004D	0F	72	RRC
004E	0B	73	DCX B
004F	02	74	STAX B ;SAVE BYTE 2
0050	DBF0	75	IN PORT ;GET BYTE 1
0052	0B	76	DCX B
0053	02	77	STAX B ;SAVE BYTE 1
0054	DBF0	78	IN PORT ;GET BYTE 0
0056	0B	79	DCX B
0057	02	80	STAX B ;SAVE BYTE 0
0058	D1	81	POP D ;RESTORE DE REGISTER PAIR
0059	C9	82	RET
		83	END

PUBLIC SYMBOLS
AMDSTR C 0000

EXTERNAL SYMBOLS

USER SYMBOLS

AMDSTR C 0000 BACK C 002A BUSY C 0004 NOERR C 0038 PORT A 00F0 PULL A 0018
SKIP1 C 0045
SKIP2 C 0047 UNDFLO C 0032

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;ROUTINE TO LOAD A FLOATING POINT NUMBER (IN INTEL FORMAT) FROM
		2	THE ADDRESS IN BC INTO THE AMC 95/6011 INTERNAL REGISTER
		3	
		4	NAME AMDLOD
		5	
00F0		7	PORT EQU 00F0H
		8	
		9	PUBLIC AMDLOD
		10	
		11	CSEG
		12	
0000	DBF1	13	AMDLOD: IN PORT+1 ;READ STATUS
0002	B7	14	ORA A ;SET FLAGS
0003	FA0000	15	C JH AMDLOD ;LOOP IF BUSY
0006	0A	16	LDAX B
0007	D3F0	17	OUT PORT ;SEND BYTE 0
0009	03	18	INX B
000A	0A	19	LDAX B
000B	D3F0	20	OUT PORT ;SEND BYTE 1
000D	03	21	INX B
000E	0A	22	LDAX B ;GET BYTE 2
000F	F680	23	ORI 80H ;SET MSB
0011	D3F0	24	OUT PORT ;SEND
0013	0A	25	LDAX B ;GET BYTE 2
0014	17	26	RAL ;MOVE SIGN TO CARRY
0015	03	27	INX B
0016	0A	28	LDAX B
0017	17	29	RAL ;GET BYTE 3
0018	17	30	RAL ;SHIFT IN EXPONENT LSB
0019	DA2800	31	JC LABEL1 ;BEGIN CONVERTING EXPONENT
001C	FE7C	32	CPI 7CH ;EXP>=2**65
001E	D23400	33	JNC LABEL2
0021	1F	34	RAR
0022	3E80	35	MVI A,80H ;IF NOT EXP=2**65
0024	1F	36	RAR
0025	C33700	37	JMP LABEL3
0028	FE7C	38	LABEL1: CPI 7CH ;EXP<2**63
002A	DA3400	39	JC LABEL2
002D	1F	40	RAR
002E	3E7E	41	MVI A,7EH ;IF NOT EXP=2**62
0030	1F	42	RAR
0031	C33700	43	JMP LABEL3
0034	C604	44	LABEL2: ADI 04 ;COMPLETE CONVERTING EXPONENT
0036	0F	45	RRC
0037	D3F0	46	LABEL3: OUT PORT ;SEND EXPONENT(BYTE 3)
0039	C9	47	RET ;RETURN
		48	END

PUBLIC SYMBOLS
AMDLOD C 0000

EXTERNAL SYMBOLS

USER SYMBOLS
AMDLOD C 0000 LABEL1 C 0028 LABEL2 C 0034 LABEL3 C 0037 PORT A 00F0

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;ROUTINE TO STORE A 16-BIT INTEGER AT THE ADDRESS IN BC
		2	; FROM THE 95/6011 INTERNAL REGISTER
		3	;
		4	NAME INTSTR
		5	;
00F0		6	PORT EQU 00F0H
		7	;
		8	PUBLIC INTSTR
		9	;
		10	CSEG
		11	;
		12	;
0000	DBF1	13	INTSTR: IN PORT+1 ;WAIT TILL NOT BUSY
0002	B7	14	ORA A
0003	FA0000	15	JM INTSTR
0006	03	16	INX B
0007	DBF0	17	IN PORT
0009	02	18	STAX B ;SAVE BYTE 1
000A	0B	19	DCX B
000B	DBF0	20	IN PORT
000D	02	21	STAX B ;SAVE BYTE 0
000E	C9	22	RET
		23	END

