CR-172 106

FINAL TECHNICAL REPORT
GT/GTRI PROJECT A-4291

NASA-JSC ANTENNA NEAR-FIELD MEASUREMENT SYSTEM

(NASA-CR-172106) NASA-JSC ANTENNA NEAR-FIELD MEASUREMENT SYSTEM Final Technical Report (Georgia Inst. of Tech.) 277 p CSCL 20N
N89-15324
Unclas
G3/32 0185238

by:

W.P. Cooke, P.G. Friederich, B.M. Jenkins,
C.R. Jameson, and J. P. Estrada

Prepared for:
NASA
Lyndon B. Johnson Space Center
Houston, Texas 77058

OCTOBER 1988

GEORGIA INSTITUTE OF TECHNOLOGY
A Unit of the University System of Georgia
Atlanta, Georgia 30332

GTRI
GEORGIA TECH RESEARCH INSTITUTE

Final Technical Report

Project A-4291

NASA-JSC Antenna Near-Field Measurement System

By

W. P. Cooke, P. G. Friederich, B. M. Jenkins,

C. R. Jameson, and J. P. Estrada

October 1988

Prepared for

NASA

Lyndon B. Johnson Space Center

Houston, Texas 77058

Prepared by

Electromagnetic Effectiveness Division

Electronics and Computer Systems Laboratory

Georgia Tech Research Institute

Georgia Institute of Technology

Atlanta, Georgia 30332

FOREWORD

The work on the NASA-Johnson Space Center near-field antenna range described in this final technical engineering report was accomplished by personnel of the Georgia Tech Research Institute (GTRI) at the Georgia Institute of Technology, Atlanta, Georgia 30332. This program was supported by the Antenna Systems Section of NASA Lyndon B. Johnson Space Center, Houston, Texas 77058, under Contract No. NAS 9-17445. This program was designated by Georgia Tech as Project A-4291. Technical direction from the sponsor was provided by Ms. Sophia Tang, Mr. D. S. Eggers and Dr. G. D. Arndt. This report covers work performed from 1 May 1985 through 31 October 1988. Mr. W. P. Cooke served as the GTRI project director.

This work was performed under the general supervision of Dr. M. E. Cram, Chief, Electromagnetic Effectiveness Division and Mr. F. L. Cain, Director, Electronics and Computer Systems Laboratory (ECSL). In addition, the authors would like to acknowledge the support and helpful discussions provided by: (1) members of ECSL, Associate Director W. B. Warren and J. A. Woody, and (2) members of the Antenna Systems Section at NASA, including Dr. G. D. Arndt and Mr. D. S. Eggers. In addition, appreciation is extended to Mr. V. L. Daughtery for his administrative assistance and careful preparation of this manuscript.

TABLE OF CONTENTS

SECTION I

INTRODUCTION

A. Background

This report describes work performed by the personnel of the Electronics and Computer Systems Laboratory (ECSL), Georgia Tech Research Institute (GTRI) at the Georgia Institute of Technology for the Antenna Systems Section, NASA-Johnson Space Center (NASA-JSC). GTRI initiated efforts in February 1983 to assist NASA-JSC in the design of a large near-field range test facility to measure the response of thermal protection system (TPS) tile-covered antennas. The scope of the first phase near-field work was to evaluate suitable near-field measurement methods, analyze hardware needs and trade-offs, develop a preliminary algorithm, and recommend a near-field measurement system [1]. A baseline measurement system was developed with preliminary specifications and requirements for the system hardware. The rectangular measurement technique was recommended because it provided stationary antenna measurement and minimum algorithm implementation cost relative to the plane-polar method.

The second phase of this program began in May 1984. Work accomplished during this phase of the program included: (1) development of a range utilization procedure, (2) continued instrumentation receiver design, (3) control algorithm development, and (4) continued data processing algorithm development. One of the recommendations was fabrication of a near-field range receiver using a front-end down converter with a HP 8510 network analyzer for phase/amplitude detection [2].

Work on the third phase of this project began in May 1985. During this phase, efforts were focused primarily on: (1) final design and fabrication of the near-field range RF measurement system, (2) near-field range control software, (3) coordination of subsystem interfaces, and (4) mechanical consultations. In addition, GTRI updated the probe compensation capabilities of the data processing algorithm. The work accomplished during this third phase is summarized next.

B. Summary

Work was completed on the near-field range control software. The control software is menu driven with several features including: (1) full control of probe position and scanning, (2) selection of receiver parameters such as frequency and power level, and (3) real time data sampling, display and storage. The capabilities of the data processing software were expanded with the addition of probe compensation. In addition, the user can process the measured data from the same computer terminal used for range control. The design of the laser metrology system was completed. It provides precise measurement of probe location during near-field measurements as well as position data for control of the translation beam and probe cart. This topic is discussed further in Section II.C.

GTRI designed, fabricated and tested a near-field range measurement sytem, (in particular a near-field range receiver) that is capable of operating over the 1-26.5 GHz frequency band. With proper selection of down converter components (such as the mixers), the near-field range receiver is capable of operation up to 60 GHz. The near-field range measurement system is designed to capture 1000 data points per second. However, depending upon the data quality desired, the system is capable of even faster sampling. It has been operated on the laboratory bench at speeds up to 4,000 measurements per second. Another feature of the measurment system is the ability to measure the near-field distribution with the antenna-under-test in either a transmission or reception mode. The measurement system features an excellent noise figure for a receiver with a 1-26.5 GHz tunable bandwidth. Typically, the receiver has a 22 dB noise figure in the 1-6 GHz, a 32 dB noise figure in 6-18 GHz range, and a 39 dB noise figure in the 18-26.5 GHz frequency range. Also, the range measurement sytem is designed, to minimize the hardware changes needed to modify range configuration. Control of the measurement system is accomplished by the computer and control panels located in the control room. More information on this topic is presented in Section II. B.

SECTION II

System Description

A. Overall Facility

The NASA-JSC antenna near-field range operates over the frequency range 1 - 26.5 GHz. It consists of a mechanical scanner which drives a field probe over a planar measurement surface, a receiver subsystem to provide amplitude and phase information about electric fields on the measurement surface, a laser-based position monitoring subsystem to track the probe and provide control inputs to the drive motors, and an HP 1000 A-900 system controller for automation of the measurement process. An overall system block diagram is provided in Figure 1.

The mechanical scanner is capable of driving the probe over a raster scan approximately forty feet by forty feet. It consists of a translation beam, or "truss", which spans the structure approximately forty feet above the ground and travels either east or west along what has been designated the X- axis. Probes are interchangeable and will be selected based on a particular application, but each will mount in the probe carrier, or "cart", which traverses the truss in either a north or south direction along the designated Y-axis. An isometric view of the overall structure is provided in Figure 2.

The antenna under test (AUT) will be mounted on the AUT table, or "platform", which is attached to a hydraulic lift. The lift rises out of the floor from a point under the center of the scan plane. Thus the vertical separation between the probe and the AUT can be controlled along the Z- axis. In addition, the AUT table is supported on the lift by three jacks and monitored with two orthogonal inclinometers which the operator can use to level the AUT surface.

The system controller is a Hewlett Packard 1000 A-900 series mini-computer running under the RTE-A operating system. It communicates with the Laser metrology subsystem via three (Model HP12006) 16-bit parallel interface cards. The interface with the Receiver subsystem consists of three (Model HP12009) GPIB interface cards, two of which are dedicated to the HP 8510B network analyzers. The third is used for remote communications with the HP 8340 synthesized sources via HP 37204 HPIB Extenders. The bus extenders are necessary because the sources are located on

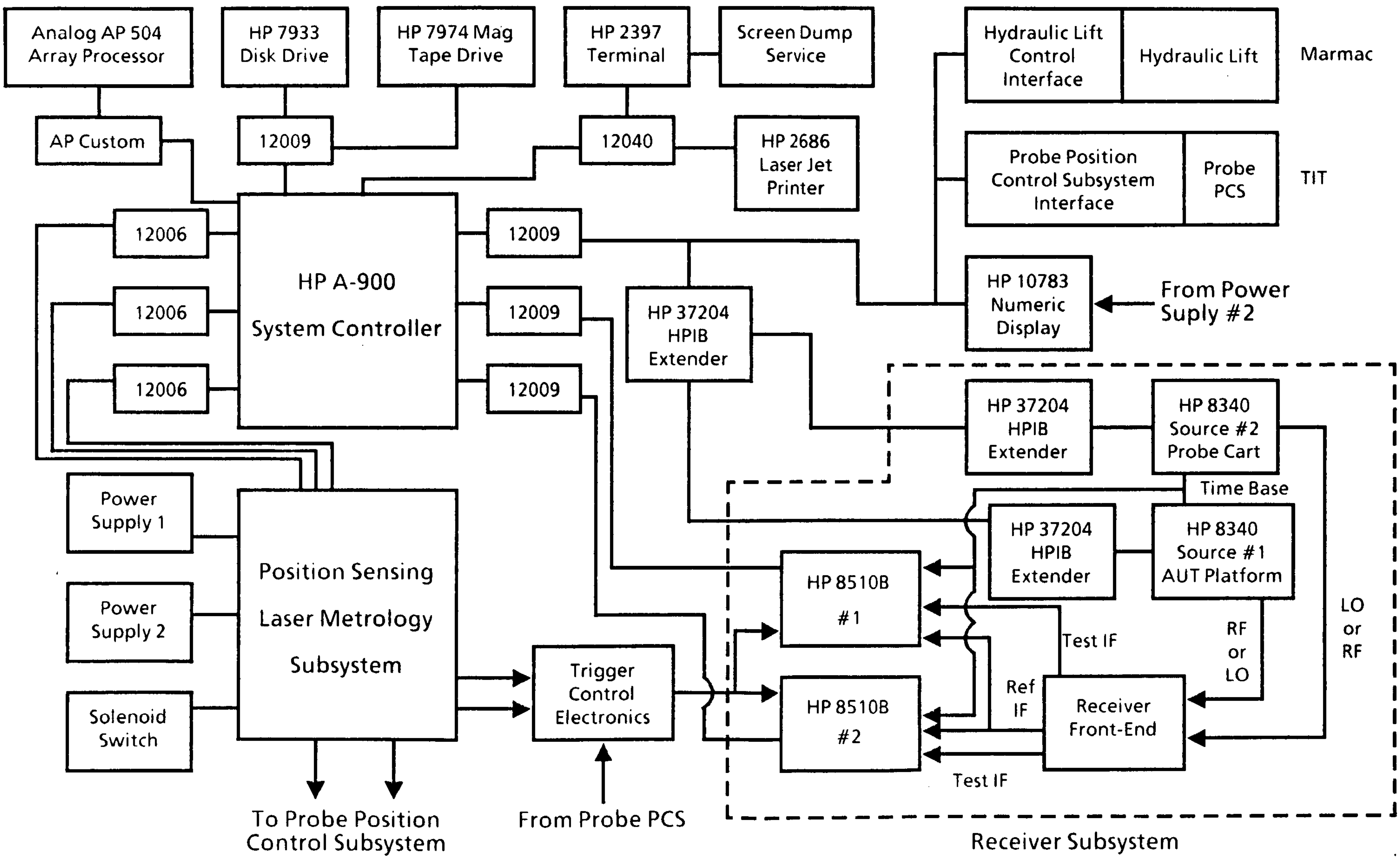


Figure 1. System Block Diagram

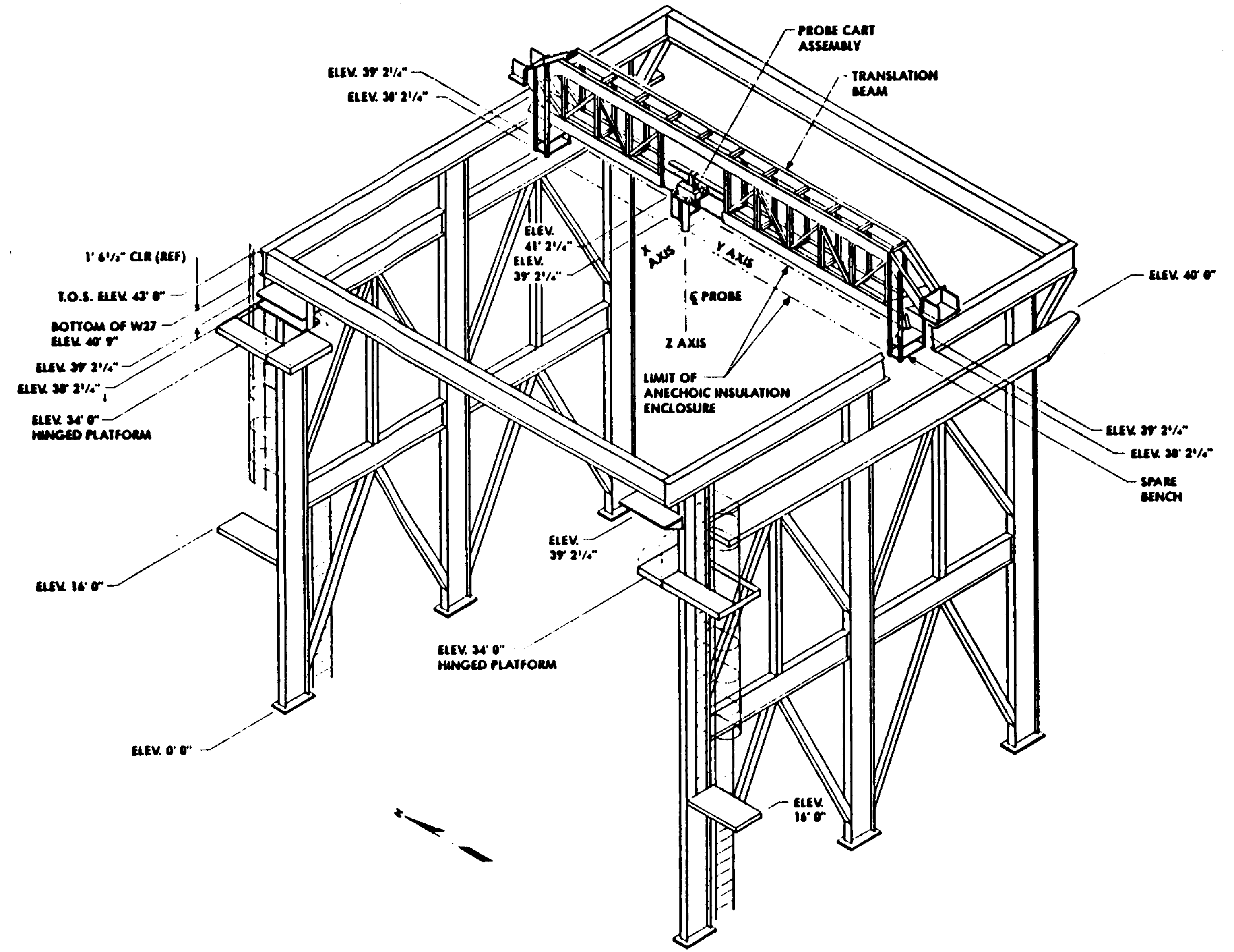


Figure 2. Isometric viewof planar scanning structure.

the probe cart and AUT platform. The third GPIB interface card is also used for communications with the Hydraulic lift control interface and the Probe Position Control interface when necessary. The GPIB communications with the Probe positioning system is accomplished using two ICS Electronics Model 4833 GPIB to Parallel converters. These units provide a number of parallel I/O lines, several of which are used to trigger data collection by the HP 8510B's via the Trigger Control Electronics. Typically, the data can be collected at rates up to 1000 points per second for a total of up to 4096 points per row scanned. Documentation on the data collection software used for control of and communication with the various subsystems is presented in a later section (see Section IV) of this report.

The probe position control subsystem is provided by Texas Integrated Technologies (TIT). One drive motor is located at each end of the truss and one is located at the probe cart. The position sensing laser metrology subsystem provides an 18-bit error signal for each motor based on a destination provided by the system controller. The error signals are converted to analog and used to drive the servo amplifiers attached to each motor. Further information about this subsystem can be obtained from TIT's reports.

The phase/amplitude receiver subsystem and the laser-based position sensing subsystem (including the trigger control electronics) are each described in greater detail in the following subsections.

B. Receiver Subsystem

B.1 Description

The near-field range receiver is a 1–26.5 GHz superheterodyne, dual-conversion system which precisely measures the phase and amplitude of the test signal relative to a reference signal. It utilizes two parallel test channels to permit simultaneous measurement of two polarizations. An overall functional diagram of the receiver system is given in Figure 3. Generically, each channel consists of an RF stage, two IF stages, and an IF processor. Specifically, the RF down converter and first IF stage are custom designed, while an HP 8510B Network Analyzer provides the second IF and IF processor functions. In the custom designed section of each channel of the receiver, the RF signal is down converted to a 20-MHz IF and then routed through an IF Control Chassis to the network analyzer. The network analyzer down converts the 20-MHz IF to a 100-kHz IF and processes this signal to obtain phase and amplitude data.

The functional diagram in Figure 3 illustrates the major components and primary signal interfaces of the receiver. The RF and LO signals are provided by two HP 8340B Synthesized Signal Generators. The Reference Receiver Front-end Down Converter samples the RF signal routed to the transmitting antenna. It down converts the sampled RF to a 20-MHz IF reference signal (IF-REF) which is routed via the IF Control Chassis to the two network analyzers. The two RF test signals (RF-1 and RF-2) from the receiving antenna are down converted to two 20-MHz IF test signals (IF-1 and IF-2) by the Test Receiver Front-end Down Converter. These two IF signals are also routed via the IF Control Chassis to the two network analyzers.

B.1.1 Test Receiver Front-end Down Converter

The Test Receiver Front-end Down Converter is illustrated in Figure 4. This down converter includes two identical parallel channels -- one for each polarization. The RF input to each channel is first routed through an electromechanical, interlocked RF switch. This switch permits the selection of either the received RF signal, a 50-ohm termination, or a Built-In-Test (BIT) signal. The output of the switch is routed through a 3-dB attenuator to the RF input of the mixer. All 3-dB attenuators, used throughout the receiver, are for impedance matching purposes.

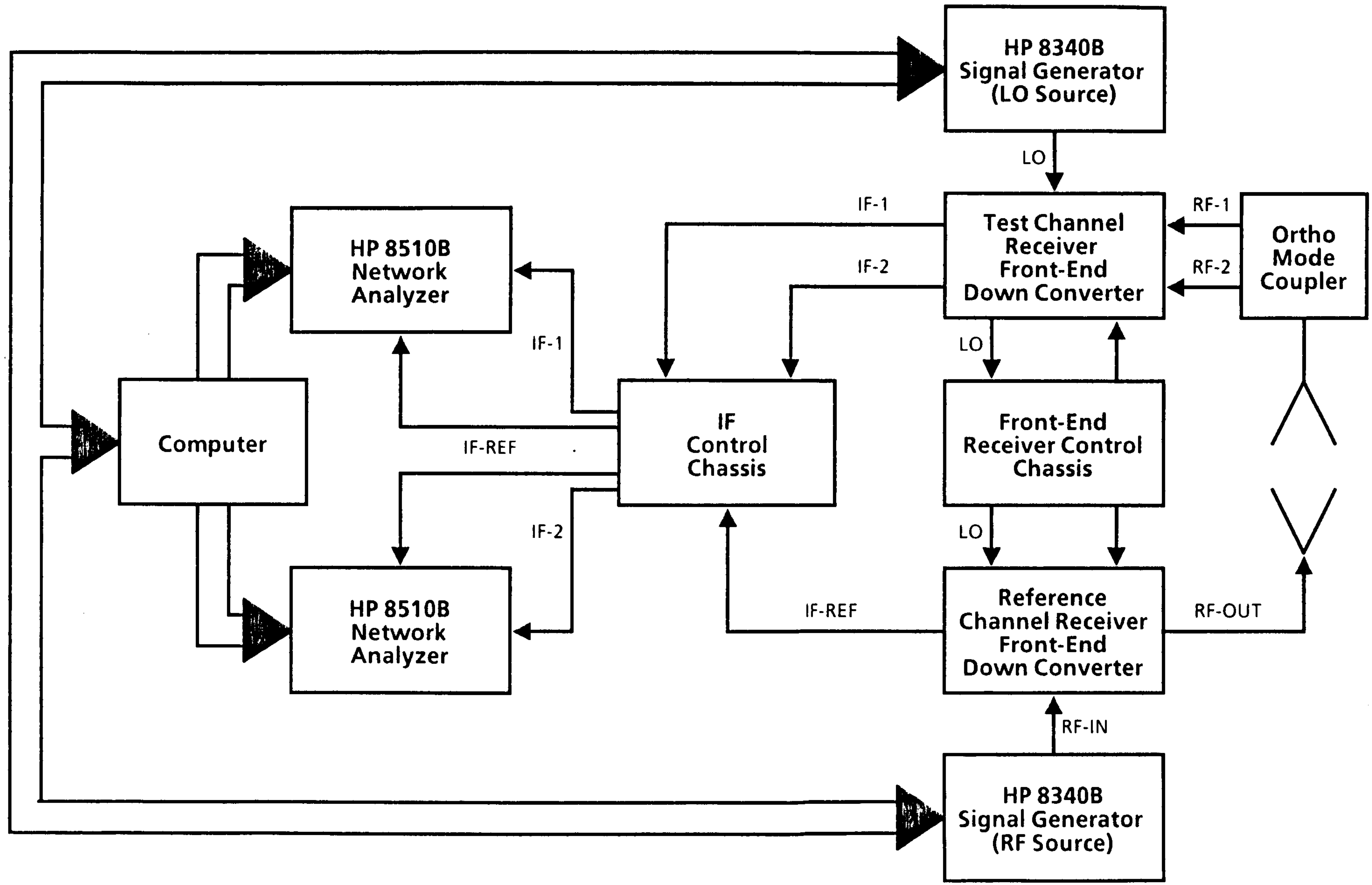


Figure 3. Functional Diagram of Near-Field Range Receiver.

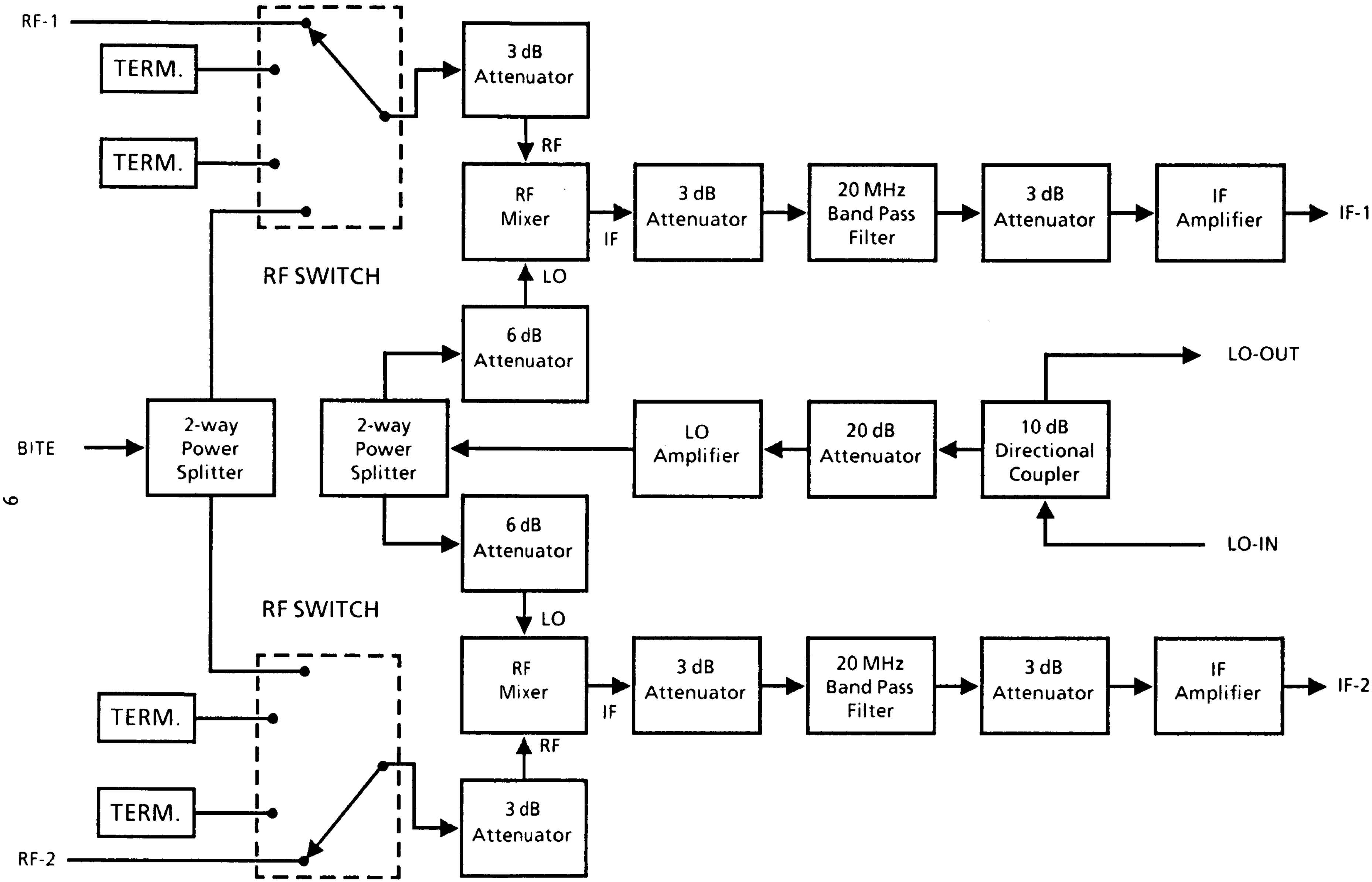


Figure 4. Test Channel Receiver Front-End Down Converter

The LO signal for the mixer is derived from the LO IN port of the down converter. Specifically, the LO IN signal is coupled through a 10-dB directional coupler to the 1–6 GHz LO amplifier via a 20-dB attenuator. The amplifier output is divided by a two-way splitter for use by each of the two polarization channels. This LO signal is attenuated by 6 dB prior to being applied to the mixer. The two attenuators (6 and 20 dB) and the LO amplifier were selected to provide the appropriate LO signal level to the mixer. The "through signal" output of the directional coupler goes to the LO OUT port of the down converter. It is then sent to the Receiver Front-end Control Chassis via coaxial cable.

Since the receiver system covers the 1–26.5 GHz frequency range, the RF mixer must be operated in both fundamental and harmonic modes to accommodate the 1–6 GHz LO amplifier. The RF frequency bands and the associated mixer modes are as follows:

1. 1–6 GHz: fundamental harmonic
2. 6–18 GHz: third harmonic
3. 18–26.5 GHz: fifth harmonic

The 20-MHz IF output of the mixer is sent through the first IF stage of the receiver system. This IF stage consists of a 20-MHz bandpass filter and a low noise amplifier with appropriate impedance matching attenuators. The output of the amplifier is connected to one of the IF output ports on the Test Receiver Front-end Down Converter (IF-1 or IF-2 depending on polarization channel). This IF signal is then sent via coaxial cable to the IF Control Chassis.

B.1.2 Reference Channel Receiver Front-end Down Converter

The Reference Channel Receiver Front-end Down Converter is illustrated in Figure 5. The major components in this down converter are the same as those in the Test Channel Receiver Front-end Down Converter except it has only one channel. The RF input to this chassis is routed from the RF-IN port through a 10-dB directional coupler to the RF-OUT port. This RF output signal is sent to the transmitting antenna via a coaxial cable. The coupled RF signal from the directional coupler is attenuated by 20 dB and is then sent to the RF input of the mixer.

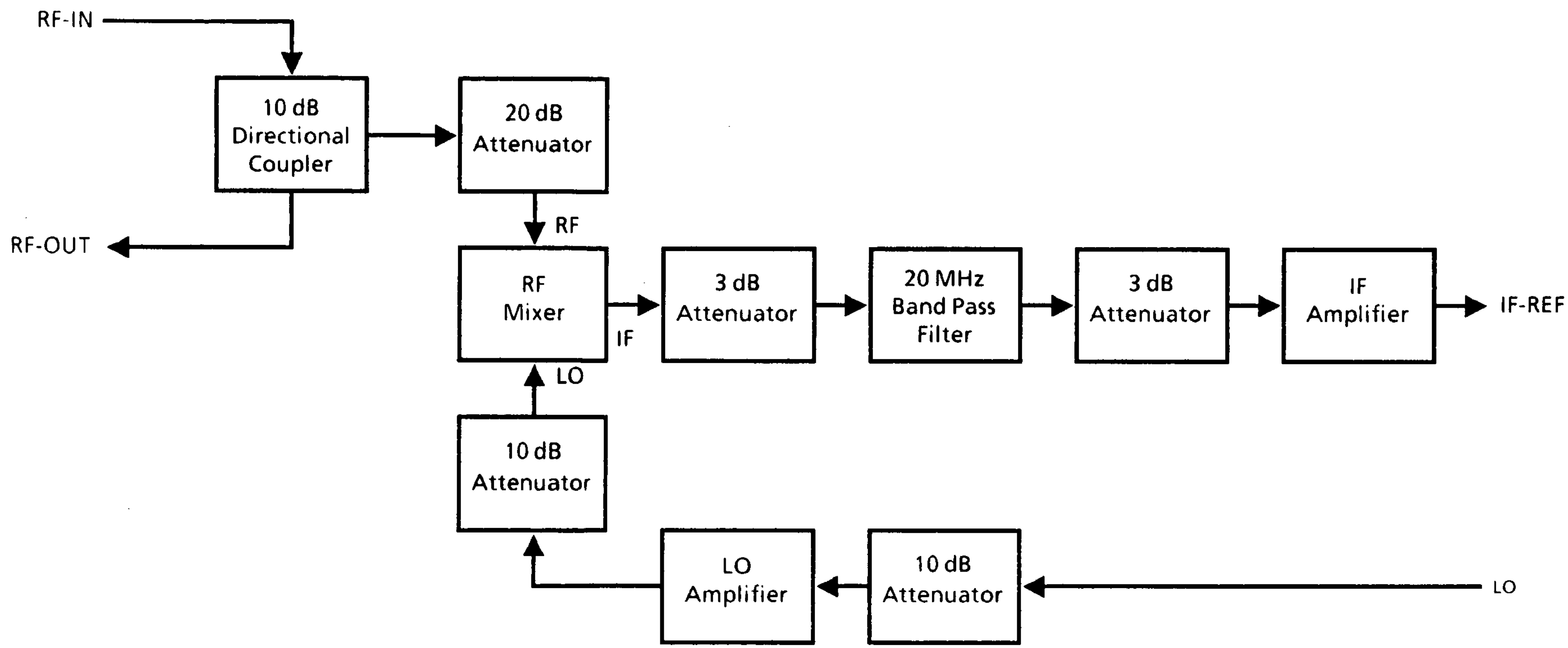


Figure 5. Reference Channel Receiver Front-End Down Converter.

The LO signal path from the LO IN port on the chassis to the LO input of the mixer consists of attenuators and an amplifier to obtain the appropriate level of LO signal for the mixer. The first IF stage connected to the IF output of the mixer is identical to the same stage in the Test Channel Receiver Front-end Down Converter. The output of the IF stage is connected to the IF-REF port which is then connected to the IF Control Chassis via a coaxial cable.

B.1.3 Front-end Receiver Control Chassis

The Front-end Receiver Control Chassis, which is diagrammed in Figure 6, has three functions. First, it conditions the LO signal as it passes from the Test Channel Receiver Front-end Down Converter to the Reference Channel Receiver Front-end Down Converter. It also amplifies the 10-MHz Time Base signal from the HP 8340B on the AUTplatform and distributes it to the network analyzers and the other HP 8340B which is located on the probe carriage. Finally, it controls the DC power to both the Test and Reference Channel Receiver Front-end Down Converters and controls the RF switches in the Test Channel Receiver Front-end Down Converter.

B.1.4 IF Control Chassis

The IF Control Chassis conditions the IF signals from the down converters and sends the resulting signals to the network analyzers. A functional diagram of this chassis is given in Figure 7.

B.2 Configurations

The near-field range can be operated in one of two configurations: 1) the AUT can be operated in the transmit mode or 2) the AUT can be operated in the receive mode. The components of the near-field range receiver must be interconnected differently for each of these configurations and the locations of the two down converters (Test and Reference) must be interchanged. The two configurations for the receiver are illustrated in Figures 8 and 9. In both figures the cabling that must be changed is shown as dotted lines while the cabling that remains unchanged in both configurations is shown as solid lines. These figures also show the location of all receiver components (Control Room, Probe Carriage, or AUT Platform) and identifies the chassis connectors to which the cables interface.

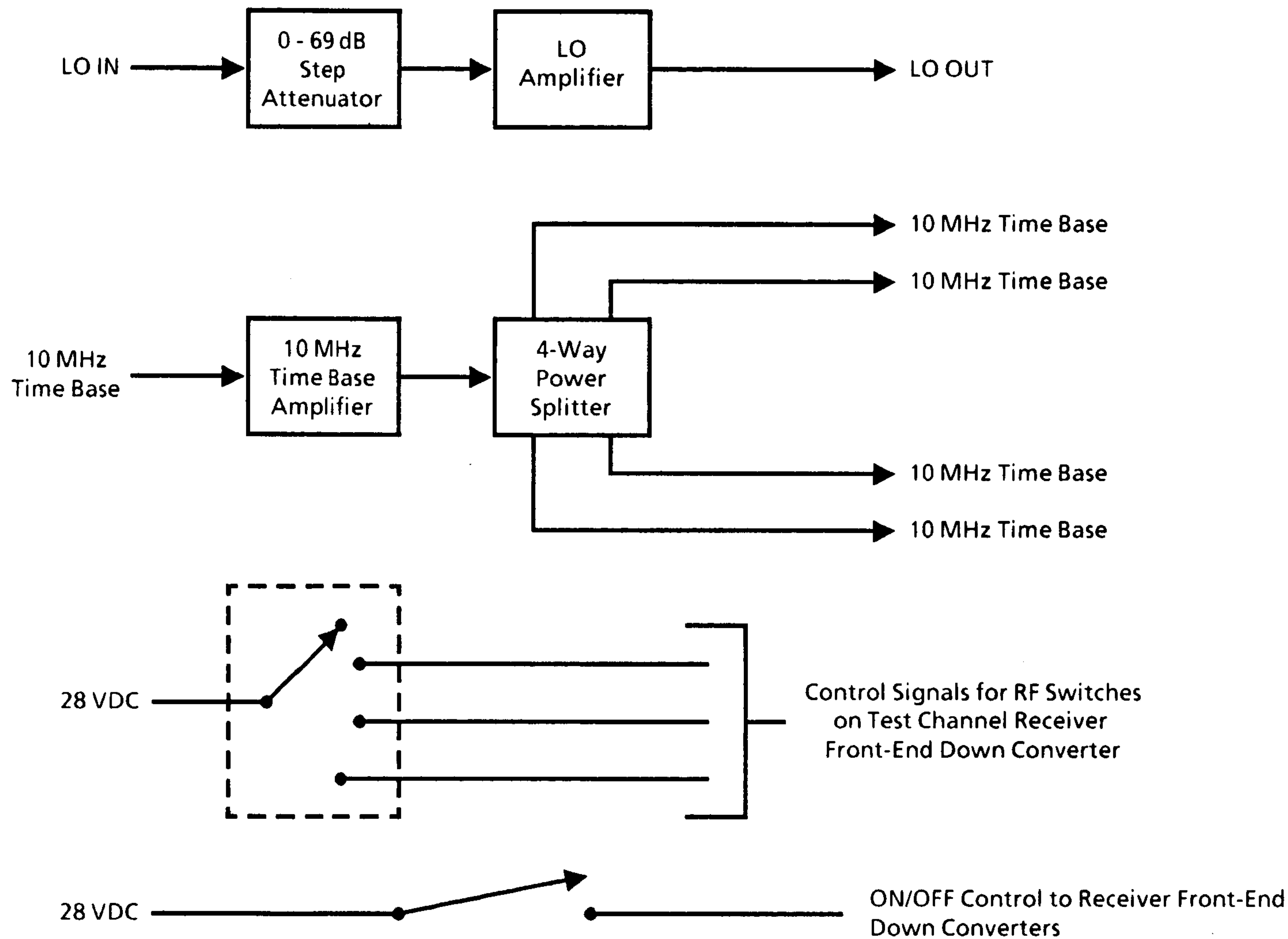


Figure 6. Front-End Receiver Control Chassis

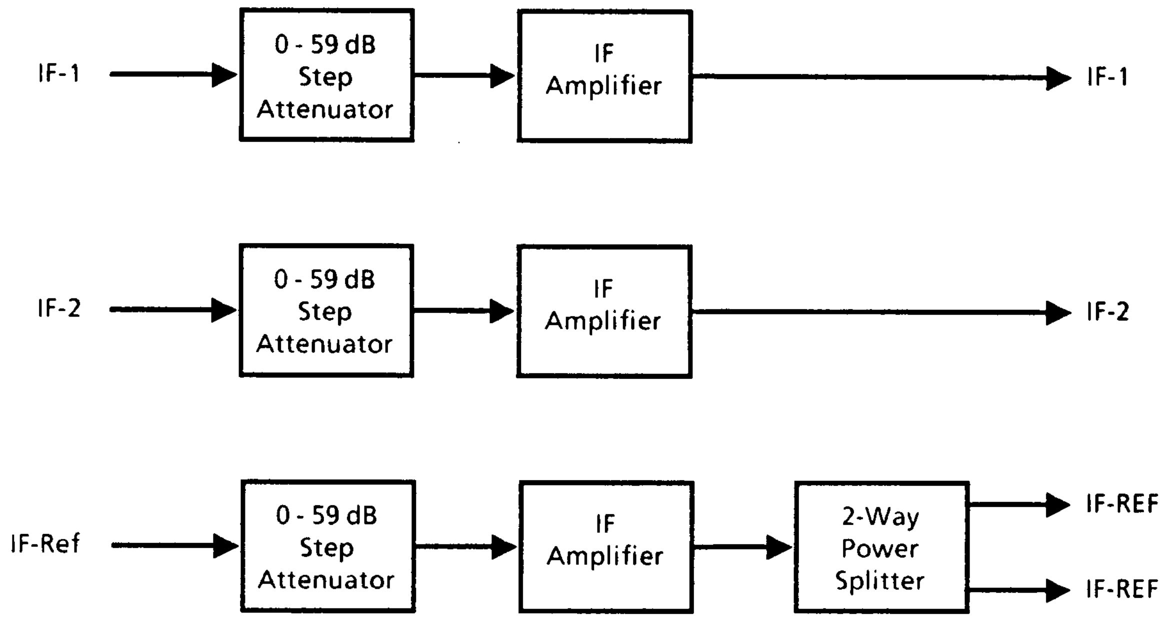


Figure 7. IF Control Chassis

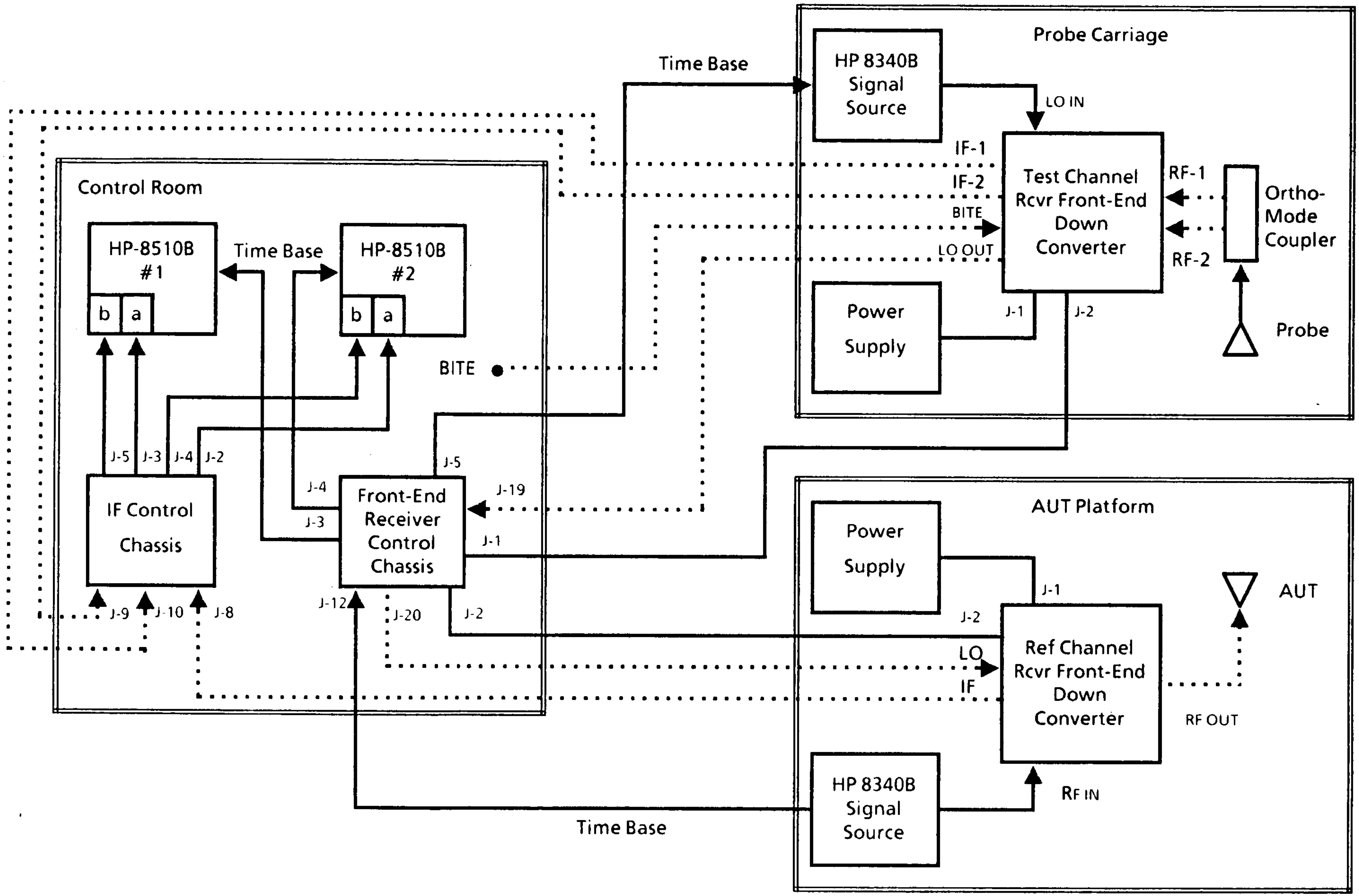


Figure 8. Interconnection Diagram with Antenna-Under-Test in Transmit Mode

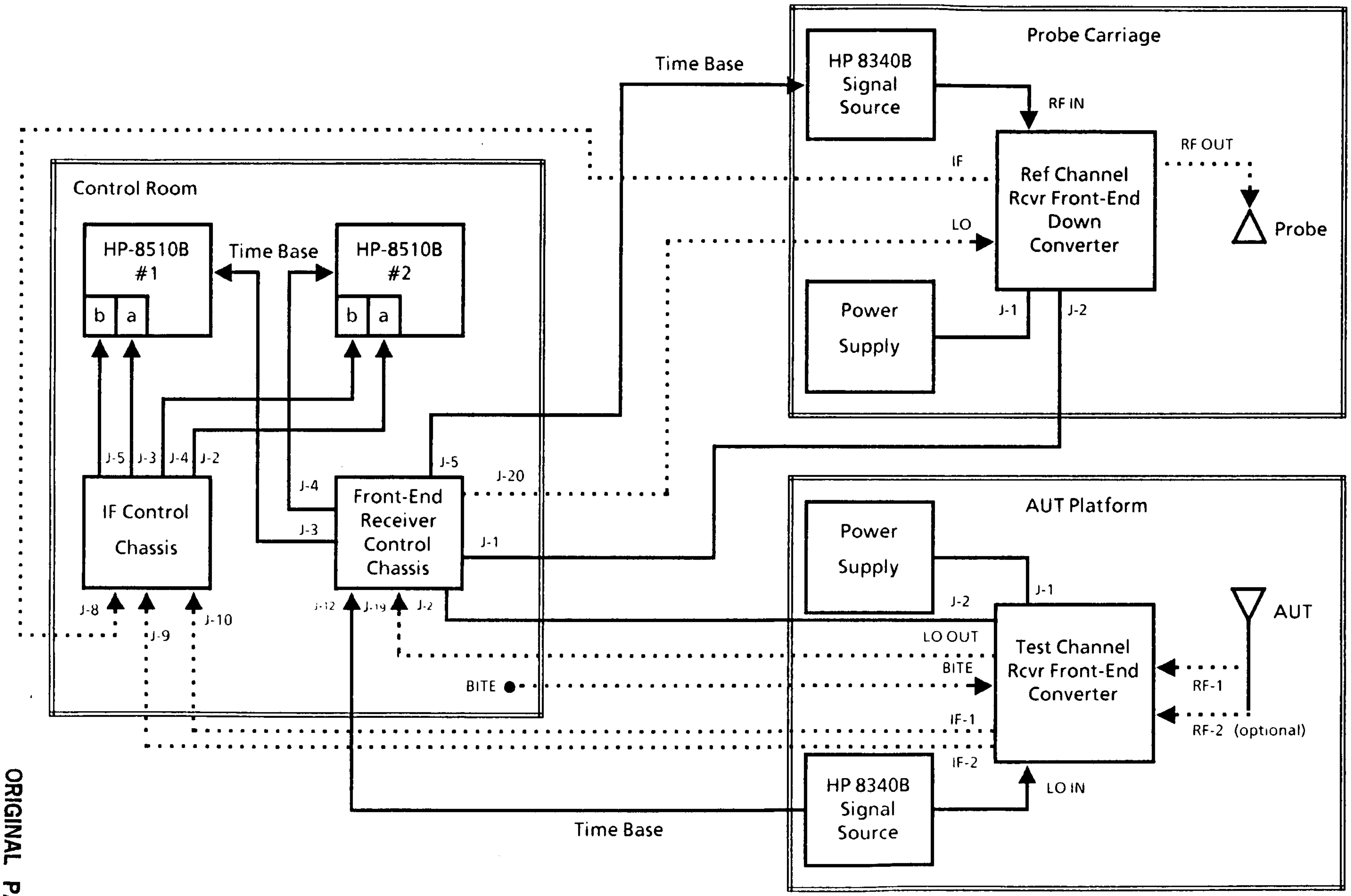


Figure 9. Interconnection Diagram with Antenna-Under-Test in Receiver Mode.

ORIGINAL PAGE IS OF POOR QUALITY

The LO cable which connects the down converter on the Probe Carriage with the Front-end Receiver Control chassis in the Control Room must flex as the Probe Carriage is moved. Phase noise due to this flexure is possible. Various system configurations to minimize the magnitude of this phase error were investigated. Although other configurations may offer lower phase noise, they are more complicated, expensive, and increase the weight on the Probe Carriage. The implemented configuration was selected after evaluating these tradeoffs. An additional feature of this configuration is flexibility. It can be modified if further reduction in phase modulation due to cable flexure is necessary.

Photographs of the major components of the receiver are given in Figures 10 through 15. Photographs of equipment fronts and backs are provided as necessary to show all interface connections and controls. The controls on the front of the Front-end Receiver Control Chassis are shown in Figure 12. The SYSTEM and RCVR POWER switches controls the DC power to this chassis and the two down converters. The RF SWITCH CONTROL is the 4-push button, interlocked switch diagrammed in Figure 6 that controls the two RF switches on the Test Channel Receiver Front-end Down Converter (See Figure 4). The LO SIGNAL ATTENUATOR (dB) control sets the level at the LO IN port of the Reference Channel Receiver Front-end Down Converter.

The controls on the front of the IF CONTROL CHASSIS are shown in Figure 14. The SYSTEM POWER switch controls the DC power to this chassis. The knobs labelled SIG CH1, SIG CH2, and REF CH control the attenuators that set the two Test Channel IF input levels and the Reference Channel IF input level, respectively, to the network analyzers.

B.3 Performance Characteristics

The measured sensitivity, noise figure, and dynamic range of the receiver system as a function of frequency are presented in Table I.

B.4 Maintenance Information

Wiring diagrams and component layout photographs for each chassis in the receiver system are provided in Appendix A. Specifications for major components on each chassis are also presented in that appendix.

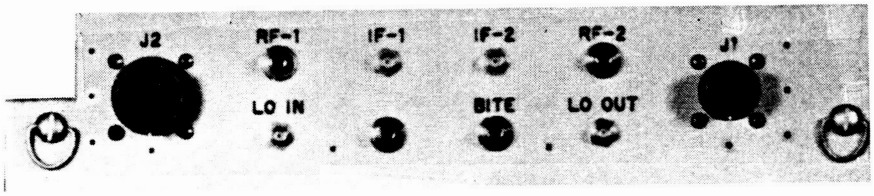


Figure 10. Front View of Test Channel Receiver Front-End Down Connector.

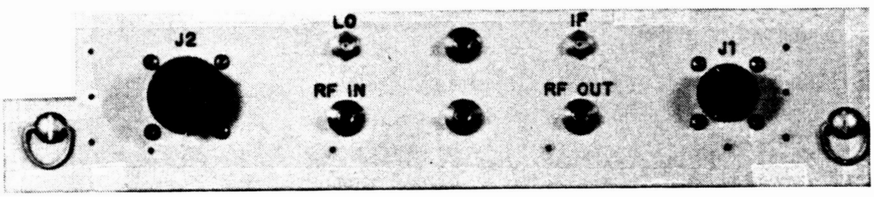


Figure 11. Front View of Reference Channel Receiver Front-End Down Converter.

ORIGINAL PAGE IS
OF POOR QUALITY

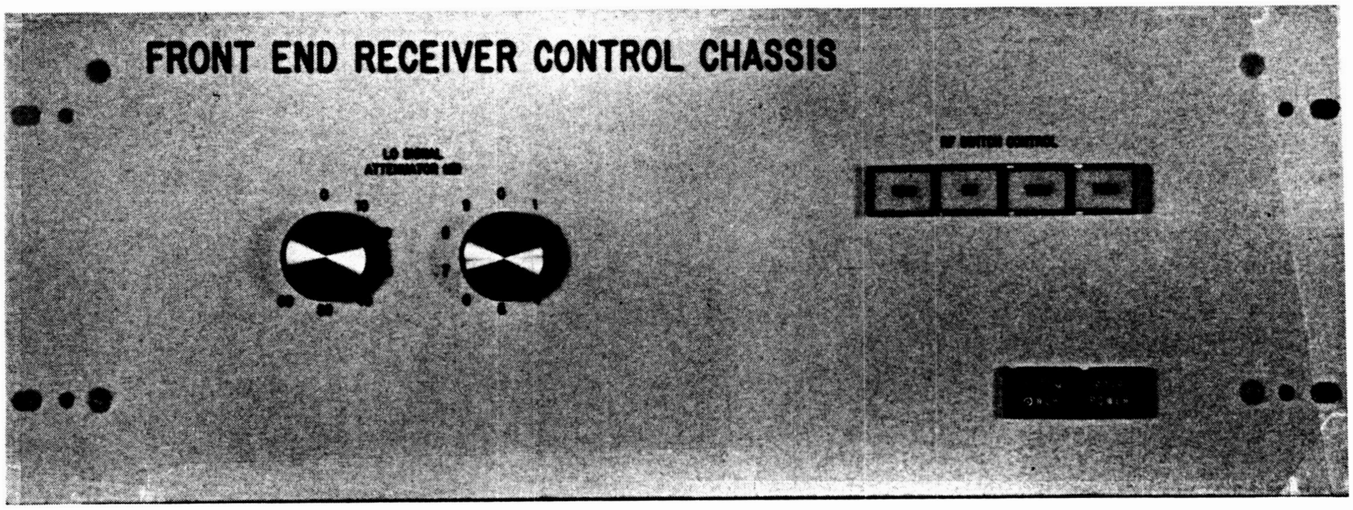


Figure 12. Front View of Front-End Receiver Control Chassis.

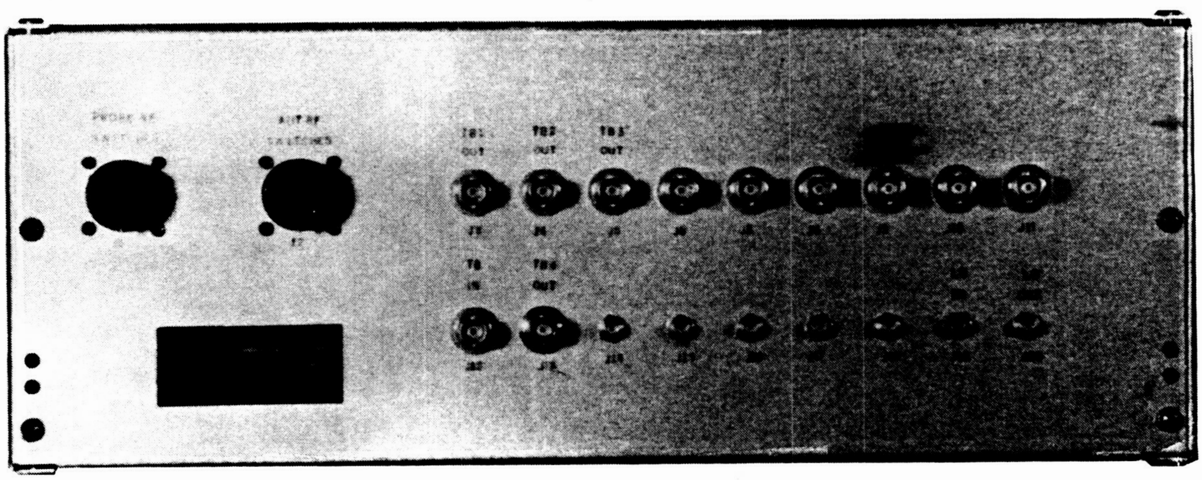


Figure 13. Rear View of Front-End Receiver Control Chassis.

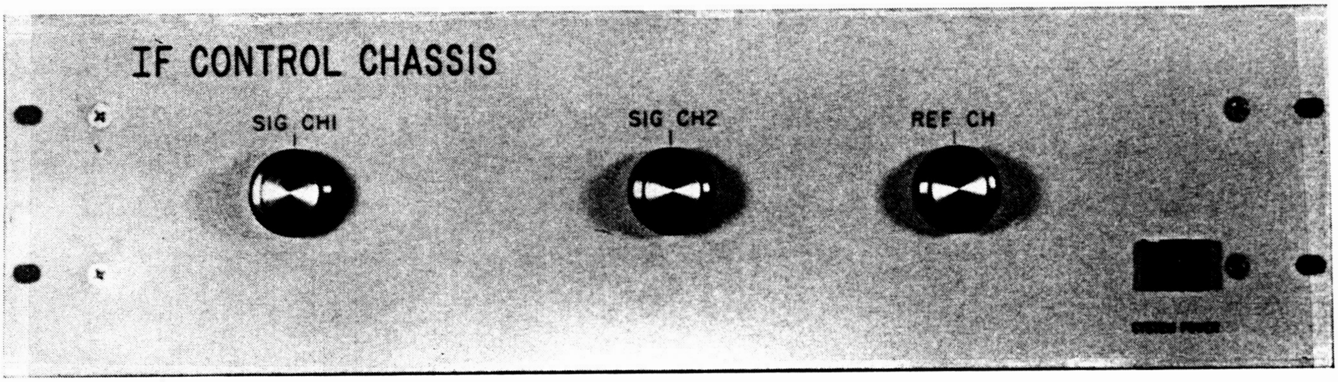


Figure 14. Front View of IF Control Chassis.

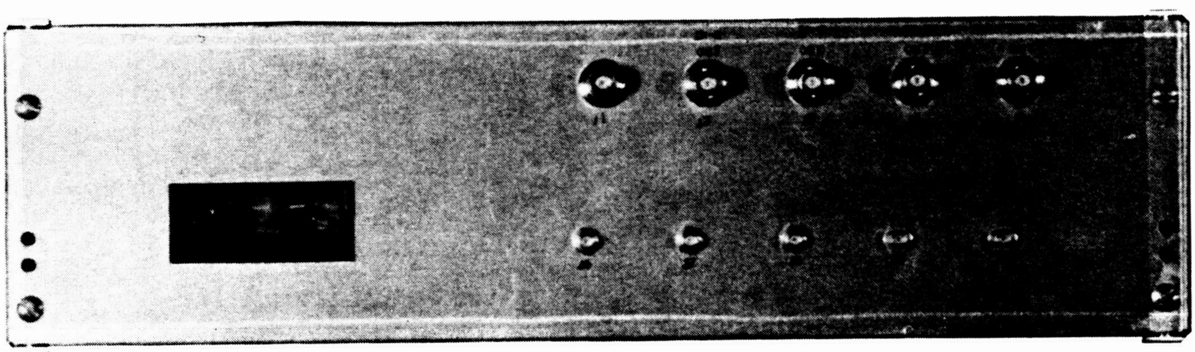


Figure 15. Rear View of IF Control Chassis.

ORIGINAL PAGE IS
OF POOR QUALITY

Table 1

NASA NEAR-FIELD RECEIVER PERFORMANCE

Parameter	Frequency (GHz)								
	2.0	5.0	8.0	12.0	16.0	18.0	20.0	23.0	26.0
Harmonic Mixing Number	1st	1st	3rd	3rd	3rd	3rd	5th	5th	5th
Sensitivity* (dBm)	-112	-113	-103	-104	-103	-100	-93	-93	-93
Measured Noise Figure (dB)	22	21	31	30	31	34	41	41	41
Calculated Noise Figure (dB)	22	22	32	32	32	32	39	39	39
Dynamic Range** (dB)	72	73	63	64	63	60	53	53	53
Channel-to-Channel Isolation (dB)	>50	>50	>50	>50	>50	>50	>40	>40	>40

* S/N = 1 with 10 KHz Video Bandwidth and no averaging.

** Based on a minimum Signal -to-Noise ratio of 30 dB and a maximum HP 8510B input level of -10 dBm.

C. Laser Metrology Subsystem

The Laser Metrology Subsystem is used to precisely monitor the position of the probe cart within the measurement plane. This task can be divided into two separate functions. The first is to aid positioning of the probe at each sample point in the data collection grid. The position error signal derived from the laser electronics associated with this first function are used to drive the servo amplifiers responsible for motion of the probe cart and translation beam. The second function is to determine deviations from the ideal sampling positions of the probe caused by unavoidable rotations and lateral deflections of the moving bodies (probe cart and translation beam). Since the algorithm used to process near field data assumes regularly spaced points on a perfectly flat measurement plane, unwanted rotations and lateral deflections of the probe degrade the accuracy of the patterns derived from the near field data. As a future task, the position information derived from this second function can be used to mathematically correct for known probe position errors. Such a capability will become important when operation of the range is extended above 26.5 GHz. This topic is discussed further in Section V.

C.1 Theory of Operation

The laser metrology components used on the NASA-JSC antenna range comprise a hybrid of two Hewlett-Packard laser measurement systems. The basic system of a laser head, optical detector/receiver, measurement electronics, and linear optics is a version of the HP 5501 system. The angular and straightness measurement optics are added from the HP 5528 system. A summary of the measurement optics is presented next. A more complete explanation can be found in references 3 and 4.

The HP 5501 laser head uses a principle called Zeeman splitting to produce a beam with two slightly different frequency components, f1 and f2. Polarizing and collimating optics adjust the two components into orthogonal linear polarizations, one vertical and one horizontal. A portion of this combined beam is sampled and directed to the reference receiver internal to the laser head. Here the two components are combined to produce interference fringing at 1.8 MHz, which is supplied to the measurement electronics as the reference signal. The measurement signal is produced when the dual component beam emitted from the head interacts with various measurement optics.

Figure 16 illustrates the operation of the linear measurement optics. A Doppler shift, caused by relative motion between the test retroreflector and the linear interferometer, produces a return beam at a frequency f2 + Δf2. When recombined with the f1 component of the beam at the optical receiver, an interference fringe is produced at a slightly different frequency from the reference signal. By monitoring the difference in fringe counts, the measurement electronics (Counter or Fast Pulse Converter cards) produces a measure of the motion of the test retroreflector with respect to the linear interferometer.

Angular motion and lateral deflections can be measured by using different optical arrangements. Figure 17 illustrates the angular optics. In this case, a Doppler shift along the axis of the laser beam will affect both frequency components equally, and thus produce no change in the interference fringing. Rotation of the angular reflector, however, will produce a positive increment in one component and a negative increment in the other, thus resulting in a measurable change in the interference fringing. To measure lateral deflection (straightness), the optics illustrated in Figure 18 are used. The interferometer consists of a Wollaston prism, which causes the beam components to diverge at a specific angle. The straightness reflector has two mirrors, each perpendicular to one of the components. When either the interferometer or the reflector moves perpendicular to the mirror axis, the path length of one beam component lengthens while the path length of the other component shortens. This produces the required change in the frequency difference between the two components.

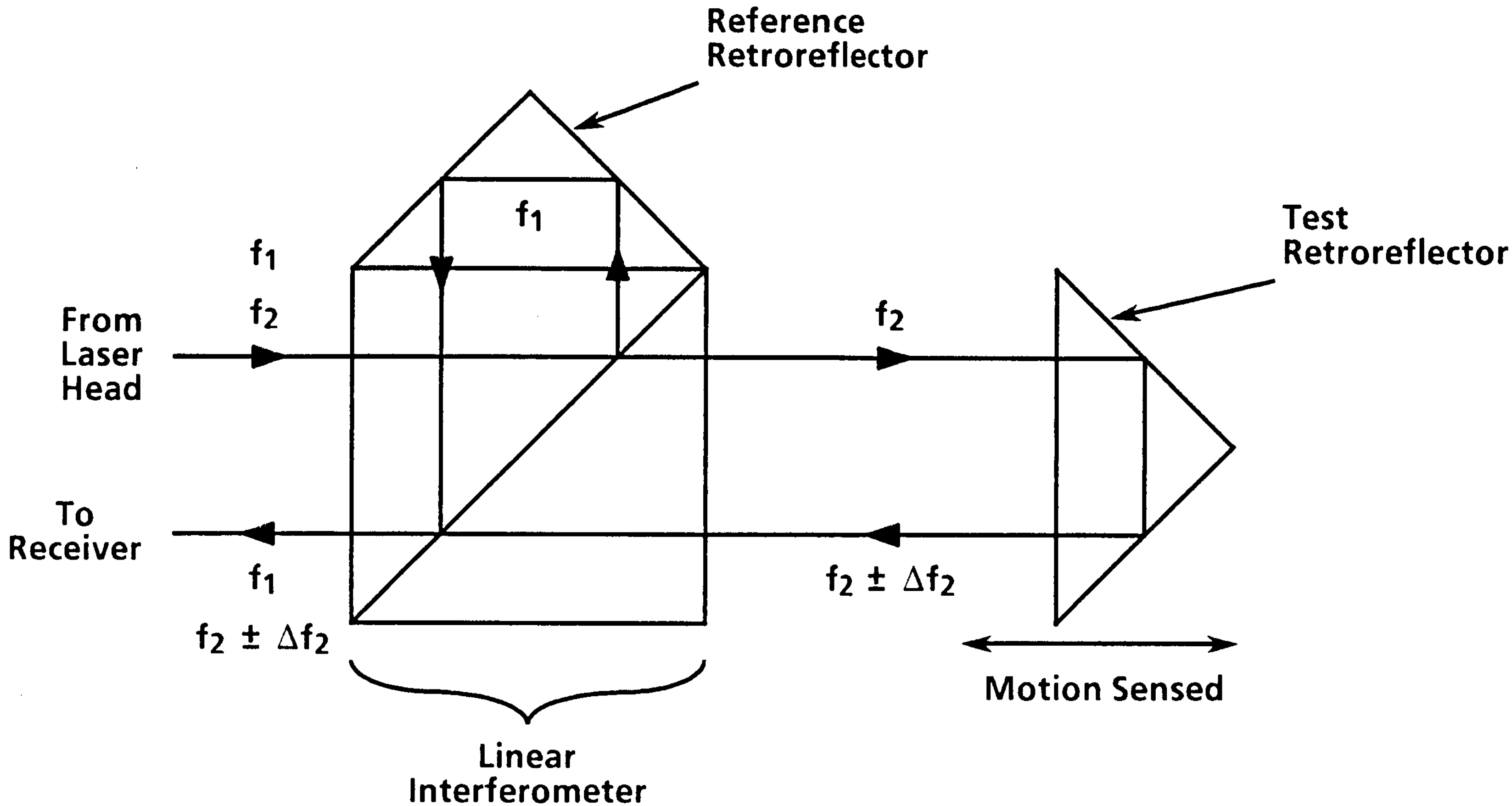


Figure 16. Linear Optics.

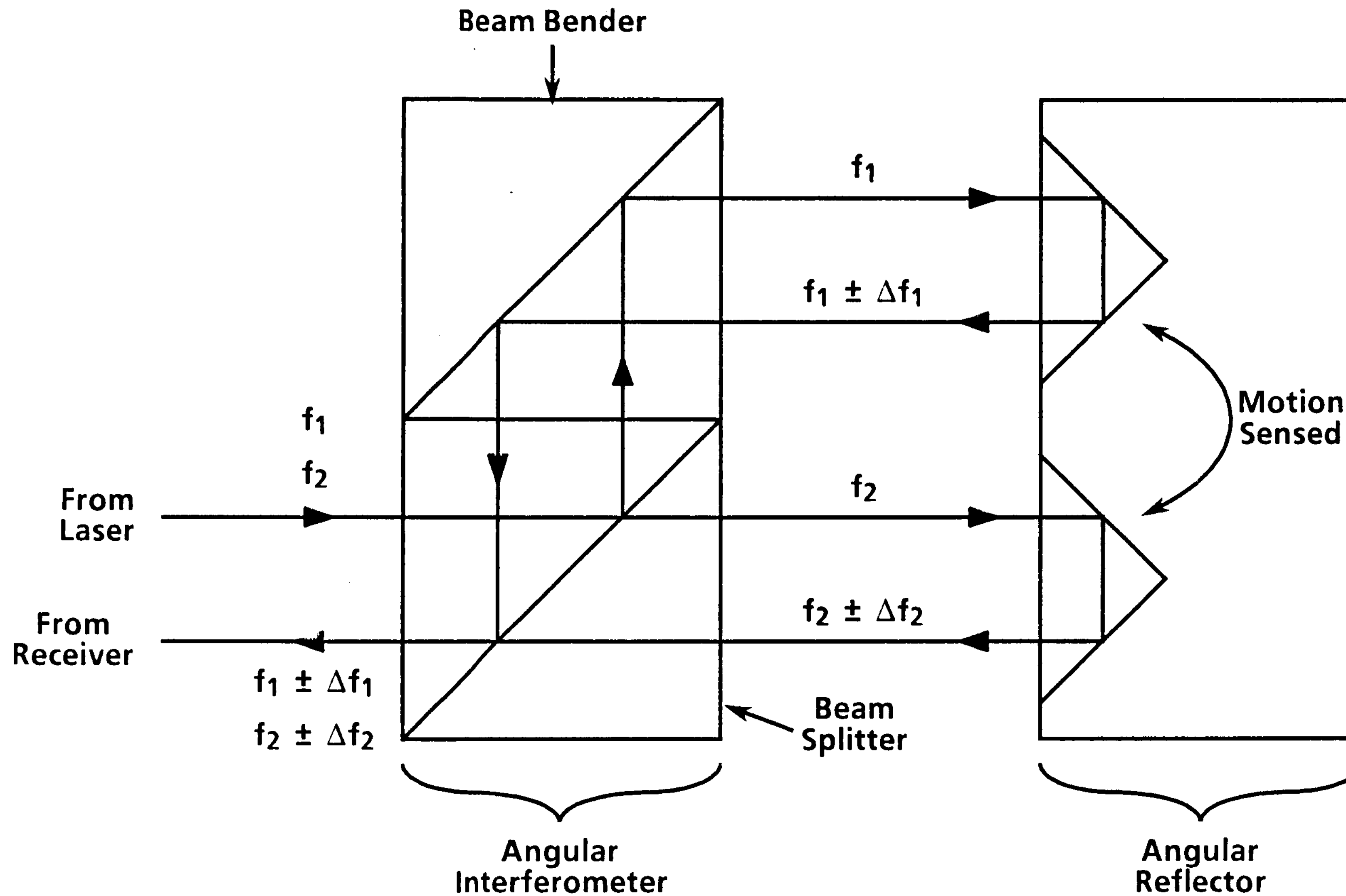


Figure 17. Angular Optics.

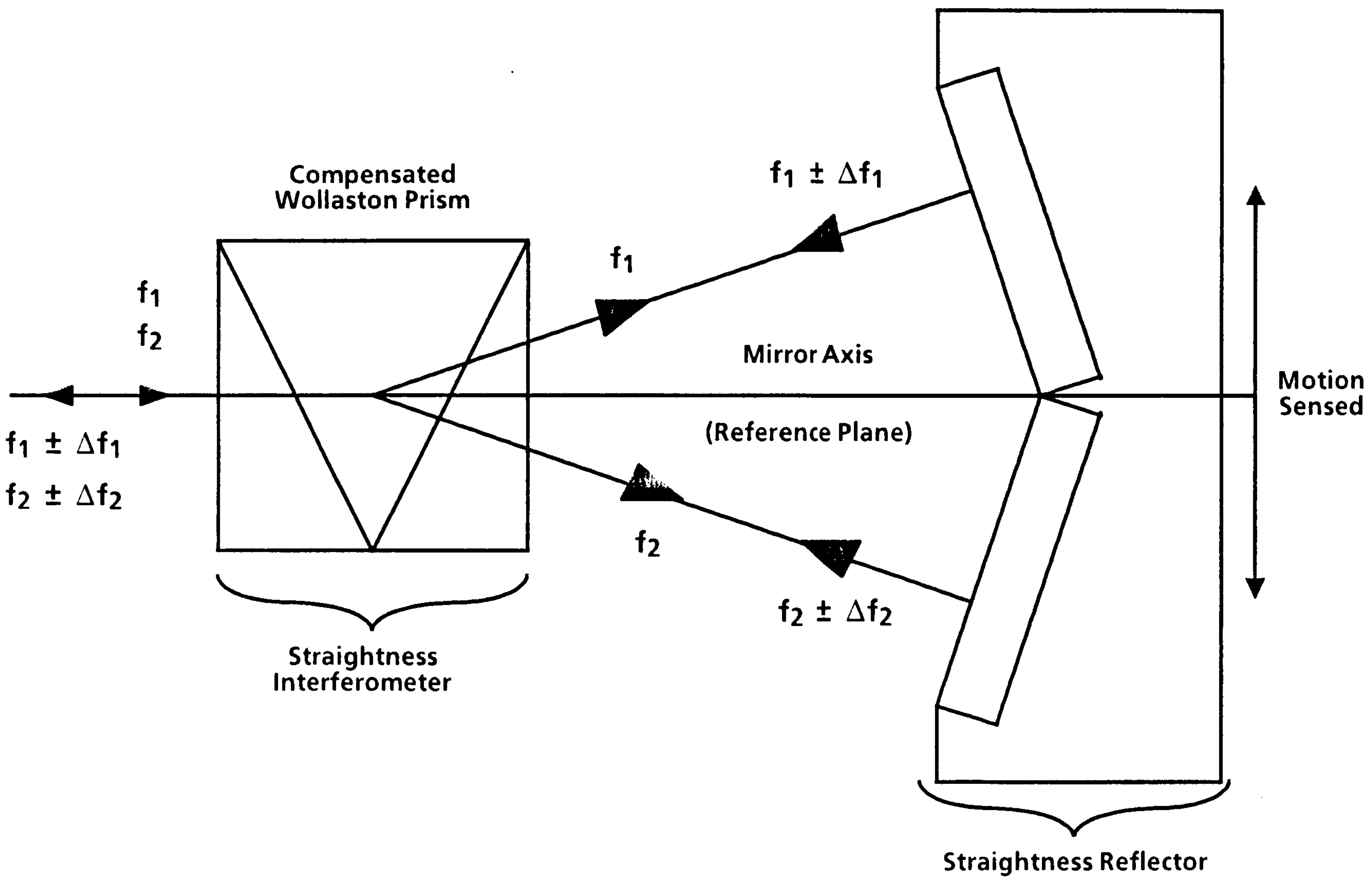


Figure 18. Straightness Optics.

C.2 Laser Optics on the NASA-JSC Range

These three types of measurement optics (linear, angular, and straightness), are used in combination to fulfill the measurement requirements of the system. All optics are mounted on laser platforms attached to the structure as shown in Figure 19. These optical benches occupy two elevation levels. On the upper level, fixed benches are located at the northwest and southwest corners of the structure. Three moving benches are found on this level, one at each end of the translation beam, and one which moves with the probe cart. On the lower level are one fixed bench located directly beneath the bench at the northwest corner, and one bench directly beneath the moving bench at the north end of the translation beam. There is also a spare bench located directly beneath the bench at the south end of the beam, but it is currently unused. In Figures 20 a - 20d, each group of measurement optics is identified on the benches it occupies.

The first measurement function listed above, that of positioning the probe at the points on the sample grid in the measurement plane, is accomplished using the linear optics groups illustrated in Figure 20a. Four laser heads (C, D, F, and I) supply the beams for four measurement signals. Receivers Xt and Xt' are used to monitor motion parallel to the X-axis of the south and north ends of the translation beams, respectively. Receiver Yc is used to monitor motion of the probe cart parallel to the Y-axis from north to south, and receiver Yc' is used for motion south to north. The motion of each end of the translation beam is monitored independently. The two signals used to monitor probe cart motion complement each other. However, a maximum speed for the probe cart of thirty inches/second was desired, which exceeds the maximum speed specified for the linear optics. Hewlett Packard representatives explained that this limit only applies to motion of the interferometer and retroreflector toward each other. Each set of probe cart optics can be used to monitor motion up to thirty inches/second in a direction which increases the separation, so probe cart motion may be characterized and controlled using two sets, each active for one direction only.

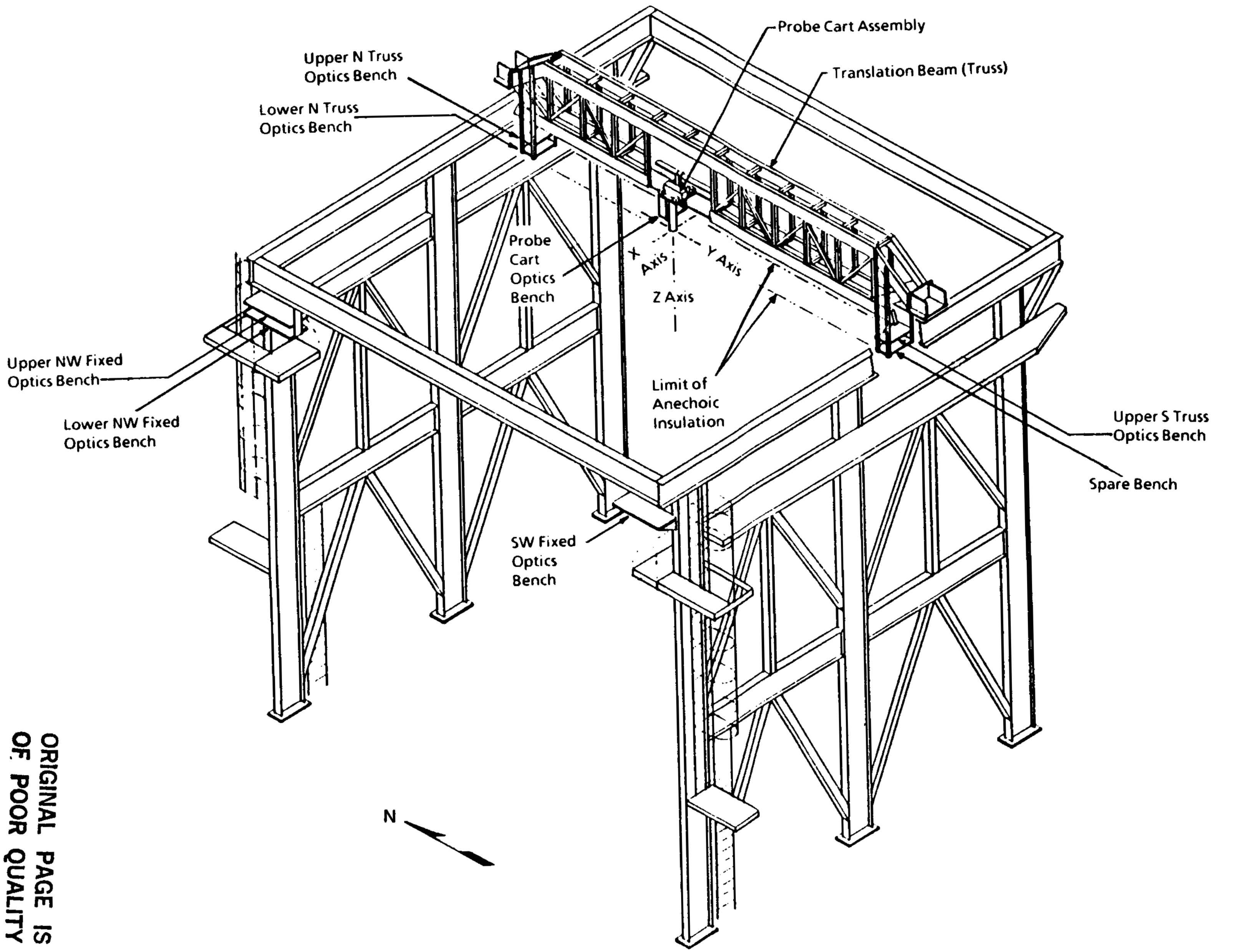


Figure 19. View of stucture showing position of benches used to support laser optics.

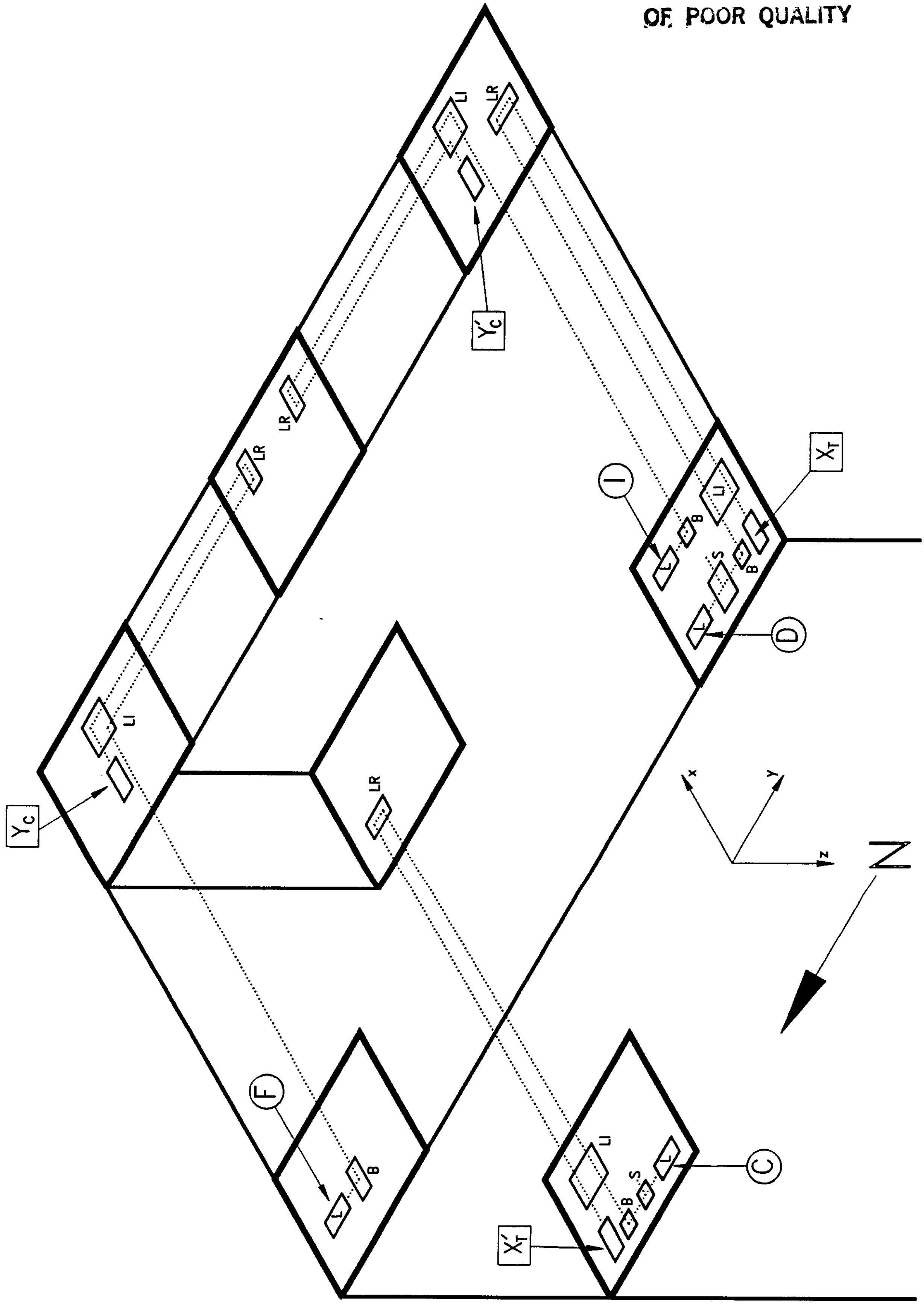


Figure 20a. Linear Position Optics.

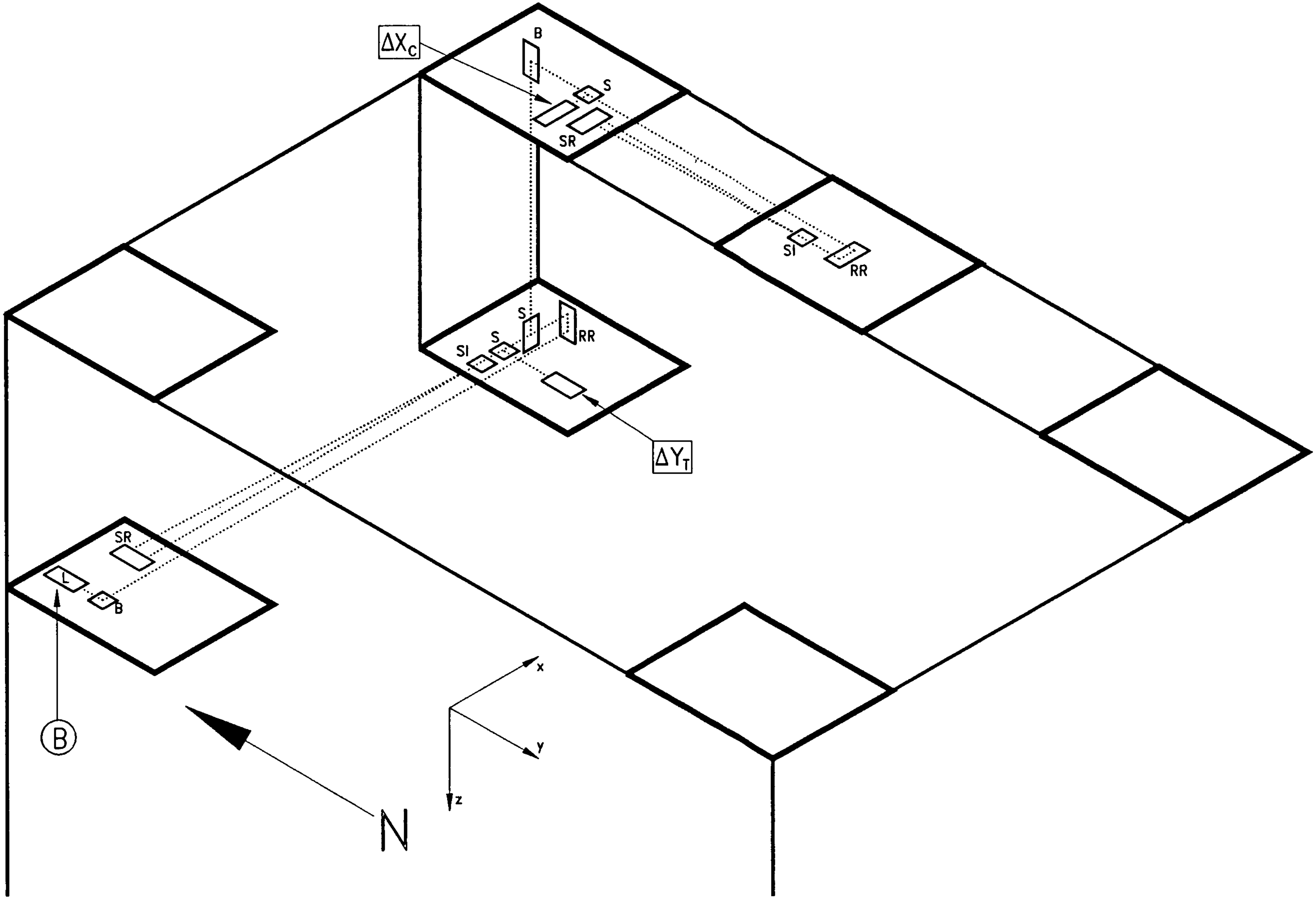


Figure 20b. Straightness Deflection Optics.

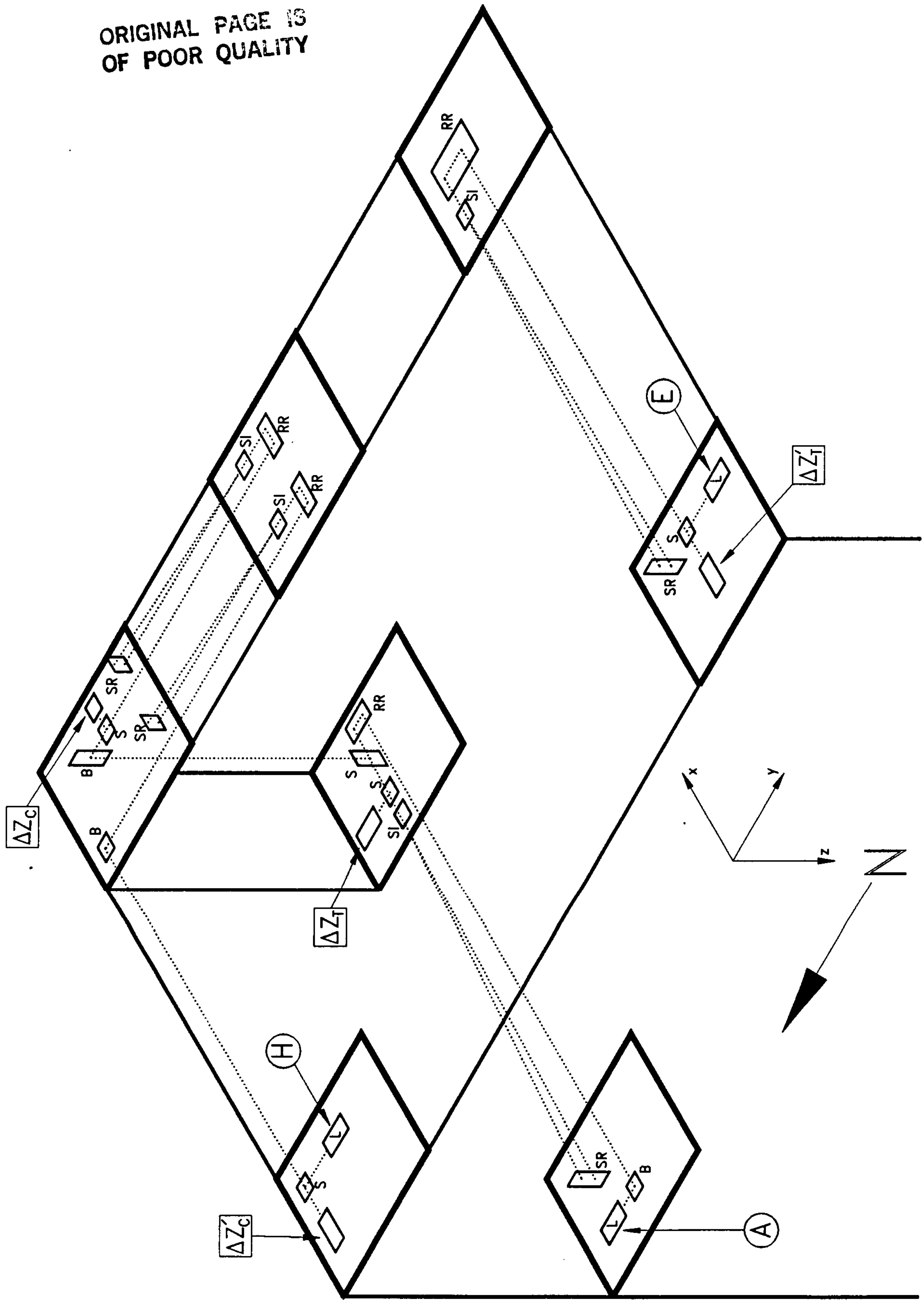


Figure 20c. Straightness Rotation Optics.

Note: Θ_{ZT} is calculated from the quantities X_T and X'_T.

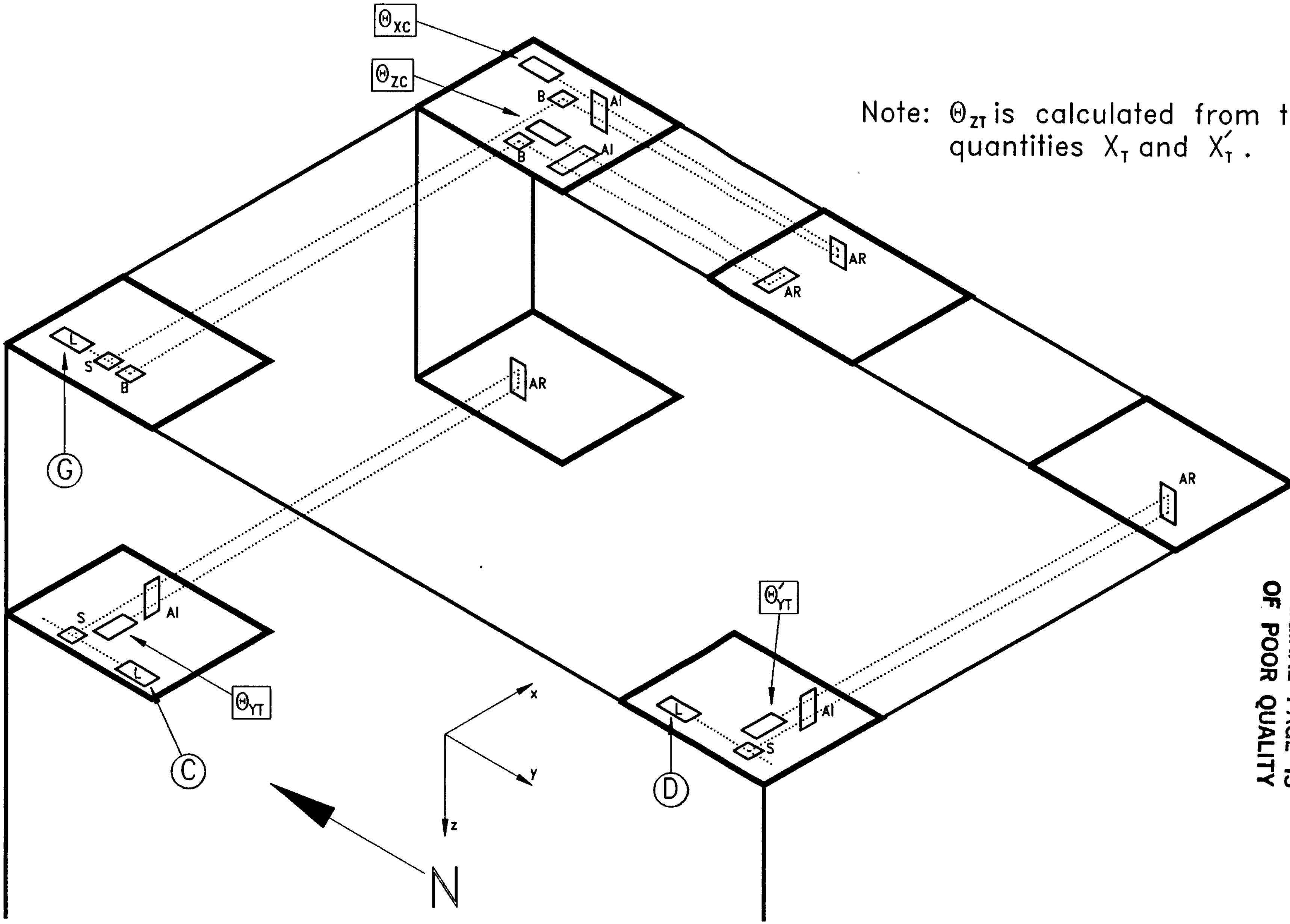


Figure 20d. Angular Rotation Optics.

C.3 Probe Position Characterization

The remaining optics illustrated in Figure 20 can be used to fulfill the second function previously identified, namely, characterizing the actual position of the probe aperture as it deviates from the ideal sampling positions in the measurement plane. This function has not yet been implemented, although it will be necessary when operation of the range is extended to 60 GHz. To do this, the probe cart and translation beams are considered rigid bodies capable of motion with six degrees of freedom (three linear and three rotational). The linear motions monitored by the optics in Figure 20a are the desired motions. The optics in Figure 20b are used to monitor undesired deflections in the horizontal measurement plane. To this end, the beam from laser head B is split and one half used to detect Y-axis deflections of the truss as it traverses the structure (via receiver ΔYt). The other half is used to detect X-axis deflections of the cart as it traverses the truss (via receiver ΔXc).

Motion in the third linear degree of freedom is obtained from the optics configuration shown in Figure 20c. Laser head A serves double duty, supplying the energy to receivers ΔZt and ΔZc. Receiver ΔZt provides information about the vertical deflection of the north end of the truss, while laser head E and receiver ΔZt' provide similar information about the south end. Together, information from the two ends yields both the vertical deflection of the truss and the rotation of the truss about the X-axis. Likewise, receiver ΔZc monitors the vertical deflection of the east side of the probe cart while laser head H and receiver ΔZc' yield the vertical deflection of the west side of the cart. Thus both the vertical motion of the cart and its rotation about the Y-axis can be determined.

In Figure 20d the use of the angular optics is shown. Laser head G supplies energy to receiver θxc via a vertical angular reflector, thus yielding the rotation of the cart around the X-axis. Laser G is also used with receiver θzc and a horizontal angular reflector to monitor rotation of the cart about the Z-axis. Rotation of the truss about the Y-axis is measured independently at each end. Laser head C and receiver θyt monitor the north end while laser head D and receiver θyt' are used at the south end. Together, information from the two receivers can be used to derive both the rotation and the twist in the truss. The last degree of freedom, rotation of the truss about the Z-axis, is obtained from receivers Xt and Xt', shown in Figure 20a.

Figure 21 shows how the signals from the laser optics are processed in the laser electronics. The actual counting of pulses in the reference and measurement signals is done in either a HP 10760 Counter card or a HP 10764 Fast Pulse Converter (FPC) card. The output of the Counter card is position information which is supplied to the system controller via the HP 10746 Binary Interface card. The output of the FPC card is supplied to a HP 10762 Comparator card. The Comparator card contains a destination register which can be loaded with a desired position from the system controller (again via the Binary interface card). The Comparator card compares position information input from the FPC card with the contents of its destination register and supplies an 18-bit digital error signal to an edge connector. The first function of the laser metrology system, that of aiding in the positioning of the probe cart and truss, is accomplished with the error signals derived from each of the receivers Xt, Xt', Yc, and Yc'. The error signals are each supplied to a Digital-to-Analog Converter driving a Controlled Motion Incorporated Servo Amplifier. One servo amp controls each drive motor in the system. Two motors drive the translation beam (one at each end) and one drives the probe cart. A multiplexer is used to select the appropriate error signal from receivers Yc and Yc' depending on the desired direction of travel of the probe.

The second function of the laser metrology system, that of characterizing probe positioning errors, should be implemented as a future task. For example, the characterization could be accomplished prior to performing an actual measurement. First the probe would be scanned along the truss approximately twenty times in each direction while monitoring the quantities associated with probe motion. An average value for each quantity as a function of the probe position along the truss would then be stored in a file. Next, the truss would be scanned approximately twenty times in each direction across the structure while monitoring the quantities associated with truss motion. An average value for each quantity as a function of the truss X-position would also be stored in the file. With a pre-determined knowledge of the (fixed) position of the probe aperture with respect to the optics mounted on the probe cart, in conjunction with the measured position errors of the truss and cart, the actual positioning errors of the probe aperture can be determined at each sample point and an appropriate correction applied to the collected data.

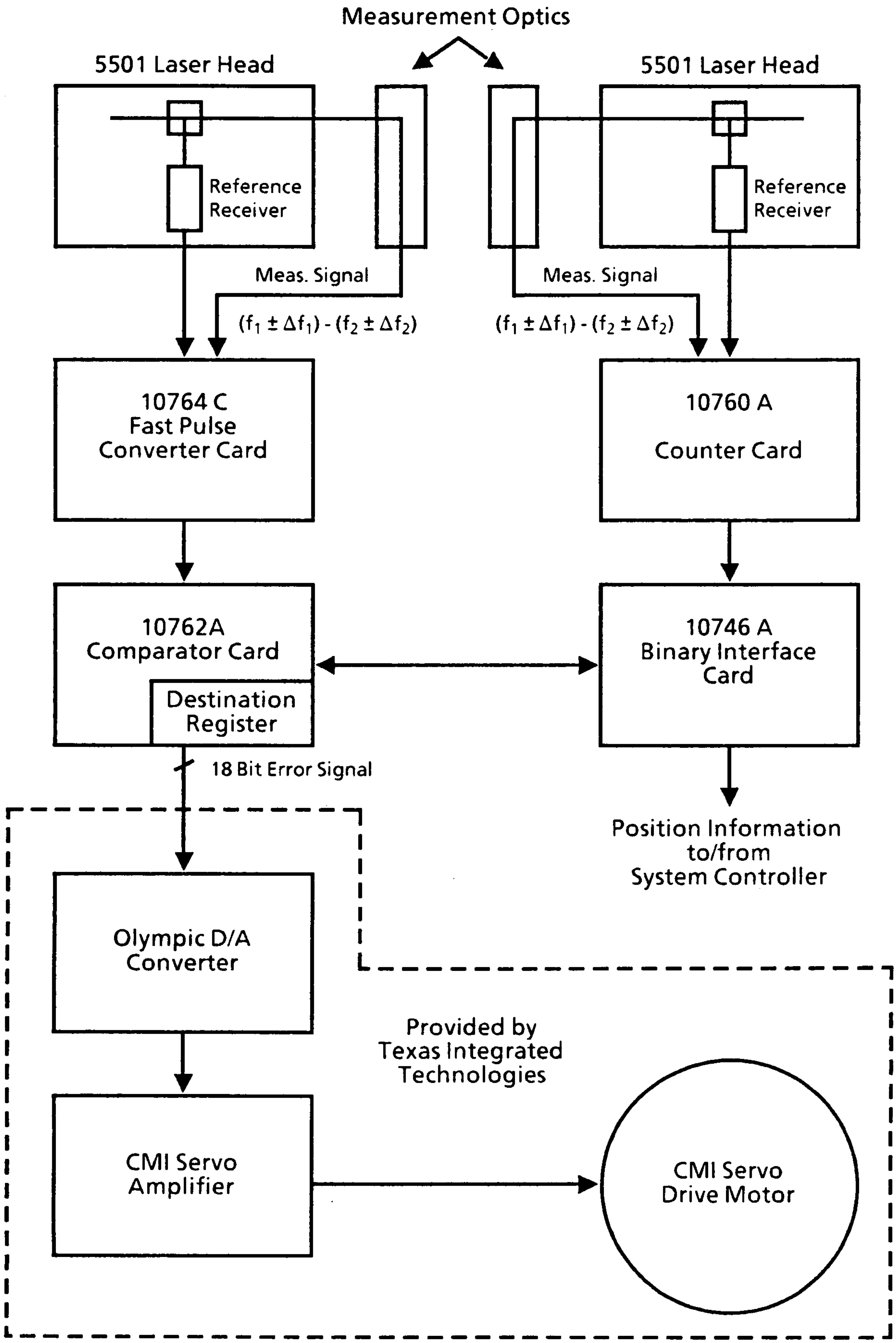


Figure 21. Laser Metrology System Block Diagram

D. Trigger control electronics

As part of its function to assist positioning of the probe, the laser metrology subsystem also provides a trigger pulse to the HP 8510's at each point in the measurement grid to initiate sampling of data. The Trigger Control Electronics select and condition the trigger pulses according to commands from the system controller. The source of the trigger pulses is an extra pair of Comparator cards in the laser electronics. Each comparator card controlling a direction of motion of the probe cart has a sister card to generate the trigger pulses. While the first comparator card contains a destination at the end of the row to be scanned, its sister card contains the destination of the next point to be sampled. As each sample point is reached, the comparator card generates a trigger pulse for the HP 8510's, the trigger control electronics set a status line to indicate the destination has been reached, and the system controller then loads the comparator card with the destination of the next sample point. The trigger control electronics also act as a multiplexer to select the trigger pulse output of the appropriate comparator card as determined by the direction of probe travel. A schematic can be found in Appendix B.

SECTION III

Near-Field Range Measurement Procedure

A. Introduction

This section is designed to assist the user of the NASA-JSC near field range by presenting a typical procedure for operation of the range. Although a particular application may require some variations, the basic steps outlined below are provided as a general guide.

B. Measurement Procedure

1. AUT Mounting

The AUT platform is supported by three hydraulic jacks which are under manual control of the operator via a panel in the control room. The platform is monitored by two inclinometers mounted perpendicular to each other on the platform with displays in the control room. Using the inclinometers and jacks, the operator should level the table after the AUT is mounted and in position.

2. Probe Selection

Probe selection will vary with the measurement application. This topic is explored in greater detail in a GTRI report [5].

3. Select Operating Mode

The near-field range can be operated with the AUT in either the transmit mode or the receive mode. The two configurations require different locations and interconnections of the receiver components. These are discussed in detail in Section II.B.2.

4. Initial Receiver System Set-Up

Turn system power on (See Section II.B.2) and allow several hours warmup time. Set attenuators and source power levels as required by operating frequency. A lookup table specifying these levels will be provided in the software at the time of installation.

5. Positioning the AUT Platform

The optimum separation distance between the scan plane of the probe aperture and the antenna under test is determined by the need to minimize multiple reflections. By moving the probe to a point of high signal level and monitoring the probe response as a function of separation distance, the operator can observe the minimum separation distance at which the multipath ripple becomes negligible. This separation distance should be confirmed at two or three distinct points in the measurement scan plane. A typical separation distance is 7-10 wavelengths.

6. Receiver Dynamic Range Alignment

Scan measurement plane to locate maximum signal level. Position probe at this point and adjust IF attenuators for signal level of -10 dBm at input to HP 8510B's. (See Section II.B.2) Note: In order to preserve relative signal levels between the two channels, both attenuators should be set to the same value.

7. Operation Under Software Control

Determine the desired scan plane dimensions and sample point spacing based on the size of the test antenna aperture and frequency of operation. Selection of a suitable aperture scan plane is described in Reference 2. Run the data collection program, program XYZ, and select the operations desired from the main menu. The program is described in greater detail in Section IV.

SECTION IV

Software Documentation

A. Introduction

This section describes both the control and data reduction software for the NASA-JSC near-field range. The control software (Program XYZ) is a menu driven algorithm. After initializing the equipment, the main menu is displayed and the user selects the required program functions. The control algorithm features are described in the next section. A listing of program XYZ is provided in Appendix C.

The data reduction software (Program NFFT) provides many options for data reduction and output format. The user prompts which describe these options are explained in detail in Section IV.C. Finally, Section IV.D provides information necessary for compiling and loading these programs. A listing of program NFFT is provided in Appendix D.

B. Program XYZ

Listed below with explanatory notes are the options available to the user from the main menu:

1. Set Source (SS)

This selection allows the user to set the RF source power level (in dBm) and frequency (in GHz). Also prompts the user for the number of polarizations to be collected. It is the user's responsibility to set the power level and frequency for the RF source and receiver, using this option, before collecting any data.

2. Initialize Scan Parameters (IN)

This selection sequentially prompts the user for the "scan parameters". The scan parameters define the scan plane over which data will be collected and also include other parameters to be stored in the data file's header record. The scan parameters must be defined before collecting any data.

3. **List and Change Scan Parameters (LC)**

This selection displays current scan parameters and allows individual parameters to be modified. This differs from "IN", where the user must step through the whole list.

4. **Examine a File (EF)**

This selection allows the user to read a column of data into the data buffer from an existing data file. The data can then be listed or plotted on the CRT. NOTE: When "EF" is used, the scan parameters are modified to match those of the file being read in. This can be used as a quick way to modify the scan parameters if you already have a data file with the parameters you desire.

5. **Column Read (CR)**

This selection collects a column of data and stores it in the buffer. If the number of polarizations specified in "SS" is equal to 2, then both polarizations will be collected. The user specifies which data column, with respect to the scan parameters, is to be read in. Once the data column has been collected, it can be listed or plotted by using the "CL" and "CP" options, respectively.

6. **Column List (CL)**

This selection lists the column of data currently stored in the buffer. The data could have been read into the buffer by using the "EF", "CR", "AR" or "CD" options.

7. **Column Plot (CP)**

This selection plots the column of data currently stored in the buffer.

8. **Move Probe (MO)**

This selection allows the user to specify an (X,Y) destination in inches, and moves the truss and cart to the destination position.

9. **Add or Replace Columns (AR)**

This selection allows the user to collect a subset (one or more columns) of a previously collected data set. The file name given here must correspond to an already existing file. Columns of data collected with this command will overwrite the corresponding columns in the existing data file. The scan parameters do not need to be set for this option since they will be modified automatically to match those of the existing data set. If the number of polarizations specified in "SS" is equal to 2, then both polarizations will be collected.

10. **Collect Data (CD)**

This selection allows the user to collect a data set using the current scan parameters. The filename specified here must not already exist (overwrite protection). If the number of polarizations specified in "SS" is equal to 2, then both polarizations will be collected.

C. Program NFFT

This program performs the Fourier transform to obtain the far-field antenna pattern from the near-field measurement. User prompts provided by the program are denoted by boldface type. The exact sequence of prompts will be determined both by the data set(s) being processed and by prior responses. The program begins with the following message:

******** **PROGRAM NFFT** ********

Default responses are shown in parentheses. When a choice is displayed, the first response is the default. Defaults may be selected with the Return key.

1. **How many polarizations will be analyzed? (1 or 2)**

Here, the default choice is one polarization, which can be selected by merely pressing the return key. The program gives the user options to process co-polar and cross-polar data together or either polarization separately. If two polarizations are used, then the parameters entered in response to the following questions apply to both files.

If one polarization was selected, the next prompt is

2. For the aperture data to be analyzed - Enter data file name:

If two polarizations are to be analyzed, the next two prompts will be

2a. For the parallel pole aperture data - Enter data file name:

2b. For the cross pole aperture data - Enter data file name:

The name(s) of data file(s) should be entered in response to each prompt.

3. Enter row numbers for starting, ending X:

4. Enter row numbers for starting, ending Y:

These prompts allow the user to operate on a rectangular subset of the aperture data array. The default is the entire data set. The user should specify the index, or row number, of the desired rows, and not the physical position.

The program will read in data points starting and ending with the specified rows.

5. Enter X thinning increment: (1)

6. Enter Y thinning increment: (1)

Enter the increment, i. The program will read in every ith point in the given dimension, beginning with the starting point from question 3 or 4. Data thus thinned will be processed faster. As the thinning increment is increased, however, more information is lost from the higher spatial frequencies (at the edges of the spectrum).

Ready to normalize the aperture data.

7. Enter the reference amplitude and phase in dB and degrees. (Use the feedthrough values if available. Default is the maximum amplitude.)

Enter the amplitude in dB and the phase in degrees. The user can normalize to reference values or input power levels if the feedthrough values are available. The program can then compute a predicted gain for the antenna under test if an open-ended waveguide probe was used. Also, two separate data collections can be more directly compared after processing. The default values used are the amplitude and phase of the peak point. If both polarizations are being processed, the maximum of the co- polar file is used for both files.

8. Enter normalized wave numbers (Kx,Ky) for the desired K-space translation: (0.,0.)

This can be used to apply a phase taper to aperture data before processing.

9a. Would you like increased resolution on the X-axis? (N/Y)

9b. Would you like increased resolution on the Y-axis? (N/Y)

If the user selects either of these options, the aperture data set is padded with zeroes until the specified dimension reaches the next power of two. The result is increased resolution in the spectrum data. In effect, the FFT is used to interpolate more points in the spectrum data. The program will loop through these two questions until the user responds negatively for both dimensions. Each affirmative response will cause the specified dimension to be increased until the program limit of 4096 points is reached.

10. Does this data set contain independent column or row measurements? (N/Y)

If the user answers yes, the data set is treated as a set of single-dimensioned arrays which are processed with a one- dimensional FFT. Thus, a number of independent, single row measurements may be stored in the same file to save on file overhead.

11a. Would you like to examine a sector of the data with greater resolution? (N/Y)

The program allows the user to view a "close-up" of a rectangular subset of the spectrum data. If the user answers yes, the following prompts are received:

11b. Enter the sector limits for Kx: (-1.,1.)

11c. Enter the sector limits for Ky: (-1.,1.)

The responses, in normalized wave numbers, tell the program where to truncate the spectrum data in K-space. The specified region will be increased so that the number of data points in each dimension is equal to the smallest power of two which completely includes the specified region. Thus, if the specified sector is larger than half the size of the original data set in both dimensions, the user will end up with the same data set and no resolution enhancement will occur. If resolution enhancement is to be applied, the program will list the targeted resolution for each axis and the following prompts will appear next:

11d. Would you like increased resolution on the X-axis? (N/Y)

11e. Would you like increased resolution on the Y-axis? (N/Y)

The program will continue to loop through these prompts until the user responds negatively for both axes. Each affirmative response will cause the specified dimension to be doubled until the program limit of 4096 points is reached.

Ready for probe correction section.

12. What direction is the first polarization? Enter angle (degrees) from Y-axis toward minus X: (0.)

The requested direction is the angle of the polarization vector of the probe with respect to the Y-axis (counterclockwise rotation is positive angle). The default is zero degrees (parallel to the Y-axis). This information is used when the program generates a theoretical probe correction, or no probe correction at all. The theoretical correction assumes an open waveguide probe. The calculated pattern for the probe uses "vertical" (Y-axis) polarization, and this angle is used to rotate the theoretical pattern. Uncorrected data is also assumed to be collected with a linearly polarized probe oriented at the given angle.

13a. Should a probe correction be used? (N/Y)

13b. Empirical or Theoretical? (E/T)

These questions are self-explanatory. A theoretical probe correction calculates the theoretical pattern of an open waveguide; an empirical probe correction requires the user to supply files containing the pattern of the probe.

13c. Enter the probe rotation: 1 for X into Y, or -1 for Y into X : (-1)

This refers to the rotation of the probe between data sets when the same probe is used in orthogonal orientations for two successive scans to collect data. "1" indicates a 90 degree rotation from the positive X-axis to the positive Y-axis; "-1" indicates a 90 degree rotation in the other direction.

13d. Enter the probe dimensions in inches. Enter large, small dimensions:

If a theoretical probe pattern is to be calculated, the program prompts the user for the probe dimensions (rectangular waveguide). The broad wall dimension is entered first.

13d. For the probe pattern (1st pole) - Enter data file name:

13e. For the probe pattern (2nd pole) - Enter data file name:

If an empirical probe correction is to be applied, the program prompts the user for the names of the files containing the probe patterns.

14a. Specify the type of output data desired:

To output the far-field pattern --
Enter "Y" for an azimuth/elevation system (conical about the Y-axis) rotated about the Z axis by a specified angle

Enter "H" **for a Huygens system rotated by a specified angle**
Enter "Z" **for a theta/phi system (conical about the Z-axis) rotated about the Z axis by a specified angle**
Or --
Enter "A" **for a physical translation of the planar aperture data,**
or Return **to output the transverse spectrum data**

After the data has been transformed and probe corrected, the user can output the results in one of three general forms. A carriage return will default to no further processing; i.e., the data will be stored as spectrum data. A response of "A" will direct the program to calculate the transverse fields on a plane parallel to the measurement aperture. Thus, for instance, one could determine the fields at the surface of a phased array for troubleshooting of element performance. Finally, a response of "Y", "H", or "Z" will direct the program to calculate the far field radiation pattern of the test antenna using one of the three polarization models. The domain of the pattern data remains direction cosines, however. A standard abcissa of rotational angle requires interpolation of corresponding pattern points.

14b. Would you like to output both polarizations? (N/Y)

If only one polarization of near-field data was collected; or in other words, if the cross polarized energy was considered negligible; the program will ask this question to allow the user the option of outputting both components of the far-field radiation pattern.

14c. Enter translation vector components in inches (X, Y, Z) : (0.,0.,0.)

14d. Enter low-pass filter radius in normalized wave-number units (Return for no filter)

These questions are asked only if a physical translation of aperture data was requested. A positive Z-component to the translation vector implies translation toward the test antenna. The filter is applied as a circle in the spectrum domain, so "low-pass" refers to spatial frequency. The default filter radius is the entire visible region, or an equivalent radius of one.

14c. What direction is the desired output polarization? Enter angle (degrees) from Y-axis toward minus X:

This is the "specified angle" referred to in question 14a. for responses "Y", "H", and "Z". The default value is the angle entered at question 12.

15. Do you want to apply a Blackman filter (N/Y)?

Ready to output spectrum data files.

16. This file contains ....data. Enter data file name:

This prompt is used if there is only one polarization of output data. The two questions below are asked if the user will output both polarizations.

16a. The first file contains..........data. Enter data file name:

16b. The second file contains.........data. Enter data file name:

In all three of the prompts numbered 16, the ellipsis is replaced by a description of the data which is to be output into that particular file. The user then enters an appropriate file name, which is used in question 17:

17. The default title for file filename is:

....

Enter a new title, or RETURN to default:

This gives the user the option of adding a descriptive title of his choosing to the header record of a file. The question is asked once for each output file.

D. Compiling and Loading

All source code for the routines in these programs is written in FORTRAN and should be compiled with the FTN7X compiler. Files which contain the source code for main routines or subroutines are distinguished by the extension ".FTN". As an example, a typical compiler invocation for the file XYZ.FTN would be:

CI> FTN7X, XYZ,, -,, S

The .FTN extension is implied; the hyphen directs the compiler to put relocatable code in default file XYZ.REL; and the "S" is for the debug option. This option, together with option "DE" when the program is linked, allows use of the system debugger for program diagnostics.

A list of the routines necessary for each program can be found in the load file (denoted by the extension ".LOD"). The load file is the command file to be used when linking the compiled routines into an executable file. It may be necessary to modify the load files if the compiled routines are located in a different directory than expected. In addition, the load files assume that the HP graphics utility subroutines are available in a library called UPLIB_CDS.LIB. Finally, the graphics subroutines in program XYZ make use of device drivers supplied by HP. Appropriate drivers must be linked into a work station program for each device which is to be used for graphics. The supplied plotting routines assume that the work station program for the user's terminal is called WSP_CDS.RUN::PROGRAMS. If a different plotting device is desired, or a different name is used for the work station program, the subroutine PLOT.FTN will have to be modified accordingly.

Section V

Conclusions and Recommendations

A. Conclusions

Completion of this program represents the successful conclusion of three consecutive projects by GTRI to develop and implement a large near-field range for the NASA-Johson Space Center. The forty foot by forty foot measurement structure features a scan plane of approximately 36 feet by 36 feet. The current RF measurement system has a tunable frequency range of 1-26.5 GHz. It has been designed so that, in the future, it can be extended up to 60 GHz. The receiver is able to obtain 1000 data points per second. Depending on data quality, it can possibly operate as high as 4000 measurements per second. The laser metrology subsystem will support probe velocities up to 30 inches per second.

B. Recommendations

The following is a list of recommendations based on the results of this program:

1. Expand data processing software to compensate for probe positioning errors.

The NASA-JSC near-field range is a planar scanner designed to measure the electric field parallel to the antenna aperture. The data processing algorithm assumes that the near-field measurements are sampled at regularly-spaced intervals on a perfectly planar rectangular lattice. However, the mechanical positioning system can not guarantee a perfectly flat or precisely spaced sampling lattice. It should be noted that the NASA-JSC near-field range has an excellent mechanical location accuracy of ±0.001 inch in the XY axis and ±0.005 inch in the Z-axis. However, the out-of-plane (z-axis) errors can become a significant source of error, particularly at millimeter wave frequencies. Methods to convert both out-of-plane errors as well as in-plane measurement errors (XY-axis) can be developed for use in the data processing software. It is recommended that software be developed to compensate for probe position errors. An example of one probe position error correction technique is K-correction. The addition of this capability will improve the accuracy of the far-field patterns computed from the near-field measurements.

2. Automate the AUT Table Leveling Procedure

Leveling of the AUT Table is accomplished manually by the range operator. This task can be automated by using the range cotrol computer. The software can be expanded so that AUT Table leveling can be accomplished by the range operator at the near-field range control console.

3. Automate the Receiver IF Attenuators.

Currently the receiver IF attenuators are manually operated. An improvement in the near-field range receiver alignment can be achieved by using computer-controlled attenuators. The range control computer could automatically set the dynamic range window to achieve the best receiver performance.

4. Add domain options to Pattern output data.

The data reduction software currently calculates pattern points evenly spaced as a function of direction cosine. However, the pattern data points are not evenly spaced as a function of angle. This can cause poor resolution in the far-out sidelobes. By addition of an interpolation algorithm, the quality of the pattern plotted as a function of angle can be improved and provide better comparison with standard output from far-field range pattern measurements.

Section VI

References

1. Cooke, W. P., Thompson, J. E., and Montgomery, J. P., "In-Situ Tile and Antenna Near-Field Measurement Systems Development," Second Interim Technical Report , Contract No. NAS 9-16353, GTRI Project A-2935, April 1984

2. Cooke, W. P., Dunn, A. G., Baugh, R. E., and Montgomery, J. P., "In-Situ Tile and Antenna Near-Field Measurement Systems Development," Final Technical Report , Contract No. NAS 9-16353, GTRI Project A-2935, August 1985

3. Application Note 197-2, Model HP 5501A, Laser and Optics Measurement System.

4. User's Guide, Model HP 5528A Laser Measurement System, April 1984.

5. Cooke, W. P., Friederich, P. G., and Jenkins, B. M. "Probe Design Considerations for the NASA-JSC Near-field Antenna Test Facility," Final Technical Report, Lockheed EMSCO Contract No. 0200118981, GTRI Project A-8029, November 1988.

This page intentionally left blank

APPENDIX A

Receiver Wiring Diagrams and Component Specifications

This page intentionally left blank

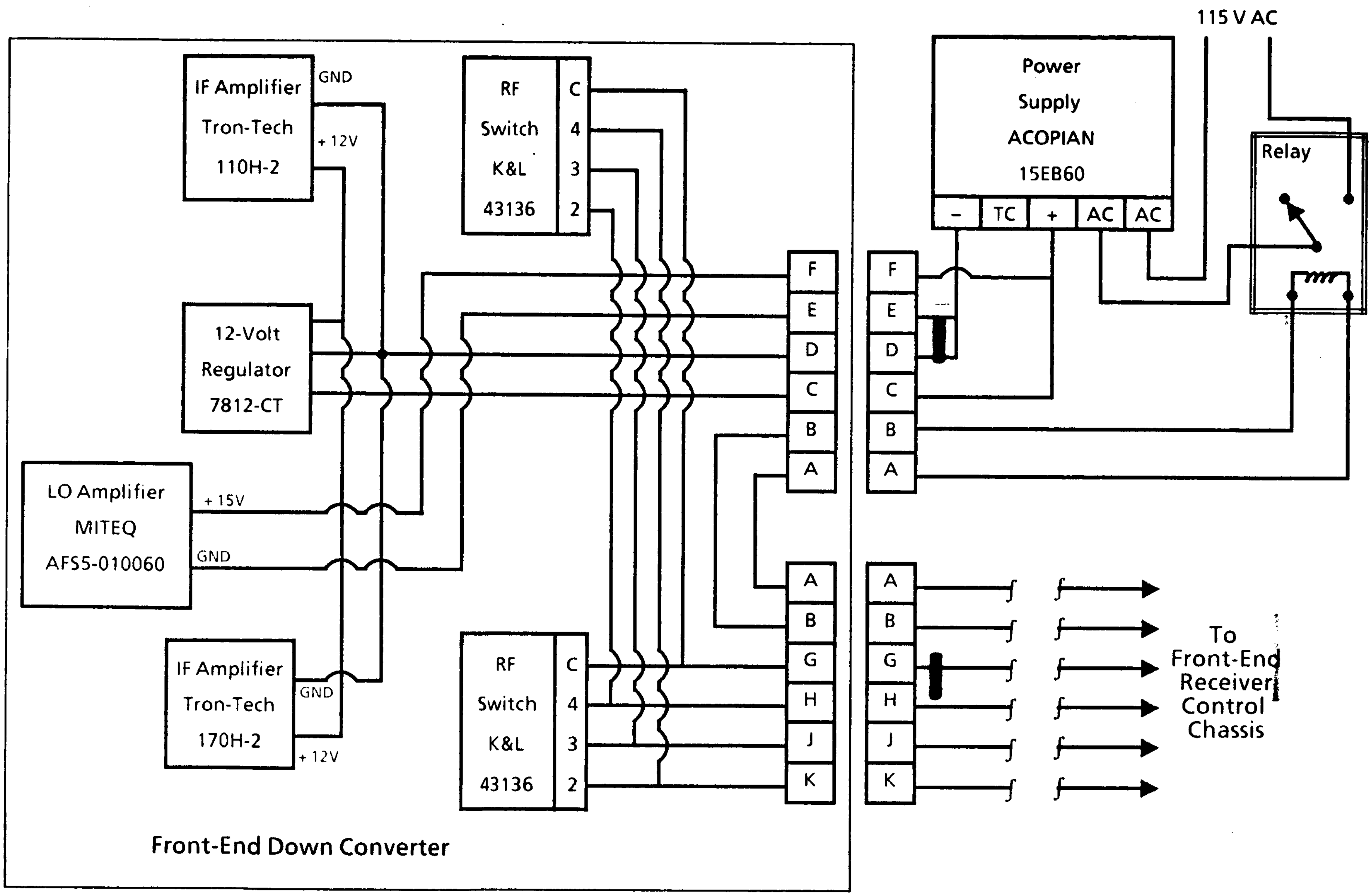


Figure A-1. Wiring Diagram of Test Channel Receiver Front-End Down Converter and Associated Power Supply.