PUBLIC SYMBOLS
INTSTR C 0000

EXTERNAL SYMBOLS

USER SYMBOLS
INTSTR C 0000 PORT A 00F0

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	ROUTINE TO GIVE A FLOATING POINT NUMBER (IN AMD FORMAT) AT THE
		2	ADDRESS IN DE TO THE AMC 95/8011 INTERNAL REGISTER
		3	
		4	NAME GIVE
		5	
00F0		7	PORT EQU 00F0H
		8	
		9	PUBLIC GIVE
		10	
		11	CSEG
		12	
0000	DBF1	13	GIVE: IN PORT+1 ;WAIT TILL NOT BUSY
0002	B7	14	ORA A
0003	FA0000	15	JM GIVE
0006	1A	16	LDAX D
0007	D3F0	17	OUT PORT ;SEND BYTE 0
0009	13	18	INX D
000A	1A	19	LDAX D
000B	D3F0	20	OUT PORT ;SEND BYTE 1
000D	13	21	INX D
000E	1A	22	LDAX D
000F	D3F0	23	OUT PORT ;SEND BYTE 2
0011	13	24	INX D
0012	1A	25	LDAX D
0013	D3F0	26	OUT PORT ;SEND BYTE 3
0015	1B	27	DCX D
0016	1B	28	DCX D
0017	1B	29	DCX D
0018	C9	30	RET
		31	END

PUBLIC SYMBOLS
GIVE C 0000

EXTERNAL SYMBOLS

USER SYMBOLS
GIVE C 0000 PORT A 00F0

ASSEMBLY COMPLETE, NO ERRORS

ALL SYMBOLS FACTORED
OF PAGE 1

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;ROUTINE TO GET A FLOATING POINT NUMBER (IN AMD FORMAT) FROM THE
		2	; AMC 95/6011 INTERNAL REGISTER AND STORE IT AT THE ADDRESS IN DE
		3	;
		4	NAME GET
		5	;
00F0		6	;
		7	PORT EQU 00F0H
		8	;
		9	PUBLIC GET
		10	;
		11	CSEG
		12	;
0000	13	13	GET: INX D ;MOVE POINTER
0001	13	14	INX D
0002	13	15	INX D
0003	DBF1	16	BSY: IN PORT+1 ;WAIT TILL NOT BUSY
0005	R7	17	ORA A
0006	FA0300	18	JH BSY
0009	DBF0	19	IN PORT
000B	12	20	STAX D ;SAVE BYTE 3
000C	1B	21	DCX D
000D	DBF0	22	IN PORT
000F	12	23	STAX D ;SAVE BYTE 2
0010	1B	24	DCX D
0011	DBF0	25	IN PORT
0013	12	26	STAX D ;SAVE BYTE 1
0014	1B	27	DCX D
0015	DBF0	28	IN PORT
0017	12	29	STAX D ;SAVE BYTE 0
0018	C9	30	RET
		31	END

PUBLIC SYMBOLS
GET C 0000

EXTERNAL SYMBOLS

USER SYMBOLS
BSY C 0003 GET C 0000 PORT A 00F0

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;ROUTINE TO LOAD A 16-BIT INTEGER AT THE ADDRESS IN BC INTO
		2	; THE AMC 95/6011 INTERNAL REGISTER
		3	;
		4	NAME INTLOD
		5	;
00F0		6	;
		7	PORT EQU 00F0H
		8	;
		9	CSEG
		10	;
		11	PUBLIC INTLOD
		12	;
0000	DBF1	13	INTLOD: IN PORT+1 ;WAIT TILL NOT BUSY
0002	B7	14	ORA A
0003	FA0000	15	JM INTLOD
0006	0A	16	LDAX B
0007	D3F0	17	OUT PORT ;SEND BYTE 0
0009	03	18	INX B
000A	0A	19	LDAX B
000B	D3F0	20	OUT PORT ;SEND BYTE 1
000D	C9	21	RET
		22	END

PUBLIC SYMBOLS
INTLOD C 0000

EXTERNAL SYMBOLS

USER SYMBOLS
INTLOD C 0000 PORT A 00F0

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	;ROUTINE TO LOAD A 32-BIT INTEGER NUMBER AT THE ADDRESS IN BC INTO
		2	; THE AMD 95/6011 INTERNAL REGISTER. BC IS INCREMENTED BY FOUR.
		3	;
		4	NAME I32LOD
		5	;
00F0		6	;
		7	PORT EQU 00F0H
		8	;
		9	PUBLIC I32LOD
		10	;
		11	CSEG
		12	;
0000	DBF1	13	I32LOD: IN PORT+1 ;WAIT TILL NOT BUSY
0002	B7	14	ORA A
0003	FA0000	15	JM I32LOD
0006	03	16	INX R
0007	03	17	INX R
0008	03	18	INX R
0009	0A	19	LDAX R
000A	D3F0	20	OUT PORT ;SEND BYTE 0
000C	0B	21	DCX R
000D	0A	22	LDAX R
000E	D3F0	23	OUT PORT ;SEND BYTE 1
0010	0B	24	DCX R
0011	0A	25	LDAX R
0012	D3F0	26	OUT PORT ;SEND BYTE 2
0014	0B	27	DCX R
0015	0A	28	LDAX R
0016	D3F0	29	OUT PORT ;SEND BYTE 3
0018	03	30	INX R
0019	03	31	INX R
001A	03	32	INX R
001B	03	33	INX R
001C	C9	34	RET
		35	END

PUBLIC SYMBOLS
I32LOD C 0000

EXTERNAL SYMBOLS

USER SYMBOLS
I32LOD C 0000 PORT A 00F0

ASSEMBLY COMPLETE, NO ERRORS