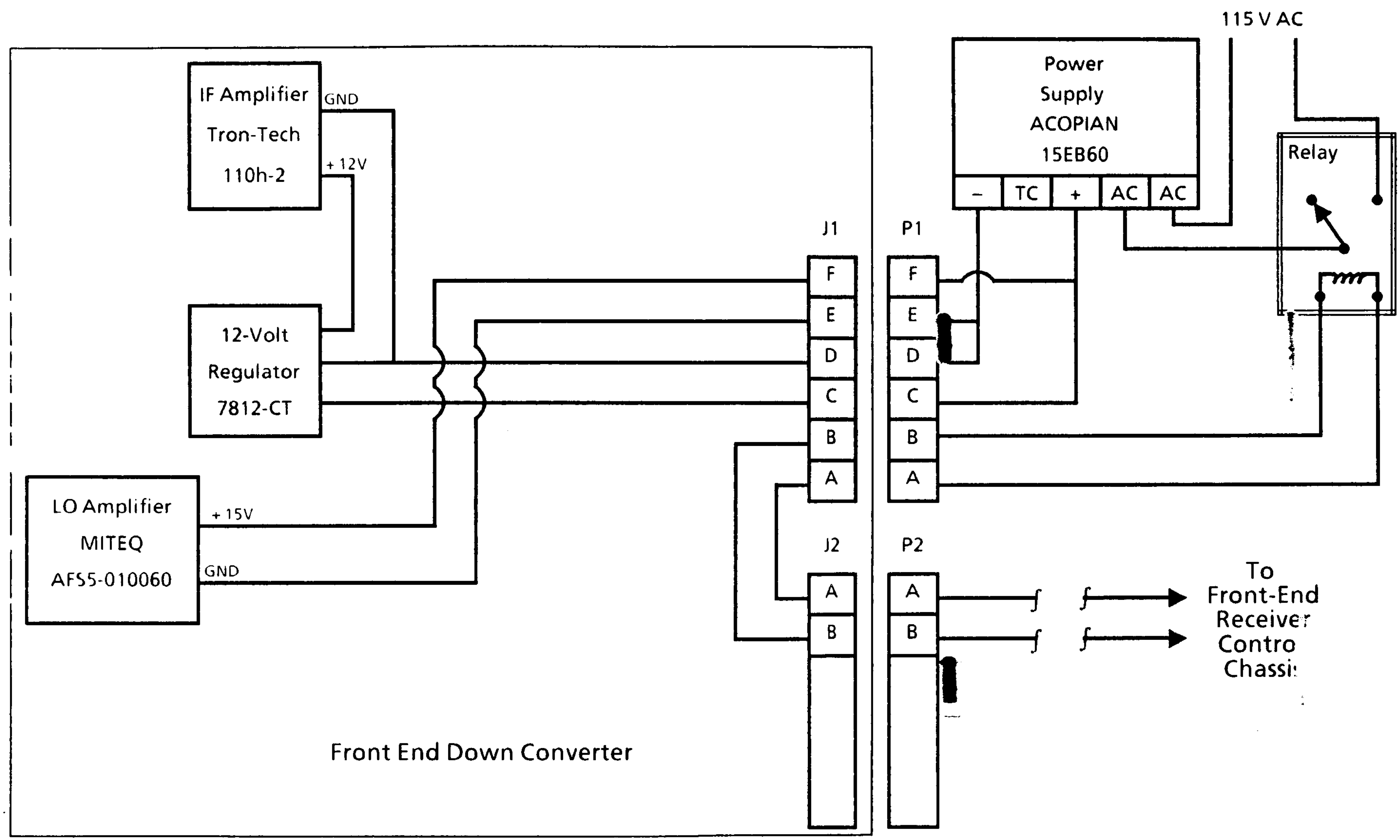


Figure A-2. Wiring Diagram of Reference Channel Receiver Front-End Down Converter and Associated Power Supply.

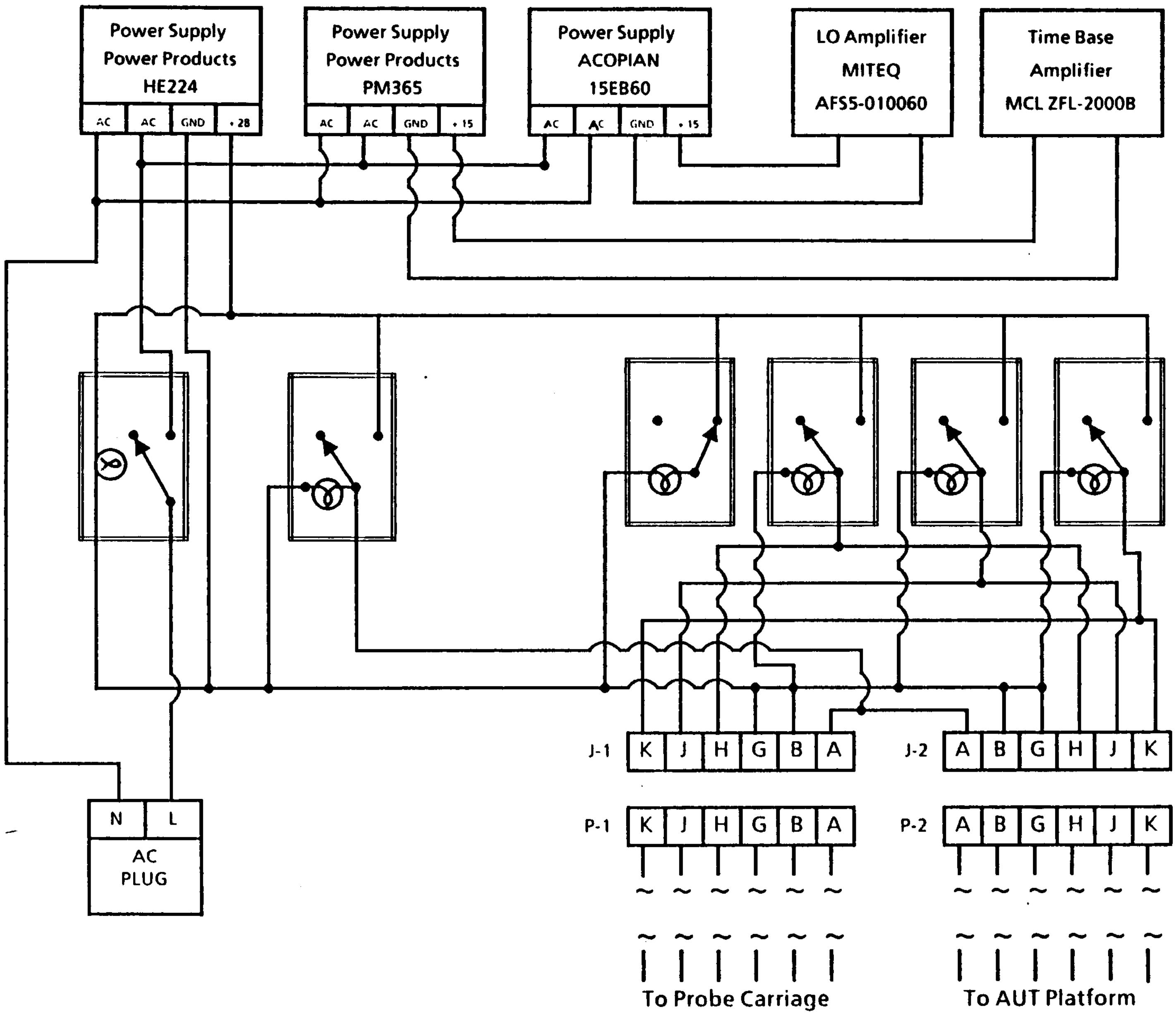


Figure A-3. Front-End Receiver Control Chassis Wiring Diagram

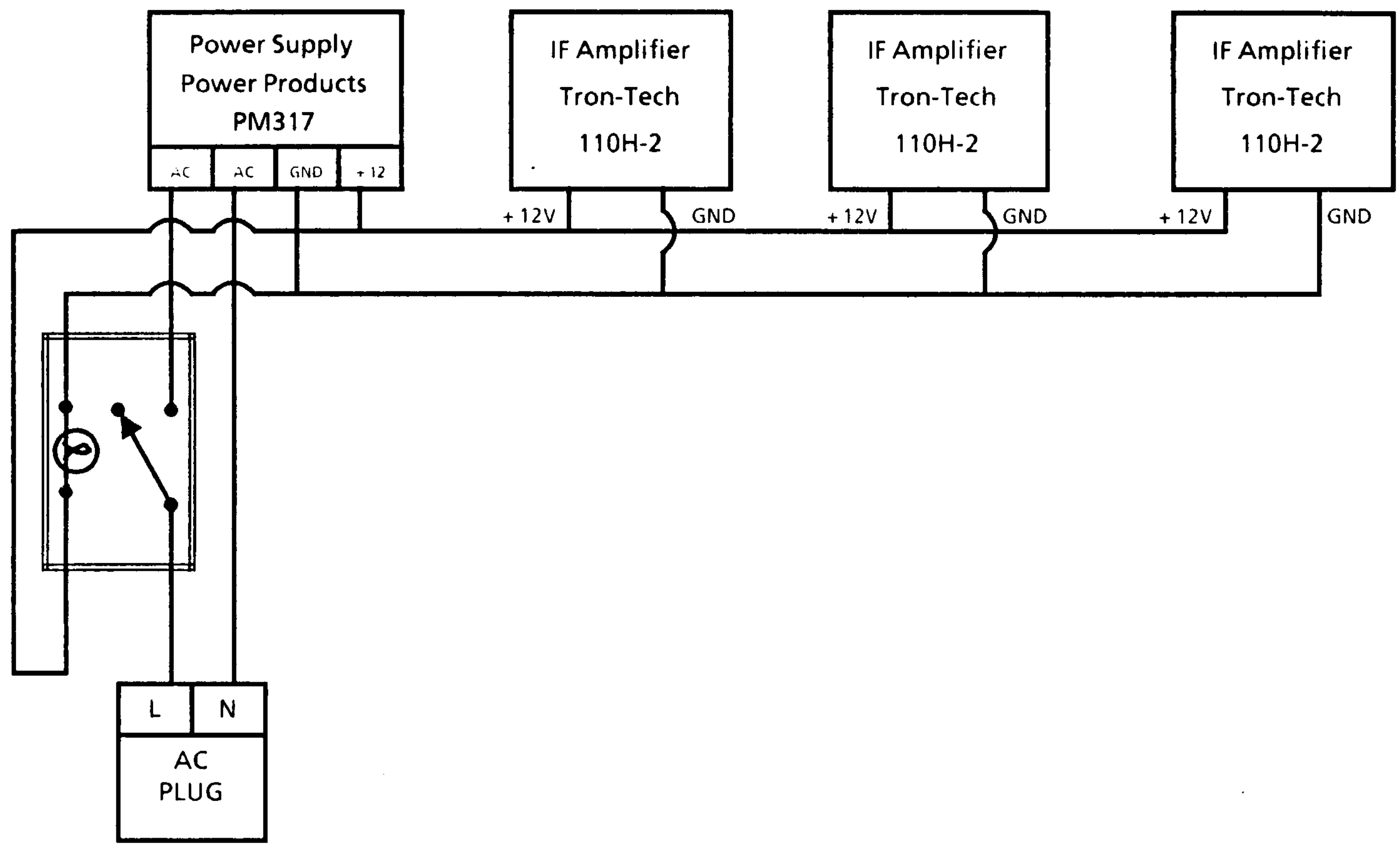


Figure A-4. Wiring Diagram of IF Control Chassis

ORIGINAL PAGE IS
OF POOR QUALITY

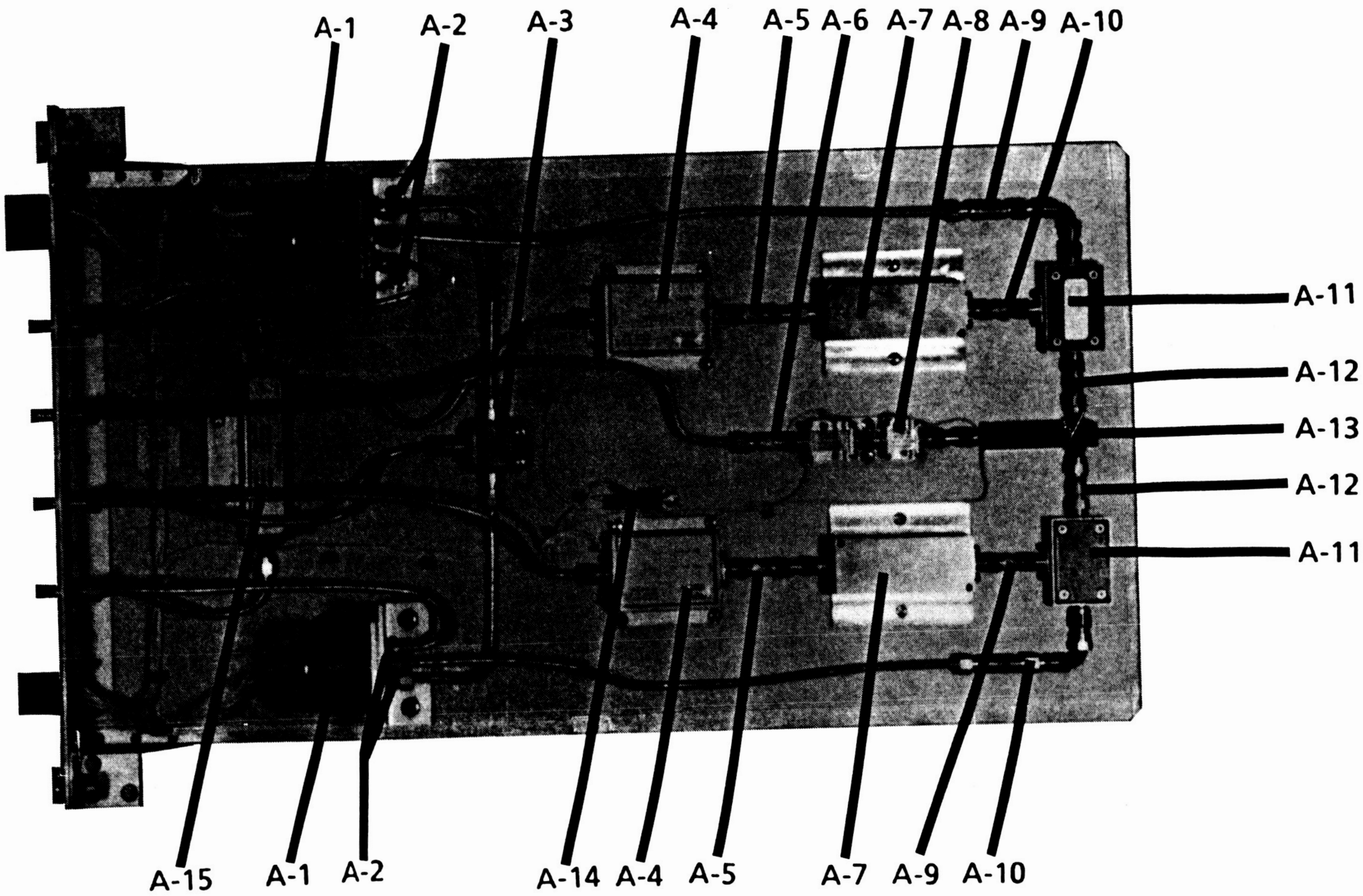


Figure A-5. Test Channel Receiver Front-End Down converter Component Layout.

TABLE A-1

TEST CHANNEL RECEIVER FRONT-END DOWN CONVERTER COMPONENT IDENTIFICATION

Photo * Designator	Description	Manufacture	Model No.	Page No.
A-1	RF Switch (2)	K&L Microwave, Inc.	43136	A-13
A-2	50 Ohm Termination (4)	Florida RF Labs	12-002	-
A-3	2-Way Power Splitter	Hewlett Packard	11667B	A-14
A-4	IF Amplifier (2)	Tron-Tech	W 110H-2	A-15
A-5	IF 3 dB Attenuator (2)	Midwest Microwave	290-03	A-16
A-6	LO 20 dB Attenuator	Midwest Microwave	290-20	A-16
A-7	IF 20 MHz Band Pass Filter (2)	K&L Microwave, Inc.	4851-20/X1-0/0	A-17
A-8	LO Amplifier	MITEQ	AFS5-010060-55-23P	A-18
A-9	RF 3 dB Attenuator (2)	Hewlett Packard	8493C	A-19
A-10	IF 3 dB Attenuator (2)	Midwest Microwave	290 M-3	A-16
A-11	RF Mixer (2)	RHG Electronics, Inc.	DMS 1-26	A-20
A-12	LO 6 dB Attenuator (2)	Midwest Microwave	290M-6	A-16
A-13	LO 2-Way Power Splitter	Omni Spectra	2089-6220-00	A-21
A-14	12 Volt Regulator	Motorola	7812 CT	A-22
A-15	LO Directional Coupler	KRYTAR	2610	A-23

* These designators identify the components shown in Figure A-5. When more than one component has the same designator, the number of the identical components is given in parenthesis in the description column.

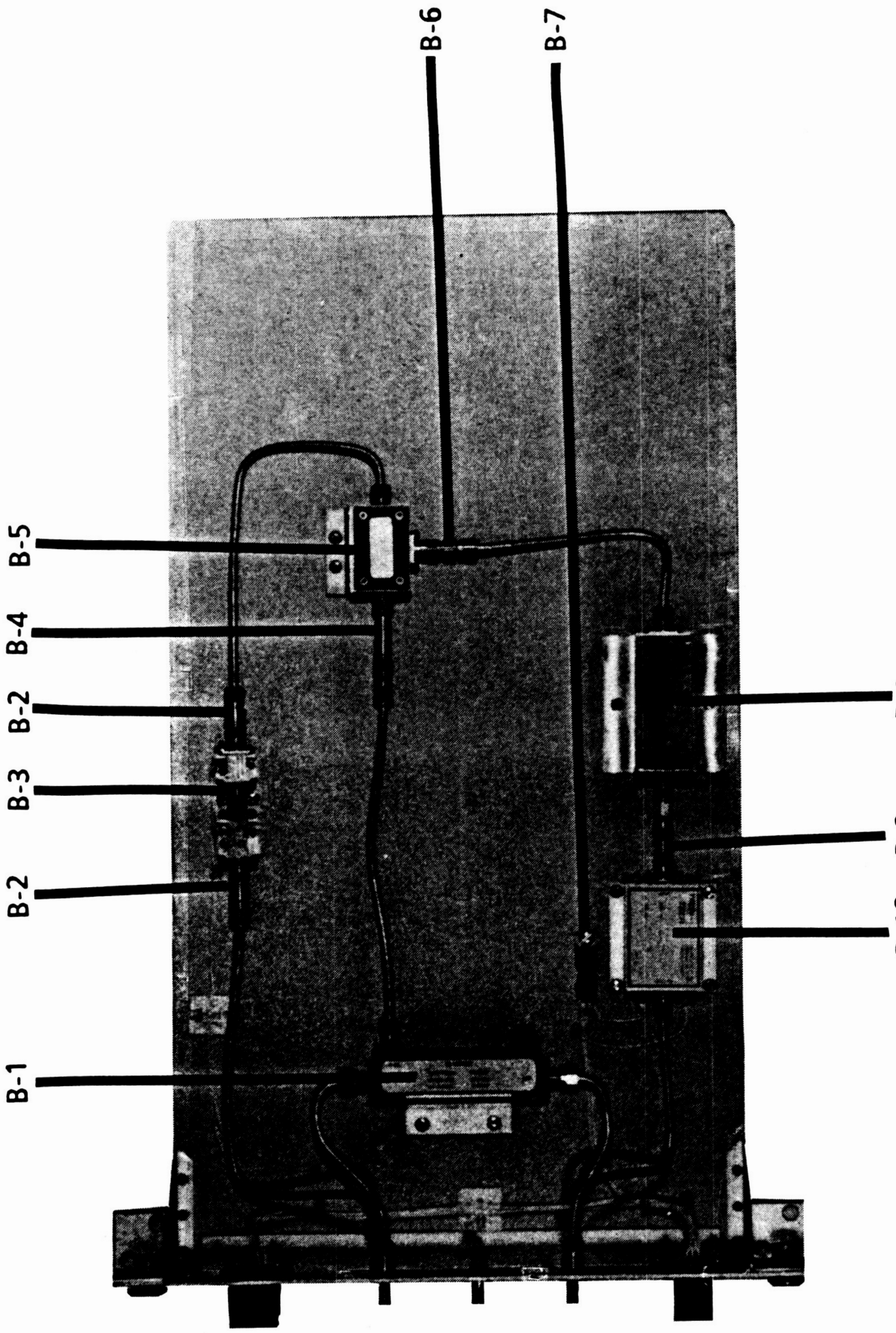


Figure A-6. Reference Channel Receiver Front-End Down Converter Component Layout.

TABLE A-2
REFERENCE CHANNEL RECEIVER FRONT-END DOWN CONVERTER COMPONENT IDENTIFICATION

Photo * Designator	Description	Manufacture	Model No.	Page No.
B-1	RF Directional Coupler	KRYTAR	2610	A-23
B-2	LO 10dB Attenuator (2)	KDI Electronics	610M	A-24
B-3	LO Amplifier	MITEQ	AF55-010060-55-2P	A-18
B-4	RF 20 dB Attenuator	Midwest Microwave	550M-20	-
B-5	RF Mixer	RHG Electronics, Inc.	DMS 1-26	A-20
B-6	IF 3 dB Attenuator	KDI Electronics	603M	A-24
B-7	12 Volt Regulator	Motorola	7812CT	A-22
B-8	IF 20-MHz Band Pass Filter	K&L Microwave, Inc.	4B51-20/XI-0/0	A-17
B-9	IF 3 dB Attenuator	Midwest Microwave	290-3	A-16
B-10	IF Amplifier	TRON-TECH	W110H-2	A-15

* These designators identify the components shown in Figure A-6. When more than one component has the same designator, the number of identical components is given in parenthesis in the description column.

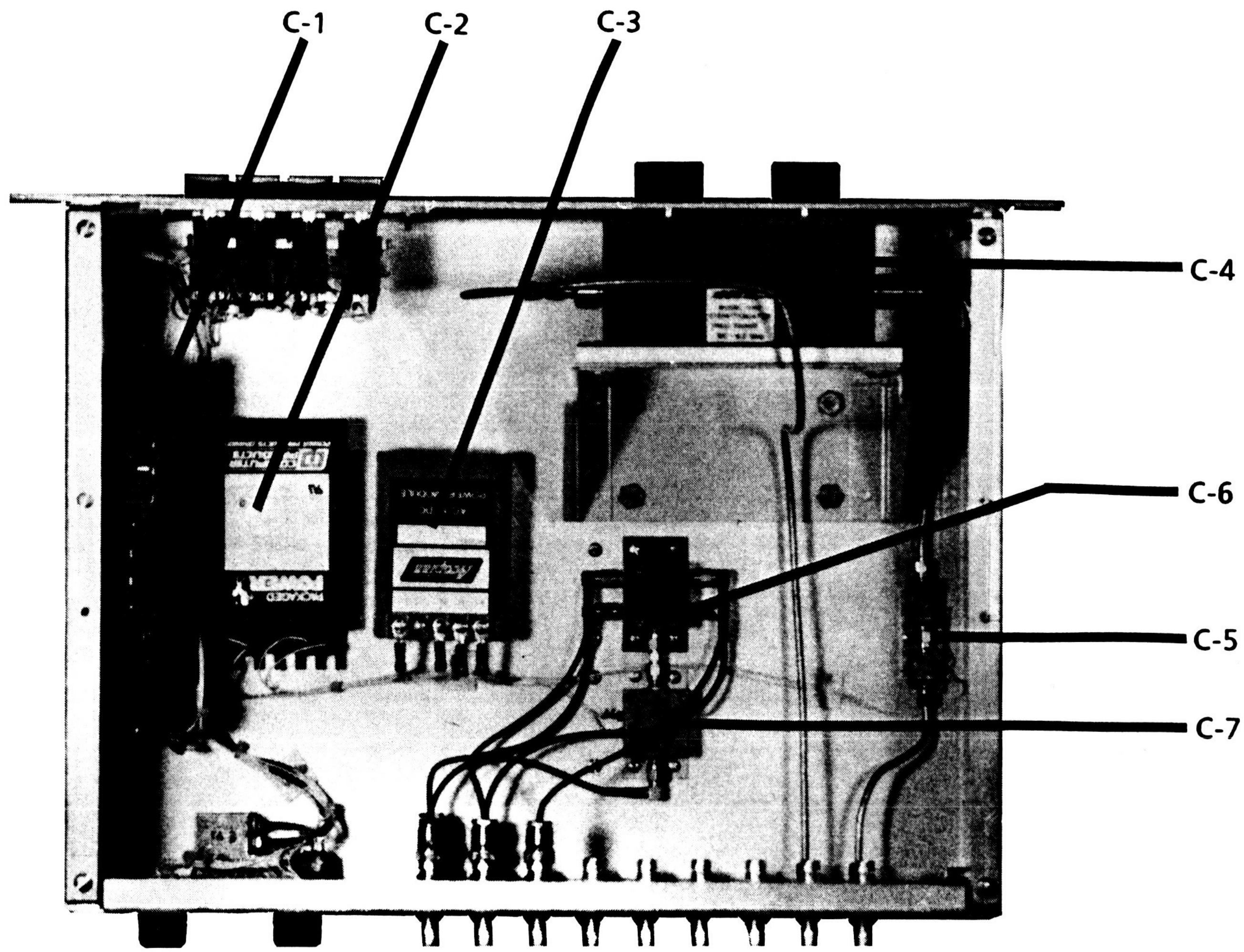


Figure A-7. Front-End Receiver Control Chassis Component Layout.

TABLE A-3
FRONT-END RECEIVER CONTROL CHASSIS COMPONENT IDENTIFICATION

Photo * Designator	Description	Manufacture	Model No.	Page No.
C-1	RF Switch Control Power Supply	Power Products	HE 224	A-25
C-2	Time Base Amplifier Power Supply	Power Products	PM 365	A-26
C-3	LO Amplifier Power Supply	ACOPIAN	15EB60	A-27
C-4	LO 0-69 dB Step Attenuator	Midwest Microwave	1044	A-28
C-5	LO Amplifier	MITEQ	AF55-010060-55-2P	A-18
C-6	Time Base 4-Way Power Splitter	Mini-Circuits	ZFSC-4-3B	A-29
C-7	Time Base Amplifier	Mini-Circuits	ZFL-2000B	A-30

* These designators identify the components shown in Figure A-7.

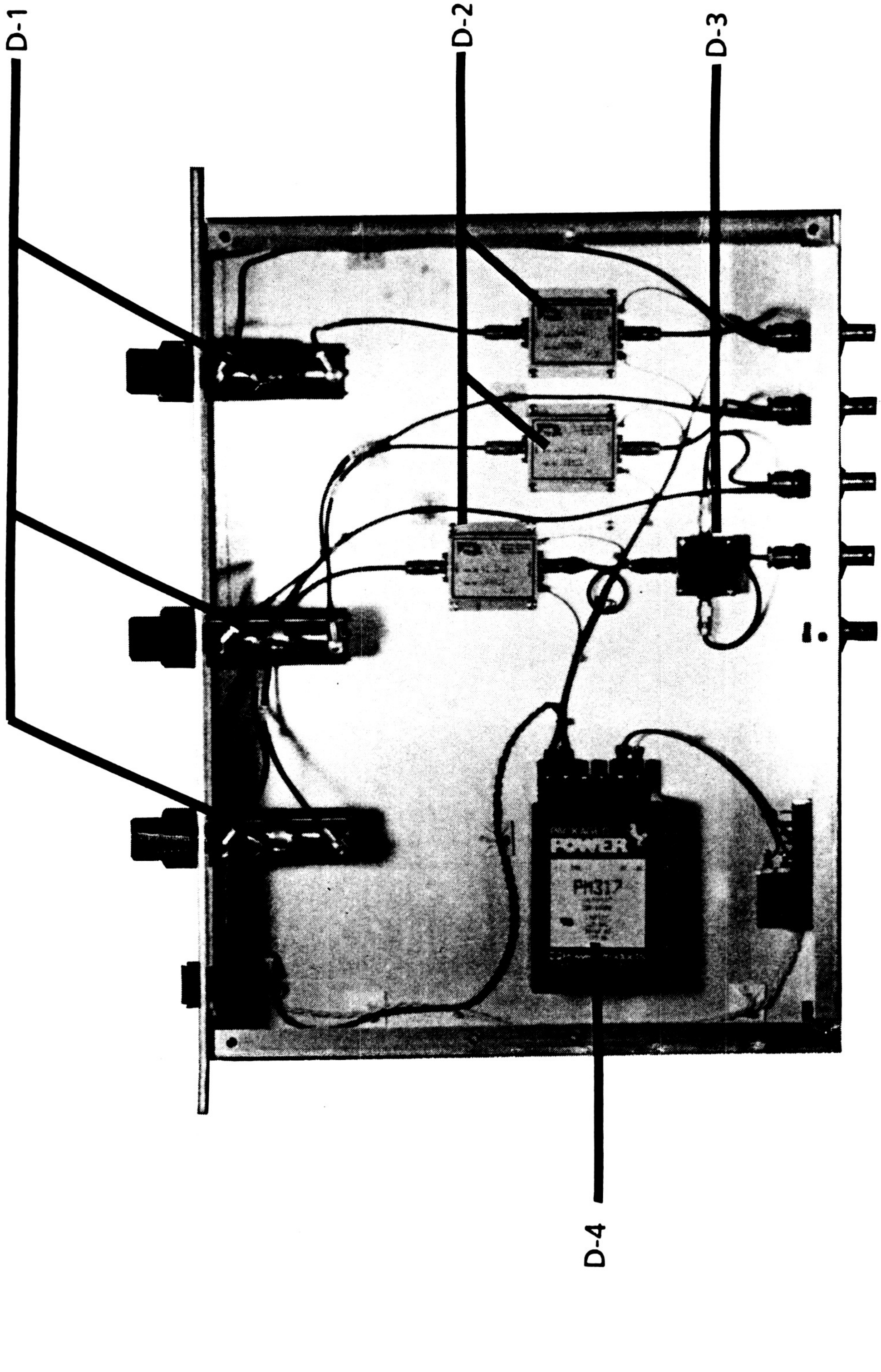
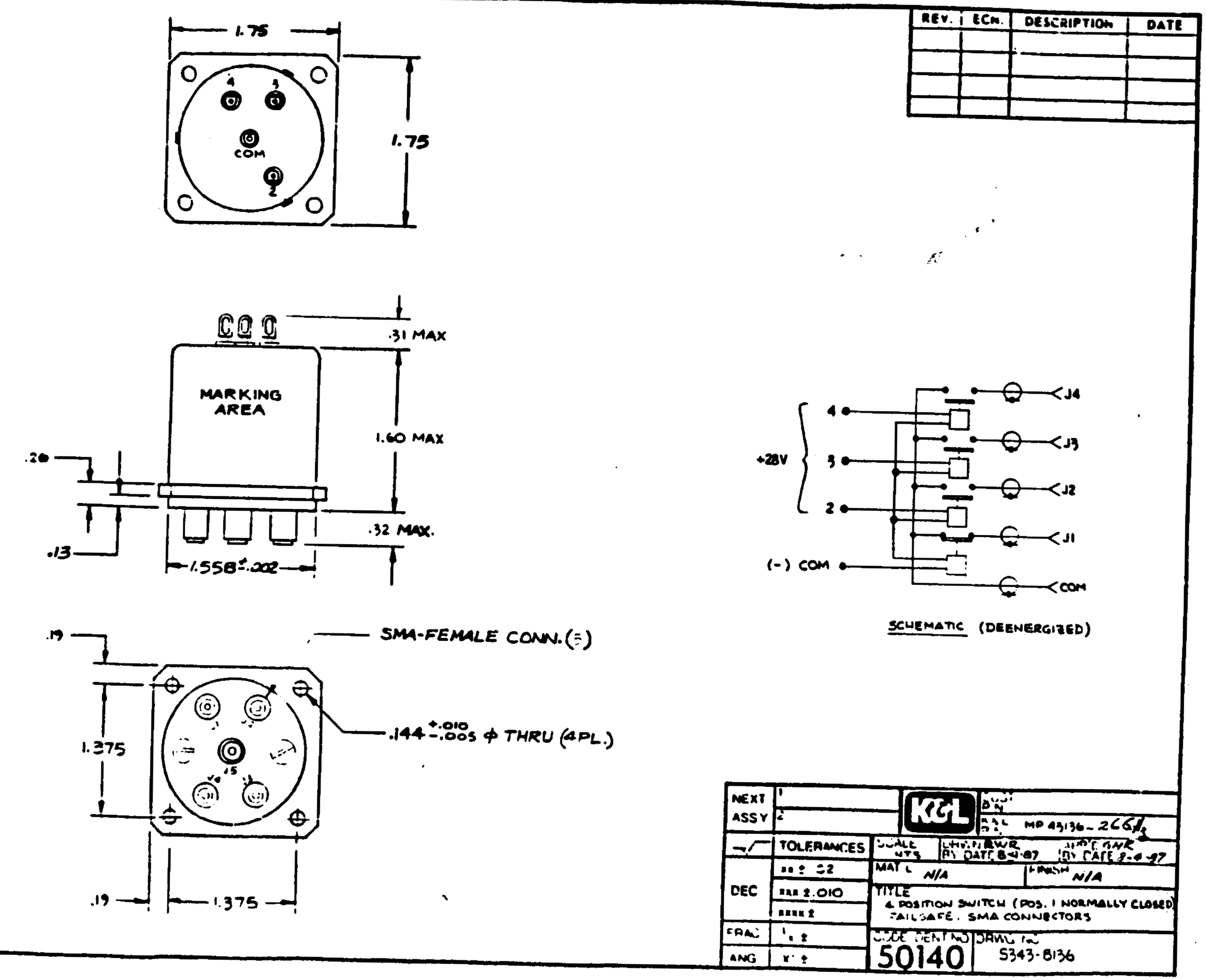


Figure A-8. IF Control Chassis Component Layout.

TABLE A-4
IF CONTROL CHASSIS COMPONENT IDENTIFICATION

Photo * Designator	Description	Manufacture	Model No.	Page No.
D-1	IF 0-59 dB Step Attenuator (3)	Weinschel Engineering	3023-100	A-31
D-2	IF Amplifier (3)	TRON-TECH	W110H-2	A-15
D-3	IF 2-Way Power Splitter	KDI Electronics	PSK-211	A-32
D-4	IF Amplifier Power Supply	Power Products	PM317	A-26

* These designatoers identify the components shown in Figure A-8. When more than one component has the same designator, the number of identical components is given in parenthesis in the Description column.



ORIGINAL PAGE IS
OF POOR QUALITY

Table 1. Specifications

Frequency Range: DC to 26.5 GHz

Maximum Input Power: +27 dBm (0.5W)

Description	Frequency (GHz)	
	DC to 18	**DC to 26.5**
Input SWR	≤1.22	≤1.29
Equivalent Output SWR (Leveling or ratio measurement)	≤1.22	≤1.22
Output Tracking (between output arms)	0.25 dB	0.40 dB

Connectors: Precision 3.5mm Female on all ports

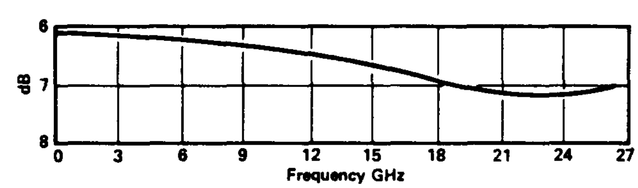
Dimensions: 47 mm wide x 40 mm high x 10 mm deep (1.85 in x 1.57 in x 0.39 in)

Shipping Weight: 0.14 kg (4.94 oz.)

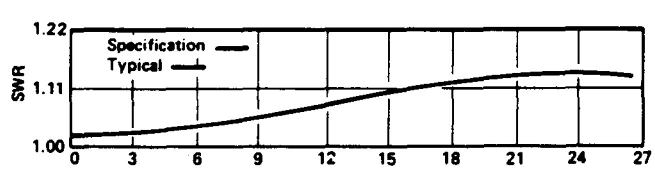
Table 2. Supplemental Characteristics

Description	Frequency (GHz)	
	DC to 18	**DC to 26.5**
Phase Tracking (between output arms), typically:	≤1.5°	≤2.5°

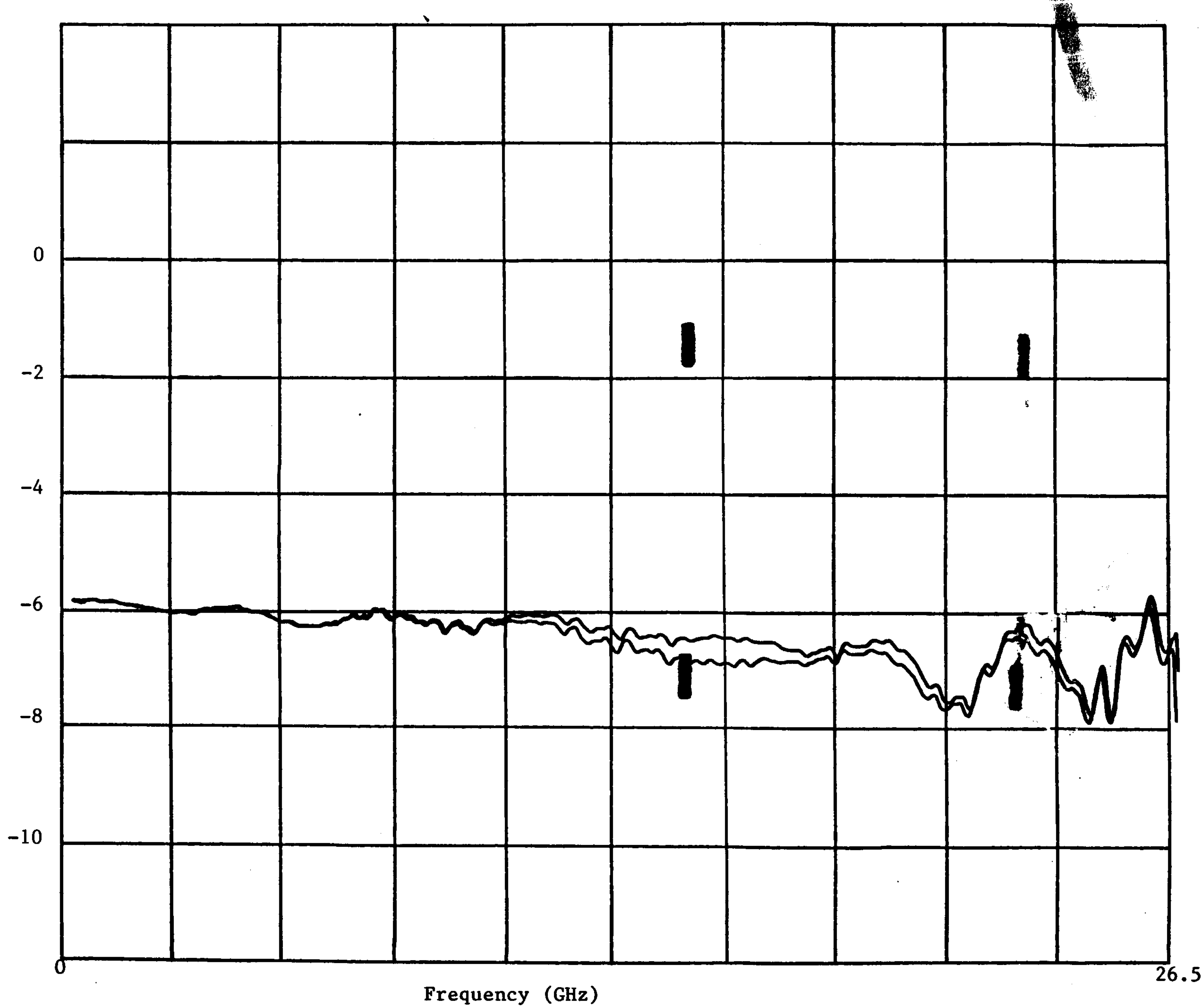
Typical insertion loss:



Leveling or ratio measurement source match:



Power Splitter
Model 11667B



TRONTECH, INC.
63 SHARK RIVER ROAD
NEPTUNE, NEW JERSEY 07753

CUSTOMER Lockheed-Emsco

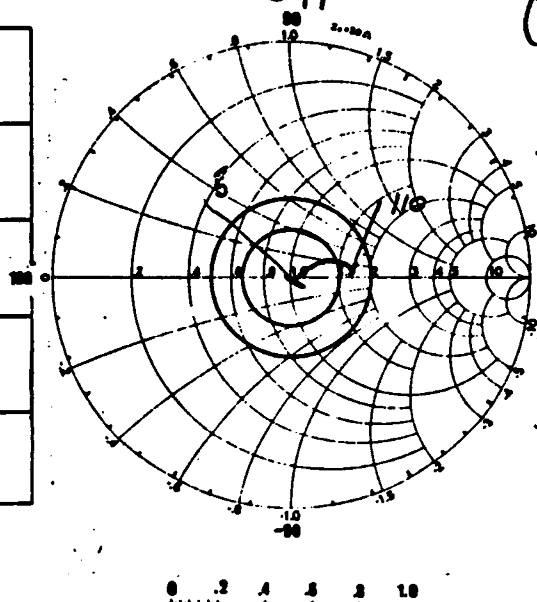
MODEL NUMBER W110H-2

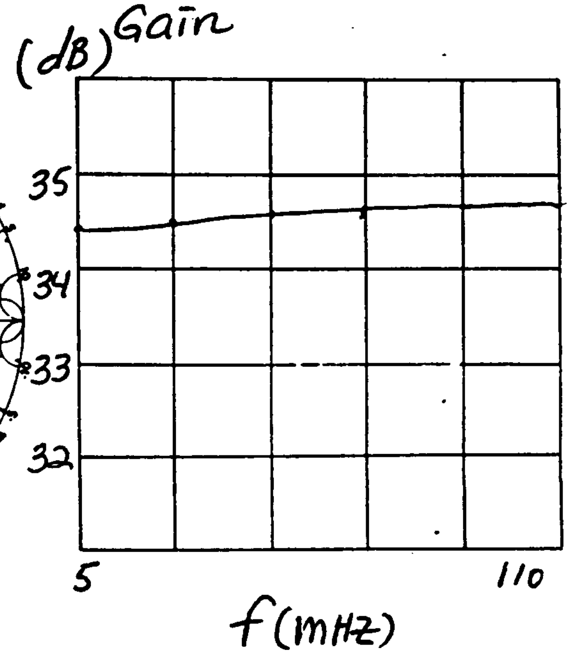
DATE 9/25/87

SERIAL NUMBER L740 01

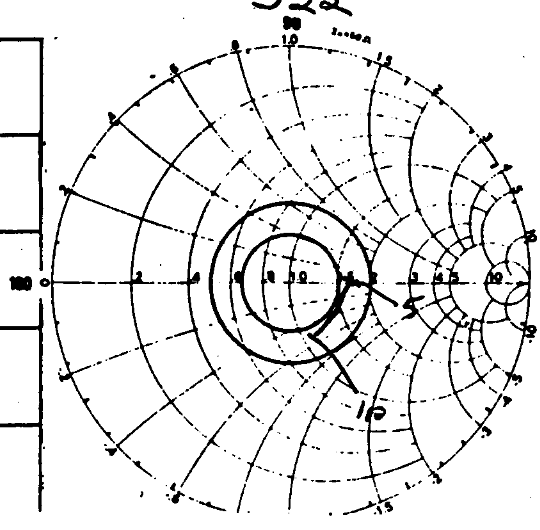
	SPEC'D.	MEASURED		SPEC'D.	MEASURED
CENT. FREQ.			VSWR IN	2:1	SEE S11
1 dB B.W.	5-110 MHz	SEE PLOT	VSWR OUT	2:1	" S22
_ dB B.W.			P/O @ 1 dB COMPR.	+5 dBm	>+7 dBm
GAIN	30 dB	" "	I_c @ +12 V		21.8 mA
FLATNESS	±.5 dB	" "			
NOISE FIG.	1.4 dB	1.4 dB			

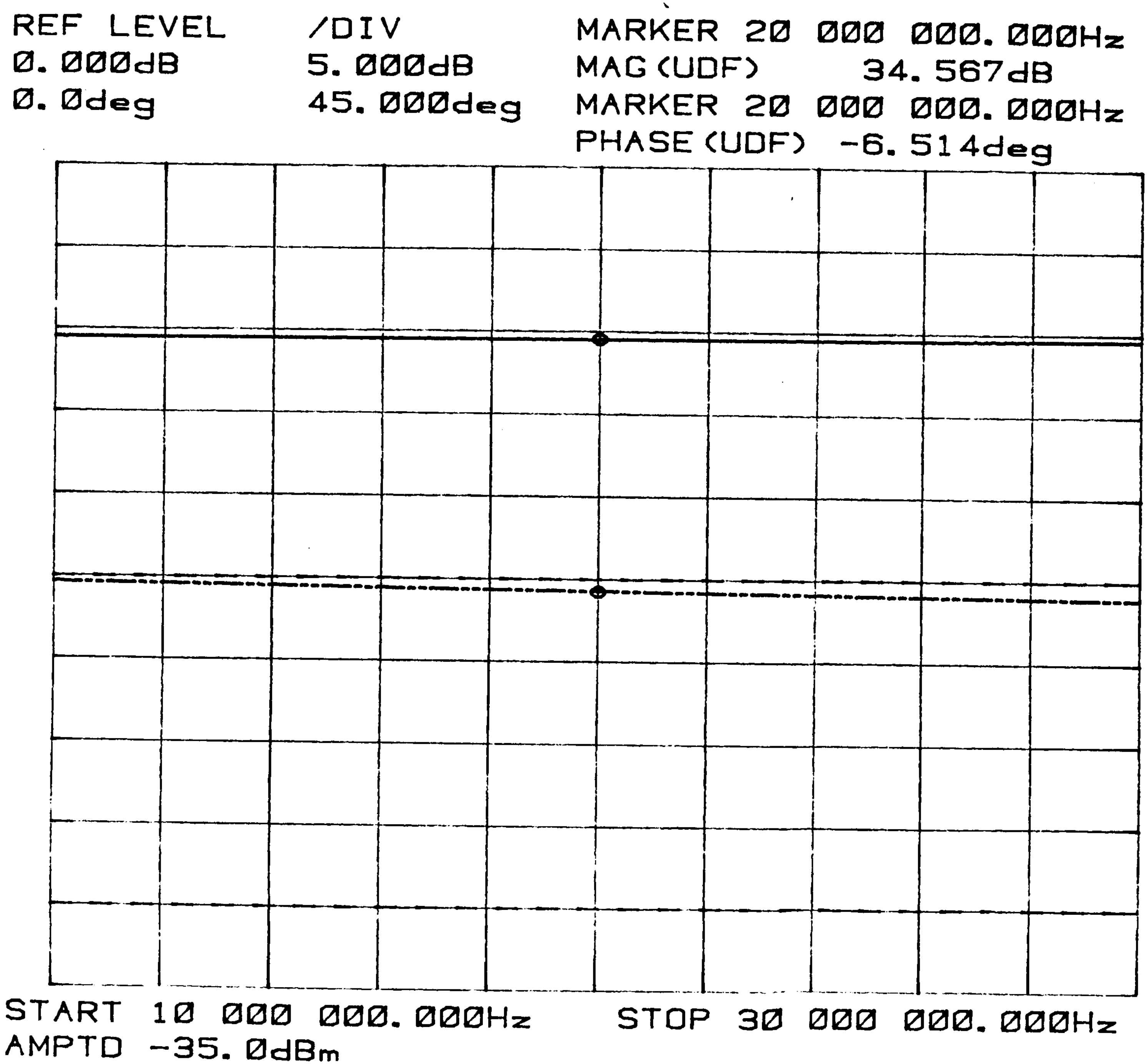
S11





S22





IF Amplifier
Model W110H-2
S/N L74001

TRONTECH, INC.
63 SHARK RIVER ROAD
NEPTUNE, NEW JERSEY 07753

ORIGINAL PAGE IS OF POOR QUALITY

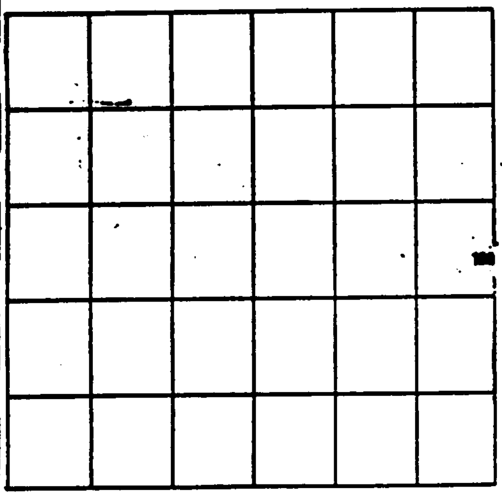
CUSTOMER Lockheed-Emsco

MODEL NUMBER W110H-2

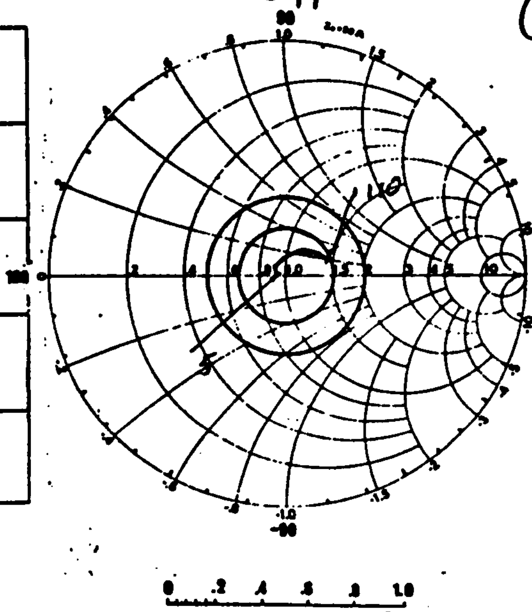
SERIAL NUMBER L740 02

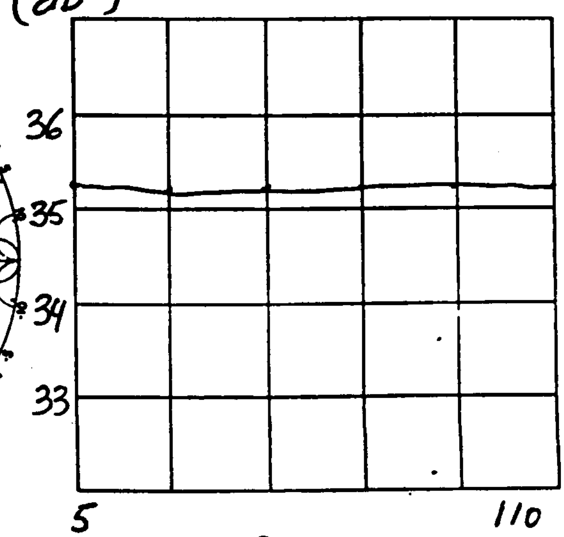
DATE 9/25/87

	SPEC'D.	MEASURED		SPEC'D.	MEASURED
CENT. FREQ.			VSWR IN	2:1	See S11
1 dB B.W.	5-110 mHz	See Plot	VSWR OUT	2:1	" S22
_ dB B.W.			P/O @ 1 dB COMPR.	+5 dBm	>+7 dBm
GAIN	30 dB	" "	I_c @ +12 V		22.3 mA
FLATNESS	±.5 dB	" "			
NOISE FIG.	1.4 dB	1.4 dB			

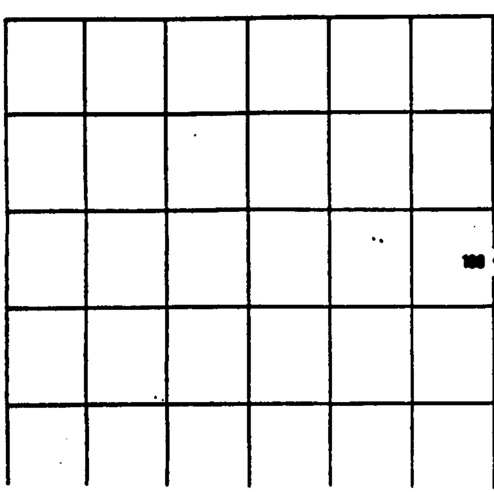


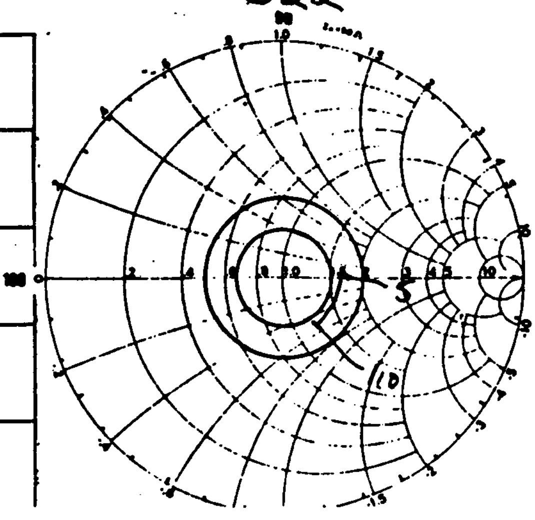
S11

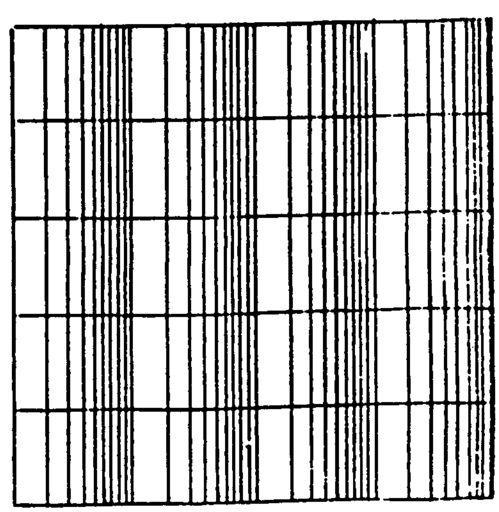




S22



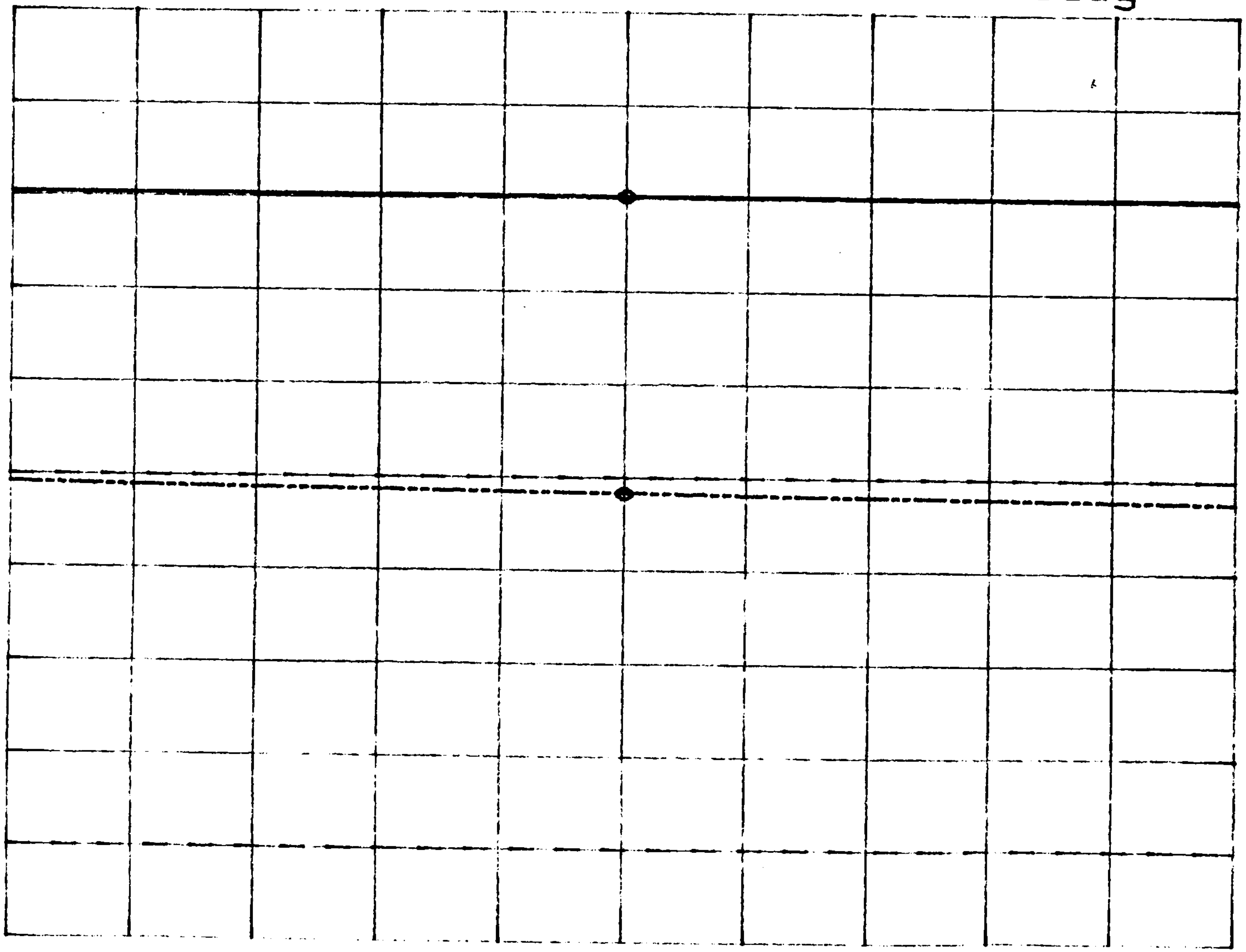




REF LEVEL 0.000dB 0.0deg

/DIV 5.000dB 45.000deg

MARKER 20 000 000.000Hz
MAG (UDF) 35.149dB
MARKER 20 000 000.000Hz
PHASE (UDF) -7.569deg



IF Amplifier
Model W110H-2
S/N L74002

START 10 000 000.000Hz STOP 30 000 000.000Hz
AMPTD -35.0dBm

TRONTECH, INC.
63 SHARK RIVER ROAD
NEPTUNE, NEW JERSEY 07753

ORIGINAL PAGE IS
OF POOR QUALITY

CUSTOMER Lockheed-Emsco

MODEL NUMBER W110H-2

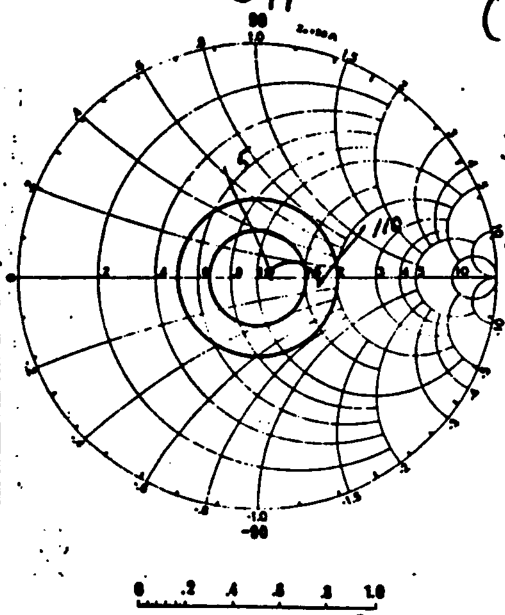
SERIAL NUMBER L74003

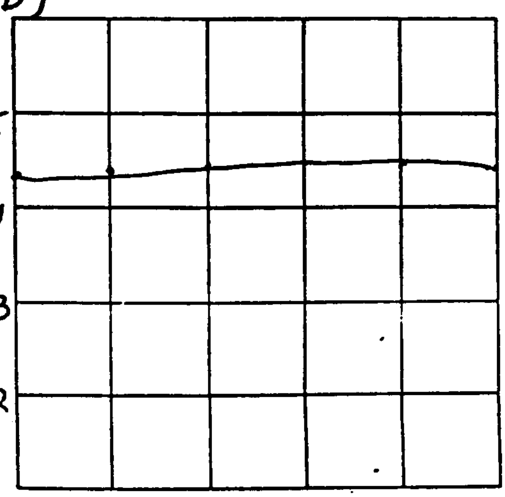
DATE 9/25/87

	SPEC'D.	MEASURED
CENT. FREQ.		
1 dB B.W.	5-110 mHz	See Plot
_ dB B.W.		
GAIN	30 dB	" "
FLATNESS	±.5 dB	" "
NOISE FIG.	1.4 dB	1.4 dB

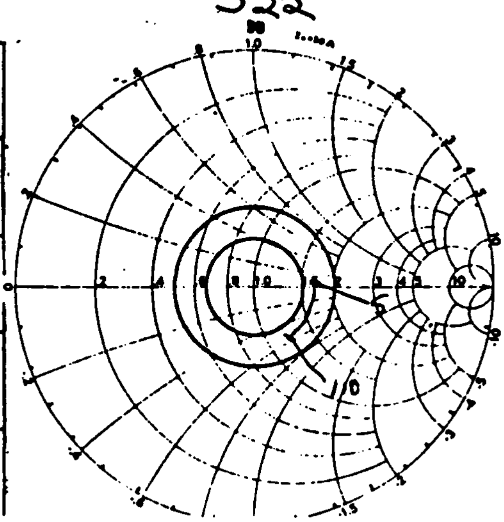
	SPEC'D.	MEASURED
VSWR IN	2:1	See S_{11}
VSWR OUT	2:1	" S_{22}
P/O @ 1 dB COMPR.	+5 dBm	7+7 dBm
I_c @ +12 V		22.1 mA

S11

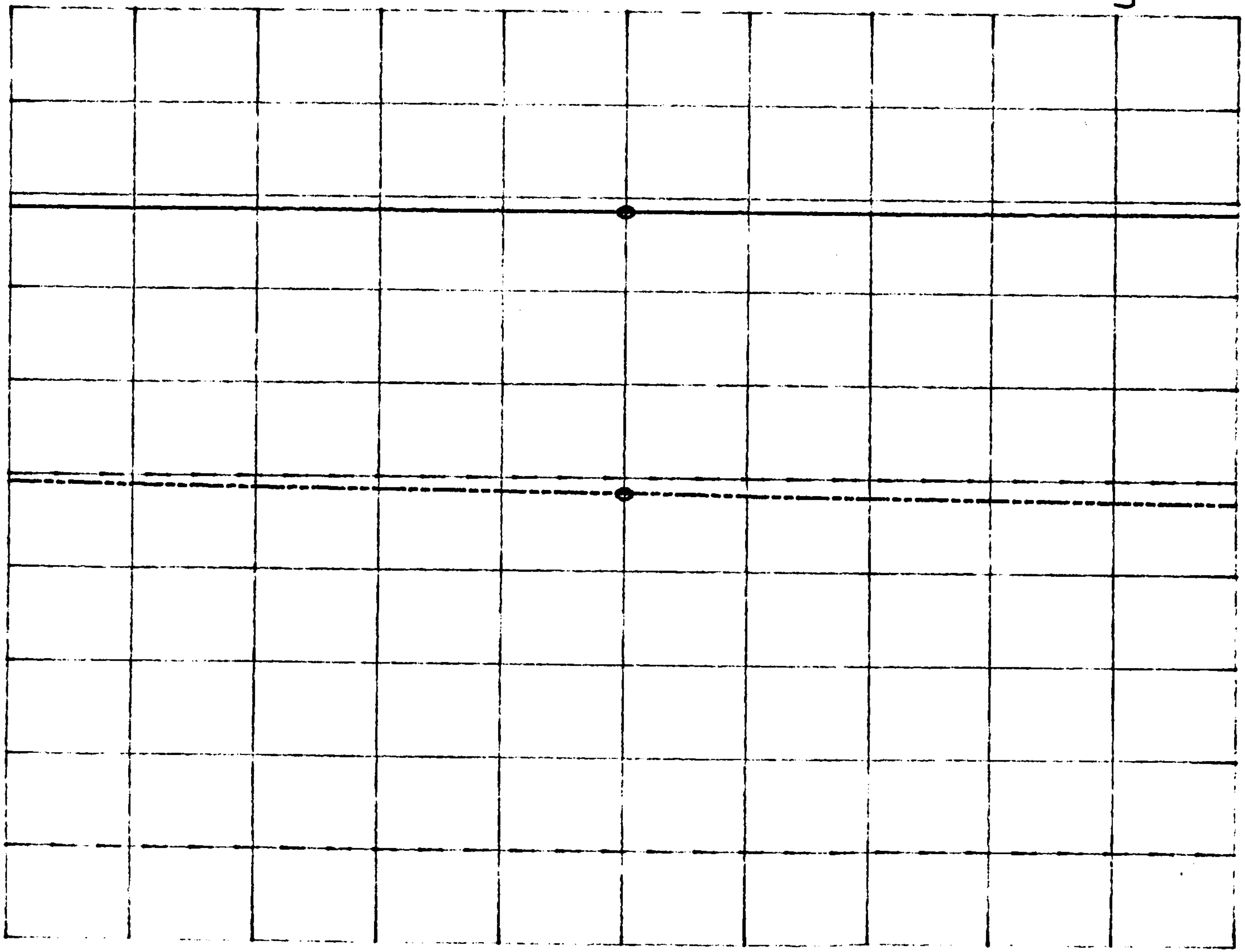




S22



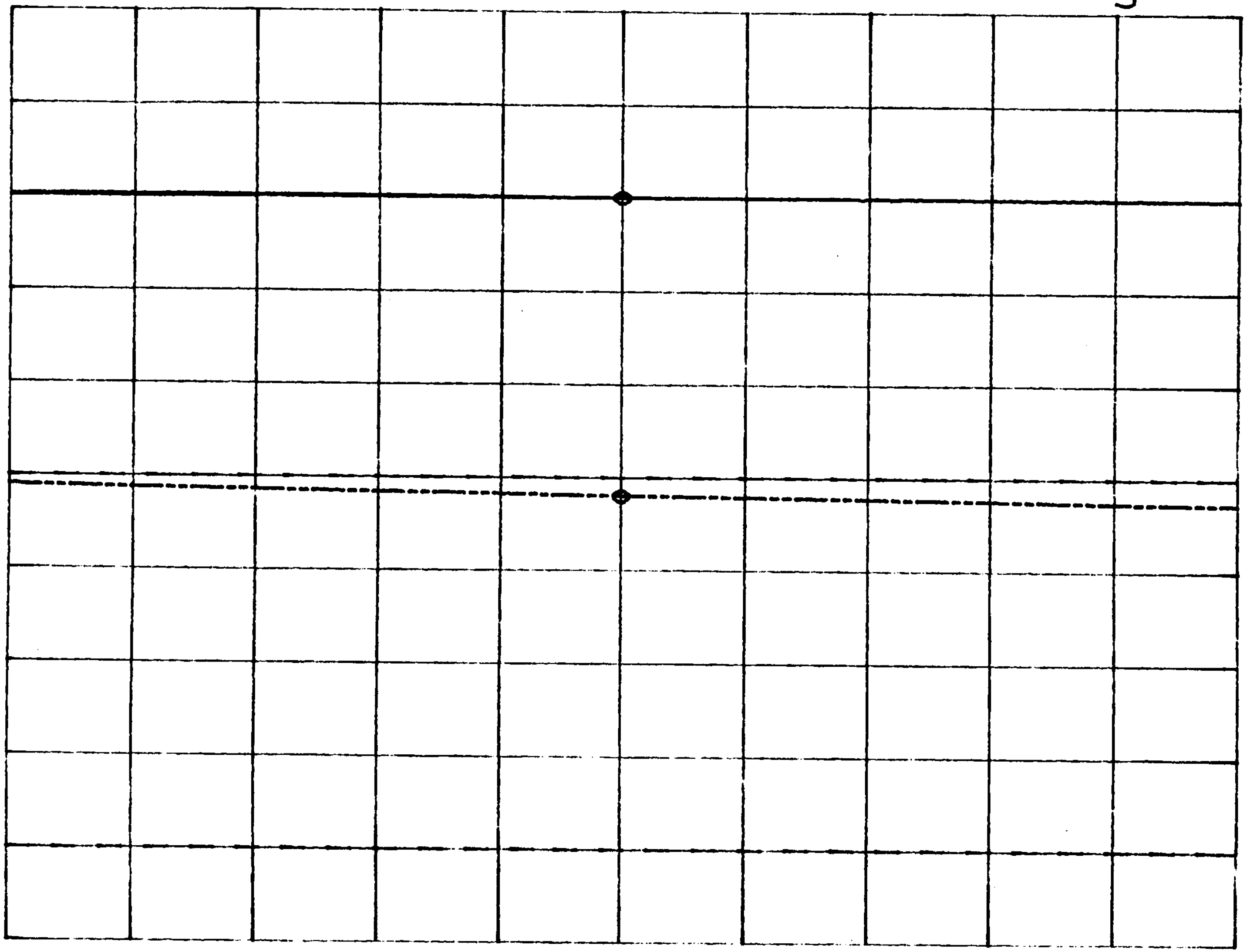
REF LEVEL /DIV MARKER 20 000 000.000Hz
0.000dB 5.000dB MAG (UDF) 34.300dB
0.0deg 45.000deg MARKER 20 000 000.000Hz
PHASE (UDF) -7.435deg

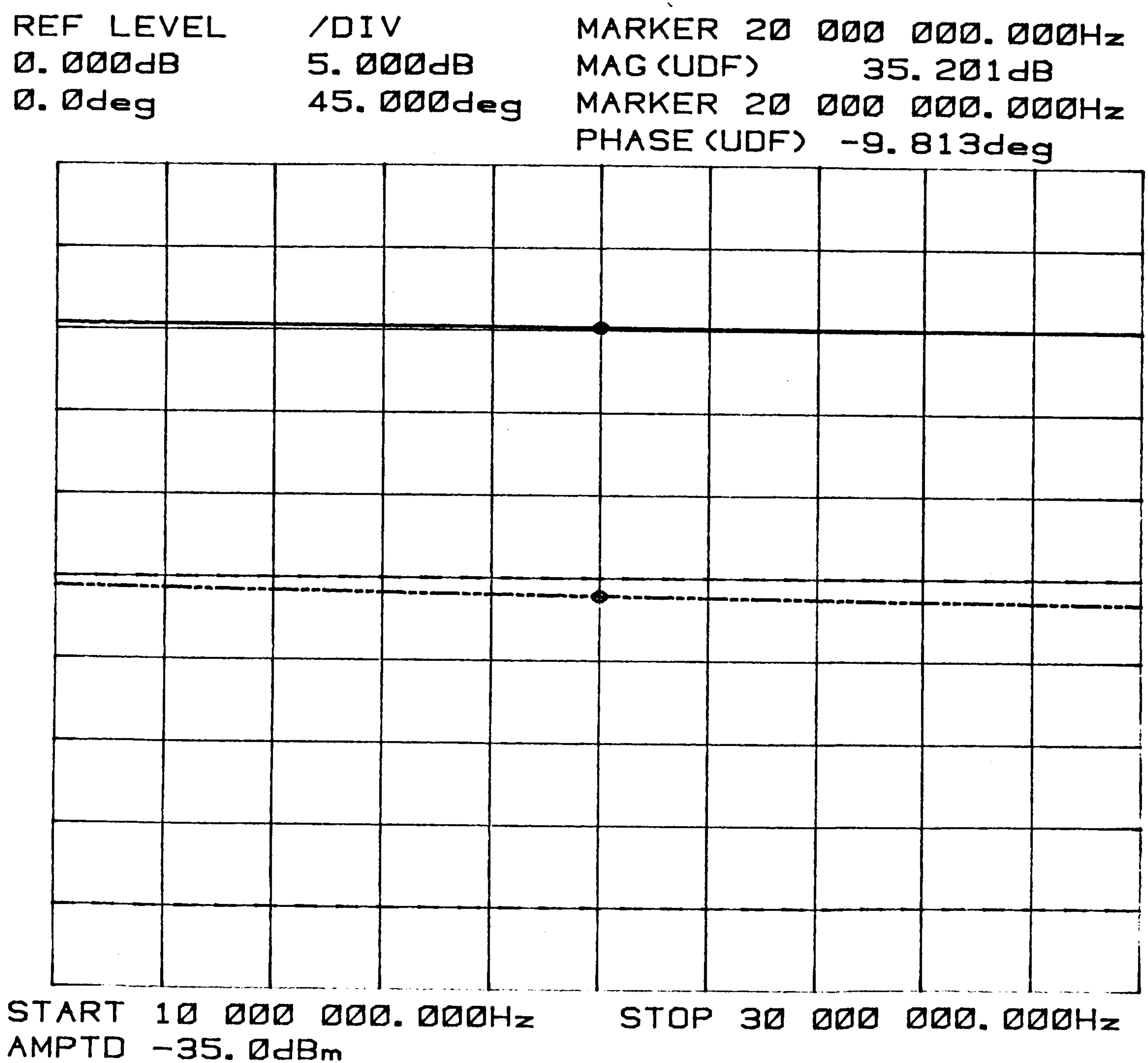


IF Amplifier
Model W110H-2
S/N L74003

START 10 000 000.000Hz STOP 30 000 000.000Hz
AMPTD -35.0dBm

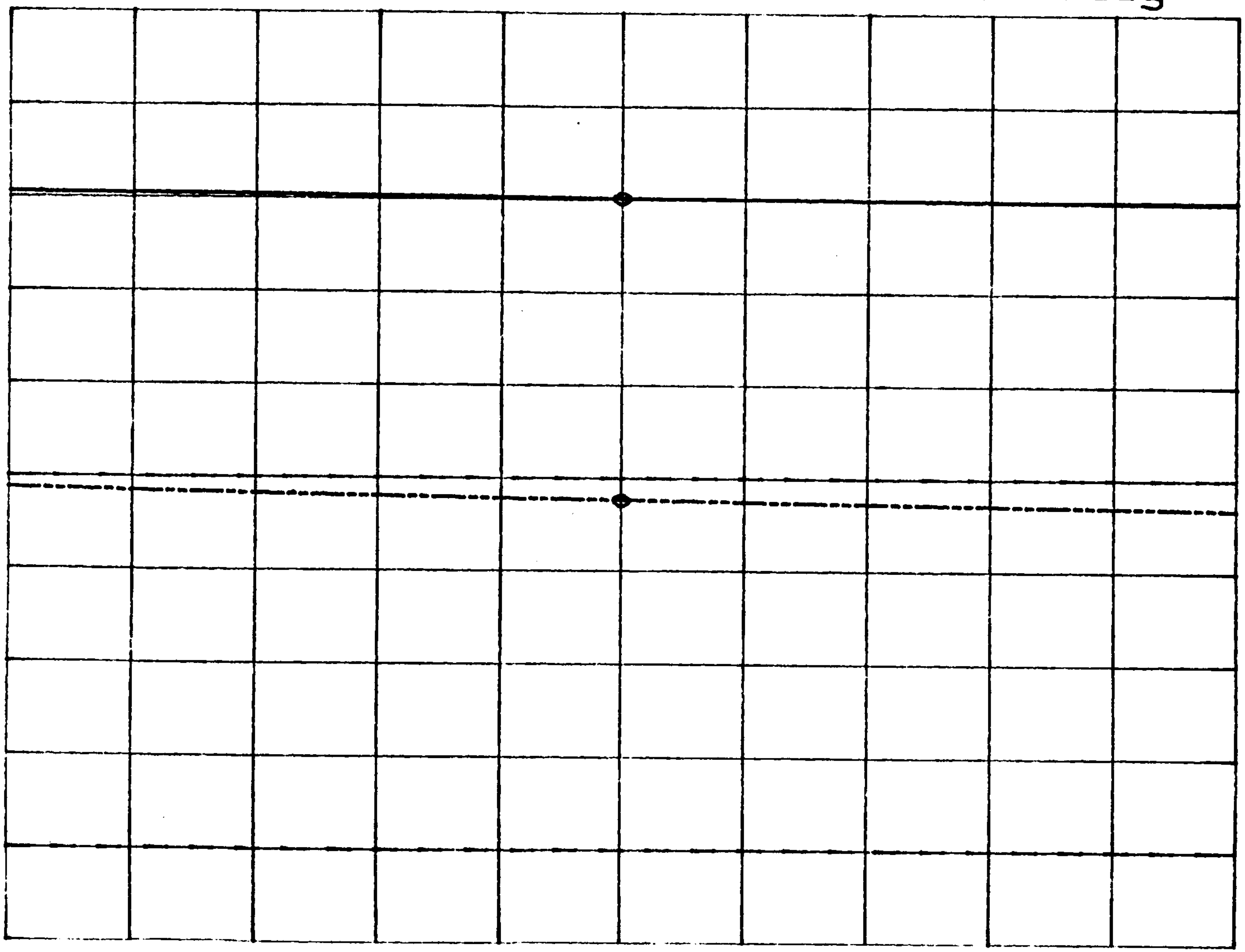
IF Amplifier
Model W110H-2
S/N L73601





IF Amplifier
Model W110H-2
S/N L73602

REF LEVEL	/DIV	MARKER 20 000 000.000Hz
0.000dB	5.000dB	MAG (UDF) 35.104dB
0.0deg	45.000deg	MARKER 20 000 000.000Hz
		PHASE (UDF) -10.455deg



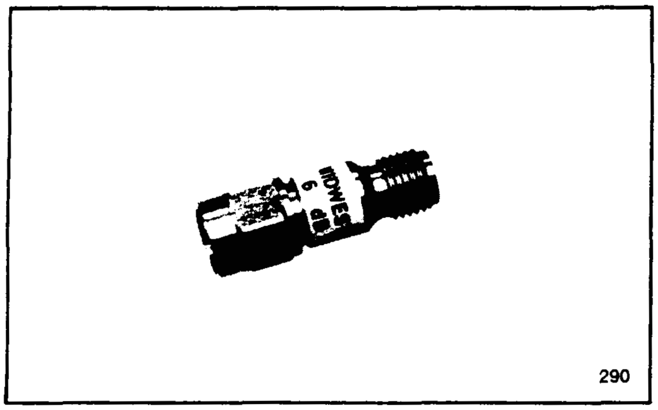
START 10 000 000.000Hz STOP 30 000 000.000Hz
AMPTD -35.0dBm

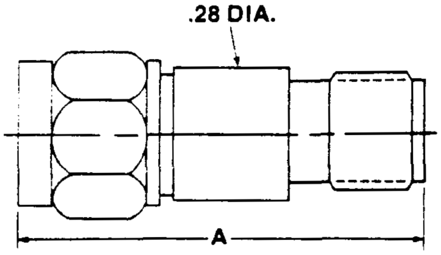
IF Amplifier
Model W110H-2
S/N L73603

FIXED ATTENUATORS SMA

ORIGINAL PAGE IS
OF POOR QUALITY

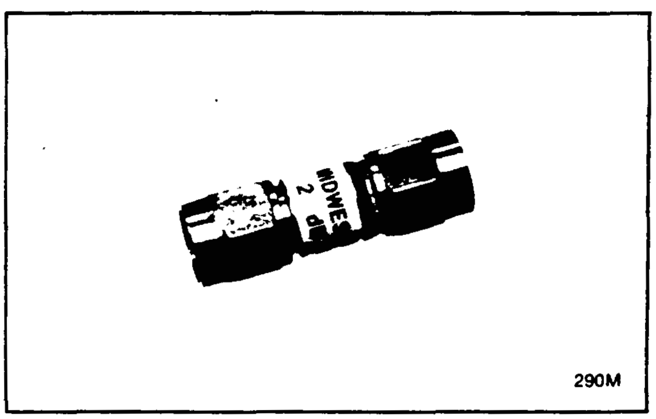
THE MINIPAD®*

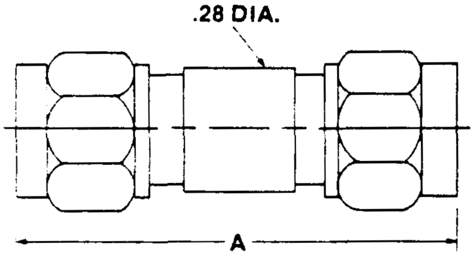

290



ATTENUATION VALUE	LENGTH A
1-12 dB	.86
13-30 dB	1.02

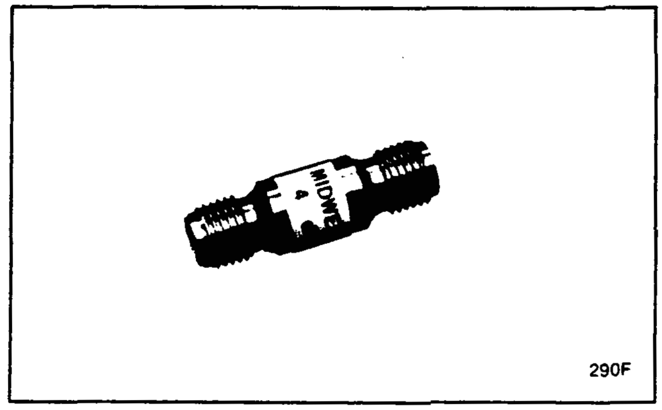
MINIPAD DOUBLE MALE

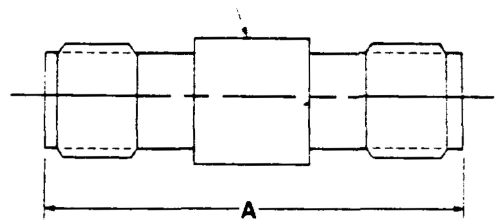

290M



ATTENUATION VALUE	LENGTH A
1-12 dB	98
13-30 dB	1.12

MINIPAD DOUBLE FEMALE


290F



ATTENUATION VALUE	LENGTH A
1-12 dB	90
13-30 dB	1.03

DC TO 18 GHz HIGH PERFORMANCE SPECIFICATIONS

MODELS 290, 290M AND 290F
FREQUENCY RANGE: DC TO 18 GHz
CONNECTOR TYPE: STAINLESS STEEL SMA PER MIL-C-39012
ATTENUATION VALUES: 1 THRU 30dB IN 1dB INCREMENTS
ATTENUATION ACCURACY: 1 - 6dB **±0.3dB** ■ 7 - 20dB **±0.5dB** ■ 21 - 30 dB **±1.0dB**
MAXIMUM VSWR: 1.07 +0.015fGHz
MAXIMUM INPUT POWER: 2 WATTS AVERAGE AT +25°C DERATED LINEARLY TO 0.5 WATTS AT +125°C
OPERATING TEMPERATURE RANGE: -65°C TO +125°C

DC TO 12.4 GHz HIGH PERFORMANCE SPECIFICATIONS

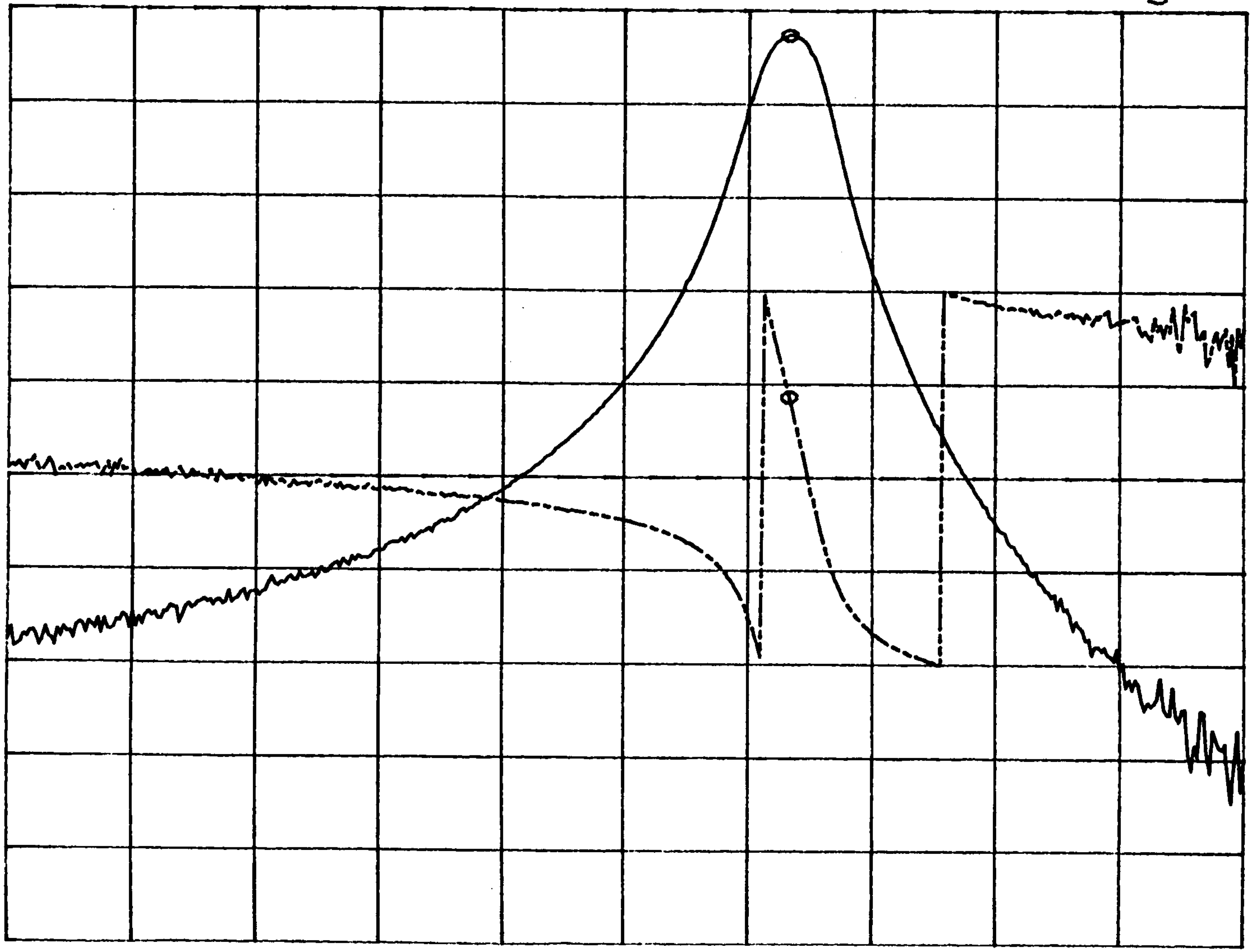
MODELS 291, 291M AND 291F
FREQUENCY RANGE: DC TO 12.4 GHz
CONNECTOR TYPE: STAINLESS STEEL SMA PER MIL-C-39012
ATTENUATION VALUES: 1 THRU 30dB IN 1dB INCREMENTS
ATTENUATION ACCURACY: 1 - 6dB **±0.3dB** ■ 7 - 20dB **±0.5dB** ■ 21 - 30dB **±1.0dB**
MAXIMUM VSWR: 1.07 +0.015fGHz
MAXIMUM INPUT POWER: 2 WATTS AVERAGE AT +25°C DERATED LINEARLY TO 0.5 WATTS AT +125°C
OPERATING TEMPERATURE RANGE: -65°C TO +125°C

* U.S. Patent number 3,824,506 applies to all Minipads.

REF LEVEL 0.000dB 0.0deg

/DIV 10.000dB 90.000deg

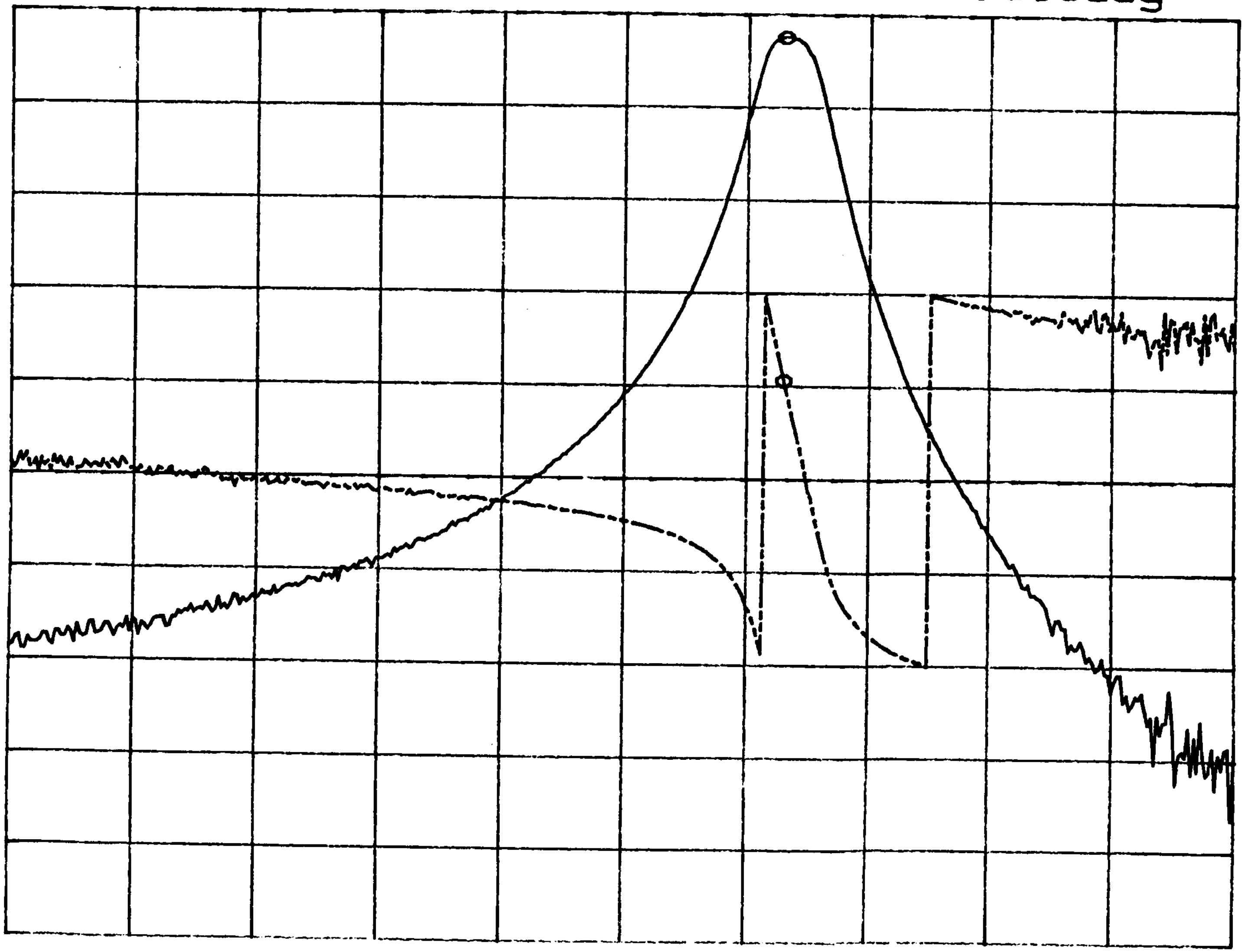
MARKER 20 041 370.400Hz
MAG(UDF) -2.615dB
MARKER 20 041 370.400Hz
PHASE(UDF) 78.512deg



K & L Model 4851
Band Pass Filter
S/N 1

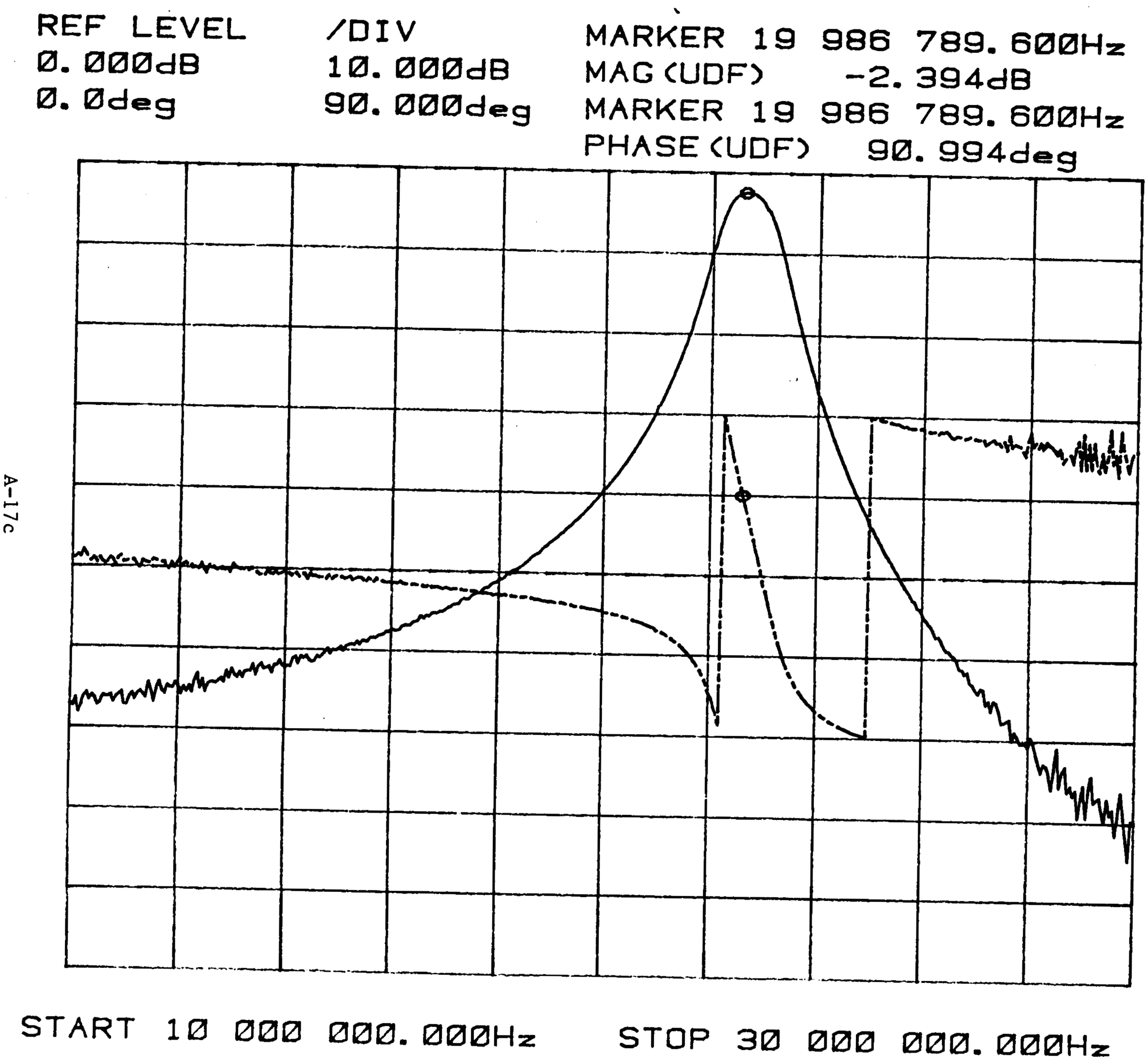
START 10 000 000.000Hz STOP 30 000 000.000Hz

REF LEVEL /DIV MARKER 19 986 789.600Hz
0.000dB 10.000dB MAG(UDF) -2.377dB
0.0deg 90.000deg MARKER 19 986 789.600Hz
PHASE(UDF) 96.516deg



K & L Model 4851
Band Pass Filter
S/N 2

START 10 000 000.000Hz STOP 30 000 000.000Hz



K & L Model 4851
Band Pass Filter
S/N 3

MITEQ
100 Davids Drive, Hauppauge, N.Y. 11788-2086
TEL: (516) 436-7400
TELEX: 6718148
FAX: 516-436-7430

PROJECT No:	P21345
MODEL No:	AFS5-010060-55-23P-32
SERIAL No:	131662
CUSTOMER:	LOCKHEED
PURCHASE ORDER No:	0200118172

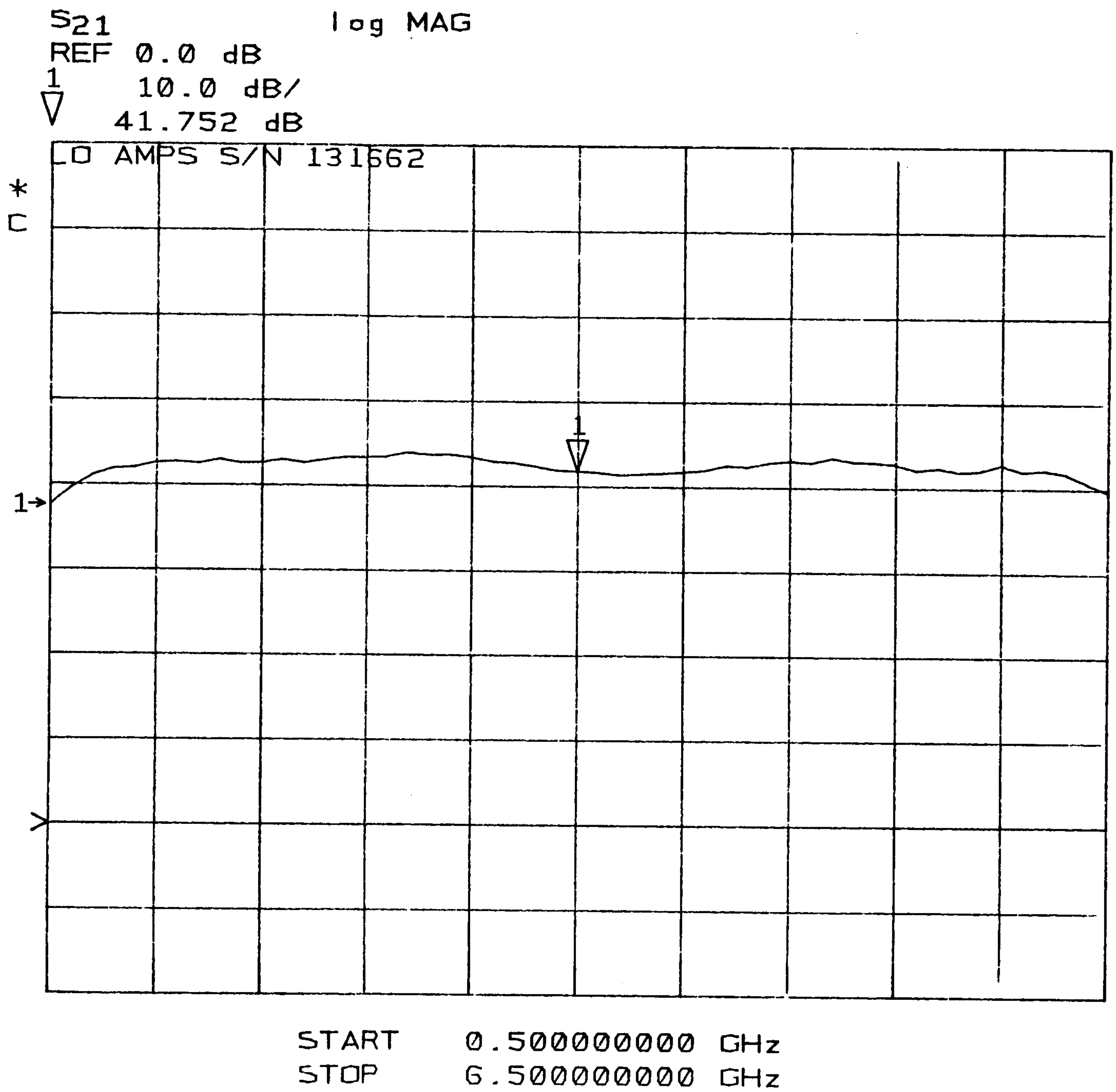
IMPORTANT - MUST USE HEAT SINK IF CASE TEMPERATURE EXCEEDS 70°C

SPECIFICATIONS: AT 23° C:			
FREQUENCY:	1.0 to 6.0 GHz	OUTPUT POWER @ 1dB GAIN COMPRESSION:	1.5-6 +23 1-1.5 +22 dBm
MIN. GAIN:	38 dB	VOLTAGE:	+15 VOLTS
MAX.GAIN FLATNESS:	+/- 1.5 dB	MEASURED CURRENT:	369 mA
MAX. VSWR INPUT:	2 :1	MAX. NOISE FIGURE:	5.5 dB
MAX. VSWR OUTPUT:	2 :1	HOUSING No:	

NOTE: TEST DATA TAKEN WITH CASE TEMP. OF 23°C

FREQUENCY (GHz)	GAIN (dB)	VSWR		NOISE FIGURE (dB)	OUTPUT POWER (dBm) (@ 1dB GAIN COMPRESSION)
		IN	OUT		
1.0	40.3	1.67	<1.22	2.19	+22
2.0	42.2	1.78	<1.22	2.26	+23
3.0	42.0	1.78	<1.22	2.34	+23
4.0	40.7	1.78	<1.22	2.36	+23
5.0	41.8	1.67	<1.22	2.42	+23
6.0	40.4	1.92	<1.22	2.56	+23

TESTED BY: Donald M.


S21
log MAG
REF 0.0 dB
1
10.0 dB/
41.752 dB
LO AMPS S/N 131662
*
C
1→
1
START 0.500000000 GHz
STOP 6.500000000 GHz

MITEQ
100 Davids Drive, Hauppauge, N.Y. 11788-2086
TEL: (516) 436-7400
TELEX: 6718148
FAX: 516-436-7430

PROJECT No:	P21345
MODEL No:	AFS5-010060-55-23P-32
SERIAL No:	131663
CUSTOMER:	LOCKHEED
PURCHASE ORDER No:	0200118172

IMPORTANT - MUST USE HEAT SINK IF CASE TEMPERATURE EXCEEDS 70°C

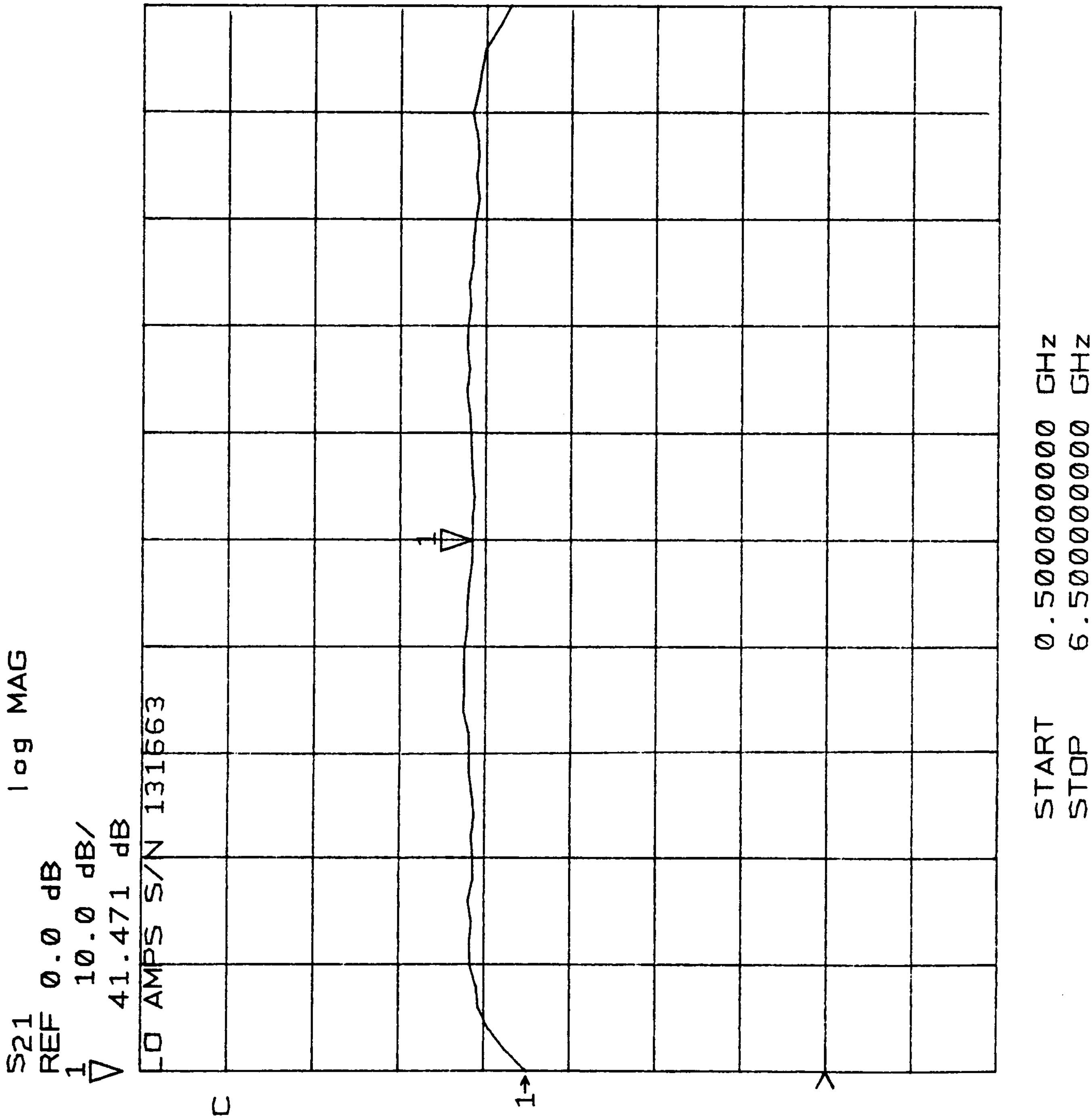
SPECIFICATIONS: AT 23° C:			
FREQUENCY:	1.0 to 6.0 GHz	OUTPUT POWER @ 1dB GAIN COMPRESSION:	1-1.5 +22 1.5-6 +23 dBm
MIN. GAIN:	38 dB	VOLTAGE:	+15 VOLTS
MAX.GAIN FLATNESS:	+/- 1.5 dB	MEASURED CURRENT:	368 mA
MAX. VSWR INPUT:	2 :1	MAX. NOISE FIGURE:	5.5 dB
MAX. VSWR OUTPUT:	2 :1	HOUSING No:	

NOTE: TEST DATA TAKEN WITH CASE TEMP. OF 23°C

FREQUENCY (GHz)	GAIN (dB)	VSWR IN	VSWR OUT	NOISE FIGURE (dB)	OUTPUT POWER (dBm) (@ 1dB GAIN COMPRESSION)
1.0	38.	1.67	<1.22	2.14	+22
2.0	40.0	1.67	<1.22	2.21	+22
3.0	40.4	1.78	<1.22	2.31	+23
4.0	40.1	1.78	<1.22	2.23	+23.5
5.0	40.4	1.67	<1.22	2.37	+23.5
6.0	38.2	1.92	<1.22	2.57	+23.5

TESTED BY: Donald Maurice
(DONALD MAURICE)

DATE: 04/28/88


S21 log MAG
REF 0.0 dB
10.0 dB/
41.471 dB
LO AMPS S/N 131563
START 0.500000000 GHz
STOP 6.500000000 GHz

MITEQ

100 Davids Drive, Hauppauge, N.Y. 11788-2086

TEL: (516) 436-7400
TELEX: 6718148
FAX: 516-436-7430

PROJECT No:	P21345
MODEL No:	AFS5-010060-55-23P-32
SERIAL No:	131664
CUSTOMER:	LOCKHEED
PURCHASE ORDER No:	0200118172

IMPORTANT - MUST USE HEAT SINK IF CASE TEMPERATURE EXCEEDS 70°C

SPECIFICATIONS: AT 23° C:			
FREQUENCY:	1.0 to 6.0 GHz	OUTPUT POWER @ 1dB GAIN COMPRESSION:	1-1.5 +22 1.5-6 +23 dBm
MIN. GAIN:	38 dB	VOLTAGE:	+15 VOLTS
MAX.GAIN FLATNESS:	+/- 1.5 dB	MEASURED CURRENT:	377 mA
MAX. VSWR INPUT:	2 :1	MAX. NOISE FIGURE:	5.5 dB
MAX. VSWR OUTPUT:	2 :1	HOUSING No:	

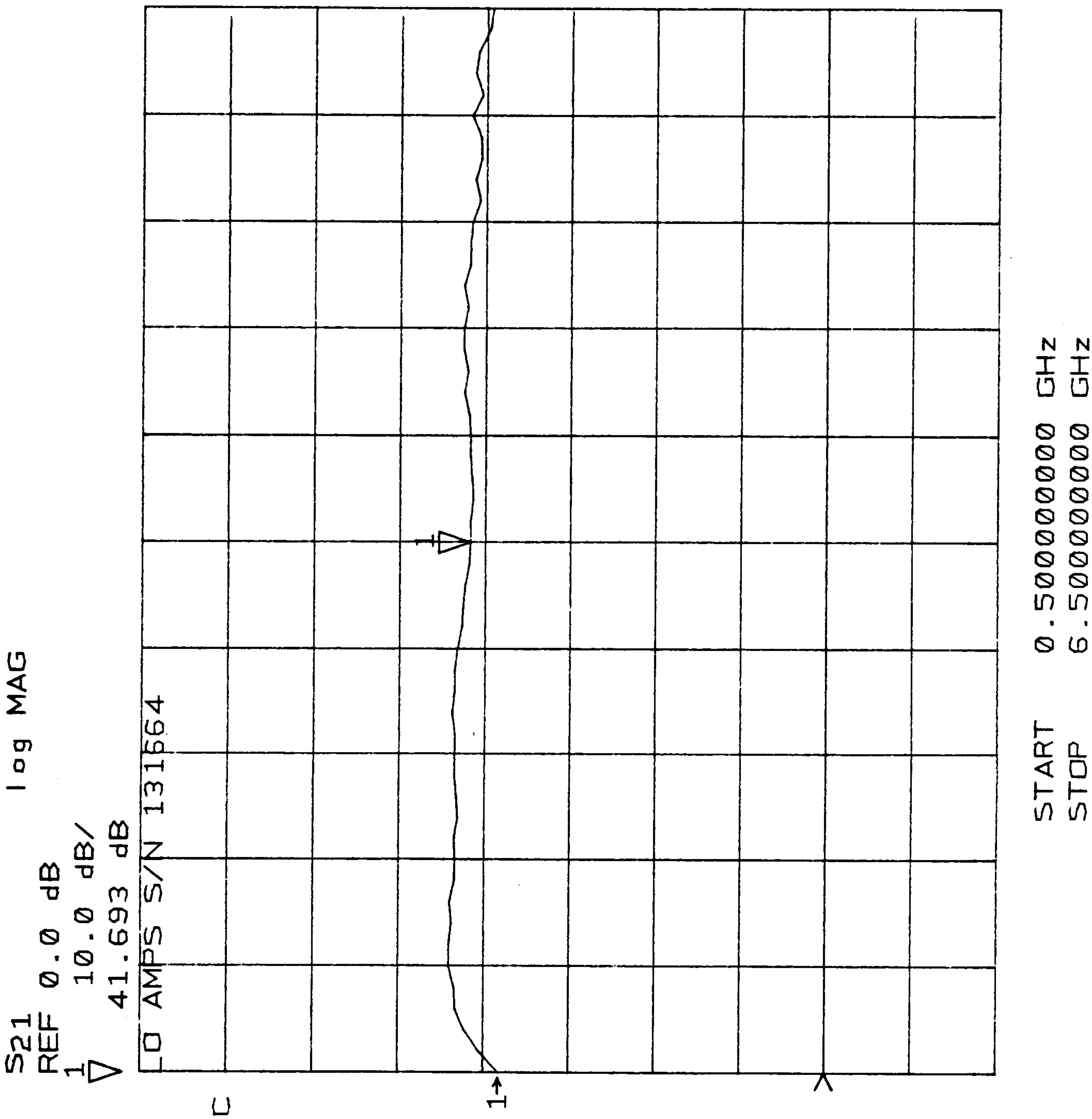
NOTE: TEST DATA TAKEN WITH CASE TEMP. OF 23°C

FREQUENCY (GHz)	GAIN (dB)	VSWR		NOISE FIGURE (dB)	OUTPUT POWER (dBm) (@ 1dB GAIN COMPRESSION)
		IN	OUT		
1.0	41.4	1.67	<1.22	2.24	+22
2.0	42.5	1.78	<1.22	2.40	+23
3.0	41.9	1.92	<1.22	2.35	+23
4.0	40.5	1.67	<1.22	2.27	+23.5
5.0	40.8	1.58	<1.22	2.34	+24
6.0	39.3	1.78	<1.22	2.73	+24

TESTED BY: Donald Maurice (DONALD MAURICE)

DATE: 04/28/88

ORIGINAL PAGE IS OF POOR QUALITY


S21
log MAG
REF 0.0 dB
1 10.0 dB/
▽ 41.693 dB
LO AMPS S/N 131564
C
1→
START 0.500000000 GHz
STOP 6.500000000 GHz

CALIBRATION DATA Cont'd

Table 1. SWR Data Uncertainties

Connector Type	Frequency Range (GHz)	SWR Data Uncertainties: Measured SWR 1.0 to 1.35	Measured SWR 1.35 to 1.86
APC 7 & Male Type N	dc to 12.4	±0.025	±0.041
	12.4 to 18.0	±0.031	±0.050
Female Type N	dc to 8.0	±0.025	±0.046
	8.0 to 12.4	±0.031	±0.063
	12.4 to 18.0	±0.042	±0.071
Male SMA	dc to 8.0	±0.031	±0.054
	8.0 to 12.4	±0.045	±0.084
	12.4 to 18.0	±0.077	±0.137
Female SMA	dc to 8.0	±0.054	±0.088
	8.0 to 12.4	±0.077	±0.132
	12.4 to 18.0	±0.122	±0.206
Male APC-3.5	dc to 10	±0.025	±0.041
	10 to 18	±0.031	±0.050
	18 to 26.5	±0.045	±0.067
Female APC-3.5	dc to 10	±0.020	±0.030
	10 to 18	±0.025	±0.037
	18 to 26.5	±0.035	±0.050

Table 2. Coaxial Attenuator Calibration Frequencies* (MHz)

100	4500	9000	13000	16750
500	5000	9500	13500	17000
1000	5500	10000	14000	17250
1500	6000	10500	14500	17500
2000	6500	11000	15000	17750
2500	7000	11500	15500	18000
3000	7500	12000	16000	(each 250 MHz
3500	8000	12400	16250	to 26.5 GHz)
4000	8500	12500	16500	

* dc to 12.4 GHz models include 26 frequencies, dc to 18 GHz models include 42 frequencies, dc to 26.5 GHz models include 67 frequencies (2 to 26.5 GHz).

Table 3. Attenuation Data Uncertainties

Attenuation (dB)	HP 8491,2,3 Attenuation Data Uncertainty (dB): 0.1 to 2.0 GHz	2 to 6 GHz	6 to 12.4 GHZ	12.4 to 18.0 GHz	18.0 to 26.5 GHz
3	±0.07	±0.06	±0.06	±0.11	±.15
6	±0.07	±0.07	±0.07	±0.11	±.15
10	±0.08	±0.07	±0.07	±0.12	±.15
20	±0.09	±0.08	±0.08	±0.13	±.15
30	±0.12	±0.11	±0.11	±0.15	±.18
40	±0.15	±0.14	±0.14	±0.21	±.25
50	±0.23	±0.23	±0.23	±0.34	n/a
60	±0.50*	±0.48*	±0.90*	±0.90*	n/a

*The uncertainties noted represent 99.7% probability values.

ORDERING INFORMATION

To order, basic model number and Option (specifies attenuation value) must be specified. Option 890 calibration data can also be ordered with the basic model number and attenuation value option.

Ordering example:

HP 8491A Option 003, Option 890

003	3 db	030	30 db
006	6 db	040	40 db
010	10 db	050	50 db
020	20 db	060	60 db

Optional calibration data

SPECIFICATIONS

Specifications describe the instruments warranted performance. Supplemental characteristics (shown in italics) are intended to provide information useful in applying the instrument by giving typical, but non-warranted, performance parameters.

FREQUENCY RANGE: HP 8491A and 8493A, dc-12.4 GHz
HP 8491B, 8493B and 8492A, dc-18 GHz
HP 8493C, dc-26.5 GHz

ATTENUATION ACCURACY:

	HP 8491A/93A	HP 8491B/93B/92A		HP 8493C	
	dc-12.4 GHz	dc-12.4	12.4-18 GHz	dc-18	18-26.5 GHz
3 dB	±0.3 dB	±0.3 dB		±0.5 dB	±1.0 dB
6 dB	±0.3 dB	±0.3 dB	±0.4 dB	±0.6 dB	
10 dB	±0.5 dB	±0.5 dB		±0.3 dB	±0.5 dB
20 dB	±0.5 dB	±0.5 dB	±1.0 dB	±0.5 dB	±0.6 dB
30 dB	±1.0 dB	±1.0 dB		±0.7 dB	±1.0 dB
	HP 8491A only	HP 8491B/92A only			
40 dB	±1.5 dB	±1.5 dB		±1.0 dB	±1.3 dB
50 dB	±1.5 dB	±1.5 dB		DNA	
60 dB	±2.0 dB	±2.0 dB		DNA	

SWR:

	HP 8491B/8493B			HP 8492A			HP 8493C		
	HP 8491A/8493A								
	dc-8GHz	8-12.4GHz	12.4-18GHz	dc-8	8-12.4	12.4-18	dc-8	8-12.4	12.4-26.5
3 dB	1.25	1.35	1.5	1.2	1.3	1.5	1.10	1.15	1.25
6 dB	1.2	1.3	1.5	1.2	1.3	1.35	1.10	1.15	1.27
10 dB	1.2	1.3	1.5	1.15	1.25	1.3	1.10	1.15	1.25
20 dB	1.2	1.3	1.5	1.15	1.25	1.3	1.10	1.15	1.25
30 dB	1.2	1.3	1.5	1.15	1.25	1.3	1.10	1.15	1.25
HP 8491A/B only			HP 8491B only						
40 dB	1.2	1.3	1.5	1.15	1.25	1.35	1.10	1.15	1.25
50 dB	1.2	1.3	1.5	1.15	1.25	1.35	DNA	DNA	DNA
60 dB	1.2	1.3	1.5	1.15	1.25	1.35	DNA	DNA	DNA

RIGINAL PAGE IS
OF POOR QUALITY

SPECIFICATIONS Cont'd

ENVIRONMENTAL

Temperature, non-operating: − 55° to + 85° C.
Temperature, operating: 0° to + 55° C.
EMC: radiated interference is within the requirements of MIL STD. 461 method REO2, VDE 0871 and CISPR Publication 11.

SUPPLEMENTAL CHARACTERISTICS

Temperature Stability: *0.0001 dB/dB/°C (all except HP 8493C)*
0.0002 dB/dB/°C (HP 8493C)
Maximum input power: 2 W avg., 100 W peak[3]
Power sensitivity: *0.001 dB/dB/W (all except HP 8943C)*
0.001 dB/W (HP 8493C)

CONNECTORS (50 Ω)		HP 8491A	HP 8493A	HP 8491B	HP 8493B	HP 8492A	HP 8493C	
		Type N[2]	SMA[1]	Type N[2]	SMA[1]	APC-7	APC-3.5	
DIMENSIONS							3,6,10,20 dB	30,40 dB
	mm	67 x 21	40 x 13	67 x 21	40 x 13	70 x 21	33.8 x 8	36.8 x 8
	Inches	2 7/16 x 13/16 Dia.	1 9/16 x 1/2 Dia.	2 7/16 x 13/16 Dia.	1 9/16 x 1/2 Dia.	2 3/4 x 13/16 Dia.	1 5/16 x 5/16 D	1 7/16 x 5/16 D
WEIGHT	Net	110 g (4oz)	30 g (1oz)	110 g (4oz)	30 g (1oz)	110 g (4oz)	8.5 g (0.3oz)	9.4 g (0.33oz)
	Shipping	220 g (8oz)	220 g (8oz)	220 g (8oz)	220 g (8oz)	220 g (8oz)	.45 kg (1 lb.)	

1 As per USASI Committee C83.2 compatible with OSM, ARM, WPM, BRM, NPM, etc.
2 Mate with MIL-C-71 or MIL-C-39012 connectors.
3 At 20° C derated to 1.3 W avg. at 55° C.

ATTENUATOR SETS

HP 11581A/11582A/11583A/11583C

A calibrated set of four HP fixed coaxial attenuators (3, 6, 10, and 20 dB) is available. Each set includes a calibration report certified traceable to the National Bureau of Standards. The reports included with the HP 11581A, 11582A and 11583A indicate the accuracy of measurement and list the attenuation and reflection coefficient at each port of the attenuator at dc, 4, 8, 12, and 18 GHz. Calibrations at other frequencies are available on request.

The HP 11583C attenuator set includes Option 890 calibration data. This option is also available for the HP 11581A, 11582A, and 11583A but must be ordered separately.

The set of four attenuators is furnished in a handsome walnut accessory case. In addition to protecting the units when not in use, the case is also a convenient storage place for the attenuators and the calibration reports.

SPECIFICATIONS

ACCURACY OF INSERTION LOSS MEASUREMENTS (S_{21}, S_{12}):

Attenuator Sets HP 11581A/11582A
- DC ± 0.01 dB
- 4 - 18 GHz ± 0.097 dB

Attenuator Set HP 11583A
- DC ± 0.01 dB
- 4 - 18 GHz ± 0.085 dB

ACCURACY OF RELECTION COEFFICIENT MEASUREMENTS (S_{11}, S_{22}):

Attenuator Sets HP 11581A/11582A
- 4 - 18 GHz $\Delta\Gamma_L < \pm 0.035$

Attenuator Set HP 11583A
- 4 - 18 GHz $\Delta\Gamma_L < \pm 0.030$

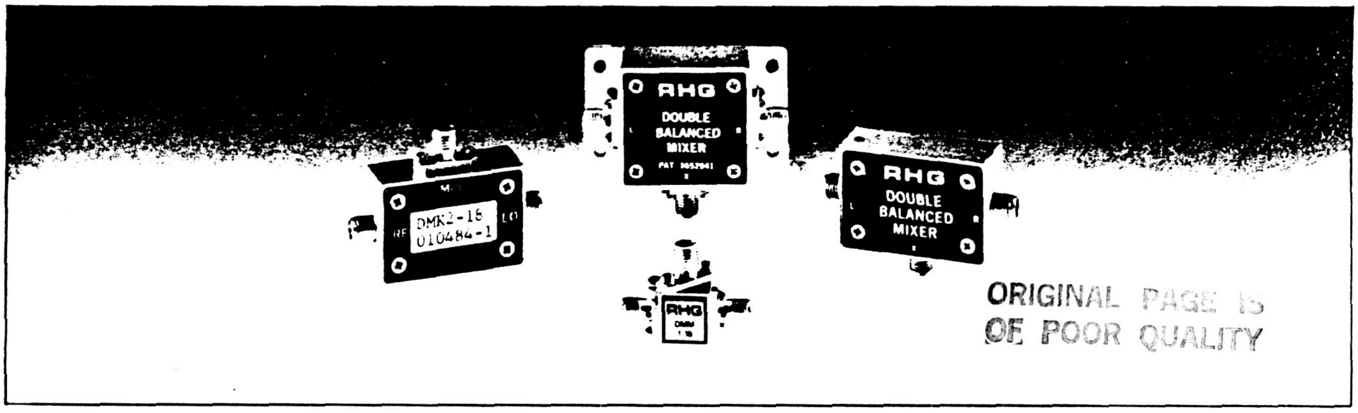
Ordering Information
HP 11581A (3, 6, 10, 20 dB values HP 8491A)
HP 11582A (3, 6, 10, 20 dB values HP 8491B)
HP 11583A (3, 6, 10, 20 dB values HP 8492A)
HP 11583C (3, 6, 10, 20 dB values HP 8493C)

For more information, call your local HP sales office listed in the telephone directory white pages. Ask for the Electronic Instrument Department, or write to Hewlett-Packard: U.S.A. — P.O. Box 10301, Palo Alto, CA 94303-0890. Europe — P.O. Box 999, 1180 AZ Amstelveen, The Netherlands. Canada — 6877 Goreway Drive, Mississauga, L4V 1M8, Ontario. Japan — Yokogawa-Hewlett-Packard Ltd., 3-29-21, Takaido-Higashi, Suginami-Ku, Tokyo 168. Elsewhere in the world, write to Hewlett-Packard Intercontinental, 3495 Deer Creek Road, Palo Alto, CA 94304.

5953-6475 DATA SUBJECT TO CHANGE Printed in U.S.A.

Double Balanced MIC Mixers

■ Biphase Modulator ■ Low Cost ■ Octave and Multioctave



Three series of double balanced mixers with conventional IF's are offered on this page. The basic "DM" series, which utilizes a rugged cast aluminum housing, is specified as a down converter for octave and multioctave frequency ranges. The DMS1-26A is a high performance, low conversion loss, multioctave model specified as a down converter in the 1 to 26 GHz frequency range.

The models described above function well as up converters, third harmonic mixers, and phase detectors. The DMK2-18 is a special version of the DMS1-26A specified as a wideband biphase modulator. The DMK2-18 uses a special diode quad with diodes selected for switching rather than mixing capability and special IF decoupling networks to produce a high performance modulator covering 2 to 18 GHz (usable 1 to 26 GHz).

SPECIFICATIONS:	**DMS1-26A**
RF/LO Range:	1 to 26 GHz
IF Range:	DC-500 MHz
Conversion Loss:	1 to 2 GHz — 8 dB typical, 9.5 dB max. 2 to 18 GHz — 6.0 dB typical, 7.0 dB max. 18 to 26 GHz — 6.5 dB typical, 8.0 dB max.
RF VSWR:	4 to 18 GHz, 2:1; 1 to 4 & 18 to 26 GHz, ~4:1
LO VSWR:	2.5:1 typical
Price:	**$650**

SPECIFICATIONS: DM SERIES
OCTAVE MODELS

Model No.	Frequency (GHz)	Typical Conv. Loss (dB)	Maximum Conv. Loss (dB)	Price
DM1-2A	1.0 to 2.0	5.5	7.0	$275
DM2-4A	2.0 to 4.0	5.5	7.0	275
DM4-8A	4.0 to 8.0	5.5	7.0	310
DM8-12A	8.0 to 12.0	6.0	7.5	345
DM12-18A	12.0 to 18.0	7.0	8.5	435

WIDE-BAND BIPHASE MODULATOR

SPECIFICATIONS:	**DMK2-18**
Frequency Range:	2 to 18 GHz
Carrier Suppression:	20 dB
Switching Speed:	3 nsec max.
Phase Balance:	±10° (from 180°)
Amplitude Balance:	±0.75 dB
Insertion Loss:	4 dB
DC Current Required:	±10 mA
Price:	**$765**

SPECIFICATIONS: DM SERIES
MULTIOCTAVE MODELS

Model No.	Frequency (GHz)	Typical Conv. Loss (dB)	Maximum Conv. Loss (dB)	Price
DM1-4A	1.0 to 4.0	5.5	7.0	$310
DM1-8A	1.0 to 8.0	5.5	7.0	325
DM1-12A	1.0 to 12.0	6.0	7.5	385
DM1-18A	1.0 to 18.0	7.0	8.5	495

NOTES: **(When not otherwise specified)**

1. LO Injection: +7 dBm to +10 dBm
2. RF/LO VSWR: 2.5:1 (typ)
3. LO/RF Isolation: 20 dB min.
4. IF Response: DC to 500 MHz.
5. Weight: DMS — 40 g (1.4 oz) max.
   DMK — 10 g (0.4 oz) max.
6. For outline drawings: See page 56.

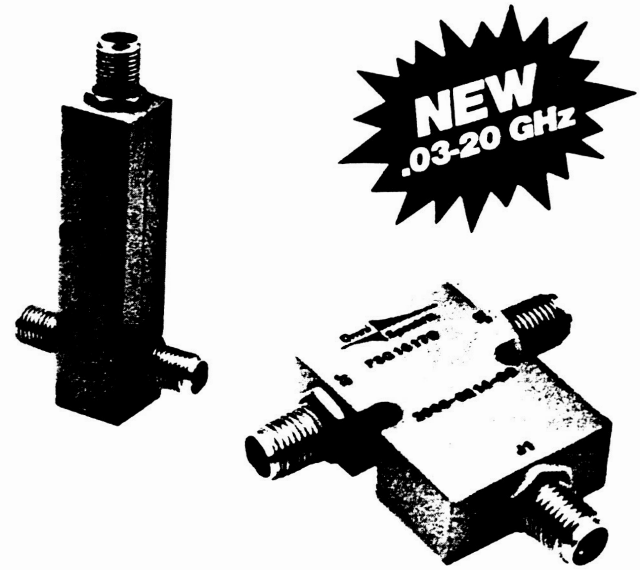
OPTIONS: **(Apply for DM and DMS series only, as noted)**

1. For improved intermodulation performance use LO injection level of +13 to +16 dBm. Add suffix "H", $80 additional.
2. Low corner noise diodes (DM series only): Reduce 1/f noise for "zero IF" applications. Add suffix "B", $65 additional.

ORIGINAL PAGE IS
OF POOR QUALITY

Omni Spectra

POWER DIVIDERS
TWO-WAY • WILKINSON
ISOLATED • ULTRA BROADBAND



- Excellent Amplitude and Phase Balance
- High Isolation Between Output Ports
- Wideband Frequency Coverage
- Low Insertion Loss
- Low VSWR
- Power: 3.0 to 10 Watts Input Maximum, with Matched Terminations
- Meets MIL-E-5400 and MIL-E-16400 Environments

These two-way in-phase stripline power dividers demonstrate excellent performance across a broad frequency spectrum. The multi-octave power dividers exhibit high isolation, low VSWR and insertion loss, excellent amplitude balance and phase balance, all combined in a small package.

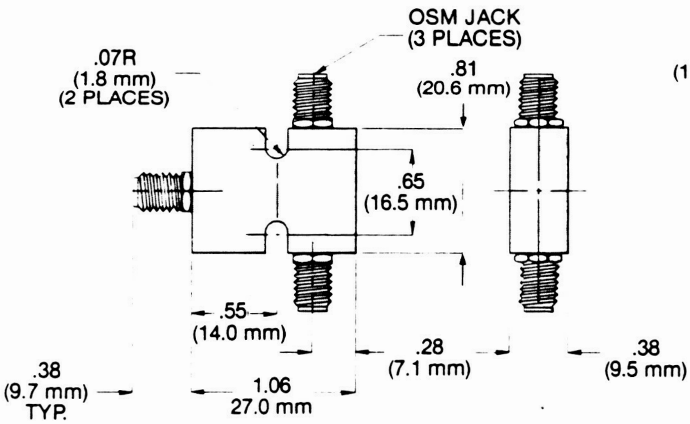


FIG. 1

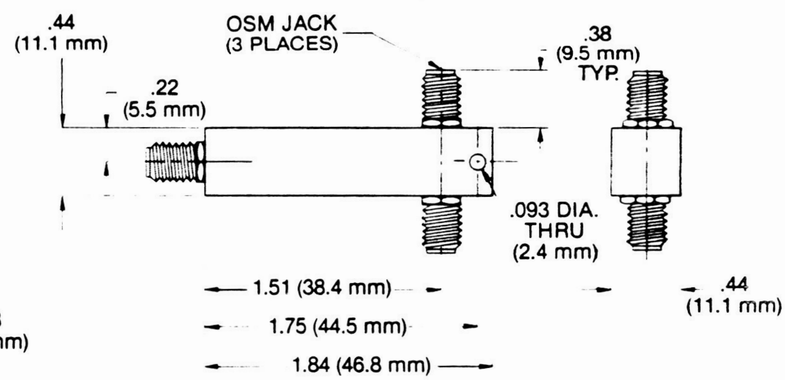


FIG. 2

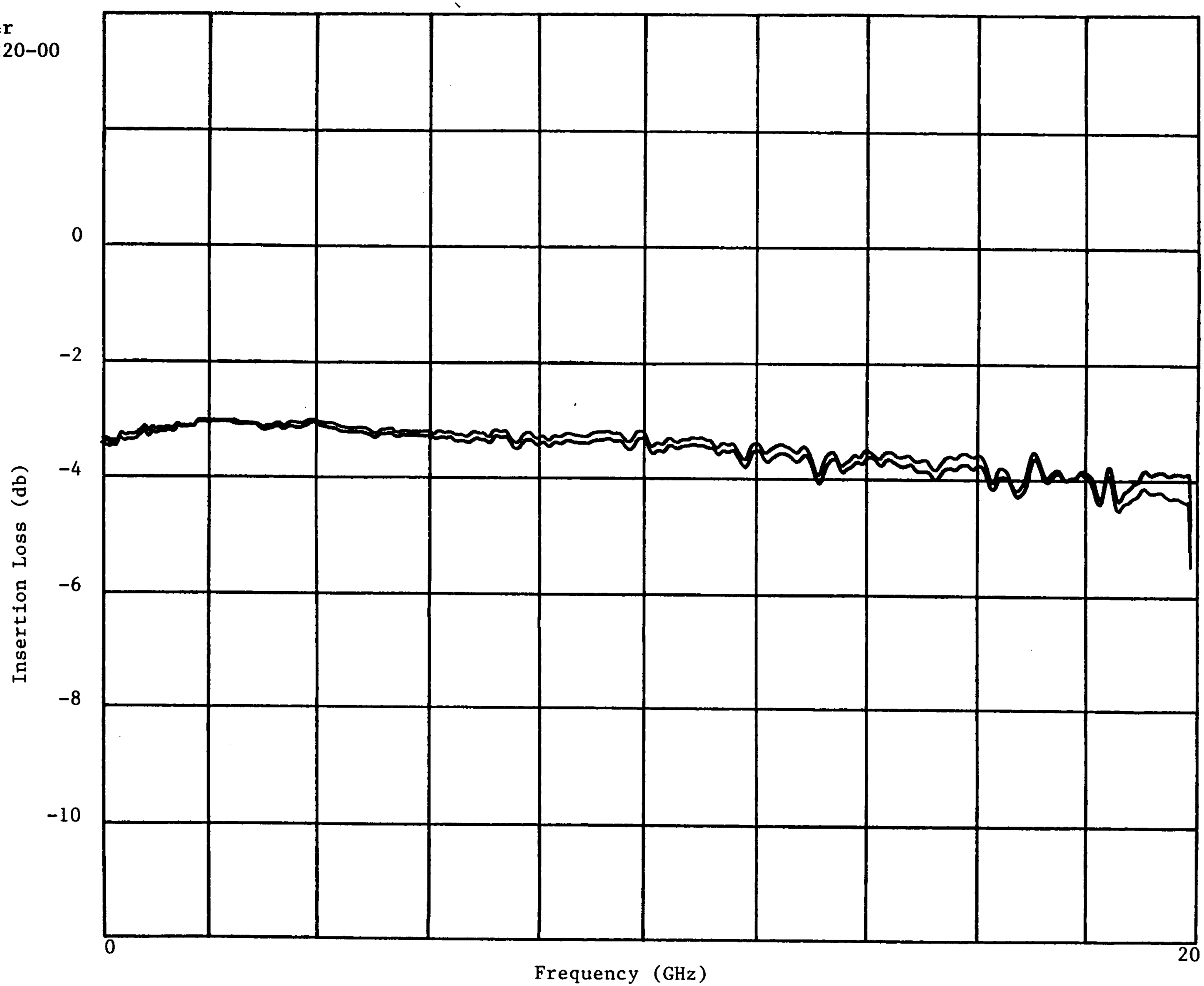
NOTE: All dimensions are ±.020, except mounting hole diameters (±.005) and mounting hole location (±.010).

SPECIFICATIONS

PART NO.	FIG.	FREQUENCY RANGE (GHz)	VSWR (max.)	ISOLATION dB (Min.)	INSERTION LOSS dB (max.)	OUTPUT UNBALANCE AMP. (dB)	OUTPUT UNBALANCE PHASE (deg.)	MAXIMUM INPUT POWER* (watts)	WEIGHT oz.	WEIGHT g
2089-6214-00	1	4.0-18.0	1.50**	18	0.9	0.3	8.0	3.0	.66	19.0
2089-6220-00	2	.03-.50	2.00	3	0.8	0.5	1.0	10.0	.72	20.5
		.50-1.0	1.93	6	0.7	0.5	1.0	10.0		
		1.0-2.0	1.70	10	0.5	0.2	1.0	10.0		
		2.0-4.0	1.50	20	0.5	0.2	1.0	10.0		
		4.0-8.0	1.50	17	0.5	0.2	1.5	10.0		
		8.0-15.0	1.50	17	0.75	0.3	2.0	10.0		
		15.0-17.0	1.80	17	0.75	0.3	3.0	10.0		
		17.0-18.0	1.80	17	1.0	0.4	4.0	10.0		
		18.0-20.0	2.00	10	1.0	0.4	5.0	10.0		

*Maximum input power with output loads of VSWR 2.0:1.
Derate to 10% of listed value when arbitrarily terminated.
**1.7:1 from 4.0 to 5.0 GHz.

Power Splitter
Model 2089-6220-00
S/N 163



ORIGINAL PAGE IS
OF POOR QUALITY

MOTOROLA

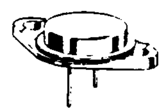
MC7800 Series

THREE-TERMINAL POSITIVE FIXED VOLTAGE REGULATORS

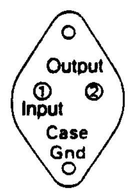
3-TERMINAL POSITIVE VOLTAGE REGULATORS

These voltage regulators are monolithic integrated circuits designed as fixed-voltage regulators for a wide variety of applications including local, on-card regulation. These regulators employ internal current limiting, thermal shutdown, and safe-area compensation. With adequate heatsinking they can deliver output currents in excess of 1.0 ampere. Although designed primarily as a fixed voltage regulator, these devices can be used with external components to obtain adjustable voltages and currents.

- Output Current in Excess of 1.0 Ampere
- No External Components Required
- Internal Thermal Overload Protection
- Internal Short-Circuit Current Limiting
- Output Transistor Safe-Area Compensation
- Output Voltage Offered in 2% and 4% Tolerance

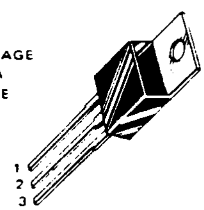


K SUFFIX
METAL PACKAGE
CASE 1
(TO-3 TYPE)



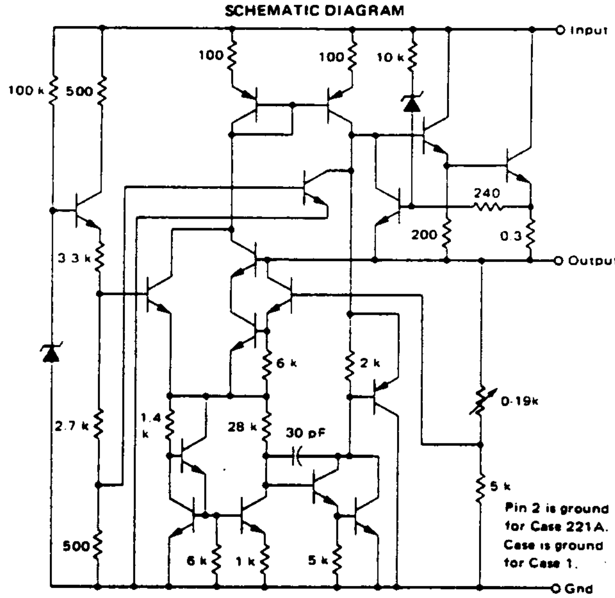
(Bottom View)

T SUFFIX
PLASTIC PACKAGE
CASE 221A
TO-220 TYPE



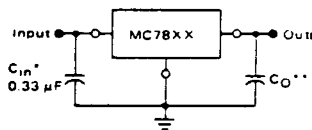
Pin 1. Input
2. Ground
3. Output

SCHEMATIC DIAGRAM



Pin 2 is ground for Case 221A. Case is ground for Case 1.

STANDARD APPLICATION



A common ground is required between the input and the output voltages. The input voltage must remain typically 2.0 V above the output voltage even during the low point on the input ripple voltage.

XX = these two digits of the type number indicate voltage.

* = C_{in} is required if regulator is located an appreciable distance from power supply filter.

** = C_O is not needed for stability; however, it does improve transient response.

XX indicates nominal voltage

ORDERING INFORMATION

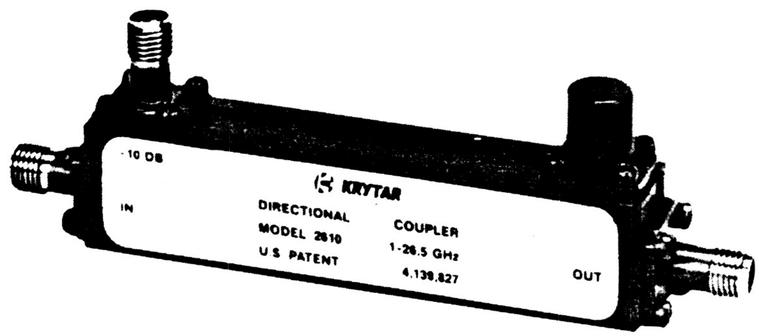
Device	Output Voltage Tolerance	Temperature Range	Package
MC78XXK MC78XXAK	4% 2%	-55 to +150°C	Metal Power
MC78XXBK	4%	-40 to +125°C	
MC78XXCK MC78XXACK	4% 2%	0 to +125°C	
MC78XXCT MC78XXACT	4% 2%		Plastic Power
MC78XXBT	4%	-40 to +125°C	

TYPE NO./VOLTAGE			
MC7805	5.0 Volts	MC7815	15 Volts
MC7806	6.0 Volts	MC7818	18 Volts
MC7808	8.0 Volts	MC7824	24 Volts
MC7812	12 Volts		

KRYTAR

MODEL 2610

1-26.5 GHz DIRECTIONAL COUPLER



SPECIFICATIONS

FREQUENCY RANGE 1-26.5 GHz

COUPLING (with respect to output)
- Nominal: 10±1 dB
- Frequency Sensitivity: ±.6 dB, 1-12.4 GHz; ±.8 dB, 1-26.5 GHz

DIRECTIVITY >14 dB, 1-12.4 GHz; >12 dB, 12.4-26.5 GHz

MAXIMUM VSWR (Any port) 1.35, 1-12.4 GHz; 1.50, 12.4-26.5 GHz

INSERTION LOSS (Includes coupled power) <1.1 dB, 1-12.4 GHz; <1.6 dB, 12.4-26.5 GHz

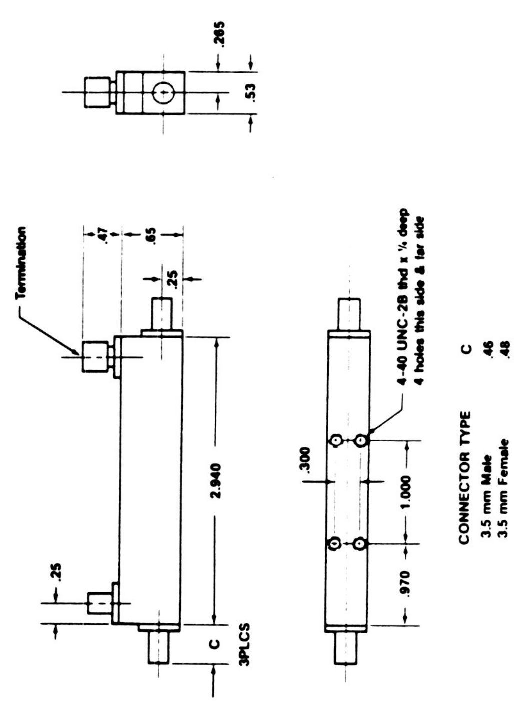
POWER RATING (input)
- Average: 20W
- Peak: 3KW

CONNECTORS 3.5mm Male or Female

WEIGHT (ounces) 2.1

PRICE $825

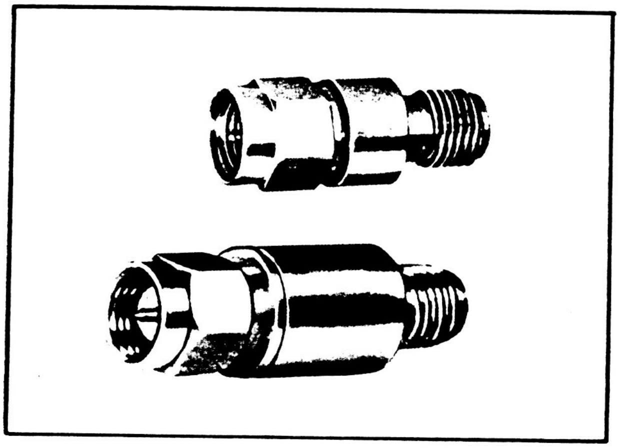
DIMENSIONS



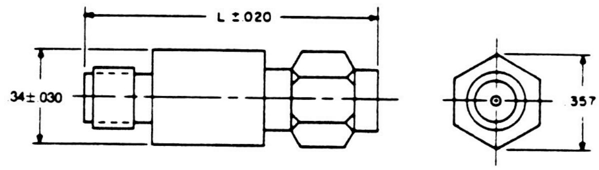
KRYTAR 1292 Anvilwood Court • Sunnyvale, California 94086 • (408) 734-5999 Issued 3/83

ORIGINAL PAGE IS
OF POOR QUALITY

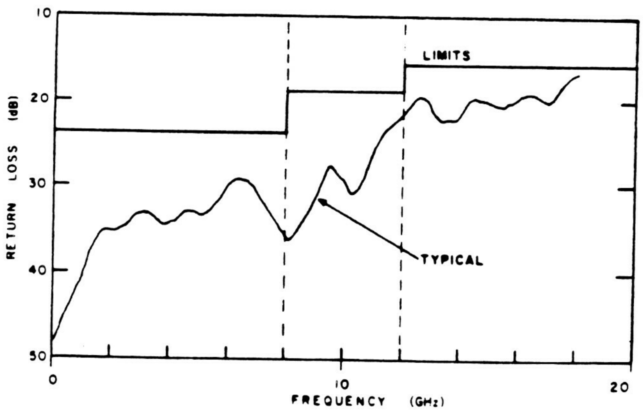
SMA COAXIAL attenuators
1 TO 30 dB • DC TO 18 GHz



A600M SERIES



Series	"L" Dimension	Attenuation Increments
A600M	0.86	1 - 10 dB
	1.02	11 - 20 dB



* standard values only others slightly higher.

HIGH PERFORMANCE

(SPACE QUALIFIED) (MEETS MIL-A-3933E)

GENERAL SPECIFICATIONS

Frequency Range:	DC to 18 GHz
Impedance:	50 ohms
Attenuation Stability:	0.0001 dB/dB/°C
Attenuation Accuracy:	1-10 dB - ±0.3 dB 11-20 dB - ±0.5 dB 21-30 dB - ±1.0 dB
VSWR (Max.):	DC - 8 GHz - 1.15:1 8 - 12 GHz - 1.25:1 12 - 18 GHz - 1.35:1
Input Power:	2 watt @ 25°C, derate to 0.5 watts @ 125°C; 200 watts peak
Operating Temperature:	-65°C + 125°C
Housing:	Stainless Steel, Passivated per QQ-P-35
Connector:	SMA, Stainless Steel per MIL-C-39012
Center Conductor:	Beryllium Copper, Gold Plated per MIL-G-45204

ORDERING INFORMATION

The Coaxial Attenuators listed are available in 1 dB increments from 1 through 30 dB. When ordering, to specify the correct part number for the desired attenuation value, select from the two basic series and add the attenuation value desired to the basic series designation.

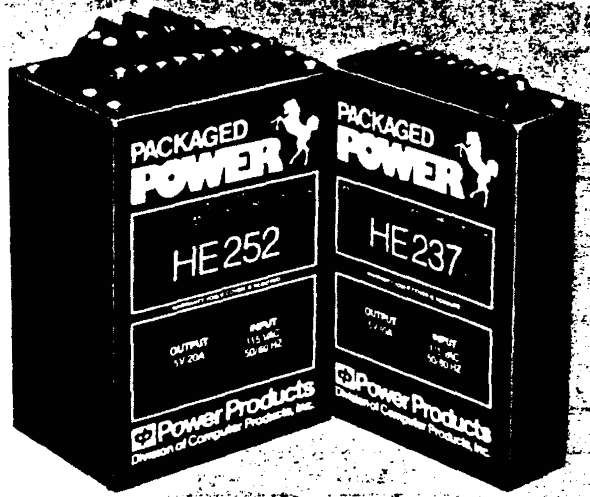
EXAMPLE: A6 10 M

A6 — Basic Series
10 — Desired dB Value
M — (SMA)

KDI ELECTRONICS, INC.
Pyrofilm & Engelmann Divisions
60 South Jefferson Road, Whippany, N.J. 07981 • TEL (201) 887-8100 • TWX (710) 986-8220 • FAX (201) 887-4645

ORIGINAL PAGE IS
OF POOR QUALITY

36-100W AC/



HE200 SERIES

- **75% Efficiency**
- **Wide Input Range**
- **Low Ripple and Noise**
- **OVP on 5-Volt Models**

The HE200 series switching power supplies consists of ten models with both single and dual output voltages. These models employ 25 KHz, pulse-width modulated switching circuitry to achieve 75% efficiency at up to 100 Watts output power. The output voltages are adjustable and line regulation is from .02% to 0.1% with load regulation from .05% to 0.1%. Output ripple and noise is held to 10 mV to 20 mV peak to peak, maximum. All outputs are short circuit protected for an indefinite time period. In addition, the 5-volt outputs are over-voltage protected by means of a crowbar circuit and they have a remote sensing feature which compensates for line drops up to 0.3 volt. There are both U.S. and international versions of each model with wide input voltage ranges of 90 to 130 VAC or 180 to 260 VAC.

ELECTRICAL SPECIFICATIONS

All Specifications Typical at Nominal Line, Full Load, and 25°C Unless Otherwise Noted.

INPUT SPECIFICATIONS

Input Voltage Range, Standard ..... 90 VAC to 130 VAC
"E" Suffix .... 180 VAC to 260 VAC
Frequency .... 47 to 450 Hz.

OUTPUT SPECIFICATIONS

Voltage Accuracy .... Adjustable
Voltage Tracking, Dual Outputs (HE215,215E) .. ±1.5%
Temperature Coefficient .... ±0.02%/°C, max.
Tracking Temp. Coefficient
Dual Outputs (HE215,215E) .. ±.005%/°C, max.
Warm-Up Drift .... 15 mV
Transient Recovery Time
5-Volt Models, 50% Load to Full Load
HE237, to 0.2% of Final Value ...... 300 μ sec.
HE252, to 0.4% of Final Value ...... 300 μ sec.
All Other Models, No Load to Full Load
HE212, to 0.5% of Final Value .... 30 μ sec.
HE215, 224, to 0.2% of Final Value ... 30 μ sec.
Hold-Up Time .... 20 msec.
Short Circuit Protection .... Continuous
Over Voltage Protection,
5V Outputs (HE237,252) .... Crowbar
Remote Sensing[1],
5V Outputs (HE237,252) ...... Up to 0.3V Drop

GENERAL SPECIFICATIONS

Efficiency 75%
Isolation Voltage .... 900 VRMS
Isolation Resistance .... 50 megohms
Switching Frequency .... 25 KHz

ENVIRONMENTAL SPECIFICATIONS

Operating Temperature Range .... 0° to +71°C
Derating, 50° to 71°C .... 2.5%/°C
Storage Temperature Range .... −25°C to +85°C
Humidity .... 20% to 95% R.H. (non-condensing)
Cooling .... Free-Air Convection

PHYSICAL SPECIFICATIONS

Dimensions, Case E .... 6.5 x 4.5 x 3.19 inches (165 x 114 x 81 mm)
Case D .... 6.5 x 4.5 x 1.50 inches (165 x 114 x 38 mm)
Weight, Case E .... 3.25 lbs. (1456 g.)
Case D .... 1.7 lbs. (762 g.)
Case Material .... Black Anodized Aluminum

NOTE:
(1) For lines up to 60 feet. Sense leads should be twisted and a large capacitor added at sense point for switching loads.

ADJUSTMENT RANGE

MODEL	OUTPUT RANGE
HE237	4.5 to 5.3V
HE252	4.5 to 5.3V
HE212	12 to 15.5V
HE215	±12 to ±15.5V
HE224	24 to 31V

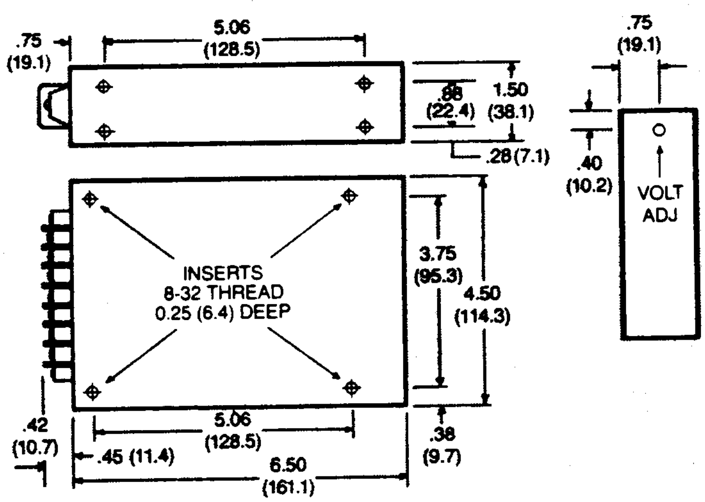
Power Products

TWO-YEAR WARRANTY

DC Cased Switchers

PACKAGED POWER™

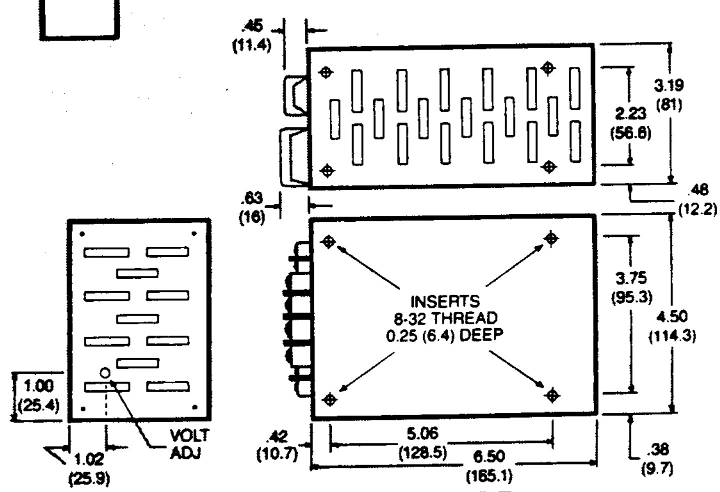
OUTPUT VOLTAGE	OUTPUT CURRENT	OVP	* REGULATION LINE	REGULATION LOAD	RIPPLE AND NOISE, MAX.	INPUT VOLTAGE	MODEL NUMBER	CASE
SINGLE OUTPUT								
5 VDC	10 A	✓	±0.1%	±0.1%	25 mV P-P (5 mV RMS)	115 VAC	HE237	D
5 VDC	10A	✓	±0.1%	±0.1%	25 mV P-P (5 mV RMS)	230 VAC	HE237E	D
5 VDC	20A	✓	±0.1%	±0.1%	50 mV P-P (13 mV RMS)	115 VAC	HE252	E
5 VDC	20A	✓	±0.1%	±0.1%	50 mV P-P (13 mV RMS)	230 VAC	HE252E	E
12 VDC to 15 VDC	3A		±.02%	±0.1%	20 mV P-P (2 mV RMS)	115 VAC	HE212	D
12 VDC to 15 VDC	3A		±.02%	±0.1%	20 mV P-P (2 mV RMS)	230 VAC	HE212E	D
24 VDC to 30 VDC	1.5 A		±.02%	±0.1%	20 mV P-P (2 mV RMS)	115 VAC	HE224	D
24 VDC 30 VDC	1.5 A		±.02%	±0.1%	20 mV P-P (2 mV RMS)	230 VAC	HE224E	D
DUAL OUTPUT								
± 12 VDC to ±15 VDC	±1.5 A		±.02%	±.05%	10 mV P-P (1.0 mV RMS)	115 VAC	HE215	D
±12 VDC to ±15 VDC	±1.5 A		±.02%	±.05%	10 mV P-P (1.0 mV RMS)	230 VAC	HE215E	D



CASE D

ALL DIMENSIONS IN INCHES (MM)

ORIGINAL PAGE IS OF POOR QUALITY



CASE E

PM300 SERIES

CHASSIS-MOUNTABLE SINGLES, DUALS & TRIPLES

- Terminal Strip Connections
- Split-Bobbin Wound
- UL Recognized
- CSA Certified

These popular chassis-mountable linear power modules feature 16 single, dual and triple output models. This series is designed for special applications where mounting on a housing or metal chassis is required. Input/output connections are made to screw terminals on a barrier terminal strip and mounting is convenient by means of four threaded inserts in the bottom of each module. Most models are UL recognized and CSA certified. For maximum safety, all power transformers are split-bobbin wound, rather than layer wound, to give total isolation with low coupling capacitance between primary and secondary. Conservative design and rating of these power modules results in reliable operation and long life. Overvoltage crowbar protection is standard on all 5V outputs for protection of logic circuitry. Standard input voltage is 115 VAC at 50 to 400 Hz; other optional inputs are 100, 220, and 240 VAC. Input/output isolation voltage is 2500 VAC and output current limiting short circuit protection is standard.

Recognized by Underwriters' Laboratories, Inc. Certified by Canadian Standards Association

Power Products

2.5-15 Watt AC

ELECTRICAL SPECIFICATIONS

All Specifications Typical at Nominal Line, Full Load, and 25°C Unless Otherwise Noted.

INPUT SPECIFICATIONS

Input Voltage Range,
Standard Models .... 105 VAC to 125 VAC
Other Models .... See Table
Frequency .... 50 to 400 Hz
Derating at 400 Hz .... Consult Factory

OUTPUT SPECIFICATIONS

Voltage Accuracy .... ±2.0%, max.
Temperature Coefficient .... ±0.02%/°C
Short-Circuit Protection .... Short Term
Over-Voltage Crowbar, 5V Outputs .... 6.2V, nom.

GENERAL SPECIFICATIONS

Isolation Voltage .... 2500 VRMS
Isolation Capacitance .... 50 pF.
Isolation Resistance .... 50 megohms

ENVIRONMENTAL SPECIFICATIONS

Operating Temperature Range .... −25°C to +71°C
Derating, 50°C to 71°C .... 2.5%/°C
Storage Temperature Range .... −25°C to +85°C
Humidity .... 20% to 95% R.H. (non-condensing)
Cooling .... Free-Air Convection

PHYSICAL SPECIFICATIONS

Dimensions, Case C1 .... 4.0 x 2.7 x 1.45 inches (102 x 69 x 37 mm)
Case C2 .... 4.0 x 2.7 x 2.00 inches (102 x 69 x 51 mm)
Weight, Case C1 .... 1.25 lbs. (567 g.)
Case C2 .... 1.80 lbs. (816 g.)
Case Material .... Non-Conductive Black Plastic

NOTE: All models are available with four optional input voltage ranges designated by the suffixes shown in table.

When ordering, specify the complete model number followed by the appropriate input voltage designation, if any. For example, PM342, PM342J, PM342D, etc.

INPUT VOLTAGE	SUFFIX
115±10VAC	(NONE)
100±10VAC	J
220±20VAC	D
240±20VAC	K

ORIGINAL PAGE IS OF POOR QUALITY

TWO-YEAR WARRANTY

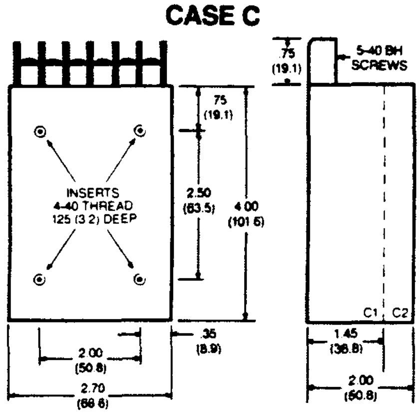
DC Linear Modules

PACKAGED POWER™

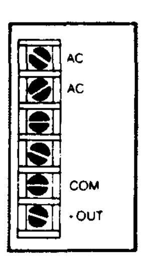
OUTPUT VOLTAGE	OUTPUT CURRENT	OVP	REGULATION LINE	REGULATION LOAD	RIPPLE AND NOISE	UL	CSA[1]	MODEL NUMBER	ALT PIN OUT	CASE
SINGLE OUTPUT										
5 VDC	500 mA	✓	±.05%	±0.1%	2.0 mV RMS	✓	✓	PM334		C1
5 VDC	1000 mA	✓	±.05%	±.15%	2.0 mV RMS	✓	✓	PM342		C1
5 VDC	2000 mA	✓	±.05%	±.15%	2.0 mV RMS	✓		PM345		C2
12 VDC	240 mA		±.05%	±.05%	1.0 mV RMS	✓		PM315		C1
12 VDC	400 mA		±.05%	±.05%	1.0 mV RMS	✓		PM316		C2
12 VDC	600 mA		±.05%	±.05%	1.0 mV RMS	✓		PM317		C2
15 VDC	200 mA		±.05%	±.05%	1.0 mV RMS	✓		PM354		C1
15 VDC	350 mA		±.05%	±.05%	1.0 mV RMS	✓		PM355		C2
15 VDC	500 mA		±.05%	±.05%	1.0 mV RMS	✓		PM356		C2
24 VDC	100 mA		±.05%	±.05%	1.0 mV RMS	✓		PM366		C1
24 VDC	200 mA		±.05%	±.05%	1.0 mV RMS	✓		PM367		C1
24 VDC	400 mA		±.05%	±.05%	1.0 mV RMS	✓		PM368		C2
DUAL OUTPUT										
±12 VDC	±120 mA		±.05%	±.05%	1.0 mV RMS	✓	✓	PM336		C1
±12 VDC	±240 mA		±.05%	±.05%	1.0 mV RMS	✓	✓	PM337		C1
±12 VDC	±400 mA		±.05%	±.05%	1.0 mV RMS	✓	✓	PM397		C2
±15 VDC	±100 mA		±.05%	±.05%	1.0 mV RMS	✓	✓	PM302		C1
±15 VDC	±200 mA		±.05%	±.05%	1.0 mV RMS	✓	✓	PM365		C1
±15 VDC	±350 mA		±.05%	±.05%	1.0 mV RMS	✓	✓	PM301		C2
±15 VDC	±500 mA		±.05%	±.05%	1.0 mV RMS	✓	✓	PM396		C2
TRIPLE OUTPUT										
5V/±12 VDC	300/±180 mA	✓	±.05%	±0.1%/±.05%	1.0 mV RMS	✓	✓	PM395		C1
5V/±12 VDC	500/±120 mA	✓	±.05%	±0.1%/±.05%	1.0 mV RMS	✓	✓	PM391		C1
5V/±12 VDC	1000/±150 mA	✓	±.02%	±0.1%/±.02%	1.0/0.5 mV RMS		✓	PM392		C2
5V/±15 VDC	300/±150 mA	✓	±.05%	±0.1%/±.05%	1.0 mV RMS	✓	✓	PM394		C1
5V/±15 VDC	500/±100 mA	✓	±.05%	±0.1%/±.05%	1.0 mV RMS	✓	✓	PM390		C1
5V/±15 VDC	1000/±150 mA	✓	±.02%	±0.1%/±.02%	1.0/0.5 mV RMS		✓	PM393		C2

NOTE: (1) All Models CSA Certified (✓) or Pending.

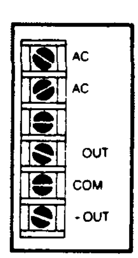
CASE C



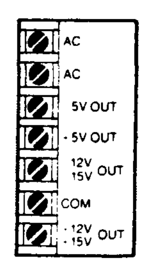
SINGLE OUTPUT MODELS



DUAL OUTPUT MODELS



TRIPLE OUTPUT MODELS



ALL DIMENSIONS IN INCHES (MM)

ORIGINAL PAGE IS OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

6

MINIATURIZED REGULATED
Terminal Strip Connections

ALL MODELS U.L. RECOGNIZED

Although small in size, these mini-modules offer high performance at modest prices. All models, with series regulated outputs ranging from 1 to 75 volts and as high as 2.5 amps, may be mounted in an area only 3.5" x 2.5". Dual output models are available with the ratings commonly required for driving op amps and other balanced loads. Terminal strip input/output connections eliminate all need for sockets or soldering. Short circuit protection, encapsulated construction, and conservative design assure long term reliability.

STANDARD FEATURES
- May be used in series
- No derating or heat sinking required
- Short circuit protected
- Small, lightweight

SPECIFICATIONS
Input Voltage: 105–125 VAC, 47 to 420 Hz, single phase.

Output Specifications: See tables.

Output Voltage Trim Adjustment: Outputs factory preset to ±2% (1 to 9 volt models) or ±1% (10 to 75 volt models) of nominal output voltage. Single output models may be trimmed to the nominal voltage rating with an external trim resistor.

Polarity: Either positive or negative terminal of a single output module may be grounded. Dual output modules have a positive/common/negative output terminal configuration.

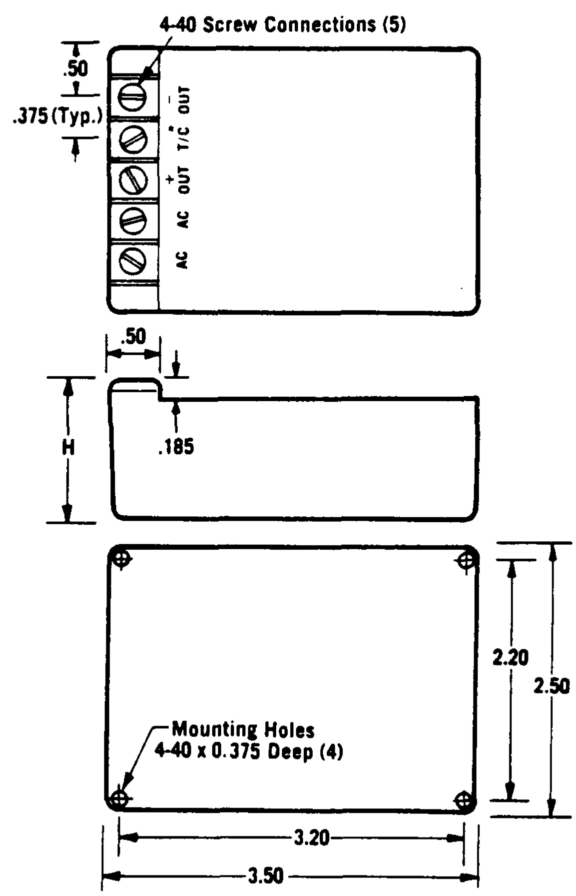
Ambient Operating Temperature: −20 to +71°C. (Model 5EB150, 0 to +71°C.) No derating required.

Storage Temperature: −55 to +85°C.

Temperature Coefficient: From 9 to 75 volts, approximate TC is .015%/°C; 1 to 8 volts, .03%/°C.

Impedance: 0.07 ohms at 1 kHz and 0.2 ohms at 10 kHz (approx.).

Optional 230 Volt Input: All models can be alternately furnished for operation on an input of 210 to 250 VAC, 47-420 Hz. To order, add suffix "–230" to model number and $10.00 to price.



*TRIM on single output modules; COMMON on duals

Case	H	Approx. Weight
EB-10	1.375	15oz
EB-13	1.625	1lb 4 oz
EB-20	2.375	2lb 1oz

ORIGINAL PAGE IS
OF POOR QUALITY

SINGLE OUTPUT MODELS

Output Voltage	Output Current Amps.	Regulation Load ±%	Regulation Line ±%	Ripple mv RMS	Price	Model	Case Size
1	.500	.4	.05	1	$ 79	1EB50	EB-10
1.5	.500	.3	.05	1	79	1.5EB50	EB-10
1.5	1.0	.5	.05	1	105	1.5EB100	EB-13
1.5	2.5	.6	.05	1	140	1.5EB250	EB-20
2	.400	.25	.05	1	79	2EB40	EB-10
3	.500	.25	.05	1	79	3EB50	EB-10
3.6	.500	.15	.05	1	79	3.6EB50	EB-10
3.6	1.0	.4	.05	1	105	3.6EB100	EB-13
3.6	2.5	.4	.05	1	140	3.6EB250	EB-20
4	.400	.15	.05	1	79	4EB40	EB-10
5	.500	.15	.05	1	79	5EB50	EB-10
5	1.0	.25	.05	1	95	5EB100	EB-13
5	1.5	.35	.1	1	110	5EB150	EB-13
5	2.0	.25	.05	1	125	5EB200	EB-20
5	2.5	.25	.05	1	140	5EB250	EB-20
6	.400	.1	.05	1	79	6EB40	EB-10
6	.550	.25	.05	1	95	6EB55	EB-10
6	1.0	.25	.05	1	110	6EB100	EB-13
6	1.75	.2	.05	1	130	6EB175	EB-20
7	.340	.1	.05	1	79	7EB34	EB-10
7	.450	.2	.05	1	95	7EB45	EB-10
7	.900	.25	.05	1	110	7EB90	EB-13
7	1.15	.2	.05	1	130	7EB115	EB-20
8	.300	.1	.05	1	79	8EB30	EB-10
8	.700	.2	.05	1	110	8EB70	EB-13
8	1.1	.2	.05	1	130	8EB110	EB-20
9	.260	.1	.05	1	79	9EB26	EB-10
9	.450	.15	.05	1	95	9EB45	EB-10
9	.850	.2	.05	1	110	9EB85	EB-13
9	1.5	.2	.05	1	135	9EB150	EB-20
10	.240	.05	.05	1	79	10EB24	EB-10
10	.400	.15	.05	1	95	10EB40	EB-10
10	.750	.2	.05	1	110	10EB75	EB-13
10	1.2	.15	.05	1	135	10EB120	EB-20
11	.220	.05	.05	1	79	11EB22	EB-10
11	.350	.15	.05	1	95	11EB35	EB-10
11	.600	.15	.05	1	110	11EB60	EB-13
11	1.0	.15	.05	1	135	11EB100	EB-20
12	.200	.05	.05	1	79	12EB20	EB-10
12	.400	.1	.05	1	95	12EB40	EB-10
12	.700	.15	.05	1	115	12EB70	EB-13
12	1.2	.2	.05	1	135	12EB120	EB-20
13	.200	.05	.05	1	79	13EB20	EB-10
13	.350	.1	.05	1	95	13EB35	EB-10
13	.600	.1	.05	1	115	13EB60	EB-13
13	1.0	.15	.05	1	135	13EB100	EB-20
14	.200	.05	.05	1	79	14EB20	EB-10
14	.300	.1	.05	1	95	14EB30	EB-10
14	.500	.1	.05	1	110	14EB50	EB-13
14	1.0	.15	.05	1	135	14EB100	EB-20
15	.200	.05	.05	1	79	15EB20	EB-10
15	.400	.1	.05	1	95	15EB40	EB-10
15	.600	.1	.05	1	110	15EB60	EB-13
15	1.0	.15	.05	1	135	15EB100	EB-20
16	.160	.05	.05	1	79	16EB16	EB-10
16	.350	.1	.05	1	100	16EB35	EB-10
16	.500	.1	.05	1	115	16EB50	EB-13
16	.900	.15	.05	1	135	16EB90	EB-20
17	.140	.05	.05	1	79	17EB14	EB-10
17	.325	.1	.05	1	100	17EB33	EB-10
17	.450	.1	.05	1	115	17EB45	EB-13
17	.750	.15	.05	1	135	17EB75	EB-20
18	.120	.05	.05	1	79	18EB12	EB-10
18	.270	.1	.05	1	95	18EB27	EB-10

Output Voltage	Output Current Amps.	Regulation Load ±%	Regulation Line ±%	Ripple mv RMS	Price	Model	Case Size
18	.400	.1	.05	1	$110	18EB40	EB-13
18	.550	.1	.05	1	130	18EB55	EB-20
19	.120	.05	.05	1	79	19EB12	EB-10
19	.250	.1	.05	1	95	19EB25	EB-10
19	.400	.1	.05	1	110	19EB40	EB-13
19	.700	.1	.05	1	135	19EB70	EB-20
20	.120	.05	.05	1	79	20EB12	EB-10
20	.200	.1	.05	1	95	20EB20	EB-10
20	.400	.1	.05	1	110	20EB40	EB-13
20	.700	.1	.05	1	135	20EB70	EB-20
21	.120	.05	.05	1	79	21EB12	EB-10
21	.175	.1	.05	1	95	21EB18	EB-10
21	.375	.1	.05	1	110	21EB38	EB-13
21	.600	.1	.05	1	130	21EB60	EB-20
22	.100	.05	.05	1	79	22EB10	EB-10
22	.150	.1	.05	1	95	22EB15	EB-10
22	.300	.1	.05	1	110	22EB30	EB-13
22	.500	.1	.05	1	130	22EB50	EB-20
23	.100	.05	.05	1	79	23EB10	EB-10
23	.200	.1	.05	1	95	23EB20	EB-10
23	.300	.1	.05	1	110	23EB30	EB-13
23	.600	.1	.05	1	135	23EB60	EB-20
24	.100	.05	.05	1	79	24EB10	EB-10
24	.200	.1	.05	1	95	24EB20	EB-10
24	.350	.1	.05	1	115	24EB35	EB-13
24	.600	.1	.05	1	135	24EB60	EB-20
25	.100	.05	.05	1	79	25EB10	EB-10
25	.190	.1	.05	1	95	25EB19	EB-10
25	.325	.1	.05	1	115	25EB33	EB-13
25	.550	.1	.05	1	135	25EB55	EB-20
26	.080	.05	.05	1	79	26EB08	EB-10
26	.170	.1	.05	1	95	26EB17	EB-10
26	.300	.1	.05	1	110	26EB30	EB-13
26	.450	.1	.05	1	130	26EB45	EB-20
27	.080	.05	.05	1	79	27EB08	EB-10
27	.160	.1	.05	1	95	27EB16	EB-10
27	.300	.1	.05	1	110	27EB30	EB-13
27	.500	.1	.05	1	135	27EB50	EB-20
28	.080	.05	.05	1	79	28EB08	EB-10
28	.150	.1	.05	1	95	28EB15	EB-10
28	.300	.1	.05	1	115	28EB30	EB-13
28	.500	.1	.05	1	135	28EB50	EB-20
30	.080	.02	.02	1	85	30EB08	EB-13
32	.070	.02	.02	1	85	32EB07	EB-13
34	.060	.02	.02	1	85	34EB06	EB-13
35	.050	.02	.02	1	85	35EB05	EB-13
36	.050	.02	.02	1	85	36EB05	EB-13
38	.040	.02	.02	1	85	38EB04	EB-13
40	.030	.02	.02	1	85	40EB03	EB-13
40	.060	.02	.02	1	105	40EB06	EB-13
42	.030	.02	.02	1	85	42EB03	EB-13
44	.030	.02	.02	1	85	44EB03	EB-13
45	.030	.02	.02	1	85	45EB03	EB-13
48	.030	.02	.02	1	85	48EB03	EB-13
48	.050	.02	.02	1	105	48EB05	EB-13
50	.030	.02	.02	1	85	50EB03	EB-13
50	.050	.02	.02	1	105	50EB05	EB-13
55	.040	.02	.02	1	105	55EB04	EB-13
60	.050	.02	.02	1	105	60EB05	EB-13
65	.050	.02	.02	1	105	65EB05	EB-13
70	.040	.02	.02	1	105	70EB04	EB-13
75	.030	.02	.02	1	105	75EB03	EB-13
185	.025	Unregulated		2V	55	NX-25B	EB-10
185	.050	Unregulated		3.5V	75	NX-50B	EB-13

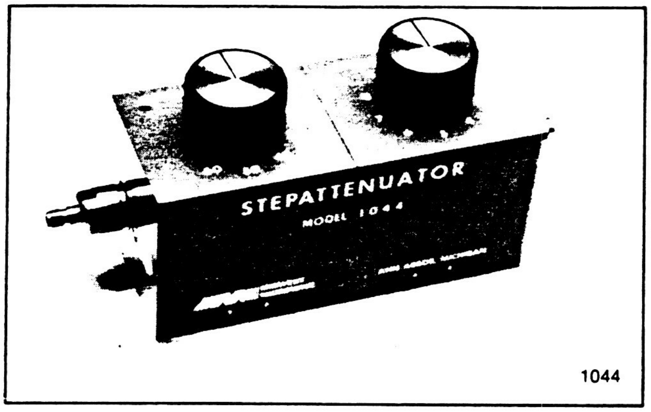
DUAL OUTPUT MODELS

Output Voltage Voltages	Current per Output Amps.	Regulation Load ±%	Regulation Line ±%	Ripple mv RMS	Price	Model	Case Size
±12	.100	.05	.05	1	$ 75	DB12-10	EB-10
±12	.150	.05	.05	1	85	DB12-15	EB-10
±12	.200	.05	.05	1	95	DB12-20	EB-10
±12	.300	.05	.05	1	115	DB12-30	EB-13
±12	.350	.05	.05	1	125	DB12-35	EB.13
±12	.500	.1	.05	1	145	DB12-50	EB-20

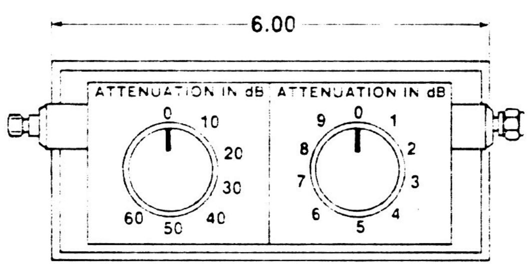
Output Voltage Voltages	Current per Output Amps.	Regulation Load ±%	Regulation Line ±%	Ripple mv RMS	Price	Model	Case Size
±15	.100	.05	.05	1	$ 75	DB15-10	EB-10
±15	.150	.05	.05	1	85	DB15-15	EB-10
±15	.200	.05	.05	1	95	DB15-20	EB-10
±15	.300	.05	.05	1	115	DB15-30	EB-13
±15	.350	.05	.05	1	125	DB15-35	EB-13
±15	.500	.1	.05	1	145	DB15-50	EB-20

STEPATTENUATORS

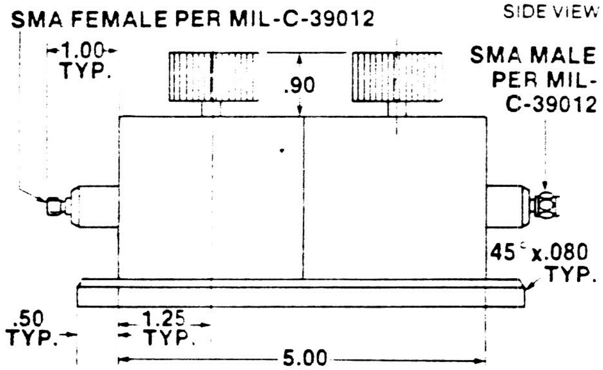
BENCH TOP
0 TO 69dB*



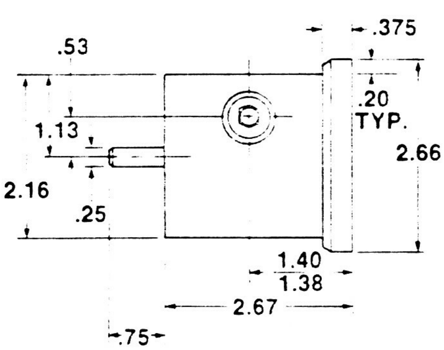
TOP VIEW



SIDE VIEW



END VIEW



0 TO 69 dB SPECIFICATIONS

MODELS 1044-4, 1044-8, 1044-12 AND 1044-18

ACCURACY OF ATTENUATION:

	DC TO 4 GHz	DC TO 8 GHz
1-9	**±0.3dB**	**±0.4dB**
10-19	**±0.8dB**	**±0.9dB**
20-29	**±1.0dB**	**±1.1dB**
30-39	**±1.2dB**	**±1.3dB**
40-49	**±1.3dB**	**±1.4dB**
50-59	**±1.4dB**	**±1.5dB**
60-69	**±1.5dB**	**±1.6dB**
	DC TO 12.4 GHz	DC TO 18 GHz
1-9	**±0.4dB**	**±0.5dB**
10-19	**±1.0dB**	**±1.0dB**
20-29	**±1.2dB**	**±1.2dB**
30-39	**±1.4dB**	**±1.4dB**
40-49	**±1.5dB**	**±1.5dB**
50-59	**±1.6dB**	**±1.6dB**
60-69	**±1.8dB**	**±1.8dB**

MAXIMUM VSWR: DC TO 4 GHz **1.35** ■ 4 to 12.4 GHz **1.50** ■ 12.4 TO 18 GHz **1.65**

MAXIMUM ZERO POSITION INSERTION LOSS: DC TO 4 GHz **0.7dB** ■ 4 TO 12.4 GHz **1.0dB** ■ 12.4 TO 18 GHz **1.5dB**

CONNECTOR TYPES: STAINLESS STEEL TYPE N, PRECISION 7MM OR SMA

MAXIMUM INPUT POWER: 2 WATTS AVERAGE

OPERATING TEMPERATURE RANGE: 0°C TO +55°C

SWITCHING REPEATABILITY: 0.05dB

SWITCHING LIFE: 1,000,000 OPERATIONS

MECHANICAL STOPS: CW AT MAXIMUM ATTENUATION ■ CCW AT MINIMUM ATTENUATION

MODEL NUMBERING SYSTEM:
MODEL 1044 IS 0 TO 69dB IN 1dB STEPS
THE MAXIMUM FREQUENCY RANGE IS SPECIFIED BY USING -4, -8, -12 OR -18
THE CONNECTOR TYPE IS SPECIFIED BY USING N, SMA OR 7MM

MODEL NUMBER EXAMPLE:

1044- 18 7MM

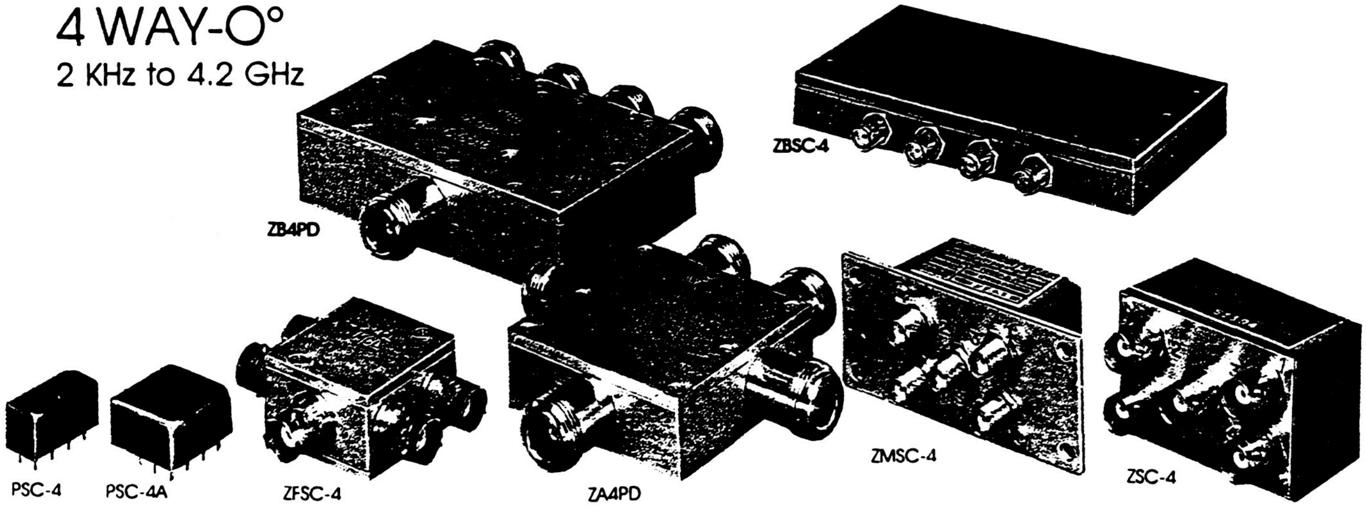
0 TO 69dB   DC TO 18 GHz   7MM CONNECTORS

* U. S. Patent number 3,824,506

50 ohms and 75 ohms

Mini-Circuits

4 WAY-0°
2 KHz to 4.2 GHz



	MODEL NO.	FREQ RANGE MHz f_L-f_U	ISOLATION dB L Typ.	L Min.	M Typ.	M Min.	U Typ.	U Min.	INSERTION LOSS, dB Above 6dB L Typ.	L Max.	M Typ.	M Max.	U Typ.	U Max.	PHASE UNBALANCE Degrees L Max.	M Max.	U Max.	AMPLITUDE UNBALANCE dB L Max.	M Max.	U Max.	PRICE, $ Ea.	Qty.
PSC-4	PSC-4-1	0.1-200	33	20	30	20	27	20	0.4	0.6	0.5	.75	0.7	1.0	4.0	6.0	8.0	.15	.20	.25	28.95	(6-49)
case A01	■ PSC-4-1-75	1-200	30	20	25	20	25	20	0.4	0.7	0.5	0.9	0.7	1.2	4.0	6.0	10.0	.15	.20	0.3	24.95	(6-49)
	PSC-4-3	0.25-250	33	20	30	20	27	20	0.4	0.7	0.5	.75	0.7	1.2	4.0	6.0	8.0	.15	.20	.25	23.95	(6-49)
	PSC-4-6	0.01-40	35	18	32	25	25	18	0.4	0.8	0.3	0.5	0.5	1.0	4.0	6.0	8.0	.10	.15	.20	29.95	(6-49)
PSC-4A	PSC-4A-4	10-1000	25	20	21	15	18	15	0.5	0.8	0.8	1.8	1.5	2.5	4.0	16.0	20.0	0.2	0.5	0.7	49.95	(6-49)
case C07	■ PSC-4A-475	10-800	30	20	33	20	25	20	0.4	0.7	0.6	0.9	1.2	1.6	—	—	—	0.2	0.4	0.8	49.95	(6-49)
ZBSC-4 case UU102	ZBSC-413	10-800	26	20	18	15	18	15	0.6	1.0	1.0	1.5	1.6	2.0	4.0	8.0	8.0	0.2	0.4	0.6	99.95	(1-9)
ZFSC-4	ZFSC-4-1	1-1000	25	20	23	18	20	15	0.4	1.2	0.6	1.5	1.6	2.5	4.0	8.0	8.0	0.2	0.4	0.7	89.95	(1-4)
case G15	ZFSC-4-1W	10-500	23	20	23	20	23	20	0.6	1.5	0.6	1.5	0.6	1.5	4.0	8.0	8.0	0.2	0.3	0.4	74.95	(1-4)
	ZFSC-4-3	10-300	32	28	38	30	38	30	0.5	0.8	0.6	0.9	0.9	1.2	4.0	6.0	8.0	0.1	0.1	0.2	69.95	(1-4)
	■ ZFSC-4375	50-90	34	30	34	30	34	30	0.3	0.8	0.3	0.8	0.3	0.8	4.0	6.0	8.0	0.15	0.15	0.15	89.95	(1-4)
	• ▲■ ZFSC-4-2-75-1	200-800	25	20	—	—	25	20	0.8	1.6	—	—	1.0	1.6	4.0	16.0	20.0	0.2	—	0.4	74.95	(1-4)
	■ ZFSC-4-175	10-1000	35	25	38	20	22	18	0.5	0.8	0.6	1.2	0.9	2.0	—	—	—	0.2	0.3	0.6	89.95	(1-4)
ZMSC-4	ZMSC-4-1	0.1-200	33	20	30	20	27	20	0.4	0.6	0.5	.75	0.7	1.0	1.0	2.0	3.0	.15	.20	.25	56.95	(4-24)
case N24	ZMSC-4-2	0.002-20	30	20	33	25	33	25	.45	.75	0.3	0.5	0.7	1.0	4.0	6.0	8.0	.15	.20	.25	69.95	(4-24)
	ZMSC-4-3	0.25-250	33	20	30	20	27	20	0.4	0.7	0.5	.75	0.7	1.2	4.0	6.0	10.0	.15	.20	.25	53.95	(4-24)
ZSC-4	ZSC-4-1	0.1-200	33	20	30	20	27	20	0.4	0.6	0.5	.75	0.7	1.0	4.0	6.0	8.0	.15	.20	.25	46.95	(4-24)
case N27	■ ZSC-4-1-75	1-200	30	20	25	20	25	20	0.4	0.7	0.5	0.8	0.7	1.2	0.4	6.0	10.0	.15	.20	0.3	46.95	(4-24)
	ZSC-4-2	0.002-20	30	20	33	25	33	25	.45	.75	0.3	0.5	0.7	1.0	4.0	6.0	8.0	.15	.20	.25	69.95	(4-24)
	ZSC-4-3	0.25-250	33	20	30	20	27	20	0.4	0.7	0.5	.75	0.7	1.2	4.0	6.0	10.0	.15	.20	.25	43.95	(4-24)
		GHz																				
ZA4PD	ZA4PD-2	1-2	25	16	25	16	25	16	0.3	1.0	0.3	1.0	0.3	1.0	—	6.0	—	0.7	0.7	0.7	79.95	(1-9)
case DD52	ZA4PD-4	2-4.2	25	16	25	16	25	16	0.5	1.0	0.3	1.0	0.3	1.0	—	16.0	—	0.8	0.8	0.8	79.95	(1-9)
ZB4PD	ZB4PD-42	1.7-4.2	23	16	23	16	23	16	0.5	1.4	0.5	1.4	0.5	1.4	—	8.0	—	0.8	0.8	0.8	99.95	(1-9)
case EE54	ZB4PD-4	3.7-4.2	24	15	24	15	24	15	0.6	1.1	0.6	1.1	0.6	1.1	—	8.0	—	0.8	0.8	0.8	89.95	(1-9)

L = low range (f_L to 10 f_L) M = mid range (10f_L to f_U/2)
• L = low range (f_L to f_U/2) U = upper range (f_U/2 to f_U)

NOTES:
■ Denotes 75 ohm models. 75 ohm BNC connectors are standard.
• Model PSC-3-1 manufactured under license protected by patent 3,428,920.
▲ On Model ZFSC-4-2-75-1, up to 15V and 15mA DC may be passed from input to all outputs.

1. For quality control procedures, see Table of Contents.
2. For environmental specifications, see Table of Contents.
3. Absolute Maximum Ratings:
   Matched power rating ZA3PD, ZA4PD, ZB4PD (10W.)
   all other models (1W)
   Internal load dissipation
   all 3-way models (0.375W) all 4-way models (0.25W)
4. For connector types and case mounting options, see case style outline drawing.
5. Prices and specifications subject to change without notice.
6. All 3-way power dividers with exception of ZA3PD are licensed under U.S. Patent 3,428,920; reissued as RE 27,299.

In Stock...Immediate Delivery

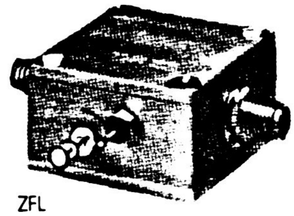
broadband linear

Power Amplifiers

up to 100mW (+20 dBm)

50 KHz to 2000 MHz

ORIGINAL PAGE IS
OF POOR QUALITY



ZFL

case style selection

outline drawings section 1

	MODEL NO.	FREQUENCY MHz	GAIN, dB			MAXIMUM POWER, dBm		DYNAMIC RANGE		VSWR		DC POWER		PRICE $	
			Min.	Flatness Max.	Control range	Output (1dB) Compression	Input (no damage)	NF, dB Typ.	Intercept pt., dBm 3rd order	In	Out	Volt.	Current	Ea.	Qty.
ZFL	ZFL-500	0.05-500	20	±1.0	—	+9	+5	5.3	+18	1.9	1.9	+15	85mA	69.95	(1-24)
	▲ ZFL-500LN	0.1-500	24	±0.3	—	+5	+5	2.9	+14	1.5:1	1.6	+15V	60mA	79.95	(1-24)
	ZFL-750	0.2-750	18	±0.5	—	+9	+5	6.0	+18	1.5:1	2:1	+15V	90mA	74.95	(1-24)
	■ ZFL-1000	0.1-1000	17	±0.6	—	+9	+5	6.0	+18	1.5:1	2:1	+15V	90mA	79.95	(1-24)
case Y 39	ZFL-1000G	10-1000	17	±1.5	30	+3	0	12	+13	2:1	2:1	+15V	90mA	199	(1-9)
	ZFL-1000LN	0.1-1000	20	±0.5	—	+3	+5	2.9	+14	1.5	2:1	+15V	60mA	89.95	(1-24)
	ZFL-2000	10-2000	20	±1.5	—	+17•	+5	7.0	+25	2:1	2:1	+15V	100mA	219	(1-9)
case SS98	ZFL-1000H	10-1000	28	±1.0	—	+20	+5	5	+33	2:1	2:1	15	150mA	219	(1-9)

NOTES:

•+15dBm below 1000 MHz.

■ ZFL-1000 output VSWR 2.8:1 maximum over 750-1000 MHz.

▲ VSWR 1.6 maximum from 0.1 to 0.2 MHz.

1. Operating temperature is −55°C to +71°C except the ZFL-2000 is −55°C to +100°C. When models ZFL-1000H and ZFL-2000 are mounted to chassis using a thermoconductive paste, their operating temperature range will be increased.
2. With no load output, derate maximum input power (no damage) by 10 dB.
3. Prices and specifications subject to change without notice.

NSN GUIDE

MCL NO.	NSN
ZFL-2000	5895-01-220-2213

ZFL-500

FREQ. (MHz)	GAIN, dB FORWARD	GAIN, dB REVERSE	LINEARITY Comp. (dB)	LINEARITY P_{out} (dBm)	NOISE FIGURE (dB)	VSWR in	VSWR out
.050	22.8	37.70	0.6	10.6	—	—	—
.104	22.9	38.80	0.6	10.9	—	1.33	1.09
.217	23.0	38.70	.7	11.0	—	1.33	1.09
.453	23.4	38.60	.8	11.4	—	1.34	1.08
.943	23.3	38.50	.7	11.3	—	1.35	1.09
1.966	23.4	38.30	.8	11.4	6.0	1.35	1.07
4.098	23.5	38.30	.8	11.4	5.9	1.35	1.09
8.541	23.5	38.20	.8	11.4	5.9	1.35	1.07
10.910	23.4	38.30	.8	11.4	6.0	1.37	1.08
22.738	23.4	38.20	.8	11.4	5.9	1.36	1.09
47.389	23.5	38.10	.8	11.4	5.8	1.36	1.07
98.767	23.6	38.00	.9	11.6	5.9	1.37	1.07
126.160	23.8	38.00	.8	11.7	5.9	1.38	1.07
161.151	23.7	38.00	.9	11.6	5.8	1.38	1.09
205.846	23.7	38.00	.9	11.6	5.8	1.36	1.10
262.938	23.6	38.00	.9	11.6	5.8	1.39	1.09
335.865	23.5	37.70	.8	11.4	5.7	1.41	1.08
429.019	23.1	37.20	.9	11.0	5.6	1.44	1.08
548.008	22.5	36.00	.7	11.4	5.6	1.50	1.11
700.000	21.4	34.90	.5	9.3	5.8	1.56	1.16

ZFL-500-LN

FREQ (MHz)	GAIN (dB)	LINEARITY Comp (dB)	LINEARITY P_{out} (dBm)	NOISE FIGURE	RETURN LOSS in	RETURN LOSS out
.100	30.90	0.72	7.65	—	33.84	16.81
.300	30.83	.89	7.62	—	27.01	16.81
.506	30.81	.87	7.92	—	26.10	17.15
1.100	30.80	.86	7.99	—	25.48	17.35
2.390	30.77	.83	7.77	—	25.30	17.43
5.190	30.76	.81	7.45	—	25.25	17.53
10.130	30.76	.80	7.39	2.52	25.19	17.56
24.460	30.74	.77	7.36	2.47	25.25	17.55
40.490	30.75	.76	7.35	2.50	25.09	17.63
50.120	30.70	.75	7.24	2.52	24.96	17.70
100.720	30.71	.72	7.16	2.56	24.00	18.01
190.970	30.67	.67	6.97	2.57	21.77	19.07
250.520	30.72	.67	6.91	2.59	20.21	20.28
322.330	30.75	.62	6.77	2.57	18.43	22.69
503.450	30.68	.56	6.52	2.65	15.38	23.08

Mini-Circuits P.O. BOX 350166, Brooklyn, New York 11235-0003 (718) 934-4500
FAX (718) 332-4661 TELEX 6852844 or 620156

ORIGINAL PAGE IS
OF POOR QUALITY

STEP ATTENUATORS FOR OEM

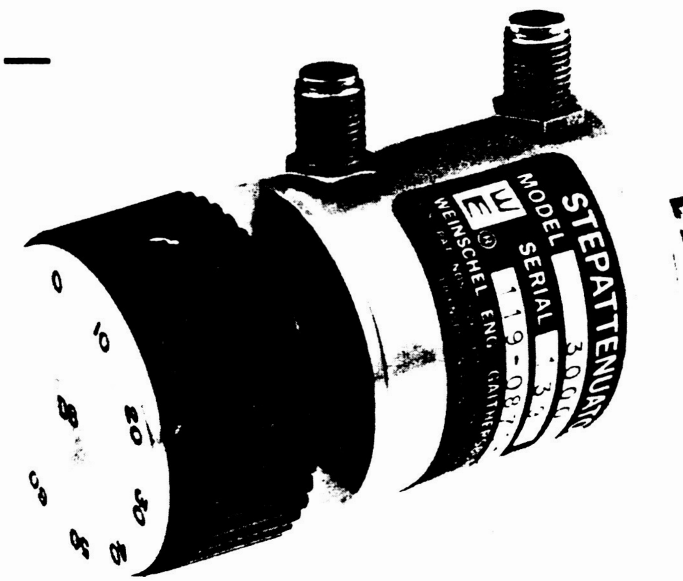
1 WATT

MODEL 3000 SERIES

DC TO 2.5 GHz
DC TO 1.25 GHz

SMA FEMALE CONNECTORS

Available in 24 attenuation ranges/steps — see specifications table.



FEATURES

LOW VSWR — Typically ©1.10 to 2.5 GHz

HIGH RELIABILITY — Repeatability better than 0.1 dB over frequency range and life. Weinschel patented[1] detent mechanism, tested to 1,000,000 operations at +75°C, operates dependably even down to −40°C.

PRODUCT UNIFORMITY — High volume fabrication techniques, including injection molding, stamping, broaching and thick film printing ensure a cost effective and uniform product.

LOW FREQUENCY SENSITIVITY — Typically 0.1 to 0.2 dB up to 2.5 GHz.

SHOCK RESISTANT — 100% spring contact system withstands mechanical and thermal shock and eliminates the need for epoxy or solder.

WIDE SELECTION — Wide choice of attenuation ranges and increments in standard stock models. Single and dual drum configurations available.

[1]U.S. Patent 4,107,634; 4,107,633

STEP ATTENUATORS FOR OEM

ORIGINAL PAGE IS OF POOR QUALITY

MODEL 3000 SERIES

SPECIFICATIONS

MODEL	ATTENUATION RANGE/STEPS	FREQ. RANGE	STEP ANGLE	CONFIGURATIONS	VSWR	MAXIMUM INSERTION LOSS	ACCURACY OF INCREMENTAL INSERTION LOSS
3001	0-50/10 dB	DC-2.5 GHz	32.7°	SINGLE	1.20:1	0.3 dB	±0.3 dB or 1% *
3002	0-60/10 dB	DC-2.5 GHz	32.7°	SINGLE	1.20:1	0.3 dB	±0.3 dB or 1% *
3003	0-70/10 dB	DC-2.5 GHz	32.7°	SINGLE	1.20:1	0.3 dB	±0.3 dB or 1% * to 60 dB, 2% to 70 dB
3004	0-80/10 dB	DC-1.25 GHz	32.7°	SINGLE	1.20:1	0.2 dB	±0.3 dB or 1% * to 60 dB, 2% to 80 dB
3005	0-90/10 dB	DC-1.25 GHz	32.7°	SINGLE	1.20:1	0.2 dB	±0.3 dB or 1% * to 60 dB, 2% to 90 dB
3006	0-100/10 dB	DC-1.25 GHz	32.7°	SINGLE	1.20:1	0.2 dB	±0.3 dB or 1% * to 60 dB, 2% * to 100 dB
3007	0-10/1 dB	DC-2.5 GHZ	32.7°	SINGLE	1.30:1	0.3 dB	±0.3 dB
3008	0-1.0/0.1 dB	DC-2.5 GHz	32.7°	SINGLE	1.30:1	0.3 dB	**
3009	0-60/1 dB	DC-2.5 GHz	32.7°	DUAL	1.35:1	0.7 dB	±0.3 dB to 10 dB, ±0.3 dB or 1% * to 60 dB
3010	0-70/1 dB	DC-2.5 GHz	32.7°	DUAL	1.35:1	0.7 dB	±0.3 dB to 10 dB, ±0.3 dB or 1% * to 60 dB, 2% to 70 dB
3011	0-80/1 dB	DC-1.25 GHz	32.7°	DUAL	1.30:1	0.5 dB	±0.3 dB to 10 dB, ±0.3 dB or 1% * to 60 dB, 2% * to 80 dB
3012	0-90/1 dB	DC-1.25 GHz	32.7°	DUAL	1.30:1	0.5 dB	±0.3 dB to 10 dB, ±0.3 dB or 1% * to 60 dB, 2% * to 90 dB
3013	0-100/1 dB	DC-1.25 GHz	32.7°	DUAL	1.30:1	0.5 dB	±0.3 dB to 10 dB , ±0.3 dB or 1% * to 60 dB 2% * to 100 dB
3014	0-110/1 dB	DC-1.25 GHz	32.7°	DUAL	1.30:1	0.5 dB	±0.3 dB to 10 dB , ±0.3 dB or 1% * to 60 dB, 2% * to 110 dB
3015	0-11/0.1 dB	DC-2.5 GHz	32.7°	DUAL	1.35:1	0.7 dB	**
3016	0-50/10 dB	DC-2.5 GHz	36°	SINGLE	1.20:1	0.3 dB	±0.3 dB or 1% *
3017	0-60/10 dB	DC-2.5 GHz	36°	SINGLE	1.20:1	0.3 dB	±0.3 dB or 1% *
3018	0-70/10 dB	DC-2.5 GHz	36°	SINGLE	1.20:1	0.3 dB	±0.3 dB or 1% * to 60 dB, 2% * to 70 dB
3019	0-80/10 dB	DC-1.25 GHz	36°	SINGLE	1.20:1	0.2 dB	±0.3 dB or 1% * to 60 dB, 2% * to 80 dB
3020	0-90/10 dB	DC-1.25 GHz	36°	SINGLE	1.20:1	0.2 dB	±0.3 dB or 1% * to 60 dB, 2% * to 90 dB
3021	0-9/1 dB	DC-2.5 GHz	36°	SINGLE	1.30:1	0.3 dB	±0.3 dB
3022	0-.9/0.1 dB	DC-2.5 GHz	36°	SINGLE	1.30:1	0.3 dB	**
3023	0-59/1 dB	DC-2.5 GHz	36°	DUAL	1.35:1	0.7 dB	±0.3 dB to 9 dB, ±0.3 dB or 1% * to 59 dB
3024	0-69/1 dB	DC-2.5 GHz	36°	DUAL	1.35:1	0.7 dB	±0.3 dB to 9 dB, ±0.3 dB or 1% * to 59 dB, 2% * to 69 dB
3025	0-79/1 dB	DC-1.25 GHz	36°	DUAL	1.30:1	0.5 dB	±0.3 dB to 9 dB , ±0.3 dB or 1% * to 59 dB, ± 2% * to 79 dB
3026	0-89/1 dB	DC-1.25 GHz	36°	DUAL	1.30:1	0.5 dB	±0.3 dB to 9 dB , ±0.3 dB or 1% * to 59 dB, ± 2% * to 89 dB
3027	0-99/1 dB	DC-1.25 GHz	36°	DUAL	1.30:1	0.5 dB	±0.3 dB to 9 dB , ±0.3 dB or 1% * to 59 dB, ±2% * to 99 dB
3028	0-9.9/0.1 dB	DC-2.5 GHz	36°	DUAL	1.35:1	0.7 dB	**
3045	0-70/10 dB	DC-2.5 GHz	45°	SINGLE	1.20:1	0.3 dB	±0.3 dB or 1% * to 60 dB, 2% * to 70 dB

*Whichever is greater.

**The change of insertion loss between adjacent positions of the 0.1 dB drum will be a minimum of 0.05 dB to a maximum of 0.15 dB to 0.2 dB maximum cumulative.

IMPEDANCE: 50 ohms, nominal

MAXIMUM RF POWER: 1 watt average, 100 watts peak with 5 μsec. maximum pulse width

POWER COEFFICIENT: <0.006 dB/dB × W

TEMPERATURE COEFFICIENT: <0.0001 dB/dB × °C

TEMPERATURE RANGE: Operating: −40°C to +65°C
Non-Operating: −54°C to +85°C

SWITCHING LIFE: 1,000,000 steps

REPEATABILITY: ±0.1 dB over frequency range and rated life

SHAFT ROTATION: ccw for increasing attenuation

CONNECTORS: Stainless steel female SMA mates with male SMA per MIL-C-39012

ROTATION STOPS: Supplied on 10 dB step drums. (Not supplied on 1 dB and 0.1 dB drums.)

INCREMENTAL PHASE SHIFT: ~0.25° per dB × f (GHz)

MATERIALS AND FINISHES: Shafting and external hardware and connector shells: CRES Type 303, PER QQ-S-764 Passivated per QQ-P-35.

Housing: AL ALLOY Gold Flash.

No fungus supporting nutrients used within or without.

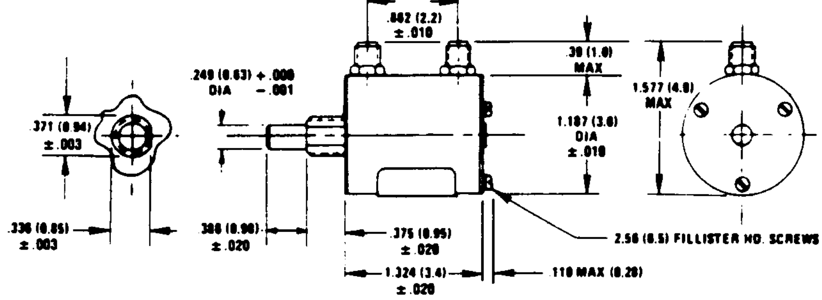
Marking: Each unit individually marked with foil type nameplate giving model number and individual serial numbers.

Acceptance Tests: Each unit is individually tested to insure performance in accordance with specifications. (No test data is supplied.)

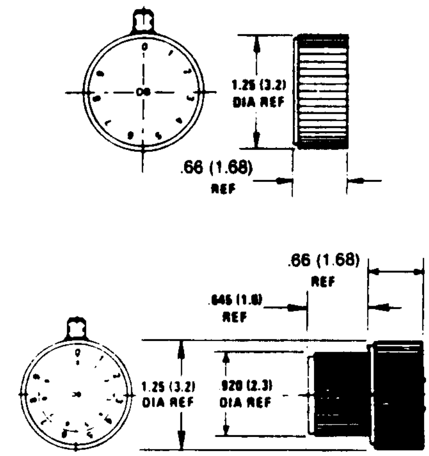
SPECIFICATIONS (cont.)

SINGLE DRUM

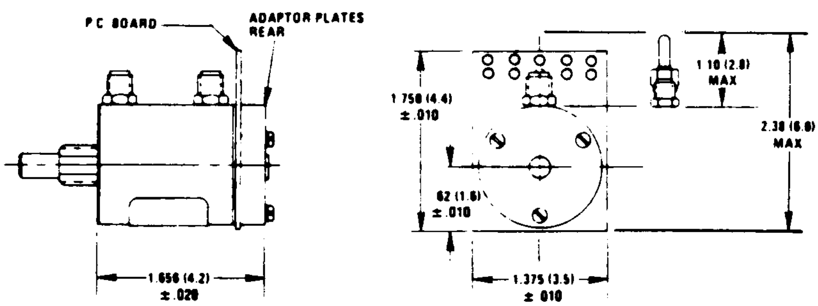
STANDARD UNIT



KNOBS

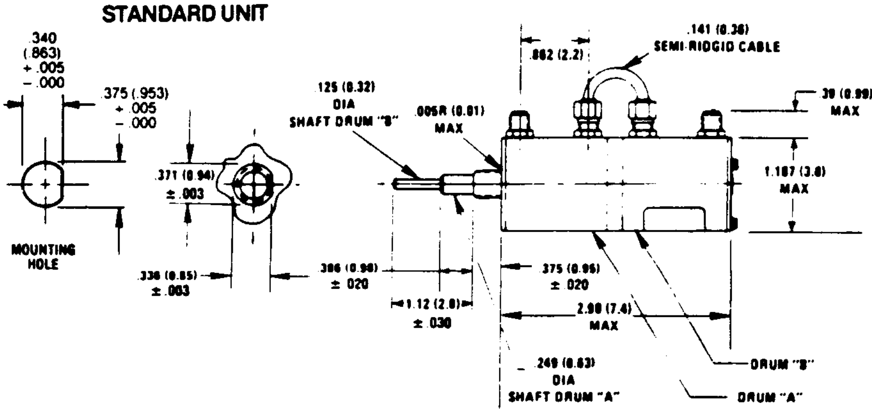


WITH INDICATOR SWITCH OPTION



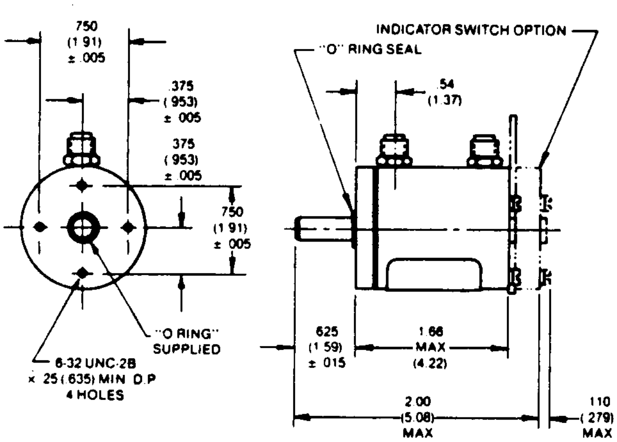
DUAL DRUM

STANDARD UNIT

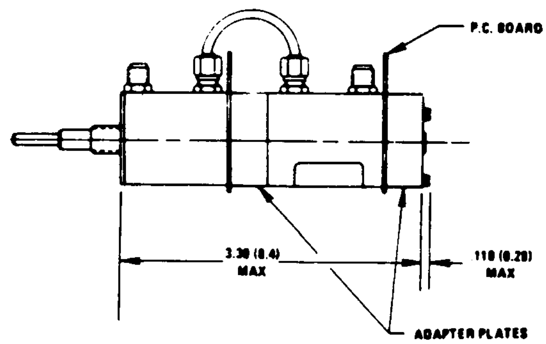


NOTES:
Dimensions in inches (cm)
For additional details not shown, contact Weinschel Engineering
For details on panel seal option, contact Weinschel Engineering

SEALED PANEL OPTION



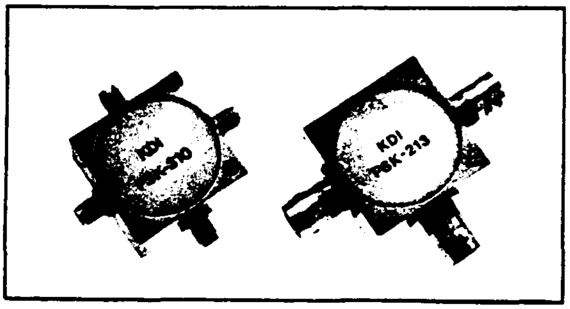
WITH INDICATOR SWITCH OPTION



ORIGINAL PAGE IS
OF POOR QUALITY

TOROIDAL power dividers

CONNECTORIZED • 2-4 OUTPUTS • 0.01-1500 MHz



SMA, BNC, TNC, and TYPE N Connectors Available.
(last letter of Model = Connector type):
M = SMA
N = TYPE N
B = BNC
T = TNC

GENERAL SPECIFICATIONS

VSWR:	1.30 Typical
Impedance:	50 Ohms Nominal
Matched Power Rating:	1 Watt Max.
Weight:	5 Grams Connector Type, 42 Grams
Temperature Range:	-55 to 100°C (operating & storage)
Finish:	Grey Paint

ENVIRONMENTAL SPECS. MIL-STD-202E

Moisture Resistance:	Method 106D
Salt Spray:	Method 101D
Vibration High Frequency:	Method 204C
Shock Test:	Method 213B

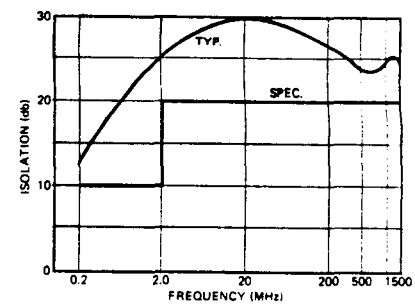
Workmanship in Accordance with MIL-STD-454D
Requirements 5 and 9

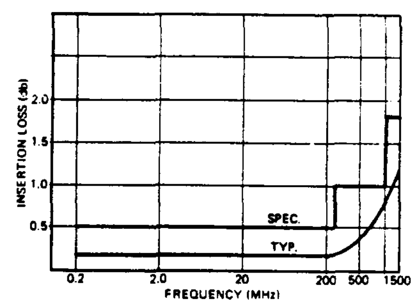
2-4 OUTPUTS

No. of Outputs	Model	Frequency (MHz)	Ins. Loss (dB Max.)	Isolation (dB Min.)	Amp. Bal. (±dB)	Phase Balance (±Deg.)	Outline
2	PSK-210	0.2-1200 1200-1500	1.3 1.8	20	0.3	9	110
	PSK-211	0.01-100	0.6	20	0.1	1	110
	PSK-212	10-1000	1.0	24	0.2	3	110
	PSK-213*	10-400	0.5	30	0.1	2	117
		400-500	0.5	25	0.1	2	117
	PSK-270**	55-85	0.5	25	0.1	1	110
3	PSK-310	0.25-500	1.2	20	0.4	4	111
	PSK-311	0.01-25	0.5	20	0.2	1	111
4	PSK-410	0.5-1000	1.8	18	0.5	9	112
	PSK-411	0.01-100	1.0	20	0.1	1	112
	PSK-413	10-500	1.0	25	0.2	4	112
	PSK-470**	55-85	1.0	25	0.1	1	112

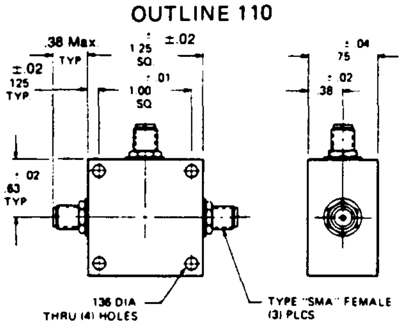
*VSWR 1.3:1 Max. **75 ohms

MODEL PSK-210
2 OUTPUT, 200 KHz - 1500 MHz

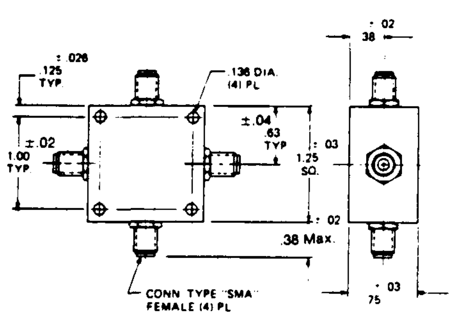


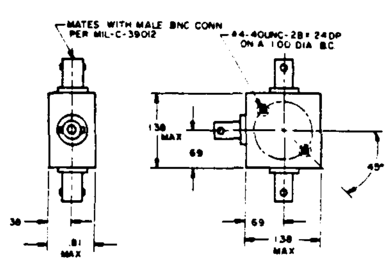


OUTLINE 110



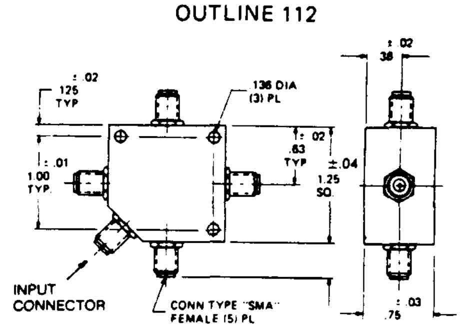
OUTLINE 111





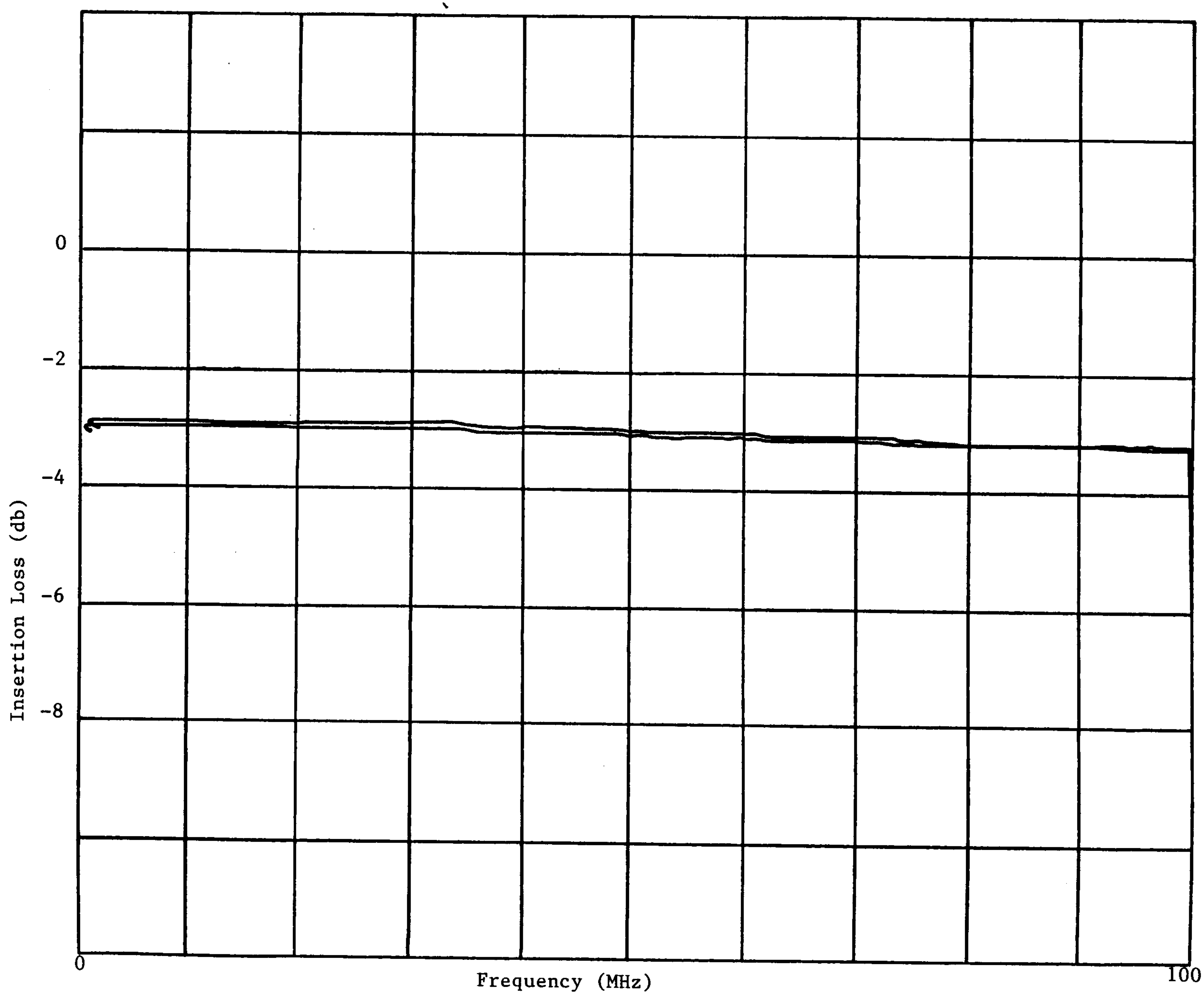
OUTLINE 117

OUTLINE 112



KDI *ELECTRONICS, INC.*
Pyrofilm & Engelmann Divisions
TEL (201) 887-8100 • TWX (710) 986-8220 • FAX (201) 887-4645 • 60 South Jefferson Road, Whippany, N.J. 07981

Power Splitter
Model PSK-211

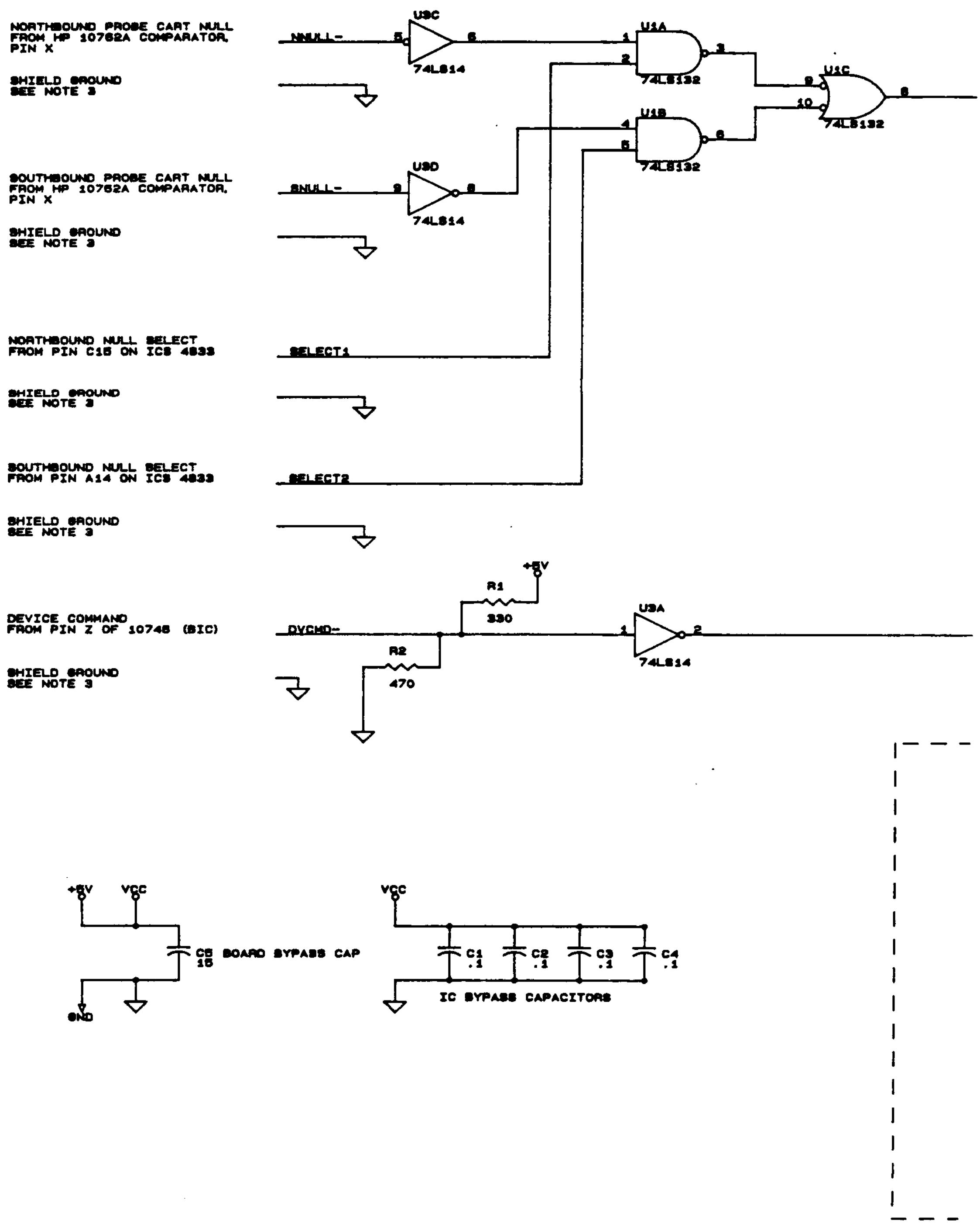


APPENDIX B

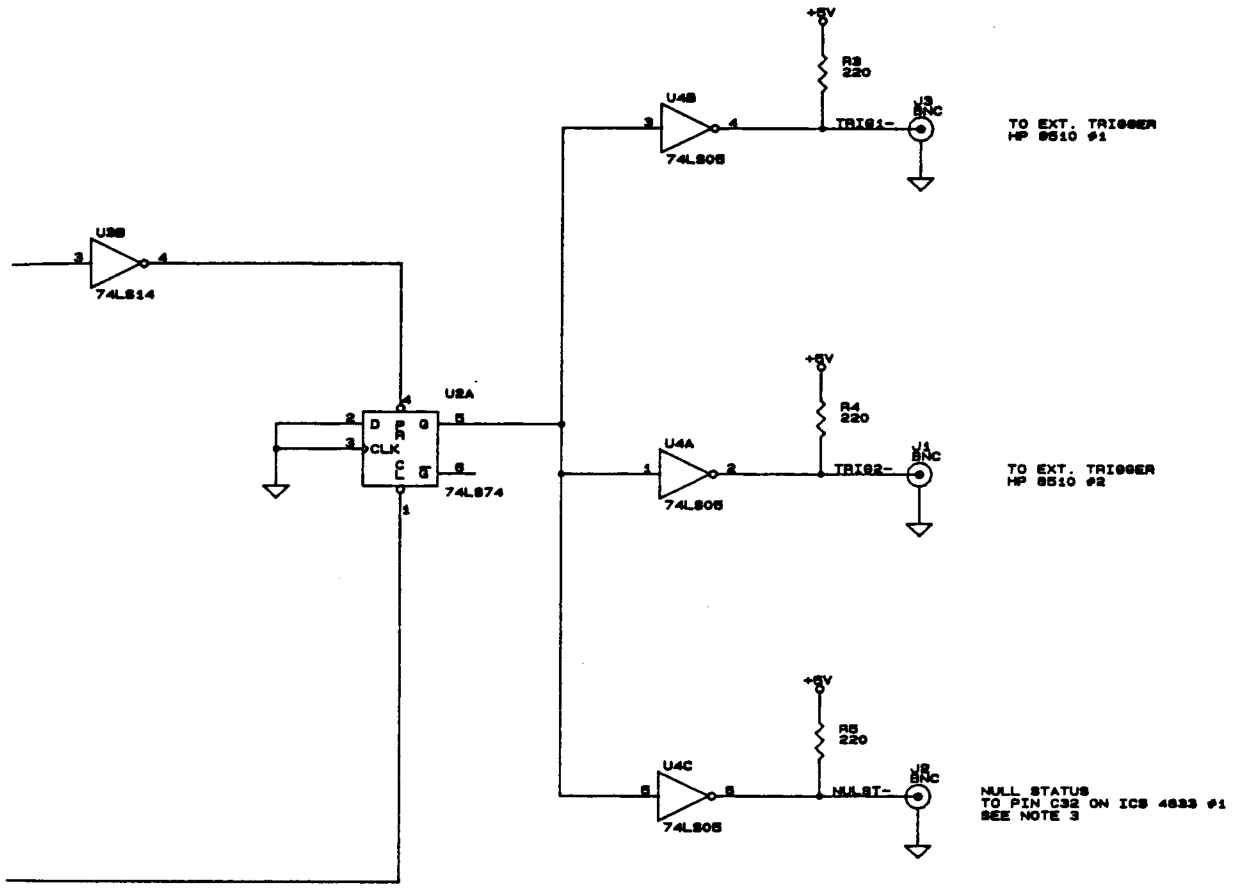
Schematic Diagram for Trigger Control Electronics

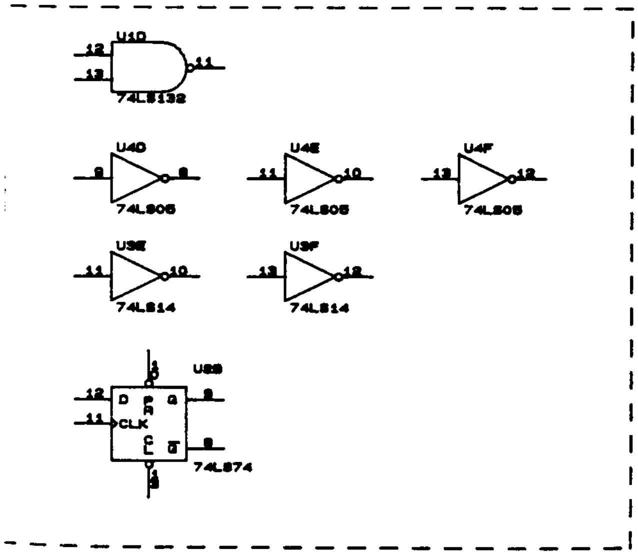
This page intentionally left blank

ORIGINAL PAGE IS
OF POOR QUALITY



ORIGINAL PAGE IS
OF POOR QUALITY





UNUSED DEVICES

NOTES:

1. CAPACITORS IN UF
2. RESISTORS IN OHMS
3. CONNECTOR TYPE IS OPTIONAL
   SIGNAL WIRE MUST BE INDIVIDUALLY SHIELDED

Title		
8510 TRIGGER SELECT SCHEMATIC		
Size C	Document Number 42910048	REV
Date: February 18, 1988	Sheet 1 of 1	

This page intentionally left blank

APPENDIX C

Program XYZ Listing

This page intentionally left blank

```
* Program XYZ
* Command file for linking loader
*
* Last Revised:  28 Nov 88

   EC
   DE

   RE XYZ.REL

   RE CLSTAT.REL
   RE COLLECT.REL
   RE COLREAD.REL
   RE DATETIME.REL::NASA
   RE DEFINE.REL::NASA

   RE DELAY.REL::NASA
   RE DRWJ.REL
   RE EFILE.REL
   RE ENCODE.REL
   RE ERRTRUSS.REL

   RE GO_HOME.REL
   RE GRIDREAD.REL
   RE HEADER.REL::NASA
   RE IERROR.REL
   RE INIT.REL

   RE LABJ.REL
   RE LASER.REL
   RE LISTBUFFER.REL
   RE LISTCHANGE.REL::NASA
   RE MAXMIN.REL

   RE MCART.REL
   RE MOVE.REL
   RE MTRUSS.REL
   RE NAMFILE.REL::NASA
   RE NEGCOL.REL

   RE PDEF.REL::NASA
   RE PLOT.REL::NASA
   RE POSCHECK.REL
   RE POSCOL.REL
   RE POSITION.REL

   RE POSWATCH.REL
   RE READWRITE.REL::NASA
   RE RECTOPOL.REL::NASA
   RE RESET.REL
   RE RMULTFIND.REL::NASA

   RE SCAN.REL
   RE SETSOURCE.REL
   RE SIDECHECK.REL
```

```
      RE SOURCE.REL
      RE STO_POSN.REL::NASA

      RE SWIPE.REL::NASA
      RE VOLIN.REL
      RE VWPTJ.REL
      RE XINIT.REL


*     Graphics Library
      SE UPLIB_CDS.LIB

*     Set Priority
      PR 89


      EN XYZ.RUN
```

```
$CDS ON
$FILES 0,3
$EMA /BUFFER/,/BUFFER2/,/POSN/

C*********************************************************************C
C                                                                     C
C     PROGRAM XYZ                  Last Revised:   6/06/88            C
C                                                                     C
C                                                                     C
C     This is the main program for data collection on the             C
C         near field antenna range (sometimes called an XYZ           C
C         range.)                                                     C
C                                                                     C
C                                                                     C
C     The following subroutines are called from the main              C
C         routine (where the entry point is different from            C
C         the subroutine name, the subroutine name follows            C
C         in parentheses):                                            C
C                                                                     C
C             COLREAD                                                 C
C             EFILE                                                   C
C             GEND  (PLOT)                                            C
C             GINIT  (PLOT)                                           C
C             GRIDREAD                                                C
C             INIT                                                    C
C             LISTBUFFER                                              C
C             LISTCHANGE                                              C
C             MOVE                                                    C
C             SCAN                                                    C
C             SRC_PWR  (SOURCE)                                       C
C             SRC_USER  (SETSOURCE)                                   C
C             SWIPE                                                   C
C             VWPTJ                                                   C
C             XINIT                                                   C
C                                                                     C
C     This list does not include system calls or subroutines          C
C         from the system libraries.                                  C
C                                                                     C
C*********************************************************************C

C     LAST REVISED 8/5/88

      PROGRAM XYZ

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /MINMAX/  AMIN, AMAX, PMIN, PMAX, MAXROW, MAXCOL
      COMMON /BUFFER/  ABUF(4096), PBUF(4096), IBUF
      COMMON /BUFFER2/  ABUF2(4096), PBUF2(4096), IBUF2
      COMMON /RECBUFF/  LBUF(8200)
      COMMON /POSN/  XPOS(4096), YPOS(4096)
      COMMON /EXP85/  AEXP(0:255)
      COMMON /ASSIGN/  IA(0:7), IB(0:7), IC(0:7), IP(0:7),
     +                   IV(0:7), IX(0:7), IY(0:7), IZ(0:7)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
```

```
      COMMON /HPIB/  I4833, J4833, I8510, J8510, I8340, J8340
      COMMON /USER/  IWRITE, IREAD
      COMMON /LASER/  CTI, VOL, DPI, CPOS
      COMMON /DATA_DIR/  DDIR

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER COM*2, DDIR*15


C   Initialization

      IWRITE=1                          ! Default LU for user prompts
      IREAD=1                           !             and input

      CALL INIT                         ! Initialize other arrays and
                                        !        equipment

      IMODE = 0
      NPOL = 2
      CALL XINIT (IMODE,1)               ! Initialize Scan parameters

      CPOS=0.                          ! Current cart position
      IAXIS=0                           ! Data collection along Y-axis

      ASP=0.                            ! Max aspect ratio
      IASP=0                            !
      IDWORK=0                          ! Plotting on terminal only
      I3D=0                             ! No 3-D plotting
      CALL GINIT(IDWORK,I3D,ASP,IASP)   ! Enable graphics


      WRITE (1,*) 'Enter the directory name for data files  ',
     +                '(default = /XYZFILES):'
      READ (1, '(A)')  DDIR
      IF (DDIR .LE. ' ')   DDIR = '/XYZFILES'


C   Enter command from user and execute (the command, not the user)

10    CALL SWIPE
      WRITE (1,19)
19    FORMAT(/ 'Enter a two letter command:' /)

      WRITE (1,99)          ! IN
      WRITE (1,699)         ! LC
      WRITE (1,299)         ! SF
      WRITE (1,1099)        ! SP
      WRITE (1,*)
      WRITE (1,899)         ! EF
      WRITE (1,399)         ! CR
      WRITE (1,499)         ! CL
      WRITE (1,599)         ! CP
      WRITE (1,*)
      WRITE (1,199)         ! MO
      WRITE (1,1199)        ! AR
```

```
      WRITE (1,799)          ! CD
      WRITE (1,*)
      WRITE (1,1299)         ! EX

      WRITE (1,29)
29    FORMAT (/,'Command?')
      READ (1,39) COM
39    FORMAT (A2)

      IF (COM .EQ. 'IN' .OR. COM .EQ. 'in') GOTO 100
      IF (COM .EQ. 'MO' .OR. COM .EQ. 'mo') GOTO 200
      IF (COM .EQ. 'SF' .OR. COM .EQ. 'sf') GOTO 300
      IF (COM .EQ. 'CR' .OR. COM .EQ. 'cr') GOTO 400
      IF (COM .EQ. 'CL' .OR. COM .EQ. 'cl') GOTO 500
      IF (COM .EQ. 'CP' .OR. COM .EQ. 'cp') GOTO 600
      IF (COM .EQ. 'LC' .OR. COM .EQ. 'lc') GOTO 700
      IF (COM .EQ. 'CD' .OR. COM .EQ. 'cd') GOTO 800
      IF (COM .EQ. 'EF' .OR. COM .EQ. 'ef') GOTO 900
      IF (COM .EQ. 'SP' .OR. COM .EQ. 'sp') GOTO 1100
      IF (COM .EQ. 'AR' .OR. COM .EQ. 'ar') GOTO 1200
      IF (COM .EQ. 'EX' .OR. COM .EQ. 'ex') GOTO 1300
      GOTO 10



99    FORMAT ('"IN"--INitialize the scan parameters')
100   CALL SWIPE
      CALL XINIT (IMODE, 0)
      GO TO 10

199   FORMAT ('"MO"--MOve the probe to a specified position')
200   CALL SWIPE
      CALL MOVE
      WRITE (1,*) 'Hit RETURN to continue'
      READ (1,39) COM
      GOTO 10

299   FORMAT ('"SF"--Set Source power and frequency')
300   CALL SWIPE
      CALL SRC_USER (FREQ, IMODE, NPOL)
      GOTO 10

399   FORMAT ('"CR"--Read a Column of data into the buffer')
400   CALL SWIPE
      CALL COLREAD (IROW,IAXIS)
      GOTO 10

499   FORMAT ('"CL"--List the Column of data in the buffer')
500   CALL SWIPE
      CALL LISTBUFFER (IROW,IAXIS,ABUF,PBUF,IBUF)
      IF (NPOL .EQ. 2) THEN
         CALL SWIPE
         WRITE (IWRITE,*) 'Second polarization: '
         CALL LISTBUFFER (IROW,IAXIS,ABUF2,PBUF2,IBUF2)
      END IF
```

```
      GOTO 10

599   FORMAT ('"CP"--Plot the Column of data in the buffer')
600   CALL VWPTJ (IROW,IAXIS,ABUF,PBUF)
      READ (1,39) COM
      IF (NPOL .EQ. 2) THEN
         CALL SWIPE
         WRITE (IWRITE,*) 'Second polarization:'
         CALL VWPTJ (IROW,IAXIS,ABUF2,PBUF2)
         READ(1,39) COM
      END IF
      GOTO 10

699   FORMAT ('"LC"--List or Change the current scan parameters')
700   CALL SWIPE
      CALL LISTCHANGE (IMODE)
      GOTO 10

799   FORMAT ('"CD"--Collect a Data set using the scan parameters')
800   CALL SWIPE
      CALL SCAN (IROW, IAXIS)
      GOTO 10

899   FORMAT ('"EF"--Examine a File for plotting or listing')
900   CALL SWIPE
      CALL EFILE (IROW, IAXIS)
      CALL LISTCHANGE (IMODE)
      GOTO 10

1099  FORMAT ('"SP"--Set Source power ')
1100  CALL SWIPE
      WRITE (1,*) 'Which source  (1 / 2)?'
      READ (1,*) ISRC
      IADR = I8340
      IF (ISRC.EQ.2)  IADR = J8340
      WRITE (1,*) 'Enter the desired power level (dBm):'
      READ (1,*) PWR
      CALL SRC_PWR (IADR, PWR)
      GOTO 10

1199  FORMAT('"AR"--Add or Replace columns of an existing file')
1200  CALL GRIDREAD (IROW, IAXIS)
      GOTO 10

1299  FORMAT ('"EX"--EXit the program')
1300  WRITE (1,*) 'Program complete.'
      CALL GEND (IDWORK)                !DISABLE WORK STATION
      END
```

```
$CDS ON
!--------------------------------------------------------------!
!                                                              !
!     SUBROUTINE CLSTAT   Last Revised:  5/30/88               !
!                                                              !
!     Initializes 8510's to collect a row of data by putting   !
!         them in the Fast CW mode (which also clears their    !
!         data buffers.)                                       !
!                                                              !
!     Subroutines called:                                      !
!         None                                                 !
!                                                              !
!--------------------------------------------------------------!

      SUBROUTINE CLSTAT (NPOL)

      COMMON /PICS/ IPIC1, IPIC2, IPIC3
      COMMON /HPIB/ I4833, J4833, I8510, J8510, I8340, J8340


      WRITE (I8510,*) 'CLES;'            ! Clear Status bytes and SRQ
      WRITE (I8510,*) 'SINP;FASC;'       ! Set up Single-point (Fast CW) mode

10    CALL STATS (I8510, ISTAT)
      IF (BTEST(ISTAT,2)) THEN          ! Wait for 8510 to be ready
          CALL TRIGR (I8510)            ! Send HPIB GET (External Trigger)
      ELSE
          GO TO 10                      ! Not ready; sample again
      END IF

      IF (NPOL .EQ. 2) THEN

          WRITE (J8510,*) 'CLES;'        ! Clear Status bytes and SRQ
          WRITE (J8510,*) 'SINP;FASC;'   ! Set up Fast CW mode

20        CALL STATS (J8510,ISTAT)
          IF (BTEST(ISTAT,2)) THEN       ! Wait for 8510
              CALL TRIGR (J8510)             ! Send HPIB GET
          ELSE
              GO TO 20                   ! Not ready
          END IF

      END IF

      RETURN
      END
```

```
$CDS ON
$EMA /BUFFER/,/BUFFER2/,/POSN/
!------------------------------------------------------------!
!                                                            !
!     SUBROUTINE COLLECT          Last Revised:  6/03/88     !
!                                                            !
!     This subroutine is used to do the actual data collection. !
!         The probe is scanned from row IBEG to row IEND.  If !
!         this represents the entire data set, then the min and !
!         max values will be accurate.                       !
!                                                            !
!     Subroutines called:                                    !
!         MTRUSS                                             !
!         POSCHECK                                           !
!         SIDECHECK                                          !
!         VOLIN                                              !
!         VWPTJ                                              !
!         WRITE_DATA  (READWRITE)                            !
!                                                            !
!------------------------------------------------------------!

      SUBROUTINE COLLECT (IBEG, IEND, IROW, NPOL, IPLOT)

      COMMON /MINMAX/  AMIN, AMAX, PMIN, PMAX, MAXY, MAXX
      COMMON /MINMAX2/  AMIN2, AMAX2, PMIN2, PMAX2, MAXY2, MAXX2
      COMMON /BUFFER/  ABUF(4095), PBUF(4095), IBUF
      COMMON /BUFFER2/  ABUF2(4095), PBUF2(4095), IBUF2
      COMMON /POSN/  XPOS(4095), YPOS(4095)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /LASER/  CTI, VOL, DPI, CPOS

      IAXIS=0               ! COLLECTING ALONG Y AXIS
      IUNIT=3                ! PRIMARY POLE UNIT #
      IUNIT2=4              ! SECONDARY POLE UNIT #

10    FORMAT(A)

      DO IROW=IBEG,IEND
          CALL VOLIN (IPIC1,VOL,IERR)  !READ IN VOL COMP. #
          TPOS=XPOS(IROW)        !TRUSS POSITION
          CALL POSCHECK            !COMPARE AGAINST ENCODERS

          CALL MTRUSS (TPOS,0)     !MOVE TRUSS
          CALL SIDECHECK (IROW)   !COLLECT ROW OF DATA

          CALL WRITE_DATA (IUNIT,IROW,2,2,ABUF,PBUF,IBUF,AMIN,AMAX,PMIN,
     +                                  PMAX,MAXY,MAXX)
          IF (NPOL .EQ. 2) CALL WRITE_DATA (IUNIT2,IROW,2,2,ABUF2,PBUF2,
     +                    IBUF2,AMIN2,AMAX2,PMIN2,PMAX2,MAXY2,MAXX2)

          IF (IPLOT .EQ. 1) CALL VWPTJ (IROW,IAXIS,ABUF,PBUF)  !PLOT DATA
      END DO
      IROW=IROW-1

      RETURN
```

```
      END
```

```
$CDS ON
$EMA /POSN/, /BUFFER/, /BUFFER2/
!-------------------------------------------------------------!
!                                                             !
!    SUBROUTINE COLREAD          Last Revised:  6/03/88       !
!                                                             !
!    This subroutine reads in one column of data and stores it !
!        in the buffer(s).                                    !
!                                                             !
!                                                             !
!    Subroutines called:                                      !
!        MTRUSS                                               !
!        POSCHECK                                             !
!        SIDECHECK                                            !
!        VOLIN                                                !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE COLREAD (IROW, IAXIS)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /BUFFER/  ABUF(4096), PBUF(4096), IBUF
      COMMON /BUFFER2/  ABUF2(4096), PBUF2(4096), IBUF2
      COMMON /POSN/  XPOS(4096), YPOS(4096)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /USER/  IWRITE, IREAD
      COMMON /LASER/  CTI, VOL, DPI, CPOS

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER ANS*4


      IAXIS = 0                ! Collecting along Y axis
      NROWS = RSCAN(3)          ! Number of rows


C   Section to input row number from user

      WRITE(IWRITE,*) 'Which column do you wish to read?'
10    READ (IREAD, '(A)') ANS
      IF (ANS .LE. ' ')  RETURN        ! Default is to quit

      READ (ANS,*) IROW
      IF ( (IROW .LT. 1) .OR. (IROW .GT. NROWS) )  THEN
          WRITE (1,*) 'Invalid column number! Try again...'
          GO TO 10
      END IF

C   Section to collect data

      CALL VOLIN (IPIC1, VOL, IERR)     ! READ VOL COMP. #
      TPOS = XPOS(IROW)            ! TRUSS POSITION
      CALL POSCHECK                 ! COMPARE TO ENCODERS
      CALL MTRUSS (TPOS,0)     ! MOVE TRUSS
```

```
      CALL SIDECHECK (IROW)   ! COLLECT COLUMN OF DATA

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!     SUBROUTINE DATETIME         Last Revised:  6/01/88      !
!                                                             !
!     This routine gets the current date and time from the system  !
!     clock and returns them in two integer arrays as follows:     !
!                                                             !
!             IDATE(1) = 2-digit year code                    !
!             IDATE(2) = month code (1-12)                    !
!             IDATE(3) = day (1-31)                           !
!             ITIME(1) = hours (0-23)                         !
!             ITIME(2) = minutes                              !
!             ITIME(3) = seconds                              !
!                                                             !
!                                                             !
!     Subroutines called:                                     !
!         None                                                !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE DATETIME (IDATE, ITIME)


      INTEGER IDATE(3), ITIME(3), ITIME11(5), IYEAR, IBUFF(15)
      CHARACTER  FBUFF*30, MONTH*4
      EQUIVALENCE (FBUFF,IBUFF)

      CALL EXEC (11,ITIME11,IYEAR)       ! Numerical time
      CALL FTIME (IBUFF)                  ! Formatted time

      IDATE(1) = IYEAR - 1900
      ITIME(1) = ITIME11(4)
      ITIME(2) = ITIME11(3)
      ITIME(3) = ITIME11(2)

      READ (FBUFF,90)  IDATE(3), MONTH

90    FORMAT (16X, I2, 2X, A4)

      IF (MONTH .EQ. 'JAN.')  IDATE(2) = 1
      IF (MONTH .EQ. 'FEB.')  IDATE(2) = 2
      IF (MONTH .EQ. 'MAR.')  IDATE(2) = 3
      IF (MONTH .EQ. 'APR.')  IDATE(2) = 4
      IF (MONTH .EQ. 'MAY ')  IDATE(2) = 5
      IF (MONTH .EQ. 'JUNE')  IDATE(2) = 6
      IF (MONTH .EQ. 'JULY')  IDATE(2) = 7
      IF (MONTH .EQ. 'AUG.')  IDATE(2) = 8
      IF (MONTH .EQ. 'SEPT')  IDATE(2) = 9
      IF (MONTH .EQ. 'OCT.')  IDATE(2) = 10
      IF (MONTH .EQ. 'NOV.')  IDATE(2) = 11
      IF (MONTH .EQ. 'DEC.')  IDATE(2) = 12

      RETURN
```

```
      END
```

```
$ CDS ON
!-------------------------------------------------------------------!
!                                                                   !
!     SUBROUTINE DEFINE            Last Revised:  6/03/88           !
!                                                                   !
!     Returns the scan parameters for a particular axis of the      !
!         given data set.  The scan parameters are the starting     !
!         position (START), sample spacing (RINC), and number       !
!         of points per row (NUMPTS).  Set IAXIS = 0 for            !
!         Y-axis cuts, 1 for X-axis cuts.                           !
!                                                                   !
!                                                                   !
!     Subroutines called:                                           !
!         None                                                      !
!                                                                   !
!-------------------------------------------------------------------!

      SUBROUTINE DEFINE (IAXIS, START, RINC, NUMPTS)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL

      CHARACTER  CAXIS*1, POL*8, CSCAN*80, NAME*15


      IF (IAXIS .EQ. 0) THEN        ! Y-axis scan parameters
          START = RSCAN(4)
          RINC = RSCAN(5)
          NUMPTS = RSCAN(6)
      ELSE                          ! X-axis scan parameters
          START = RSCAN(1)
          RINC = RSCAN(2)
          NUMPTS = RSCAN(3)
      END IF

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------------!
!                                                                   !
!     SUBROUTINE DELAY              Last Revised:  5/20/88          !
!                                                                   !
!     This subroutine kills time in a loop for the requested        !
!        number of milliseconds (INTERVAL).  The resolution         !
!        is 10 msec.                                                !
!                                                                   !
!                                                                   !
!     Subroutines called:                                           !
!        None                                                       !
!                                                                   !
!-------------------------------------------------------------------!

      SUBROUTINE DELAY(INTERVAL)

      INTEGER *4 Itime,ElapsedTime


      CALL ResetTimer                    ! Set ElapsedTime to 0
      ITIME=ElapsedTime()

      DO WHILE (Itime .LT. INTERVAL)
         ITIME=ElapsedTime()
      END DO

      RETURN
      END
```

```
$CDS ON
$EMA /POSN/
!-------------------------------------------------------------!
!                                                             !
!    SUBROUTINE DRWJ              Last Revised:  6/03/88      !
!                                                             !
!                                                             !
!    This subroutine does the actual plotting of the data in  !
!        the buffer.                                          !
!            IGR = 0 - Plot amplitude data                    !
!                = 1 - Plot phase data                        !
!            NP  - Number of points to be plotted             !
!                                                             !
!                                                             !
!                                                             !
!    Subroutines called:                                      !
!        PDEF                                                 !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE DRWJ (IGR, IAXIS, NP, BUFR)

      EMA  BUFR(4096)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /POSN/  XPOS(4096), YPOS(4096)
      COMMON /AMP/  VHI, VLO, YMAX, YMIN

      CHARACTER  CAXIS*1, POL*8, CSCAN*80, NAME*15


      CALL JCOLR(IGR+2)
      H=PDEF(IAXIS,1)        !STARTING PT
      V=BUFR(1)        !STARTING VALUE
      IF (IGR .EQ. 0 .AND. V .LT. VLO) V=VLO
      CALL J2MOV (H,V)     !MOVE TO FIRST PT
      DO J=2,NP
          H=PDEF(IAXIS,J)      !HORIZONTAL VALUE
          V=BUFR(J)        !VERTICAL VALUE
          IF (IGR .EQ. 0 .AND. V .LT. VLO) V=VLO
          CALL J2DRW(H,V)     !DRAW LINE TO NEXT PT
      END DO

      RETURN
      END
```

```
$CDS ON
$EMA /BUFFER/, /BUFFER2/
!--------------------------------------------------------------!
!                                                              !
!    SUBROUTINE EFILE            Last Revised:  6/06/88        !
!                                                              !
!    This subroutine opens a data file and allows the user to  !
!        specify one row or column at a time to be read into   !
!        memory.  Each row can then be plotted or listed on    !
!        the terminal screen.                                  !
!            IROW identifies the row of data currently         !
!                    in the buffer.                            !
!            IAXIS specifies along which axis the data was     !
!                    collected (only Y-axis scans are          !
!                    implemented).                             !
!                                                              !
!    Subroutines called:                                       !
!        HEADREAD  (HEADER)                                    !
!        LISTBUFFER                                            !
!        NAMFILE                                               !
!        READ_DATA  (READWRITE)                                !
!        STO_POSN                                              !
!        SWIPE                                                 !
!        VWPTJ                                                 !
!                                                              !
!--------------------------------------------------------------!

C     LAST REVISED 8/5/88

      SUBROUTINE EFILE (IROW, IAXIS)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                IDATE(3), ITIME(3), NPOL
      COMMON /BUFFER/  ABUF(4096), PBUF(4096), IBUF
      COMMON /BUFFER2/  ABUF2(4096), PBUF2(4096), IBUF2
      COMMON /USER/  IWRITE, IREAD
      COMMON /DATA_DIR/  DDIR

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER ANS*2, DDIR*15


      IUNIT=3
      CALL NAMFILE (IUNIT,0,DDIR)                   !OPEN OLD FILE
      CALL HEADREAD (IUNIT,IRDAT)
      CALL STO_POSN                  !STORE POSITION COORDINATES
      IF (CAXIS .EQ. 'Y') THEN    !STORED BY COLUMNS
          NPTS=RSCAN(3)
          IAXIS=0
      ELSE
          NPTS=RSCAN(6)
          IAXIS=1                             !STORED BY ROWS
      END IF

10    IF (CAXIS .EQ. 'X') THEN
```

```
          WRITE (IWRITE,*) 'Enter row number (RETURN to stop)'
      ELSE
          WRITE (IWRITE,*) 'Enter column number (RETURN to stop)'
      END IF

      READ (IREAD, '(A)') ANS
      IF (ANS .LE. ' ') THEN
          CLOSE (IUNIT)
          RETURN                    !QUIT
      END IF
      READ (ANS,*) IROW
      IF (IROW.LT.1 .OR. IROW.GT.NPTS)  THEN
          WRITE (1,*) 'Invalid column number!  Try again...'
          GO TO 10
      END IF


      CALL READ_DATA (IUNIT,IROW,IRDAT,2,ABUF,PBUF,IBUF)  !READ IROW INTO BUFF

      WRITE (IWRITE,*) 'Enter 0 to Plot the data,'
      WRITE (IWRITE,*) '      1 to List the data on the terminal, or'
      WRITE (IWRITE,*) '      2 to do Both.'
      WRITE (IWRITE,*) '      RETURN defaults to 0'
      READ (IREAD,20) ANS

      ICHOICE=0
      IF (ANS .GT. ' ') READ (ANS,*) ICHOICE

      IF (ICHOICE.EQ.0 .OR. ICHOICE.EQ.2)  THEN
          CALL VWPTJ (IROW,IAXIS,ABUF,PBUF)        !PLOT ROW
          READ (IREAD,20)  ANS
          CALL SWIPE
      END IF

      IF (ICHOICE.EQ.1 .OR. ICHOICE.EQ.2)  THEN
          CALL LISTBUFFER (IROW,IAXIS,ABUF,PBUF,IBUF)     !LIST ROW
      END IF

      GOTO 10

20    FORMAT (A)

      END
```

```
$CDS ON
!------------------------------------------------------------!
!                                                            !
!    SUBROUTINE ENCODE           Last Revised:  5/23/88      !
!                                                            !
!    This subroutine reads positions of cart & truss ends from  !
!        the rotary encoders.  Arguments are as follows:     !
!            CEPOS-- Cart encoder position output            !
!            TSEPOS-- Truss south end encoder position output   !
!            TNEPOS-- Truss north end encoder position output   !
!                                                            !
!                                                            !
!    Subroutines called:                                     !
!        None                                                !
!                                                            !
!------------------------------------------------------------!

      SUBROUTINE ENCODE(CEPOS,TSEPOS,TNEPOS)

      COMMON /PICS/ IPIC1, IPIC2, IPIC3
      COMMON /HPIB/ I4833, J4833, I8510, J8510, I8340, J8340

      INTEGER*4 DPOSN, DBUF(2), CEPOS, TSEPOS, TNEPOS
      LOGICAL  SIGN


C   Read probe cart position

      CALL EXEC (1, I4833, DBUF, -3)
      CALL MVBITS (DBUF, 12, 18, DPOSN, 0)
      SIGN = BTEST(DBUF(1), 30)
      IF (SIGN)  CEPOS = -1*DPOSN


C   Read truss positions

      CALL EXEC (1, J4833, DBUF, -5)
      CALL MVBITS (DBUF, 12, 18, DPOSN, 0)    ! Get North posn
      SIGN = BTEST(DBUF(1), 30)
      IF (SIGN)  TNEPOS = -1*DPOSN

      CALL MVBITS (DBUF(2), 24, 8, DPOSN, 0)  ! Get South posn
      CALL MVBITS (DBUF, 0, 10, DPOSN, 8)
      SIGN = BTEST(DBUF(1), 10)
      IF (SIGN)  TSEPOS = -1*DPOSN

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------------!
!                                                                   !
!     SUBROUTINE ERRTRUSS          Last Revised:  6/04/88           !
!                                                                   !
!     This subroutine prints a warning if the program ever         !
!         detects that the laser position indicators and the       !
!         rotary encoders have conflicting readings.               !
!             IERR - Error code from laser                          !
!             TEPOS - Truss position according to encoder          !
!             TPOS - Truss position according to laser             !
!                                                                   !
!                                                                   !
!     Subroutines called:                                           !
!         None                                                      !
!                                                                   !
!-------------------------------------------------------------------!

      SUBROUTINE ERRTRUSS (IERR, TEPOS, TPOS)

      CHARACTER ANS*2


      IF (IERR .EQ. 0  .OR.  IERR .EQ.5) THEN

          WRITE (1,*) 'WARNING: Laser position indicators do not agree'
          WRITE (1,*) '         with rotary encoders on position of'
          WRITE (1,*) '         translation beam!!'
          WRITE (1,*)
          WRITE (1,*) '      Encoder = ', TEPOS
          WRITE (1,*) '      Laser   = ', TPOS

          STOP

      END IF

      WRITE (1,*) 'WARNING: Laser Error ', IERR, ' occurred on read',
     +                        ' of truss position at ', TPOS

      WRITE (1,*) ' Hit RETURN to continue.'
      READ (1,*)  ANS

      RETURN



      END
```

```
$CDS ON
!-----------------------------------------------------------------!
!                                                                 !
!     SUBROUTINE GO_HOME          Last Revised:  6/06/88          !
!                                                                 !
!     This subroutine checks that the encoders and lasers agree   !
!         on the current truss position, and if so, moves the     !
!         probe cart and truss to the home position.  It assumes  !
!         that the encoders were reset to zero at the home        !
!         position some time prior to the subroutine call.        !
!                                                                 !
!                                                                 !
!     Subroutines called:                                         !
!         ENCODE                                                  !
!         MCART                                                   !
!         MTRUSS                                                  !
!         TRUSS_CHECK  (POSCHECK)                                 !
!         VOLIN                                                   !
!                                                                 !
!-----------------------------------------------------------------!

      SUBROUTINE GO_HOME (CPOS)

      COMMON /USER/  IWRITE, IREAD
      COMMON /PICS/  IPIC1, IPIC2, IPIC3


      CALL TRUSS_CHECK               ! Compare with encoders
      CALL VOLIN (IPIC1,VOL,IERR)     ! Read VOL compensation
      CALL MTRUSS (0.,1)             ! Move truss

      CALL ENCODE (CEPOS, TSEPOS, TNEPOS)

      IF ( ABS( CPOS-CEPOS ) .LE. .01) THEN
          CALL MCART (0., 0., 1)        ! Encoder agrees w/ expected
      ELSE
          CPOS = CEPOS                 ! Assume cart position not yet
          CALL MCART (0., 0., 1)    !      initialized
      END IF

      RETURN
      END
```

```
$CDS ON
$EMA /BUFFER/,/BUFFER2/,/POSN/
!----------------------------------------------------------------------!
!                                                                      !
!     SUBROUTINE GRIDREAD          Last Revised:  6/03/88              !
!                                                                      !
!     This subroutine is used to add or replace rows in an already     !
!         existing data set.  The scan parameters are read from        !
!         the header record of the primary-pole data file.  If         !
!         two poles were collected, the user is prompted for the       !
!         second file name and both poles are added or replaced.       !
!         NOTE:  Program HILO  should be run on any file               !
!             modified with this subroutine to insure the              !
!             max and min values are accurate.                         !
!                                                                      !
!     Subroutines called:                                              !
!         COLLECT                                                      !
!         DATETIME                                                     !
!         HEADREAD  (HEADER)                                           !
!         HEADWRITE  (HEADER)                                          !
!         MAXMIN                                                       !
!         NAMFILE                                                      !
!                                                                      !
!----------------------------------------------------------------------!

C     LAST REVISED 8/5/88

      SUBROUTINE GRIDREAD (IROW, IAXIS)


      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                    IDATE(3), ITIME(3), NPOL
      COMMON /USER/  IWRITE, IREAD
      COMMON /DATA_DIR/  DDIR
      COMMON /BUFFER/ABUF(4096),PBUF(4096),IBUF
      COMMON /BUFFER2/ABUF2(4096),PBUF2(4096),IBUF2
      COMMON /POSN/XPOS(4096),YPOS(4096)
      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER COM*1, DDIR*15


      IAXIS=0                ! COLLECTING ALONG Y AXIS
      IUNIT=3                 ! PRIMARY POLE UNIT #
      IUNIT2=4              ! SECONDARY POLE UNIT #

10    FORMAT(A)

      WRITE (IWRITE,*) 'NOTE: Program HILO should be run on any file'
      WRITE (IWRITE,*) 'updated with this subroutine.  Would you like'
      WRITE (IWRITE,*) 'to run HILO automatically ? (Y/N)'
      READ (IREAD,10) COM
      IHILO=1
      IF (COM .EQ. 'N' .OR. COM .EQ. 'n') IHILO=0

      CALL NAMFILE (IUNIT,0,DDIR)     ! Open primary file
```

```
      CALL HEADREAD (IUNIT,IRDAT)     !  and read scan parameters
C     NPOL = 1
C     IPOL = ICHAR(POL) - 48
C     IF (IPOL.EQ.1 .OR. IPOL.EQ.2)  NPOL=2

      IF (NPOL .EQ. 2) THEN
          CALL NAMFILE (IUNIT2,0,DDIR)    ! OPEN SECONDARY POLE FILE
          CALL HEADREAD (IUNIT2,IRDAT)
          CALL DATETIME (IDATE,ITIME)      ! READ DATE AND TIME
          CALL HEADWRITE (IUNIT2,IRDAT)   ! UPDATE DATE AND TIME
      END IF


      CALL HEADREAD (IUNIT,IRDAT)   ! Get scan parameters from header
      CALL DATETIME (IDATE,ITIME)      ! READ DATE AND TIME
      CALL HEADWRITE (IUNIT2,IRDAT)  !UPDATE DATE AND TIME

      IPLOT=0
      WRITE (IWRITE,*) 'Shoud each row be plotted ',
     +                      'after it is collected? (N/Y)'
      READ (IREAD,10) COM
      IF (COM .EQ. 'Y' .OR. COM .EQ. 'y') IPLOT=1


      IF (CAXIS .EQ. 'Y') THEN
          NROWS=RSCAN(3)            !NUMBER OF DATA COLUMNS IN FILE
          IAXIS=0
      ELSE
          NROWS=RSCAN(6)
          IAXIS=1
      END IF

17    WRITE (IWRITE,*) 'Enter first data column to be collected:'
      READ (IREAD,*,ERR=17) IBEG
      IF (IBEG .LT. 1 .OR. IBEG .GT. NROWS) GOTO 17

19    WRITE (IWRITE,*) 'Enter last data column to be collected:'
      READ (IREAD,*,ERR=19) IEND
      IF (IEND .LT. IBEG .OR. IEND .GT. NROWS) GOTO 19

      CALL COLLECT (IBEG, IEND, IROW, NPOL, IPLOT)

      IF (IHILO .EQ. 1) CALL MAXMIN (IUNIT,1)   !MAX AND MIN INFO. FOR PRIM.
      CLOSE(IUNIT)
      IF (NPOL .EQ. 2) THEN
          IF (IHILO .EQ. 1) CALL MAXMIN (IUNIT2,2)  !GET MAX AND MIN FOR SEC.
          CLOSE(IUNIT2)
      END IF

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!    SUBROUTINE HEADER            Last Revised:  6/03/88      !
!                                                             !
!    Entry points:                                            !
!        HEADREAD                                             !
!        HEADWRITE                                            !
!                                                             !
!    This routine reads or writes the header record of a data !
!        file depending on which entry point is used.         !
!            IUNIT - Unit number of the data file.            !
!            IRDAT - Indicates whether amplitude and/or       !
!                    phase information is stored in the file. !
!                                                             !
!                                                             !
!    Subroutines called:                                      !
!        None                                                 !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE HEADER

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /MINMAX/  AMIN, AMAX, PMIN, PMAX, MAXY, MAXX
      COMMON /USER/  IWRITE, IREAD

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15


      ENTRY HEADWRITE (IUNIT,IRDAT)     ! To write the header record

      INQUIRE(UNIT=IUNIT,IOSTAT=IERR,ERR=17,RECL=IRECLB)  !GET RECORD LENGTH
      NDUM=(IRECLB-168)/2        !NUMBER OF DUMMY VAR. TO WRITE OUT
      WRITE(UNIT=IUNIT,IOSTAT=IERR,ERR=17,REC=1) RSCAN,CAXIS,POL,CSCAN,
     +        NAME,IDATE,ITIME,AMIN,AMAX,PMIN,PMAX,MAXY,MAXX,IRDAT,
     +                NPOL,(IDUM,I=1,NDUM)

17    IF (IERR .GT. 0) THEN
          WRITE(IWRITE,*) 'ERROR ',IERR,' WRITING HEADER'
          PAUSE
      END IF

      RETURN

C

      ENTRY HEADREAD(IUNIT,IRDAT)           ! To read the header record

      READ (UNIT=IUNIT,IOSTAT=IERR,ERR=27,REC=1) RSCAN,CAXIS,POL,CSCAN,
     +       NAME,IDATE,ITIME,AMIN,AMAX,PMIN,PMAX,MAXY,MAXX,IRDAT,
     +       NPOL

27    IF (IERR .GT. 0) THEN
```

```
    WRITE(IWRITE,*) 'ERROR ',IERR,' READING HEADER'
END IF

RETURN


END
```

```
$CDS ON
!-------------------------------------------------------------------!
!                                                                   !
!     FUNCTION IERROR              Last Revised:  5/26/88           !
!                                                                   !
!                                                                   !
!     Function to determine if error has occured in the laser       !
!         metrology system, and if so, whether or not it is         !
!         recoverable.                                              !
!                                                                   !
!         IERR -- upon entry contains the upper eight bits of the   !
!             most significant word read in from the binary         !
!             interface card                                        !
!                                                                   !
!         IPIC -- LU of the PIC                                     !
!                                                                   !
!         IERROR -- upon exit contains error code.                  !
!                 = 0 if no error has occured                       !
!                 = 1 if measurement error has occured              !
!                 = 2 if reference error has occured                !
!                 = 3 if overflow error has occured                 !
!                 = 4 if VOL error has occured                      !
!                 = 5 if destination has been reached               !
!                                                                   !
!                                                                   !
!     Subroutines called:                                           !
!         None                                                      !
!                                                                   !
!-------------------------------------------------------------------!

      FUNCTION IERROR (IERR, IPIC)


      IF (IERR .LT. 240) THEN
          IERROR=0                  ! Upper 4 bits <> 1111 (No error)

      ELSE
          IERR=NOT(IERR)

          IF (BTEST(IERR,0)) THEN
              WRITE(1,*) 'Measurement Error at LU ', IPIC, '!'
              IERROR=1

          ELSE IF (BTEST(IERR,1)) THEN
              WRITE(1,*) 'Reference Error at LU ', IPIC, '!'
              IERROR=2

          ELSE IF (BTEST(IERR,2)) THEN
              WRITE(1,*) 'Overflow Error'   ! Recoverable error
              CALL EXEC(1,IPIC,IJUNK,1,0)   ! Finish previous read
              CALL EXEC(2,IPIC,240,1,0)     ! Reset error bits
              IERROR=3

          ELSE IF (BTEST(IERR,3)) THEN
              WRITE(1,*) 'VOL Error'
```

```
            CALL EXEC(1,IPIC,IJUNK,1,0)   ! Finish previous read
            CALL EXEC(2,IPIC,240,1,0)     ! Reset error bits
            IERROR=4

        ELSE
            WRITE(1,*) 'Comparator match'
            CALL EXEC(1,IPIC,IJUNK,1,0)   ! Finish previous read
            CALL EXEC(2,IPIC,240,1,0)     ! Reset error bits
            IERROR=5                      ! Comparator within tolerance
        END IF
    END IF

    RETURN
    END
```

```
$CDS ON
!-------------------------------------------------------------------!
!                                                                   !
!     SUBROUTINE INIT              Last Revised:  6/06/88           !
!                                                                   !
!     This subroutine initializes arrays and equipment used by      !
!        data-collection program XYZ.                               !
!                                                                   !
!                                                                   !
!     Subroutines called:                                           !
!        GO_HOME                                                    !
!        RESET                                                      !
!                                                                   !
!-------------------------------------------------------------------!

      SUBROUTINE INIT

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /EXP85/  AEXP(0:255)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /HPIB/  I4833, J4833, I8510, J8510, I8340, J8340
      COMMON /USER/  IWRITE, IREAD
      COMMON /LASER/  CTI, VOL, DPI, CPOS
      COMMON /DATA_DIR/  DDIR
      COMMON /TITLE/  CTITL(10)

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER CBUF*76, CTITL*28

      DIMENSION IBUF(40)

      EQUIVALENCE (IBUF,CBUF)


C     Initialize LU variables

      I4833 = 46
      J4833 = 47
      I8510 = 29
      J8510 = 35
      I8340 = 49
      J8340 = 48

C     See BLOCK DATA LASER for other variable initializations

C     Initialize ICS 4833 HPIB-Parallel Interface Units

10    FORMAT ('N', I2, ',TB1,TP1,TH0,E1,I1,V0123456789ABCDEF,O0,',
     +        'LB0,LP1,LH0,S1,A0,B0,R1,X1,C1,M00')

      I=7
      WRITE (CBUF,10) I
      CALL EXEC (2, I4833, IBUF, -76, 12)
```

```
      I=10
      WRITE (CBUF,10) I
      CALL EXEC (2, J4833, IBUF, -76, 12)


C     Reset laser system

      CALL GO_HOME (CPOS)         ! Move to home position
      CALL RESET(IPIC1)           ! Zero laser counters
      CALL RESET(IPIC2)
      CALL RESET(IPIC3)

C     Set up exponent table for FORM1 (8510 internal format)

      DO I=0,127
         AEXP(I)=2.**(I-15)
      END DO

      DO I=128,255
         AEXP(I)=2.**(I-271)
      END DO


C     Initialize titles

      CTITL(1)='1. Starting X=              '         ! RSCAN(1)
      CTITL(2)='2. X increment=             '         ! RSCAN(2)
      CTITL(3)='3. # X Pts=                 '         ! RSCAN(3)
      CTITL(4)='4. Starting Y=              '         ! RSCAN(4)
      CTITL(5)='5. Y increment=             '         ! RSCAN(5)
      CTITL(6)='6. # Y pts=                 '         ! RSCAN(6)
      CTITL(7)='7. Freq.(GHz)=              '         ! RSCAN(7)
      CTITL(8)='8. # Poles to collect =     '         ! NPOL
      CTITL(9)='9. Polarization(8 char max)='         ! POL
      CTITL(10)='10. Title (70 char max)=   '         ! CSCAN

      CAXIS = 'Y'                                     ! Scan axis

      RETURN
      END
```

```
$CDS ON
$EMA /POSN/
!-------------------------------------------------------------!
!                                                             !
!     SUBROUTINE LABJ             Last Revised:  6/03/88      !
!                                                             !
!                                                             !
!     This subroutine draws grids and labels on plots.        !
!                                                             !
!         IPRT = 0  - Plot amplitude values                   !
!              = 1  - Plot phase values                       !
!         IAXIS = 0  - Plot data from a Y-axis cut            !
!               = 1  - Plot data from a X-axis cut            !
!         IROW     - Row or column being plotted              !
!         P0       - Starting position                        !
!         P1       - Ending position                          !
!                                                             !
!                                                             !
!     Subroutines called:                                     !
!         None                                                !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE LABJ (IPRT, IAXIS, IROW, P0, P1)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /POSN/  XPOS(4096), YPOS(4096)
      COMMON /AMP/  VHI, VLO, YMAX, YMIN

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER CROW*4, XLAB*6, YLAB*6, TEMP*80
      INTEGER X(3), Y(3), LROW(2), ITITL(40)
      EQUIVALENCE (TEMP,ITITL), (CROW,LROW), (XLAB,X), (YLAB,Y)


      CALL  JDFNT(2,0.0,19,19HFONT2.DAT::GRAPHICS,0)   !DEFINE FONT FILE
      CALL JFONT(2)                                    !ACCESS FONT FILE
      SPACE=(P1-P0)/10.       !BREAK INTERVAL INTO TENTH
      XSIZ=SPACE/8                          !CHARACTER WIDTH
      CTR=(P1+P0)/2         !CENTER OF CHART
      POS=P0-SPACE        !INITIAL POSITION
10    FORMAT(F6.1)                  !FORMAT FOR  LABELS
20    FORMAT(I6)
      IF (IPRT .EQ. 1) GOTO 100     !PLOT PHASE OR AMP?


C  Section for drawing X-axis grid and labels for amplitude plots

      CALL JJUST(.5,0.0)            !JUSTIFIED BOTTOM CENTER
      YSIZ=(VHI-VLO)/20.            !CHARACTER HEIGHT
      CGAP=XSIZ/5.
      CALL JCSIZ(XSIZ,YSIZ,CGAP)
      CALL JCOLR(7)
      DO I=1,11
```

```
          POS=POS+SPACE
          CALL J2MOV(POS,VHI)          !MOVE TO RIGHT PT ON X AXIS
          CALL J2DRW(POS,VLO)          !DRAW GRID MARK
          WRITE(XLAB,10) POS             !PUT VALUE IN XLAB
          GAP=VHI+(VHI-VLO)/30
          CALL J2MOV(POS,GAP)            !MOVE TO LABEL PT
          CALL JTEXM(6,X)              !WRITE OUT LABELS
      END DO
      CALL JJUST(.5,0.)          !JUSTIFIED BOTTOM CENTER
      CALL J2MOV(CTR,YMIN)            !BOTTOM CENTER OF VIEWPORT
      CALL JTEXM(22,22HAMPLITUDE VS. POSITION)  !AMPLITUDE TITLE


C  Section for drawing Y-axis grid and labels for amplitude plots

      RINC=(VHI-VLO)/9.               !BREAK UP INTO 10 INTERVALS
      R=VHI+RINC
      CALL JJUST(1.0,.5)              !JUSTIFIED CENTER RIGHT
      DO L=1,10
          R=R-RINC                      !MAKE L REAL NO.
          CALL J2MOV(P0,R)        !MOVE TO Y AXIS
          CALL J2DRW(P1,R)       !DRAW Y GRID LINE
          WRITE (YLAB,10) R             !PUT VALUE IN YLAB
          V=P0-XSIZ            !LEAVE ROOM FOR ONE BLANK OFF AXIS
          CALL J2MOV(V,R)              !MOVE TO Y LABEL PT.
          CALL JTEXM(6,Y)               !WRITE OUT LABELS
      END DO

      GOTO 200


C  Section for drawing X-axis grid and labels for phase plots

100   CALL JCOLR(7)
      YSIZ=270./8.                          !CHARACTER WIDTH
      CGAP=XSIZ/5.
      CALL JCSIZ(XSIZ,YSIZ,CGAP)       !SET CHARACTER SIZE
      DO I=1,11
          POS=POS+SPACE              !INCREASE X POSITION
          CALL J2MOV(POS,-180.)          !MOVE TO PT AT BOTTOM OF GRAPH
          CALL J2DRW(POS,180.)           !DRAW X GRID LINE
      END DO
      CALL JJUST(.5,0.)                  !JUSTIFIED BOTTOM CENTER
      CALL J2MOV(CTR,-230.)              !BOTTOM CENTER
      CALL JTEXM(18,18HPHASE VS. POSITION)  !WRITE OUT PHASE TITLE


C  Section for drawing Y-axis grid and labels for phase plots

      CALL JJUST(1.0,.5)              !JUSTIFIED CENTER RIGHT
      DO L=-180,180,90
          R=L                           !MAKE L REAL NO.
          CALL J2MOV(P0,R)        !MOVE TO PT ON Y AXIS
          CALL J2DRW(P1,R)      !DRAW Y GRID LINE
          WRITE(YLAB,20) L              !PUT VALUE IN YLAB
```

```
          V=PO-XSIZ              !LEAVE SPACE OFF AXIS
          CALL J2MOV(V,R)               !MOVE TO RIGHT PT
          CALL JTEXM(6,Y)               !WRITE LABELS
      END DO


C  Section for writing title

      CALL JJUST(.5,0.)        !JUSTIFIED BOTTOM CENTER
      I3=INDEX(CSCAN,'   ')   !# ACTUAL CHAR. IN CSCAN
      IF (I3 .EQ. 0) I3=80
      I4=INDEX(NAME,'  ')      !# ACTUAL CHAR. IN FILENAME
      IF (I4 .EQ. 0) I4=15
      I=I3+I4+2               !TOTAL # OF CHARACTERS
      IF (I .GT. 80) I=80
      TEMP=NAME(1:I4)//': '//CSCAN(1:I3)
      CALL J2MOV(CTR,255.)    !MOVE TO TOP CENTER OF VIEWPORT
      CALL JTEXM(I,ITITL)     !WRITE OUT TITLE

      CALL J2MOV(CTR,210.)
      IF (IAXIS .EQ. 0) THEN
          CALL JTEXM(7,7HCOLUMN )     !WRITE OUT COLUMN
      ELSE
          CALL JTEXM(4,4HROW )       !OR ROW HEADER
      END IF
      CALL JJUST(0.,0.)
      WRITE (CROW,'(I4)') IROW           !PUT VALUE IN CHAR. VARIABLE
      CALL JTEXM(4,LROW)


200   RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!    BLOCK DATA LASER            Last Revised:  6/03/88       !
!                                                             !
!                                                             !
!                                                             !
!    For use with HP 5501 laser metrology system.             !
!                                                             !
!    This block data routine assigns values to instructions   !
!        for the laser electronics cards in the three 10740A  !
!        coupler boxes.  It is assumed that the 10746A binary !
!        interface cards (BIC's) are set for positive logic   !
!        (high = true).  Each instruction is associated with  !
!        a backplane card address in the 10740A coupler.  The !
!        interpretation of the instruction may vary depending !
!        on the card located at that address.  For example,   !
!        suppose that in one coupler a counter card has been  !
!        assigned address "A", while in a second coupler a    !
!        comparator card has been assigned address "A".  The  !
!        instruction "3A" (which can be applied by outputting !
!        the value IA(3)) to the second coupler would cause the !
!        comparator card to load its destination register.  The !
!        same instruction to the first coupler would have no  !
!        effect, since instruction "3" is not implemented for the !
!        counter.  A complete list of instructions, their values, !
!        and their meanings to different cards, can be found  !
!        in the 5501A Laser Transducer System  Operating and  !
!        Service Manual, section 4.9.  The manual shows each  !
!        instruction as two characters, a numeral and a letter. !
!        The numeral indicates an operation to be performed   !
!        and the letter indicates the card address.  Thus,    !
!        "5Z" represents operation 5 to be performed by card Z. !
!        In the software, the value of this instruction is stored !
!        in array element IZ(5).  The other array elements are !
!        defined similarly.                                   !
!    Some commands require additional parameters (consult the !
!        manual).  Address P will always correspond to the    !
!        binary interface cards (10746A).  The addresses of the !
!        remaining cards are as follows:                      !
!                                                             !
!                                                             !
!        Coupler #1 ------ addressed via PIC at LU # 54       !
!                                                             !
!            Counter #1 (10760)----------------address X      !
!                (Delta Zc')                                  !
!            Counter #2 (10760)----------------address Y      !
!                (Delta Zt')                                  !
!            Counter #3 (10760)----------------address Z      !
!                (Delta Zc, Delta Zt)                         !
!            Counter #4 (10760)----------------address A      !
!                (Delta Xc, Delta Yt)                         !
!            Counter #5 (10760)----------------address B      !
!                (Theta Xc)                                   !
!            Counter #6 (10760)----------------address C      !
```

```
!            (Theta Zc)                                          !
!        Compensator interface (10755)-----address V             !
!            (for Velocity of Light compensation)                !
!                                                                !
!                                                                !
!    Coupler #2 ------ addressed via PIC at LU # 55              !
!                                                                !
!        Counter #1 (10760)----------------address C             !
!            (Theta Yt)                                          !
!        Counter #2 (10760)----------------address Z             !
!            (Theta Yt')                                         !
!        Comparator #1 (10762)-------------address A             !
!            (Xt - North end of truss)                           !
!        Comparator #2 (10762)-------------address X             !
!            (Xt' - South end of truss)                          !
!        Fast pulse converter #1 (10764)---address A             !
!            (to Comparator #1)                                  !
!        Fast pulse converter #2 (10764)---address X             !
!            (to Comparator #2)                                  !
!                                                                !
!                                                                !
!    Coupler #3 ------ addressed via PIC at LU # 56              !
!                                                                !
!        Comparator #1 (10762)-------------address X             !
!            (Yc - Southbound Cart)                              !
!        Comparator #2 (10762)-------------address Y             !
!            (8510 Trigger for Southbound cart)                  !
!        Comparator #3 (10762)-------------address Z             !
!            (Yc' - Northbound Cart)                             !
!        Comparator #4 (10762)-------------address A             !
!            (8510 Trigger for Northbound cart)                  !
!        Fast pulse converter #1 (10764)---address X             !
!            (to Comparator #1)                                  !
!        Fast pulse converter #1 (10764)---address Y             !
!            (to Comparator #2)                                  !
!        Fast pulse converter #2 (10764)---address Z             !
!            (to Comparator #3)                                  !
!        Fast pulse converter #2 (10764)---address A             !
!            (to Comparator #4)                                  !
!                                                                !
!                                                                !
!  ********************List of variables*****************        !
!                                                                !
!       IPIC1----lu of PIC that communicates with coupler #1     !
!       IPIC2----lu of PIC that communicates with coupler #2     !
!       IPIC3----lu of PIC that communicates with coupler #3     !
!       I4833----lu of ICS 4833 HPIB Adapter #1                  !
!       J4833----lu of ICS 4833 HPIB Adapter #2                  !
!       I8510----lu of primary pole 8510                         !
!       J8510----lu of secondary pole 8510                       !
!       I8340----lu of source #1                                 !
!       J8340----lu of source #2                                 !
!                                                                !
!       IA( )------array of binary values for instructions       !
!              to address A in the coupler (e.g.,IA(1)=1A)       !
```

```
!        The other array variables are similar                   !
!                                                                !
!                                                                !
!----------------------------------------------------------------!

      BLOCK DATA LASER


      COMMON /ASSIGN/  IA(0:7), IB(0:7), IC(0:7), IP(0:7),
     +                 IV(0:7), IX(0:7), IY(0:7), IZ(0:7)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /HPIB/  I4833, J4833, I8510, J8510, I8340, J8340
      COMMON /LASER/  CTI, VOL, DPI, CPOS


      DATA IA /16,17,18,19,20,21,22,23/
      DATA IB /32,33,34,35,36,37,38,39/
      DATA IC /48,49,50,51,52,53,54,55/
      DATA IP /0,1,2,3,4,5,6,7/
      DATA IV /96,97,98,99,100,101,102,103/
      DATA IX /128,129,130,131,132,133,134,135/
      DATA IY /144,145,146,147,148,149,150,151/
      DATA IZ /160,161,162,163,164,165,166,167/

      DATA IPIC1, IPIC2, IPIC3 /54, 55, 56/

      DATA CTI, DPI /6.23E-6, 0./

      END
```

```
$CDS ON
$EMA /BUFFER/, /BUFFER2/
!-----------------------------------------------------------------!
!                                                                 !
!     SUBROUTINE EFILE             Last Revised:  6/06/88         !
!                                                                 !
!     This subroutine opens a data file and allows the user to    !
!         specify one row or column at a time to be read into     !
!         memory.  Each row can then be plotted or listed on      !
!         the terminal screen.                                    !
!             IROW identifies the row of data currently           !
!                     in the buffer.                              !
!             IAXIS specifies along which axis the data was        !
!                     collected (only Y-axis scans are            !
!                     implemented).                               !
!                                                                 !
!     Subroutines called:                                         !
!         HEADREAD  (HEADER)                                      !
!         LISTBUFFER                                              !
!         NAMFILE                                                 !
!         READ_DATA  (READWRITE)                                  !
!         STO_POSN                                                !
!         SWIPE                                                   !
!         VWPTJ                                                   !
!                                                                 !
!-----------------------------------------------------------------!

C     LAST REVISED 8/5/88

      SUBROUTINE EFILE (IROW, IAXIS)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                    IDATE(3), ITIME(3), NPOL
      COMMON /BUFFER/  ABUF(4096), PBUF(4096), IBUF
      COMMON /BUFFER2/  ABUF2(4096), PBUF2(4096), IBUF2
      COMMON /USER/  IWRITE, IREAD
      COMMON /DATA_DIR/  DDIR

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER ANS*2, DDIR*15


      IUNIT=3
      CALL NAMFILE (IUNIT,0,DDIR)                       !OPEN OLD FILE
      CALL HEADREAD (IUNIT,IRDAT)
      CALL STO_POSN                  !STORE POSITION COORDINATES
      IF (CAXIS .EQ. 'Y') THEN    !STORED BY COLUMNS
          NPTS=RSCAN(3)
          IAXIS=0
      ELSE
          NPTS=RSCAN(6)
          IAXIS=1                             !STORED BY ROWS
      END IF

10    IF (CAXIS .EQ. 'X') THEN
```

```
    WRITE (IWRITE,*) 'Enter row number (RETURN to stop)'
ELSE
    WRITE (IWRITE,*) 'Enter column number (RETURN to stop)'
END IF

READ (IREAD, '(A)') ANS
IF (ANS .LE. ' ') THEN
    CLOSE (IUNIT)
    RETURN                    !QUIT
END IF
READ (ANS,*) IROW
IF (IROW.LT.1 .OR. IROW.GT.NPTS)  THEN
    WRITE (1,*) 'Invalid column number!  Try again...'
    GO TO 10
END IF


CALL READ_DATA (IUNIT,IROW,IRDAT,2,ABUF,PBUF,IBUF)  !READ IROW INTO BUFF

WRITE (IWRITE,*) 'Enter 0 to Plot the data,'
WRITE (IWRITE,*) '      1 to List the data on the terminal, or'
WRITE (IWRITE,*) '      2 to do Both.'
WRITE (IWRITE,*) '      RETURN defaults to 0'
READ (IREAD,20) ANS

ICHOICE=0
IF (ANS .GT. ' ') READ (ANS,*) ICHOICE

IF (ICHOICE.EQ.0 .OR. ICHOICE.EQ.2)  THEN
    CALL VWPTJ (IROW,IAXIS,ABUF,PBUF)        !PLOT ROW
    READ (IREAD,20)  ANS
    CALL SWIPE
END IF

IF (ICHOICE.EQ.1 .OR. ICHOICE.EQ.2)  THEN
    CALL LISTBUFFER (IROW,IAXIS,ABUF,PBUF,IBUF)     !LIST ROW
END IF

GOTO 10

20      FORMAT (A)

END
```

```
$CDS ON
!-----------------------------------------------------------!
!                                                           !
!     SUBROUTINE LISTCHANGE        Last Revised:  5/19/88   !
!                                                           !
!     This subroutine will list the scan parameters and accept  !
!         changes from the user.                            !
!                                                           !
!                                                           !
!     Subroutines called:                                   !
!         SWIPE                                             !
!         STO_POSN                                          !
!                                                           !
!-----------------------------------------------------------!

      SUBROUTINE LISTCHANGE

      COMMON /PARAM/RSCAN(7),CAXIS,POL,CSCAN,NAME,IDATE(3),ITIME(3)
      COMMON /TITLE/CTITL
      COMMON /USER/IWRITE,IREAD

      CHARACTER CTITL(10)*28,CAXIS*1,POL*8,CSCAN*80,NAME*15,ANS*2


C   Print out scan parameters

10    CALL SWIPE

      WRITE(IWRITE,*) '                SCAN PARAMETERS'
      WRITE(IWRITE,*)

      DO I=1,7
         WRITE(IWRITE,*) CTITL(I),RSCAN(I)
      END DO

      CAXIS= 'Y'

      WRITE(IWRITE,*) CTITL(8),CAXIS
      WRITE(IWRITE,*) CTITL(9),POL

      WRITE(IWRITE,*) CTITL(10),CSCAN

      WRITE(IWRITE,*) '---------------------------------------------'
      WRITE(IWRITE,*)


C   Get changes from user

      WRITE(IWRITE,*) 'Enter the number of any parameter you wish to '
      WRITE(IWRITE,*) 'change (hit RETURN if everything is correct)'
      READ(IREAD,20) ANS
      IF (ICHAR(ANS) .EQ. 32) THEN
         CALL STO_POSN
         RETURN
      ELSE
```

```
            READ (ANS,*) IOPT
        END IF

13      IF (IOPT .LE. 0  .OR.  IOPT .GT. 10)  THEN
            GOTO 10
        ELSE
17          WRITE(IWRITE,*) CTITL(IOPT),'?'
            IF (IOPT .EQ. 9) READ(IREAD,20) POL      !POLARIZATION
            IF (IOPT .EQ. 10) READ(IREAD,20) CSCAN       !TITLE
            IF (IOPT .LT. 8) THEN
                READ(IREAD,*) RSCAN(IOPT)          !READ INTO REAL ARRAY
                IF (RSCAN(IOPT) .LT. 0) GOTO 17
                IF (IOPT .EQ. 3 .OR. IOPT .EQ. 6) THEN
                    IF (RSCAN(IOPT) .NE. INT(RSCAN(IOPT))) GOTO 17
                END IF
            END IF
        END IF

        GOTO 10                          !ANY MORE CHANGES?

20      FORMAT (A)

        END
```

```
$CDS ON
$EMA /BUFFER/, /BUFFER2/
!--------------------------------------------------------------------!
!                                                                    !
!     SUBROUTINE MAXMIN            Last Revised:  6/06/88            !
!                                                                    !
!     This subroutine finds the maximum and minimum amplitude        !
!         and phase values in a data set and writes them into        !
!         the header record.                                         !
!                                                                    !
!     Subroutines called:                                            !
!         HEADREAD  (HEADER)                                         !
!         HEADWRITE  (HEADER)                                        !
!         READ_DATA  (READWRITE)                                     !
!                                                                    !
!--------------------------------------------------------------------!

      SUBROUTINE MAXMIN (IUNIT, IPOL)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /MINMAX/  AMIN, AMAX, PMIN, PMAX, MAXY, MAXX
      COMMON /MINMAX2/  AMIN2, AMAX2, PMIN2, PMAX2, MAXY2, MAXX2
      COMMON /BUFFER/  ABUF(4096), PBUF(4096), IBUF
      COMMON /BUFFER2/  ABUF2(4096), PBUF2(4096), IBUF2

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15


      CALL HEADREAD (IUNIT,IRDAT)       !GET OLD FILE AND READ HEADER

      IF (CAXIS .EQ. 'X') THEN
          MAX1=RSCAN(3)
          MAX2=RSCAN(6)         !DATA STORED BY ROWS
      ELSE
          MAX1=RSCAN(6)        !DATA STORED BY COLUMNS
          MAX2=RSCAN(3)
      END IF

      IF (IPOL.EQ.1) THEN

          AMAX=-100.
          AMIN=100.
          PMAX=-180.
          PMIN=180.

          DO IROW=1,MAX2                               ! Check row-by-row
              CALL READ_DATA (IUNIT,IROW,IRDAT,IRDAT,ABUF,PBUF,IBUF)

              DO I=1,MAX1
                  IF (ABUF(I) .GT. AMAX) THEN
                      AMAX=ABUF(I)               ! highest amplitude
                      IF (CAXIS .EQ. 'Y') THEN
                          MAXY=IROW          ! Position coordinates of
                          MAXX=I             !     data point with
```

```
                ELSE                    !     the highest
                    MAXY=I              !     amplitude
                    MAXX=IROW           !
                END IF
            END IF
            IF (ABUF(I) .LT. AMIN) AMIN=ABUF(I)      ! lowest amp
            IF (PBUF(I) .LT. PMIN) PMIN=PBUF(I)      ! lowest phase
            IF (PBUF(I) .GT. PMAX) PMAX=PBUF(I)      ! highest phase
        END DO
    END DO

ELSE

    AMAX2=-100.
    AMIN2=100.
    PMAX2=-180.
    PMIN2=180.

    DO IROW=1,MAX2                              ! Check row-by-row
        CALL READ_DATA (IUNIT,IROW,IRDAT,IRDAT,ABUF2,PBUF2,IBUF2)

        DO I=1,MAX1
            IF (ABUF2(I) .GT. AMAX) THEN
                AMAX=ABUF2(I)               ! highest amplitude
                IF (CAXIS .EQ. 'Y') THEN
                    MAXY2=IROW          ! Position coordinates of
                    MAXX2=I             !     data point with
                ELSE                    !     the highest
                    MAXY2=I             !     amplitude
                    MAXX2=IROW          !
                END IF
            END IF
            IF (ABUF2(I) .LT. AMIN) AMIN=ABUF2(I)      ! lowest amp
            IF (PBUF2(I) .LT. PMIN) PMIN=PBUF2(I)      ! lowest phase
            IF (PBUF2(I) .GT. PMAX) PMAX=PBUF2(I)      ! highest phase
        END DO
    END DO

END IF

CALL HEADWRITE(IUNIT,IRDAT)         ! Update header record

RETURN
END
```

```
$CDS ON
!-----------------------------------------------------------------!
!                                                                 !
!     SUBROUTINE MCART             Last Revised:  6/03/88         !
!                                                                 !
!     Subroutine to move probe cart to desired position.          !
!                                                                 !
!     Parameter definitions:                                      !
!        DCPOS--Desired cart position                             !
!        CPOS--Current cart position                              !
!        TPOS--Truss position                                     !
!        IDIS = 1  Display position on terminal                   !
!             = 0  No display                                     !
!                                                                 !
!                                                                 !
!     Subroutines called:                                         !
!        POSOUT  (POSITION)                                       !
!        POSWATCH                                                 !
!                                                                 !
!-----------------------------------------------------------------!

      SUBROUTINE MCART(DCPOS,TPOS,IDIS)

      COMMON /ASSIGN/  IA(0:7), IB(0:7), IC(0:7), IP(0:7),
     +                  IV(0:7), IX(0:7), IY(0:7), IZ(0:7)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /HPIB/  I4833, J4833, I8510, J8510, I8340, J8340
      COMMON /LASER/  CTI, VOL, DPI, CPOS


10    IDIR=SIGN(1.,DCPOS-CPOS)          ! +1 FOR SOUTH TRAVEL,-1 FOR NORTH
      POFF=ABS(DCPOS-CPOS)              ! OFFSET FROM CURRENT POSITION
      IF (IDIR .EQ. 1) THEN
         BADR=IX(0)                   ! MOVING SOUTH
      ELSE
         BADR=IZ(0)                   ! MOVING NORTH
      END IF

      CALL EXEC (2,IPIC3,BADR,1,0)        !RESET COMPARATOR
      CALL POSOUT (IPIC3,POFF,15)        !OUTPUT POS. TO BIC
      CALL EXEC (2,IPIC3,BADR+3,1,0)      !LOAD UP CART DEST. REGISTER
      CALL EXEC (2,IPIC3,BADR+1,1,0)     !SAMPLE TO START DIGITAL DIFF.

C  Section to switch trigger MUX and enable motion

      IF (IDIR .EQ. 1) THEN
         WRITE (I4833,*) '01'           ! SWITCH MUX TRIGGER SOUTH BOUND
         WRITE (I4833,*) '11'           ! ENABLE CART MOTION
      ELSE
         WRITE (I4833,*) '02'           ! SWITCH MUX NORTH BOUND
         WRITE (I4833,*) '12'           ! ENABLE CART MOTION
      END IF


C  Section to monitor position
```

```
      CALL POSWATCH(IPIC3,BADR,TPOS,DCPOS,1,IDIS,IERR)
      IF (IERR .NE. 0 .AND. IERR .NE. 5) GOTO 10

      CPOS=DCPOS              !UPDATE CURRENT CART POSITION
      WRITE (I4833,*) '0'        !DISABLE CART MOTION

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!    SUBROUTINE MOVE              Last Revised:  6/06/88      !
!                                                             !
!    This subroutine prompts the user for the X & Y coordinates !
!        of the desired position and then moves truss and     !
!        probe cart to the destination.                       !
!                                                             !
!                                                             !
!    Subroutines called:                                      !
!        POSCHECK                                             !
!        VOLIN                                                !
!        MCART                                                !
!        MTRUSS                                               !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE MOVE

      COMMON /USER/  IWRITE, IREAD
      COMMON /PICS/  IPIC1, IPIC2, IPIC3


10    WRITE (IWRITE,*) 'Enter desired X position: '
      READ (IREAD,*,ERR=10) TPOS
20    WRITE (IWRITE,*) 'Enter desired Y position: '
      READ (IREAD,*,ERR=20) DCPOS

      CALL POSCHECK                ! Compare with encoders
      CALL VOLIN (IPIC1,VOL,IERR)  ! Read VOL compensation
      CALL MTRUSS (TPOS,1)         ! Move truss
      CALL MCART (DCPOS,TPOS,1)    ! Move cart

      WRITE (IWRITE,*) 'X= ',TPOS,',   Y= ',DCPOS

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!     SUBROUTINE MTRUSS            Last Revised:  6/03/88     !
!                                                             !
!     Subroutine to move truss to desired position            !
!                                                             !
!         TPOS--Desired truss position                        !
!         CPOS--Cart position                                 !
!         IDIS = 1 Display position on terminal               !
!              = 0 No display                                 !
!                                                             !
!                                                             !
!     Subroutines called:                                     !
!         POSOUT  (POSITION)                                  !
!         POSWATCH                                            !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE MTRUSS (TPOS, IDIS)

      COMMON /ASSIGN/  IA(0:7), IB(0:7), IC(0:7), IP(0:7),
     +                   IV(0:7), IX(0:7), IY(0:7), IZ(0:7)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /HPIB/  I4833, J4833, I8510, J8510, I8340, J8340


C   Section to load up comparators

10    CALL POSOUT (IPIC2,TPOS,15)      ! Output position to BIC
      CALL EXEC (2,IPIC2,IA(3),1,0)     ! Load truss north end
      CALL EXEC (2,IPIC2,IX(3),1,0)     ! Load truss south end
      CALL EXEC (2,IPIC2,IA(1),1,0)    ! Sample to start digital diff.
      CALL EXEC (2,IPIC2,IX(1),1,0)    !   "    "    "     "     "

      WRITE (I4833,*) '2'      ! Enable truss

C   Section to monitor truss position.  Looks for null on north end of
C     truss first, then checks south end.

      CALL POSWATCH(IPIC2,IA(0),TPOS,TPOS,0,IDIS,IERR)
      IF (IERR .NE. 0 .AND. IERR .NE. 5) GOTO 10
      CALL POSWATCH(IPIC2,IX(0),TPOS,TPOS,0,IDIS,IERR)
      IF (IERR .NE. 0 .AND. IERR .NE. 5) GOTO 10

      WRITE (I4833,*) '0'      ! Disable truss

30    RETURN
      END
```

```
$CDS ON
!---------------------------------------------------------------!
!                                                               !
!     SUBROUTINE NAMFILE          Last Revised:  6/03/88        !
!                                                               !
!     This subroutine opens a datafile for subsequent reads or  !
!         writes.  IUNIT is the unit number to be associated with !
!         the file.  ISTATUS is the status of the file:         !
!             ISTATUS = 0  -  New file                          !
!                     = 1  -  Old file                          !
!                     = 2  -  Status unknown                    !
!             DDIR is the data directory, if other than         !
!                     ::XYZFILES                                !
!                                                               !
!         LGBUF is a library subroutine to enlarge I/O buffer size. !
!         NOTE:  the buffer array LBUF must not be in EMA under any !
!             circumstances.                                    !
!         NOTE:  if CDS is used, then either the common block   !
!             /RECBUFF/ must be declared in the main program and !
!             this subroutine, or the call to LGBUF must be made !
!             in the main program (in which case /RECBUFF/ is not !
!             required.)                                        !
!                                                               !
!                                                               !
!     Subroutines called:                                       !
!         DATETIME                                              !
!                                                               !
!---------------------------------------------------------------!

C     SUBROUTINE NAMFILE (IUNIT, ISTATUS, DDIR)
      SUBROUTINE NAMFILE (IUNIT, ISTATUS)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                IDATE(3), ITIME(3), NPOL
      COMMON /RECBUFF/  LBUF(8200)
      COMMON /USER/  IWRITE, IREAD

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER DDIR*16, INFILE*30, STAT*7


C     NP = PCOUNT()           ! Number of parameters passed
C     IF (NP .LT. 3)  DDIR = '/XYZFILES '

C     ID = INDEX (DDIR, ' ') - 1    ! Length of string
C     IF (ID .LE. 0) ID=16

5     WRITE (IWRITE,*) 'Enter data file name:'
      READ (IREAD,20) NAME
20    FORMAT(A)
C     INFILE = DDIR(1:ID)// '/' // NAME
      INFILE = NAME//'::XYZFILES'

      IF (ISTATUS .EQ. 0) STAT='OLD    '
      IF (ISTATUS .EQ. 1) STAT='NEW    '
```

```
      IF (ISTATUS .EQ. 2) STAT='UNKNOWN'

      IF (STAT .EQ. 'NEW') THEN
          NPTS=RSCAN(6)
          IF (CAXIS .EQ. 'X') NPTS=RSCAN(3)
          IRECLB=(NPTS*4)+2  !RECORD LENGTH(BYTES)--AMP OR PHASE AND STATUS
          IF (IRECLB .LT. 180) IRECLB=180  !INSURE ENOUGH ROOM FOR HEADER REC.
          CALL DATETIME (IDATE,ITIME)
      ELSE
          INQUIRE(FILE=INFILE,IOSTAT=IERR,ERR=65,RECL=IRECLB) !READ RECORD LTH
      END IF


      OPEN(UNIT=IUNIT,FILE=INFILE,ACCESS='DIRECT',FORM='UNFORMATTED',
     +    RECL=IRECLB,IOSTAT=IERR,ERR=65,STATUS=STAT)


65    IF (IERR .GT. 0) THEN
          WRITE(IWRITE,*) 'ERROR ',IERR,' ON OPENING FILE'
          GOTO 5
      ELSE
          CALL LGBUF (LBUF,IRECLB/2)       !ENLARGE I/O BUFFER TO #BYTES/2
      END IF

      RETURN

      END
```

```
$CDS ON
$EMA /BUFFER/, /BUFFER2/, /POSN/
!--------------------------------------------------------------!
!                                                              !
!     SUBROUTINE NEGCOL            Last Revised:  6/03/88      !
!                                                              !
!     Collects data in a negative direction (probe moving      !
!        northbound).                                          !
!                                                              !
!                                                              !
!     Subroutines called:                                      !
!        CLSTAT                                                !
!        MCART                                                 !
!        POSOUT  (POSITION)                                    !
!        POSWATCH                                              !
!        RECTOPOL                                              !
!                                                              !
!--------------------------------------------------------------!

C     LAST REVISED 8/5/88

      SUBROUTINE NEGCOL (IROW)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /BUFFER/  ABUF(4096), PBUF(4096), IBUF
      COMMON /BUFFER2/  ABUF2(4096), PBUF2(4096), IBUF2
      COMMON /POSN/  XPOS(4096), YPOS(4096)
      COMMON /EXP85/  AEXP(0:255)
      COMMON /ASSIGN/  IA(0:7), IB(0:7), IC(0:7), IP(0:7),
     +                   IV(0:7), IX(0:7), IY(0:7), IZ(0:7)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /HPIB/  I4833, J4833, I8510, J8510, I8340, J8340
      COMMON /LASER/  CTI, VOL, DPI, CPOS

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      COMPLEX CDAT
      DIMENSION IDBUF(3,4096), IDBUF2(3,4096)
      LOGICAL NULL
      INTEGER CLASS1, CLASS1W, CLASS2, CLASS2W, AREG, BREG


      ICONT1 = I8510+100b     ! Control words for 8510 EXEC calls.  100b
      ICONT2 = J8510+100b     !     is the code for normal binary format
      NP = RSCAN(6)         ! # of points to be sampled in data column
      IFC1 = I8510 + 5100b               ! Control code for a
      IFC2 = J8510 + 5100b               !     10 msec IFC on HPIB


10    DCPOS = YPOS(NP)                     ! Start at last point
      CALL MCART (DCPOS,0.,0)                ! Move cart to start


C  Section to set up the "motion Comparator" to control the probe scan by
C     loading the destination register with the location of the last point.
```

```
      CALL EXEC (2,IPIC3,IZ(0),1,0)       ! Reset motion comparator
      DEST = ABS( CPOS-YPOS(1) )         ! Offset to beginning of data row
      CALL POSOUT (IPIC3,DEST,15)       ! Output destination to BIC
      CALL EXEC (2,IPIC3,IZ(3),1,0)      ! Load destination to motion Comp.
      CALL EXEC (2,IPIC3,IZ(1),1,0)     ! Start generating digital difference


C  Section to prepare "sample Comparator" and 8510(s) for data  collection.
C     The destination register of the sample comparator is loaded with the
C     location of the next data point to be sampled.  As each point is
C     reached the comparator's null output automatically triggers the
C     8510(s) (via the Trigger Control Electronics) to sample a data point.
C     A Class Read is used to get data from 8510's so this program does not
C     monitor the 8510's during a scan.  The null line from the comparator
C     is monitored so the program knows when to load the location of the
C     next data point to be sampled.

      CALL EXEC (2,IPIC3,IA(0),1,0)       ! Reset sample Comparator

      CALL EXEC (3, IFC1, 1)        ! Clear both interface cards
      CALL EXEC (3, IFC2, 1)        !

      CLASS1W = 0                  ! Initialize Class Numbers
      CLASS2W = 0                  !

      CALL CLRQ (1, CLASS1W)     ! Reserve class numbers from the system
      CALL CLRQ (1, CLASS2W)     !       for 8510 class reads

      CLASS1W = IBSET( CLASS1W, 13 )     ! Set "Save Class Number" bit
      CLASS2W = IBSET( CLASS2W, 13 )     !

      CLASS1 = IBSET( CLASS1W, 15 )    ! Set "No Wait" bit
      CLASS2 = IBSET( CLASS2W, 15 )    !

      CALL CLSTAT (NPOL, I8510, J8510)     ! Set 8510's in Fast CW mode

      CALL EXEC (17, ICONT1, IDBUF1, NP*3, 0, 0, CLASS1)   ! Do Class Read
      CALL ABREG (AREG, BREG)                              ! and check for
      LOCATION=1                                           ! errors
      IF (AREG.NE.0) GO TO 999                             !

      IF (NPOL .EQ. 2) THEN
         CALL EXEC (17, ICONT2, IDBUF2, NP*3, 0, 0, CLASS2)  ! Ditto, for
         CALL ABREG (AREG, BREG)                             ! 2nd 8510
         LOCATION=2                                          !
         IF (AREG.NE.0) GO TO 999                            !
      END IF


C  Section to do the actual data collection

      WRITE (IWRITE,*) 'Collecting data for Column ',IROW

      WRITE (I4833,*) '02'         ! Select Northbound
```

```
      WRITE (I4833,*) '12'          ! Enable motion

      DO I=NP,1,-1
          POFF = ABS(CPOS-YPOS(I))              ! Offset to sample point
          CALL POSOUT (IPIC3,POFF,15)         ! Output dest. to BIC
          CALL EXEC (2,IPIC3,IA(3),1,0)         ! Load up Sample Comparator
          CALL EXEC (2,IPIC3,IA(1),1,0)       ! Start digital difference

C         CALL POSWATCH (IPIC3,IA(0),0.,POFF,1,0,IERR) ! Wait
C         IF (IERR .NE. 0 .AND. IERR .NE. 5) THEN
C             WRITE (IWRITE,*) 'Error reading laser position.  Move to'
C             WRITE (IWRITE,*) '    Home postion and start over.'
C             STOP
C         END IF

          NULL = .FALSE.
          DO WHILE (.NOT. NULL)
              CALL EXEC (1, I4833, INULL, -1)  ! Read null from Sample comp.
              NULL = BTEST (INULL, 7)          ! Null line at bit 7
          END DO

      END DO


C  Section to check for successful completion of column scan

      CALL POSWATCH (IPIC3, IZ(0), 0., DEST, 1, 0, IERR)
      IF (IERR .NE. 0 .AND. IERR .NE. 5) THEN
          WRITE (IWRITE,*) 'ERROR in scan of column # ', IROW
          CALL CLRQ (2, CLASS1W)
          CALL CLRQ (2, CLASS2W)
          GO TO 10
      END IF

      WRITE(I4833,*) '0'        ! Disable motion
      CPOS=YPOS(1)              ! New cart position


C  Section to get the last point(s) from the Class Reads

      CALL EXEC (21, CLASS1W, IDBUF1, NP*3)     ! Class Get
      CALL ABREG (AREG, BREG)
      LOCATION =3
      IF (AREG .LT. 0) GO TO 999

      IF (NPOL .EQ. 2) THEN
          CALL EXEC (21, CLASS2W, IDBUF2, NP*3)     ! Class Get
          CALL ABREG (AREG, BREG)
          LOCATION =4
          IF (AREG .LT. 0) GO TO 999
      END IF


C  Section to convert data to amplitude/phase format
```

```
      DO I=1,NP
          J = NP-I+1

          EX = AEXP( IAND( IDBUF(3,I), 255 ) )      ! Exponent
          RE = IDBUF(2,I)*EX                        ! Real part
          RIM = IDBUF(1,I)*EX                       ! Imaginary part
          CDAT = CMPLX(RE,RIM)              ! Convert to complex form and
          CALL RECTOPOL (CDAT,AMP,PHSE)     !    then to amplitude, phase
          ABUF(J) = AMP                        ! Store in buffers
          PBUF(J) = PHSE                       !

          IF (NPOL .EQ. 2) THEN
              EX = AEXP( IAND( IDBUF2(3,I), 255 ) )   ! Do the same
              RE = IDBUF2(2,I)*EX                     !    for the
              RIM = IDBUF2(1,I)*EX                    !    second
              CDAT = CMPLX(RE,RIM)                    !    pole data
              CALL RECTOPOL(CDAT,AMP,PHSE)            !
              ABUF2(J)=AMP                            !
              PBUF2(J)=PHSE                           !
          END IF
      END DO

      RETURN

999   WRITE (IWRITE,*) 'ERROR on Class Read or Get at location ',
     +                    LOCATION

      STOP


      END
```

```
$CDS ON
$EMA /POSN/
!-------------------------------------------------------------------!
!                                                                   !
!     FUNCTION PDEF                   Last Revised:  6/01/88        !
!                                                                   !
!     This function returns the location of point IPT.              !
!         If LBUF = 0 program returns Y-position;                   !
!                 = 1    "        "    X-position.                  !
!                                                                   !
!                                                                   !
!     Subroutines called:                                           !
!         None                                                      !
!                                                                   !
!-------------------------------------------------------------------!

      FUNCTION PDEF (LBUF, IPT)

      COMMON /POSN/ XPOS(4096), YPOS(4096)


      IF (LBUF .EQ. 0) PDEF=YPOS(IPT)
      IF (LBUF .EQ. 1) PDEF=XPOS(IPT)

      RETURN
      END
```

```
$CDS ON
!--------------------------------------------------------------------!
!                                                                    !
!     SUBROUTINE PLOT              Last Revised:  5/19/88            !
!                                                                    !
!                                                                    !
!         Entry Points:   GINIT                                      !
!                         GEND                                       !
!                                                                    !
!                                                                    !
!     This subroutine affects the work station for graphics          !
!         output via AGP.  If entry point GINIT is used, the         !
!         work station is initialized; if entry point GEND is        !
!         used, the work station is disabled.  IDWORK may take       !
!         the following values:                                      !
!             IDWORK = 0      Terminal only                          !
!                    = 1      Plotter only                           !
!                    = 2      Terminal and plotter                   !
!                                                                    !
!         If IDWORK = 2, the user will be prompted for possible      !
!         rotation of the logical display limits.  The other         !
!         arguments have the following meanings:                     !
!             I3D = 1  enable 3-D graphics                           !
!                 = 0  no 3-D graphics                               !
!                                                                    !
!             IASP = 0 prompt user for no distortion                 !
!                  = 1 use given aspect ratio                        !
!                                                                    !
!             ASP = 0  use the maximum aspect ratio                  !
!                 <>0  set aspect ratio for no distortion            !
!                                                                    !
!                                                                    !
!                                                                    !
!     Subroutines called:                                            !
!         None                                                       !
!                                                                    !
!--------------------------------------------------------------------!

      SUBROUTINE PLOT

      COMMON /USER/IWRITE,IREAD

      CHARACTER C*1,CP*1


      ENTRY GINIT(IDWORK,I3D,ASP,IASP)  !INITIALIZE WORK STATION

10    FORMAT(A)

      CALL JBEGN                   !INITIALIZE AGP
      IF (IDWORK .GT. 0) THEN
         WRITE(IWRITE,*) 'ENTER PAPER SIZE(A=SMALL,B=LARGE)',
     +                   '--CR DEFAULTS'
         READ (IREAD,10) CP
         IF (CP .NE. 'B' .AND. CP .NE. 'b') THEN
```

```
          WRITE(53,*) 'PS 4 ;'      !SET SMALL PAPER SIZE
          CP='A'
        ELSE
          WRITE(53,*) 'PS 0 ;'    !SET LARGE PAPER SIZE
          CP='B'
        END IF
        WRITE(IWRITE,*) 'WOULD YOU LIKE TO ROTATE THE COORDINATE',
     +                  ' SYSTEM OF THE PLOTTER(N/Y)?'
        READ(IREAD,10) C
        IROTATE=0
        IF (C .EQ. 'Y' .OR. C .EQ. 'y') IROTATE=IBSET(IROTATE,8)
      END IF

      IF (IASP .NE. 1) THEN       !DETERMINE ASPECT RATIO
        IF (ASP .EQ. 0) THEN     !USE DEFAULTS
          IF (IDWORK .EQ. 0) THEN
            ASP=.762793          !ASPECT RATIO FOR TERMINAL
          ELSE
            IF (CP .EQ. 'B') THEN
              ASP=.6229          !FOR LARGE PAPER
              IF (BTEST(IROTATE,8)) ASP=1./.6229
            ELSE
              ASP=.75            !FOR SMALL PAPER
              IF (BTEST(IROTATE,8)) ASP=1./.75
            END IF
          END IF
        ELSE
          WRITE(IWRITE,*) 'WOULD YOU LIKE TO SET THE ASPECT RATIO',
     +                    'FOR NO DISTORTION(N/Y)?'
          READ (IREAD,10) C
          IF (C .NE. 'Y' .AND. C .NE. 'y') ASP=.762793
        END IF
      END IF

      IF (ASP .LT. 1.0) THEN
        XSIZE=1.0
        YSIZE=ASP
      ELSE
        XSIZE=1./ASP
        YSIZE=1.0
      END IF
      CALL JASPK(XSIZE,YSIZE)           !SET ASPECT RATIO
      IF (I3D .EQ. 1) CALL JHAND(1)  !MAKE IT A RIGHT-HANDED COORD. SYS.
      IF (IDWORK .NE. 1) THEN
        CALL JDINT(1,22,22HWSP_CDS.RUN::PROGRAMS ,1,0) !WSP FOR 2397A
        CALL JIERR(IERR,ILEV,IND,INFO)          !SEE IF ERROR OCCURED
        IF (IERR .GT. 0) THEN                   !IF SO ,CHANGE WSP
          CALL JDINT(1,24,24HWSP23_CDS.RUN::PROGRAMS ,1,0) !WSP FOR 2623A
        END IF

        CALL JWON(1)                 !ENABLE GRAPHICS OUTPUT

        DO I=1,6
          CALL JEDEV(1,I,1)          !ENABLE LOGICAL DEVICES
        END DO
```

```
      END IF

      IF (IDWORK .GT. 0) THEN
         CALL JDINT(2,24,24HWSPPEN_CDS.RUN::PROGRAMS,53,IROTATE)
         CALL JWON(2)                              !FOR PLOTTER
      END IF


      RETURN

C

      ENTRY GEND(IDWORK)   !DISABLE WORK STATION

      IF (IDWORK .NE. 1) THEN
         DO I=1,6                      !FOR TERMINAL
            CALL JDDEV(1,I)            !DISABLE LOGICAL DEVICES
         END DO

         CALL  JWOFF(1)               !DISABLE GRAPHICS OUTPUT
         CALL  JWEND(1)               !DISABLE WORK STATION
      END IF

      IF (IDWORK .GT. 0) THEN
         CALL JWOFF(2)
         CALL JWEND(2)        !DO SAME FOR PLOTTER
      END IF

      CALL JEND

      RETURN



      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!     SUBROUTINE POSCHECK          Last Revised:  6/03/88     !
!                                                             !
!     Entry points:                                           !
!         TRUSS_CHECK                                         !
!                                                             !
!     This subroutine compares the position measured by the   !
!         encoders to the position measured by the laser system.  !
!         If they are not in agreement, the lasers are reset.  !
!                                                             !
!                                                             !
!     Subroutines called:                                     !
!         ENCODE                                              !
!         ERRTRUSS                                            !
!         POSIN  (POSITION)                                   !
!         VOLIN                                               !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE POSCHECK

      COMMON /ASSIGN/  IA(0:7), IB(0:7), IC(0:7), IP(0:7),
     +                  IV(0:7), IX(0:7), IY(0:7), IZ(0:7)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /LASER/  CTI, VOL, DPI, CPOS

      CHARACTER ANS*2


      CALL VOLIN(IPIC1,VOL,IERR)      ! Read VOL compensation

      CALL ENCODE(CEPOS,TSEPOS,TNEPOS)   ! Read encoder positions


      IF (ABS(CEPOS-CPOS) .GT. .01) THEN
         WRITE (1,*) 'WARNING:  Encoder reading disagrees with ',
     +              'expected cart position!'
         WRITE (1,*) '          Hit RETURN to accept encoder ',
     +              'reading and continue...'
         READ (1,*) ANS
         CPOS = CEPOS
      END IF

      GO TO 10                ! Skip redundant reads

C

      ENTRY TRUSS_CHECK


      CALL VOLIN(IPIC1,VOL,IERR)      ! Read VOL compensation

      CALL ENCODE(CEPOS,TSEPOS,TNEPOS)   ! Read encoder positions
```

```
10    CALL EXEC(2,IPIC2,IA(1),1,0)
      CALL EXEC(2,IPIC2,IA(2),1,0)          ! Sample truss North end
      CALL POSIN(IPIC2,TNPOS,IERR)          !    and input position

      IF (IERR .NE. 0 .AND. IERR .NE. 5
     +              .OR. ABS(TNEPOS-TNPOS) .GT. .01)  THEN
         CALL ERRTRUSS (IERR, TNEPOS, TPOS)
         GO TO 10
      END IF

20    CALL EXEC(2,IPIC2,IX(1),1,0)
      CALL EXEC(2,IPIC2,IX(2),1,0)          ! Sample truss South end
      CALL POSIN(IPIC2,TSPOS,IERR)          !    and input position

      IF (IERR .NE. 0 .AND. IERR .NE. 5
     +              .OR. ABS(TSEPOS-TSPOS) .GT. .01)  THEN
         CALL ERRTRUSS (IERR, TSEPOS, TPOS)
         GO TO 20
      END IF

      RETURN
      END
```

```
$CDS ON
$EMA /BUFFER/, /BUFFER2/, /POSN/
!-------------------------------------------------------------!
!                                                             !
!    SUBROUTINE POSCOL            Last Revised:  6/03/88      !
!                                                             !
!    Collects data in a positive direction (probe moving      !
!       southbound).                                          !
!                                                             !
!                                                             !
!    Subroutines called:                                      !
!       CLSTAT                                                !
!       MCART                                                 !
!       POSOUT  (POSITION)                                    !
!       POSWATCH                                              !
!       RECTOPOL                                              !
!                                                             !
!-------------------------------------------------------------!

C     LAST REVISED 8/5/88

      SUBROUTINE POSCOL (IROW)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /BUFFER/  ABUF(4096), PBUF(4096), IBUF
      COMMON /BUFFER2/  ABUF2(4096), PBUF2(4096), IBUF2
      COMMON /POSN/  XPOS(4096), YPOS(4096)
      COMMON /EXP85/  AEXP(0:255)
      COMMON /ASSIGN/  IA(0:7), IB(0:7), IC(0:7), IP(0:7),
     +                   IV(0:7), IX(0:7), IY(0:7), IZ(0:7)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /HPIB/  I4833, J4833, I8510, J8510, I8340, J8340
      COMMON /LASER/  CTI, VOL, DPI, CPOS

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      COMPLEX CDAT
      DIMENSION IDBUF(3,4096), IDBUF2(3,4096)
      LOGICAL NULL
      INTEGER CLASS1, CLASS1W, CLASS2, CLASS2W, AREG, BREG


      ICONT1 = I8510+100b     ! Control words for 8510 EXEC calls.  100b
      ICONT2 = J8510+100b     !     is the code for normal binary format
      NP = RSCAN(6)          ! # of points to be sampled in data column

      IFC1 = I8510 + 5100b              ! Control code for a
      IFC2 = J8510 + 5100b              !    10 msec IFC on HPIB


10    DCPOS = YPOS(1)                  ! Start at first point
      CALL MCART (DCPOS,0.,0)           ! Move cart to start

C  Section to set up the "motion Comparator" to control the probe scan by
```

```
C     loading the destination register with the location of the last point.

      CALL EXEC (2,IPIC3,IX(0),1,0)       ! Reset motion comparator
      DEST = ABS( CPOS-YPOS(NP) )        ! Offset to end of data row
      CALL POSOUT (IPIC3,DEST,15)      ! Output destination to BIC
      CALL EXEC (2,IPIC3,IX(3),1,0)      ! Load destination to motion Comp.
      CALL EXEC (2,IPIC3,IX(1),1,0)     ! Start generating digital difference


C  Section to prepare "sample Comparator" and 8510(s) for data  collection.
C     The destination register of the sample comparator is loaded with the
C     location of the next data point to be sampled.  As each point is
C     reached the comparator's null output automatically triggers the
C     8510(s) (via the Trigger Control Electronics) to sample a data point.
C     A Class Read is used to get data from 8510's so this program does not
C     monitor the 8510's during a scan.  The null line from the comparator
C     is monitored so the program knows when to load the location of the
C     next data point to be sampled.

      CALL EXEC (2,IPIC3,IY(0),1,0)       ! Reset sample Comparator

      CALL EXEC (3, IFC1, 1)       ! Clear both interface cards
      CALL EXEC (3, IFC2, 1)       !

      CLASS1W = 0                 ! Initialize Class Numbers
      CLASS2W = 0                 !

      CALL CLRQ (1, CLASS1W)     ! Reserve class numbers from the system
      CALL CLRQ (1, CLASS2W)     !      for 8510 class reads

      CLASS1W = IBSET( CLASS1W, 13 )     ! Set "Save Class Number" bit
      CLASS2W = IBSET( CLASS2W, 13 )     !

      CLASS1 = IBSET( CLASS1W, 15 )    ! Set "No Wait" bit
      CLASS2 = IBSET( CLASS2W, 15 )    !

      CALL CLSTAT (NPOL, I8510, J8510)     ! Set 8510's in Fast CW mode

      CALL EXEC (17, ICONT1, IDBUF1, NP*3, 0, 0, CLASS1)  ! Do Class Read
      CALL ABREG (AREG, BREG)                             ! and check for
      LOCATION=1                                          ! errors
      IF (AREG.NE.0) GO TO 999                            !

      IF (NPOL .EQ. 2) THEN
          CALL EXEC (17, ICONT2, IDBUF2, NP*3, 0, 0, CLASS2)  ! Ditto, for
          CALL ABREG (AREG, BREG)                             ! 2nd 8510
          LOCATION=2                                          !
          IF (AREG.NE.0) GO TO 999                            !
      END IF


C  Section to do the actual data collection

      WRITE (IWRITE,*) 'COLLECTING DATA FOR COLUMN ',IROW
```

```
      WRITE (I4833,*) '01'          ! Select Southbound
      WRITE (I4833,*) '11'          ! Enable motion

      DO I=1,NP
         POFF = ABS(CPOS-YPOS(I))             ! Offset to sample point
         CALL POSOUT (IPIC3,POFF,15)        ! Output dest. to BIC
         CALL EXEC (2,IPIC3,IY(3),1,0)        ! Load up Sample Comparator
         CALL EXEC (2,IPIC3,IY(1),1,0)      ! Start digital difference

C        CALL POSWATCH (IPIC3,IY(0),0.,POFF,1,0,IERR) ! Wait
C        IF (IERR .NE. 0 .AND. IERR .NE. 5) THEN
C           WRITE (IWRITE,*) 'Error reading laser position.  Move to'
C           WRITE (IWRITE,*) '    Home postion and start over.'
C           STOP
C        END IF

         NULL = .FALSE.
         DO WHILE (.NOT. NULL)
            CALL EXEC (1, I4833, INULL, -1)  ! Read null from Sample comp.
            NULL = BTEST (INULL, 7)          ! Null line at bit 7
         END DO

      END DO


C  Section to check for successful completion of column scan

      CALL POSWATCH (IPIC3,IX(0),0.,DEST,1,0,IERR)
      IF (IERR .NE. 0 .AND. IERR .NE. 5) THEN
         WRITE (IWRITE,*) 'ERROR in scan of column # ', IROW
         CALL CLRQ (2, CLASS1W)
         CALL CLRQ (2, CLASS2W)
      END IF

      WRITE (I4833,*) '0'       ! Disable motion
      CPOS = YPOS(NP)            ! New cart position


C  Section to get the last point(s) from the Class Reads

      CALL EXEC (21, CLASS1W, IDBUF1, NP*3)     ! Class Get
      CALL ABREG (AREG, BREG)
      LOCATION =3
      IF (AREG .LT. 0) GO TO 999

      IF (NPOL .EQ. 2) THEN
         CALL EXEC (21, CLASS2W, IDBUF2, NP*3)     ! Class Get
         CALL ABREG (AREG, BREG)
         LOCATION =4
         IF (AREG .LT. 0) GO TO 999
      END IF


C  Section to convert data to amplitude/phase format
```

```
      DO I=1,NP
          EX = AEXP( IAND( IDBUF(3,I), 255 ) )       ! Exponent
          RE = IDBUF(2,I)*EX                        ! Real part
          RIM = IDBUF(1,I)*EX                       ! Imaginary part
          CDAT = CMPLX(RE,RIM)             ! Convert to complex form and
          CALL RECTOPOL (CDAT,AMP,PHSE)    !    then to amplitude, phase
          ABUF(I) = AMP                       ! Store in buffers
          PBUF(I) = PHSE                      !

          IF (NPOL .EQ. 2) THEN
              EX = AEXP( IAND( IDBUF2(3,I), 255 ) )    ! Do the same
              RE = IDBUF2(2,I)*EX                      !    for the
              RIM = IDBUF2(1,I)*EX                     !    second
              CDAT = CMPLX(RE,RIM)                     !    pole data
              CALL RECTOPOL(CDAT,AMP,PHSE)             !
              ABUF2(I)=AMP                             !
              PBUF2(I)=PHSE                            !
          END IF
      END DO

      RETURN

999   WRITE (IWRITE,*) 'ERROR on Class Read or Get at location ',
     +                       LOCATION

      STOP

      END
```

```
$CDS ON
!-----------------------------------------------------------------------!
!                                                                       !
!     SUBROUTINE POSITION          Last Revised:  6/04/88               !
!                                                                       !
!     Entry points:                                                     !
!         POSIN                                                         !
!         POSOUT                                                        !
!                                                                       !
!                                                                       !
!     Subroutine to read/write position information from/to binary     !
!         interface card in the laser system (10746 BIC) via the        !
!         12006 PIC in the A900 controller.  Entry POSIN is used        !
!         for a position read, entry POSOUT for a destination           !
!         write.                                                        !
!                                                                       !
!         IPIC -- lu of the PIC to be used                              !
!         CTI -- conversion factor, wavelengths to inches               !
!                 (depends on resolution)                               !
!         VOL -- velocity of light compensation factor                  !
!         DPI -- deadpath in inches                                     !
!         DPW -- deadpath in wavelengths                                !
!         POS -- compensated position value in inches                   !
!         IWL -- number of wavelength counts; binary value of           !
!                bits 0-27 of interface card (32-bit integer)           !
!         ITOL -- tolerance value for bits 28-31                        !
!         IDEC -- decimal data extracted from bits 28-31                !
!         RDEC -- recovered decimal point information                   !
!         IERR -- 0 if no error or recoverable error occured            !
!                 1 if irrecoverable error occurred                     !
!                                                                       !
!                                                                       !
!                                                                       !
!     Subroutines called:                                               !
!         IERROR                                                        !
!                                                                       !
!-----------------------------------------------------------------------!

      SUBROUTINE POSITION

      COMMON /LASER/  CTI, VOL, DPI, CPOS

      INTEGER*4 IWL,INTOWL


      ENTRY POSIN (IPIC, POS, IERR)


      CALL EXEC (2,IPIC,3,1,0)       ! Set BIC for data output to computer
      CALL EXEC (1,IPIC,IBUF,1,0)     ! Read most significant word

      IERR=ISHFT(IBUF,-8)            ! Extract error information
      IERR=IERROR(IERR,IPIC)           ! Do error check
      IF (IERR .NE. 0) RETURN       ! Irrecoverable error occurred
                                    !     or destination reached
```

```
      IDEC=ISHFT(IBUF,-12)       ! Get decimal information
      RDEC=2-IDEC

      IWL=0                           ! Initialize
      CALL MVBITS (IBUF,0,12,IWL,16)   ! Move 12 data bits into upper word
      CALL EXEC (1,IPIC,IBUF,1,0)    ! Read least significant word
      CALL MVBITS (IBUF,0,16,IWL,0)   ! Combine with upper bits

      DPW = DPI/CTI                ! Deadpath in wavelengths
      WLTOIN = (IWL-160)*(10.**RDEC)   ! Subtract 160 and apply decimal info.
      POS = (DPW+WLTOIN)*(VOL*CTI)-DPI    ! Convert to inches

      RETURN

C

      ENTRY POSOUT(IPIC, POS, ITOL)


      CALL EXEC (2,IPIC,4,1,0)     ! Prepare BIC to input data from computer
      DPW = DPI/CTI                        ! Deadpath, in wavelengths
      IWL = (POS+DPI)/(VOL*CTI)-DPW +160    ! Convert to wavelengths
      CALL MVBITS (ITOL,0,4,IWL,28)       ! Specify tolerance
      CALL MVBITS (IWL,16,16,IBUF,0)       ! Load upper 16 bits for output
      CALL EXEC (2,IPIC,IBUF,1,0)      ! Output upper word
      CALL MVBITS (IWL,0,16,IBUF,0)      ! Load lower 16 bits for output
      CALL EXEC (2,IPIC,IBUF,1,0)     ! Output lower word
      CALL EXEC (2,IPIC,2,1,0)       ! Transfer BIC data to backplane

      RETURN


      END
```

```
$CDS ON
!-----------------------------------------------------------!
!                                                           !
!    SUBROUTINE POSWATCH          Last Revised:  6/04/88    !
!                                                           !
!    Subroutine to "watch the position".  Monitors the comparator  !
!        at address BADR of the coupler associated with LU IPIC    !
!        until the probe cart or translation beam reaches its      !
!        destination (within the specified tolerance.)             !
!                                                           !
!        IPIC -- LU for communication with the coupler box         !
!                    containing the comparator                     !
!        BADR -- Base address of the comparator                    !
!        BADR+1 -- Instructs the comparator to load counter        !
!                    contents into its output buffer               !
!        BADR+2 -- Instructs the comparator to write its output    !
!                    buffer to the coupler backplane               !
!        TPOS -- Current truss position (inches)                   !
!        CPOS -- Current cart position (inches)                    !
!        DPOS -- Desired position along axis of motion             !
!        IMOVE  = 0  for truss motion  (x-axis)                    !
!               = 1  for cart motion  (y-axis)                     !
!        IDIS  = 1  display position on terminal screen            !
!              = 0  no display                                     !
!                                                           !
!                                                           !
!    Subroutines called:                                    !
!        POSIN  (POSITION)                                  !
!        POSCHECK                                           !
!                                                           !
!-----------------------------------------------------------!

      SUBROUTINE POSWATCH (IPIC, BADR, TPOS, DPOS, IMOVE, IDIS, IERR)

      COMMON /HPIB/  I4833, J4833, I8510, J8510, I8340, J8340
      COMMON /LASER/  CTI, VOL, DPI, CPOS


17    FORMAT('X= ',F7.3,10X,'Y= ',F7.3)

      IDIR=SIGN(1.,DPOS-CPOS)     !ONLY SIGNIFICANT FOR CART MOTION

10    CALL EXEC (2,IPIC,BADR+1,1,0)      !SAMPLE POSITION
      CALL EXEC (2,IPIC,BADR+2,1,0)
      CALL POSIN (IPIC, POS, IERR)

      IF (IERR .EQ. 5) THEN
          RETURN              !COMPARATOR W/IN TOLERANCE

      ELSE IF (IERR .EQ. 0) THEN

          IF (IDIS .EQ. 1) THEN
              IF (IMOVE .EQ. 0) THEN
                  POST=POS              !TRUSS POSITION
                  POSC=CPOS             !CART POSITION
```

```
          ELSE
              POST=TPOS               !TRUSS POSITION
              POSC=CPOS+(IDIR)*POS    !CART POSITION
          END IF
          WRITE(1,17) POST,POSC     !DISPLAY POSITION
      END IF

      GOTO 10                !KEEP WATCHING

  ELSE

      WRITE (I4833,*) '00'        ! Disable any motion

      IF (IMOVE .EQ. 1) THEN
          WRITE (1,*) 'ERROR ',IERR,' on read of probe cart laser!'
      ELSE
          CALL POSCHECK           ! Try once more
      END IF

  END IF

  RETURN
  END
```

```
$CDS ON
!-----------------------------------------------------------------!
!                                                                 !
!     SUBROUTINE READWRITE          Last Revised:  6/04/88        !
!                                                                 !
!     Entry points:                                               !
!         READ_DATA                                               !
!         WRITE_DATA                                              !
!                                                                 !
!     Depending on which entry point is used, this routine reads  !
!         a row of data from, or writes a row of data to, a data  !
!         file.                                                   !
!             IUNIT - Unit number of data file                    !
!             IROW - Number of the row or column to be transferred !
!             IRDAT = 0  - only amplitude is recorded             !
!                   = 1  - only phase is recorded                 !
!                   = 2  - amplitude and phase are recorded       !
!             IDATA = 0 - only amplitude information is transferred !
!                   = 1 - only phase information is transferred   !
!                   = 2 - both amplitude and phase are transferred !
!                                                                 !
!                                                                 !
!     Subroutines called:                                         !
!         None                                                    !
!                                                                 !
!-----------------------------------------------------------------!

      SUBROUTINE READWRITE

      EMA ABUF(4096), PBUF(4096)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /USER/  IWRITE, IREAD

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15

C

      ENTRY READ_DATA (IUNIT, IROW, IRDAT, IDATA, ABUF, PBUF, IBUF)

      IF (CAXIS .EQ. 'X' ) THEN       !DATA COLLECTED ALONG X AXIS
          NPTS=RSCAN(3)       !# X PTS
      ELSE                                 !DATA COLLECTED ALONG Y AXIS
          NPTS=RSCAN(6)       !# Y PTS
      END IF

C  Section for reading data from a file

      IF (IRDAT .NE. 2) THEN       !ONLY AMP OR PHASE STORED
          IF (IDATA .NE. IRDAT) WRITE(IWRITE,*) 'WARNING----',
     +                    'DATA REQUESTED WAS NOT RECORDED'
          IREC=1+IROW             !RECORD #
          IF (IDATA .EQ. 0) READ(UNIT=IUNIT,IOSTAT=IERR,ERR=99,REC=
```

```
     +                               IREC) (ABUF(M),M=1,NPTS),IBUF
          IF (IDATA .EQ. 1) READ(UNIT=IUNIT,IOSTAT=IERR,ERR=99,REC=
     +                               IREC) (PBUF(M),M=1,NPTS),IBUF
      ELSE                             !AMPLITUDE AND PHASE STORED
          IREC=2+2*(IROW-1)        !RECORD #
          IF (IDATA .NE. 1) READ(UNIT=IUNIT,IOSTAT=IERR,ERR=99,REC=IREC)
     +                                          (ABUF(M),M=1,NPTS),IBUF
          IF (IDATA .NE. 0) READ(UNIT=IUNIT,IOSTAT=IERR,ERR=99,REC=IREC+
     +                                         1) (PBUF(M),M=1,NPTS),IBUF
      END IF

      RETURN

C

      ENTRY WRITE_DATA (IUNIT, IROW, IRDAT, IDATA, ABUF, PBUF, IBUF,
     +                      AMIN, AMAX, PMIN, PMAX, MAXY, MAXX)


      IF (CAXIS .EQ. 'X' ) THEN       !DATA COLLECTED ALONG X AXIS
          NPTS=RSCAN(3)        !# X PTS
      ELSE                                  !DATA COLLECTED ALONG Y AXIS
          NPTS=RSCAN(6)        !# Y PTS
      END IF

C  Section to determine maximum and minimum amplitudes and phases

      IF (IROW .EQ. 1) THEN
          AMIN=100.
          AMAX=-100.
          PMIN=180.          !INITIALIZE THE MAX AND MINS
          PMAX=-180.
      END IF

      DO I=1,NPTS
          IF(ABUF(I) .GT. AMAX) THEN
              AMAX=ABUF(I)             !AMPLITUDE MAX
              IF (CAXIS .EQ. 'X' ) THEN
                  MAXY=IROW
                  MAXX=I                           !SECTION TO DETERMINE
              ELSE                                   !MAX AND MINS
                  MAXY=I
                  MAXX=IROW
              END IF
          END IF
          IF (ABUF(I) .LT. AMIN) AMIN=ABUF(I)    !AMP MIN
          IF (PBUF(I) .GT. PMAX) PMAX=PBUF(I)    !PHASE MAX
          IF (PBUF(I) .LT. PMIN) PMIN=PBUF(I)    !PHASE MIN
      END DO


C  Section for writing data to a file

      IF (IRDAT .NE. 2) THEN       !ONLY AMP OR PHASE STORED
          IREC=1+IROW              !RECORD #
```

```
        IF (IRDAT .EQ. 0) WRITE(UNIT=IUNIT,IOSTAT=IERR,ERR=98,REC=
     +                          IREC) (ABUF(M),M=1,NPTS),IBUF
        IF (IRDAT .EQ. 1) WRITE(UNIT=IUNIT,IOSTAT=IERR,ERR=98,REC=
     +                          IREC) (PBUF(M),M=1,NPTS),IBUF
      ELSE                         !AMPLITUDE AND PHASE STORED
        IREC=2+2*(IROW-1)       !RECORD #
        IF (IDATA .NE. 1) WRITE(UNIT=IUNIT,IOSTAT=IERR,ERR=98,REC=
     +                            IREC)  (ABUF(M),M=1,NPTS),IBUF
        IF (IDATA .NE. 0) WRITE(UNIT=IUNIT,IOSTAT=IERR,ERR=98,REC=
     +                          IREC+1) (PBUF(M),M=1,NPTS),IBUF
      END IF

      RETURN


C  Section for error messages

98    WRITE (IWRITE,*) 'ERROR ',IERR,' WRITING ROW ',IROW,' TO FILE ',
     +                                                    NAME
      RETURN

99    WRITE (IWRITE,*) 'ERROR ',IERR,' READING ROW ',IROW,' FROM FILE ',
     +                                                    NAME
      RETURN

      END
```

```
!--------------------------------------------------------------!
!                                                              !
!     SUBROUTINE RECTOPOL          Last Revised:  5/30/88      !
!                                                              !
!     Converts a complex number in rectangular form (DATA) into !
!         equivalent amplitude and phase.  Amplitude (AMP) is  !
!         returned in dB and phase (PHASE) is returned in degrees. !
!                                                              !
!     Subroutines called:                                      !
!         None                                                 !
!                                                              !
!--------------------------------------------------------------!

      SUBROUTINE RECTOPOL (DATA, AMP, PHASE)

      COMPLEX DATA


      PI=3.1415927

      X = REAL(DATA)
      Y = AIMAG(DATA)
      AMP = SQRT( X**2 + Y**2 )

      IF (AMP .EQ. 0.) THEN
         PHASE = 0.
      ELSE
         PHASE = ATAN2(Y,X)            ! Phase in radians
      ENDIF
      PHASE = PHASE * 180./PI          ! Phase in degrees

      IF (AMP .LE. 0.00001) THEN
          AMP = -100.
      ELSE
          AMP = 20. * ALOG10(AMP)      ! Amplitude in dB
      END IF

      RETURN
      END
```

```
$CDS ON
!------------------------------------------------------------------!
!                                                                  !
!     SUBROUTINE RESET              Last Revised:  6/01/88         !
!                                                                  !
!     This subroutine resets the laser electronics in the          !
!         coupler box attached to the PIC at LU IPIC.              !
!                                                                  !
!                                                                  !
!     Subroutines called:                                          !
!         None                                                     !
!                                                                  !
!------------------------------------------------------------------!

      SUBROUTINE RESET(IPIC)


      IPRAM1 = 63
      ICNT = IPIC + 4000B
      CALL EXEC (3,ICNT,IPRAM1)     !CONFIGURE PIC CONTROL REGISTER

      CALL EXEC (2,IPIC,0,1,0)
      CALL EXEC (2,IPIC,0,1,0)    !SEND RESET COMMAND TO BINARY
      CALL EXEC (2,IPIC,0,1,0)    !INTERFACE CARD

      RETURN
      END
```

```
$CDS ON
!----------------------------------------------------------------!
!                                                                !
!    FUNCTION RMULTFIND          Last Revised:  6/01/88          !
!                                                                !
!    Entry points:                                               !
!        RMULTUP                                                 !
!        RMULTDOWN                                               !
!                                                                !
!    Entry point RMULTUP returns the smallest integer multiple   !
!        of FACTR greater than or equal to RVAR.  Entry point    !
!        RMULTDOWN returns the greatest integer multiple of FACTR !
!        less than or equal to RVAR.                             !
!                                                                !
!                                                                !
!    Subroutines called:                                         !
!        None                                                    !
!                                                                !
!----------------------------------------------------------------!

      FUNCTION RMULTFIND()

C

      ENTRY RMULTUP (RVAR, FACTR)


      RMULTUP=RVAR

      IF (RVAR/FACTR .NE. INT(RVAR/FACTR)) THEN
          IF (RVAR .GE. 0) THEN
              RMULTUP=INT((RVAR+FACTR)/FACTR)*FACTR   !FOR POSITIVE #'S
          ELSE
              RMULTUP=INT(RVAR/FACTR)*FACTR            !FOR NEGATIVE #'S
          END IF
      END IF

      RETURN

C

      ENTRY RMULTDOWN (RVAR, FACTR)


      RMULTDOWN=RVAR

      IF (RVAR/FACTR .NE. INT(RVAR/FACTR)) THEN
          IF (RVAR .GE. 0) THEN
              RMULTDOWN=INT(RVAR/FACTR)*FACTR         !FOR POSITIVE #'S
          ELSE
              RMULTDOWN=INT((RVAR-FACTR)/FACTR)*FACTR  !FOR NEGATIVE #'S
          END IF
      END IF

      RETURN
```

```
      END
```

```
$CDS ON
$EMA /BUFFER/, /BUFFER2/, /POSN/
!-------------------------------------------------------------!
!                                                             !
!    SUBROUTINE SCAN            Last Revised:  6/04/88        !
!                                                             !
!    This subroutine is used to collect a whole data set      !
!        according to the scan parameters and store the data  !
!        in a file.                                           !
!            IROW - Counter indicating which row is currently !
!                    stored in the buffer                     !
!            IAXIS = 0 for data collection along Y-axis       !
!                    (only mode currently implemented)        !
!            CPOS - current cart position                     !
!            NPOL - number of poles to be collected (1 or 2)  !
!                                                             !
!    Subroutines called:                                      !
!        COLLECT                                              !
!        DATETIME                                             !
!        HEADREAD  (HEADER)                                   !
!        HEADWRITE  (HEADER)                                  !
!        NAMFILE                                              !
!                                                             !
!-------------------------------------------------------------!

C     LAST REVISED 8/5/88

      SUBROUTINE SCAN (IROW, IAXIS)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /MINMAX/  AMIN, AMAX, PMIN, PMAX, MAXY, MAXX
      COMMON /MINMAX2/  AMIN2, AMAX2, PMIN2, PMAX2, MAXY2, MAXX2
      COMMON /BUFFER/  ABUF(4096), PBUF(4096), IBUF
      COMMON /BUFFER2/  ABUF2(4096), PBUF2(4096), IBUF2
      COMMON /POSN/  XPOS(4096), YPOS(4096)
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /HPIB/  I4833, J4833, I8510, J8510, I8340, J8340
      COMMON /USER/  IWRITE, IREAD
      COMMON /LASER/  CTI, VOL, DPI, CPOS

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER COM*2


      IAXIS=0              !COLLECTING ALONG Y AXIS
      IUNIT=3               !PRIMARY POLE UNIT #
      IUNIT2=4             !SECONDARY POLE UNIT #
10    FORMAT(A)

      CALL DATETIME (IDATE,ITIME)     !READ DATE AND TIME

      CALL NAMFILE (IUNIT,1)    !OPEN PRIMARY FILE
      CALL HEADWRITE (IUNIT,2)     !STORE HEADER INFO.
      AMIN1=100.
```

```
      AMAX1=-100.
      PMIN1=180.          !INITIALIZE PRIMARY MAX AND MINS
      PMAX1=-180.

      IF (NPOL .EQ. 2) THEN
         CALL NAMFILE (IUNIT2,1)        !OPEN SECONDARY POLE FILE
         WRITE (IWRITE,*) 'Enter label for 2nd polarization:'
         READ (IREAD,10) POL
         CALL HEADWRITE (IUNIT2,2)      !STORE HEADER INFO.
         AMIN2=100.
         AMAX2=-100.
         PMIN2=180.                !INITIALIZE  MAX AND MINS
         PMAX2=-180.
      END IF

      IPLOT=0
      WRITE (IWRITE,*) 'Should each row be plotted ',
     +                  'after it is collected? (N/Y)'
      READ (IREAD,10) COM
      IF (COM .EQ. 'Y' .OR. COM .EQ. 'y') IPLOT=1

      IF (CAXIS .EQ. 'Y') THEN
         NROWS=RSCAN(3)            !NUMBER OF DATA COLUMNS TO COLLECT
         IAXIS=0
      ELSE
         NROWS=RSCAN(6)
         IAXIS=1
      END IF

      CALL COLLECT (1, NROWS, IROW, NPOL, IPLOT)

      CALL HEADREAD (IUNIT,IRDAT)  !GET PRIM. FILE NAME AND POL.

      AMIN=AMIN1
      AMAX=AMAX1
      PMIN=PMIN1        ! UPDATE MIN AND MAX INFO.
      PMAX=PMAX1
      MAXY=MAXY1
      MAXX=MAXX1

      CALL HEADWRITE(IUNIT,2)    !STORE CORRECT MAX AND MIN INFO.
      CLOSE(IUNIT)

      IF (NPOL .EQ. 2) THEN
         CALL HEADREAD(IUNIT2,IRDAT)    !GET SECONDARY FILE NAME AND POL.

         AMIN=AMIN2
         AMAX=AMAX2            !MAX AND MINS
         PMIN=PMIN2
         PMAX=PMAX2
         MAXY=MAXY2
         MAXX=MAXX2

         CALL HEADWRITE(IUNIT2,2)     !STORE CORRECT MAX AND MIN INFO.
         CLOSE(IUNIT2)
```

```
      END IF

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!    SUBROUTINE SETSOURCE             Last Revised:  6/06/88  !
!                                                             !
!    This subroutine sets the frequency and power level of the !
!        two sources.  The arguments are:                     !
!            FREQ = Operating frequency                       !
!            IMODE = 0   Probe receiving                      !
!                    1   Probe transmitting                   !
!            NPOL = Number of poles being collected (1 or 2)  !
!                                                             !
!                                                             !
!    Subroutines called:                                      !
!        SOURCE                                               !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE SETSOURCE (FREQ, IMODE, NPOL)

      COMMON /HPIB/  I4833, J4833, I8510, J8510, I8340, J8340
      COMMON /PICS/  IPIC1, IPIC2, IPIC3
      COMMON /USER/  IWRITE, IREAD

      CHARACTER CSTRING*40,ANS*1


10    FORMAT('CENT ',F8.5,' GHz;USER1;CHAN1')
15    FORMAT(A)


      GO TO 100                   ! Skip user prompts

C

      ENTRY SRC_USER (FREQ, IMODE, NPOL)

90    WRITE (1,*) 'Enter the desired operating frequency (GHz): '
      READ (1,*)  FREQ

      WRITE (1,*) 'Will the TEST Antenna be transmitting or receiving'
     +                             ,' (T/R)?'
      READ (1, '(A1)')  ANS
      IMODE = 0
      IF (ANS.EQ.'R' .OR. ANS.EQ.'r')    IMODE = 1



100   IF (FREQ .LT. 1.0) THEN
          WRITE (1,*) 'WARNING: System not set up to operate below ',
     +                    '1.0 GHz!'
          GO TO 90

      ELSE IF (FREQ .LE. 6.0) THEN
          RF_PWR = 10.0
```

```
            LO_PWR = -20.0
            LO_FRQ = (FREQ-.02)

        ELSE IF (FREQ .LE. 18.0) THEN
            RF_PWR = 10.0
            LO_PWR = -20.0
            LO_FRQ = (FREQ-.02) / 3.

        ELSE IF (FREQ .LE. 26.5) THEN
            RF_PWR = 10.0
            LO_PWR = -20.0
            LO_FRQ = (FREQ-.02) / 5.

        ELSE IF (FREQ .GT. 26.5) THEN
            WRITE (1,*) 'WARNING: System not set up to operate above ',
     +                  '26.5 GHz!'
            GO TO 90

        END IF


        IF (IMODE.EQ.0) THEN
            CALL SOURCE (I4830, RF_PWR, FREQ)
            CALL SOURCE (J4830, LO_PWR, LO_FRQ)
        ELSE
            CALL SOURCE (J4830, RF_PWR, FREQ)
            CALL SOURCE (I4830, LO_PWR, LO_FRQ)
        END IF


        WRITE (CSTRING,10) FREQ
        WRITE (I8510,*) CSTRING                     !SET UP 8510'S
        IF (NPOL .EQ. 2) WRITE (J8510,*) CSTRING


        RETURN
        END
```

```
$CDS ON
$EMA /POSN/
!-------------------------------------------------------------!
!                                                             !
!     SUBROUTINE SIDECHECK        Last Revised:  6/04/88      !
!                                                             !
!     This subroutine checks to see which side of the scanner !
!         the probe is closest to, then calls the appropriate !
!         subroutine to collect a row of data.                !
!                                                             !
!                                                             !
!     Subroutines called:                                     !
!         POSCOL                                              !
!         NEGCOL                                              !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE SIDECHECK (IROW)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                  IDATE(3), ITIME(3), NPOL
      COMMON /POSN/  XPOS(4095), YPOS(4095)
      COMMON /LASER/  CTI, VOL, DPI, CPOS

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15


      NPTS = RSCAN(6)             ! # of pts in column to be collected
      DSTART = ABS( CPOS-YPOS(1) )   ! Distance to start of data column
      DEND = ABS( CPOS-YPOS(NPTS) )    ! Distance to end of data column

      IF (DSTART .LE. DEND) THEN
          CALL POSCOL (IROW)          ! Scan forward from first pt.
      ELSE
          CALL NEGCOL (IROW)          ! Scan backward from last pt.
      END IF

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!    SUBROUTINE SOURCE           Last Revised:  6/04/88       !
!                                                             !
!                                                             !
!    Entry points:                                            !
!        SOURCE                                               !
!        SRC_PWR                                              !
!                                                             !
!                                                             !
!    This subroutine sets the CW frequency and power level for !
!        an HP 8340 synthesizer.  If entry point SRC_PWR is    !
!        used, just the power level is set.  The arguments     !
!        have the following meaning:                           !
!            IADDR = LU of the source to be set                !
!            PWR = desired power level from source (dBm)       !
!            FREQ = operating frequency of source (GHz)        !
!                                                             !
!                                                             !
!    Subroutines called:                                      !
!        None                                                 !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE SOURCE (IADDR, PWR, FREQ)

      CHARACTER  CFREQ*12, CPWR*10


      WRITE (CFREQ,'("CW",F8.5,"GZ")') FREQ
      WRITE (IADDR,'(A)',ERR=999) CFREQ

C

      ENTRY SRC_PWR (IADDR, PWR)

      WRITE (CPWR,'("PW",F6.2,"DB")') PWR
      WRITE (IADDR,'(A)',ERR=999) CPWR

      RETURN

999   WRITE (1,*) 'WARNING: Error setting source', IADDR
      PAUSE

      RETURN

      END
```

C-3

```
$CDS ON
$EMA /POSN/
!------------------------------------------------------------!
!                                                            !
!     SUBROUTINE STO_POSN          Last Revised:  6/04/88    !
!                                                            !
!     This subroutine calculates X and Y coordinates of points !
!         on the data sampling grid, and stores them in arrays !
!         XPOS and YPOS.                                     !
!                                                            !
!                                                            !
!     Subroutines called:                                    !
!         None                                               !
!                                                            !
!------------------------------------------------------------!

      SUBROUTINE STO_POSN

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /POSN/  XPOS(4096), YPOS(4096)

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15


      DO I=1,RSCAN(3)
          XPOS(I)=RSCAN(1) + (I-1)*RSCAN(2)     ! X coordinates
      END DO

      DO I=1,RSCAN(6)
          YPOS(I)=RSCAN(4) + (I-1)*RSCAN(5)     ! Y coordinates
      END DO


      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!     SUBROUTINE SWIPE           Last Revised:  5/19/88       !
!                                                             !
!     This subroutine clears the terminal display.            !
!                                                             !
!     Subroutines called:                                     !
!         None                                                !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE SWIPE


      CHARACTER*4 A,G,U


      A=CHAR(27)//'H'//CHAR(27)//'J'     !Clear Alpha display
      G=CHAR(27)//'*da'                  !Clear Graphics display
      U=CHAR(27)//'&j@'                  !Clear User Keys display

      WRITE(1,5) A,G,U

5     FORMAT (3A4)

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------------!
!                                                                   !
!     SUBROUTINE VOLIN            Last Revised:  5/20/88            !
!                                                                   !
!     Subroutine to read temp. compensation coefficient via the     !
!         PIC.                                                      !
!                                                                   !
!     IPIC -- LU of the PIC to be read from                         !
!     IDATA -- binary value of data bits 0-27 from interface        !
!              card (32-bit integer)                                !
!     IDEC -- binary value of the decimal data extracted from       !
!             bits 28-31                                            !
!     IERR -- 0 if no error or recoverable occured.                 !
!             1 if irrecoverable error occured.                     !
!     VOL -- Velocity-of-light compensation calculated from         !
!            IDEC and IDATA                                         !
!                                                                   !
!                                                                   !
!     Subroutines called:                                           !
!         DELAY                                                     !
!                                                                   !
!-------------------------------------------------------------------!

      SUBROUTINE VOLIN(IPIC,VOL,IERR)

      INTEGER*4,IDATA


      CALL EXEC(2,IPIC,98,1,0)  !TAKE NEW READING
      CALL DELAY(500)           !WAIT FOR MEAS. TO BE COMPLETE
10    CALL EXEC(2,IPIC,98,1,0)  !SAMPLE COMP. READING
      CALL EXEC(2,IPIC,3,1,0)   !PREPARE BIC TO OUTPUT DATA
      CALL EXEC(1,IPIC,IBUF,1,0)    !READ MOST SIGNIFICANT WORD IN
      IDATA=0                   !INITIALIZE
      IERR=ISHFT(IBUF,-8)           !EXTRACT ERROR INFORMATION
      IERR=IERROR(IERR,IPIC)            !GO TO ERROR CHECKING ROUTINE
      IF (IERR .EQ. 4) GOTO 10     !VOL ERROR
      IDEC=ISHFT(IBUF,-12)       !GET DECIMAL INFORMATION
      RDEC=2-IDEC
      CALL MVBITS(IBUF,0,12,IDATA,16)     !MOVE 12 DATA BITS INTO UPPER WORD
      CALL EXEC(1,IPIC,IBUF,1,0)    !READ LEAST SIGNIFICANT WORD IN
      CALL MVBITS(IBUF,0,16,IDATA,0)  !STORE IN ONE WORD
      VOL=IDATA*(10.**RDEC)

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!     SUBROUTINE VWPTJ            Last Revised:  6/04/88      !
!                                                             !
!                                                             !
!     This subroutine sets the viewport and window for plotting !
!         purposes.  It also calls other subroutines to set   !
!         up the grid and do the plot.                        !
!             IROW     - Row or column to be plotted          !
!             IAXIS  = 0  - Plot data collected on Y-axis cut !
!                    = 1  - Plot data collected on X-axis cut !
!                                                             !
!                                                             !
!     Subroutines called:                                     !
!         DEFINE                                              !
!         DRWJ                                                !
!         LABJ                                                !
!         PDEF                                                !
!         RMULTUP  (RMULTFIND)                                !
!         SWIPE                                               !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE VWPTJ (IROW, IAXIS, ABUF, PBUF)

      EMA  ABUF(4096), PBUF(4096)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                  IDATE(3), ITIME(3), NPOL
      COMMON /AMP/  VHI, VLO, YMAX, YMIN

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      REAL AR(2)


      CALL SWIPE                         !CLEAR ALPHA DISPLAY
      CALL JNEWF                         !CLEAR GRAPHICS DISPLAY
      CALL JIWS(1,254,0,2,IDUM,AR)    !GET LOGICAL DISPLAY ASPECT RATIO
      VMAXX=1.0
      VMAXY=1.0
      IF (AR(1) .LT. 1.0) THEN     !DETERMINE MAX X AND Y BOUNDS
         VMAXY=AR(1)               !    GIVEN THE ASPECT RATIO
      ELSE
         VMAXX=1.0/AR(1)
      END IF
      YLOW=VMAXY/8*5               !LOWER Y RANGE

      CALL DEFINE(IAXIS,START,RINC,NP)       !GET # OF PTS.
      P0=PDEF(IAXIS,1)                 !GET STARTING PT.
      P1=PDEF(IAXIS,NP)                !GE ENDING PT.

      SPACE=(P1-P0)/10.            !BREAK INTO TENTHS
      XMIN=P0-SPACE
      XMAX=P1+SPACE
```

```
      VHI=-1000.
C     VLO=1000.                     !INITIAL VALUES

      DO I=1,NP
         IF (ABUF(I) .GT. VHI) VHI=ABUF(I)  !HIGHEST AMP VALUE
C        IF (ABUF(I) .LT. VLO) VLO=ABUF(I)  !LOWEST AMP VALUE
      END DO

      VHI=RMULTUP(VHI,5.)          !ROUND UP TO NEAREST 10 DB
      VLO=VHI-45.          !LOWEST AMP VALUE

      YMIN=VLO-((VHI-VLO)*.1)
      YMAX=VHI+((VHI-VLO)*.2)

C  Section for setting viewport and window for amplitude plot

      CALL JWIND(XMIN,XMAX,YMIN,YMAX)        !LIMITS FOR AMP PLOT
      CALL JVIEW(0.,VMAXX,0.,YLOW)       !SET VIEWPORT TO LOWER 5 EIGHTS
      CALL LABJ(0,IAXIS,IROW,P0,P1)     !DRAW GRID AND LABELS FOR AMP
      CALL DRWJ(0,IAXIS,NP,ABUF)        !DRAW PLOT FOR AMP

C  Section for setting viewport and window for phase plot

      CALL JWIND(XMIN,XMAX,-270.,300.)    !LIMITS FOR PHASE PLOT
      CALL JVIEW(0.,VMAXX,YLOW,VMAXY)    !UPPER 3 EIGHTHS
      CALL LABJ(1,IAXIS,IROW,P0,P1)     !DRAW GRID AND LABELS FOR PHASE
      CALL DRWJ(1,IAXIS,NP,PBUF)                  !DRAW PLOT FOR PHASE

      CALL JMCUR                    !MAKE PICTURE CURRENT

      RETURN
      END
```

```
$CDS ON
!---------------------------------------------------------------!
!                                                               !
!     SUBROUTINE XINIT            Last Revised:  6/06/88        !
!                                                               !
!     This subroutine initializes the scan parameters after     !
!         prompting the user to input desired values.  The      !
!         position buffer is updated and the sources are set    !
!         via calls to STO_POSN and SETSOURCE, respectively.    !
!                                                               !
!                                                               !
!     Subroutines called:                                       !
!         LISTCHANGE                                            !
!         SETSOURCE                                             !
!         STO_POSN                                              !
!                                                               !
!---------------------------------------------------------------!

      SUBROUTINE XINIT (IMODE, INIT)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                IDATE(3), ITIME(3), NPOL
      COMMON /USER/  IWRITE, IREAD
      COMMON /TITLE/  CTITL(10)

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER ANS*8, CTITL*28


C   If called from initialization routine, ask user

      IF (INIT .NE. 0) THEN
          WRITE (IWRITE,*) 'Do you wish to set the scan parameters?',
     +                     ' (N/Y)'
          READ (IREAD,*) ANS
          IF (ANS .NE.'Y' .and. ANS .NE. 'y')  RETURN
      END IF


C   Get scan parameters from user

      DO I=1,7
13        WRITE (IWRITE,*) CTITL(I),'?'
          READ (IREAD,*) RSCAN(I)              ! Read scan parameters
          IF (RSCAN(I) .LT. 0) GOTO 13
          IF (I .EQ. 3 .OR. I .EQ. 6) THEN
              RSCAN(I) = INT( RSCAN(I) )       ! # of rows or # of columns
          END IF
      END DO

      WRITE (IWRITE,*)  CTITL(8), '?'          ! # of poles to collect
      READ (IREAD,*) NPOL

      WRITE (IWRITE,*) CTITL(9),'?'            ! polarization description
      READ (IREAD,10) POL                      !    (8 chars)
```

```
      WRITE (IWRITE,*) CTITL(10),'?'          ! data set description
      READ (IREAD,10) CSCAN                   !    (80 chars)

      WRITE (IWRITE,*) 'Will the TEST Antenna be Transmitting or',
     +                   ' Receiving  (T/R)?'
      READ (IREAD,10) ANS
      IMODE = 0
      IF (ANS.EQ.'R' .OR. ANS.EQ.'r')  IMODE = 1


10    FORMAT (A)

C   Call routines to store position coordinates in buffer /POSN/
C        and set freq, power levels on sources

      FREQ = RSCAN(7)
      CALL SETSOURCE (FREQ, IMODE, NPOL)
      CALL STO_POSN                     !COMPUTE POSITION BUFFERS
      CALL LISTCHANGE (IMODE)           !EXAMINE PARAMETERS

      RETURN
      END
```

APPENDIX D

Program NFFT Listing

This page intentionally left blank

```
* Load file for Program NFFT
* Last Revision:   14 NOV 88

EC
DE

RE NFFT.REL

RE ARRAY_DUMP.REL
RE ARRAY_FILL.REL
RE BLACKMAN.REL
RE BLOWUP.REL
RE CONVERT.REL

RE CORREC.REL
RE DATETIME.REL::NASA
RE DUMP_FILTER.REL
RE EEU.REL
RE EHU.REL

RE EXPAND.REL
RE FFT2.REL
RE GOWAVGD.REL
RE GETPAT.REL
RE GRIDSET.REL::NASA

RE HEADER.REL::NASA
RE NAMFILE.REL::NASA
RE NFNORM.REL
RE PCALC.REL
RE PCORR.REL

RE POLAR.REL
RE POWRT.REL
RE READWRITE.REL::NASA
RE S1OTO1.REL
RE SEPARATE.REL

RE SEPTRANS.REL
RE SINX.REL
RE SWIPE.REL::NASA
RE TESTP2.REL
RE TRANSLATE.REL

RE XYTHUY.REL
RE XYTYCON.REL
RE XYTZCON.REL
RE XYZOPEN.REL::NASA

LI %FRPLS::FTN7X
LI $FCDS::VCPLUS

EMA 1022
VM,65000
WS,1022
```

EN NFFT.RUN

```
$cds on
$files 2,7
$ema /buffer/
      PROGRAM NFFT (INPUT)

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                 C
C     This program reduces two-dimensional near field data.       C
C          Written by J.P. Montgomery  Feb 1976                   C
C          Extensively modified starting Mar 1986                 C
C          by  P.G. Friederich  and  V.K. Tripp                   C
C                                                                 C
C     Program Last Revised:       05 AUG 1988                     C
C                                                                 C
C                                                                 C
C     References for this program (especially the probe           C
C         correction section):                                    C
C                                                                 C
C     Kerns, D.M., Plane-wave Scattering-Matrix Theory of         C
C         Antennas and Antenna-Antenna Interactions, NBS          C
C         Monograph 162, June 1981.                               C
C                                                                 C
C     Montgomery, J.P., "Near-Field Measurement Equation and      C
C         Terminology", Texas Instruments Memo, 2 December 1975.  C
C                                                                 C
C     Tripp, V.K., "Probe Correction of Arbitrary Polarization",  C
C         GTRI Technical Memo, 2 September 1986.                  C
C                                                                 C
C                                                                 C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC


      COMPLEX SDATA(4096)
      COMPLEX DATA(4096,4096),DATA2(4096,4096),BFILT(4096,4096)
      EMA DATA,DATA2,DUMMY,BFILT,SDATA
      COMPLEX CJ,DUMMY,A0
      CHARACTER*80 TITLE,CANS,TEMP,CTIT,CTIT2
      CHARACTER*15 INPUT,COFILE,XFILE,FNAME,FNAME2
      CHARACTER*1 CFILT
      INTEGER DBUFF(15)
      LOGICAL REPEAT
      LOGICAL SINGLE

      COMMON /PARAM/ RSCAN(7),CAXIS,POL,TITLE,NAME,IDATE(3),ITIME(3)
      COMMON /MINMAX/ AMIN,AMAX,PMIN,PMAX,JMAX,IMAX
      COMMON /BUFFER/ABUF(4096),PBUF(4096),IBUF
      COMMON /USER/ IWRITE,IREAD
      COMMON /WVGE/ A,B,AK0
      COMMON /LIMIT/ NX0,NX1,NY0,NY1
      COMMON /TRANS/TX,TY,TZ,FILTER,SXINC,SYINC
      COMMON /RECBUFF/LBUF(8200)

      INTEGER*4  TIME0,TIME1,TIME2,TIME3,TIME4,TIME5
      INTEGER*4  ElapsedTime
```

```
      CHARACTER  NAME*15,CAXIS*1,POL*8,COPOL*8,XPOL*8,BELL*1

      CALL ResetTimer

C     Unit numbers for files:
C
C         Unit 2  -   Aperture data, 1st probe rotation
C         Unit 3  -   Aperture data, 2nd probe rotation
C         Unit 4  -   Spectrum data, 1st probe rotation
C         Unit 5  -   Spectrum data, 2nd probe rotation
C         Unit 6  -   Output file for debugging information
C         Unit 8  -   Pattern data for probe correction (1st rotation)
C         Unit 9  -   Pattern data for probe correction (2nd rotation)
C         Unit 11 -   Input file for unattended run
C         Unit 13 -   Output file for aperture Blackman filter
C         Unit 14 -   Output file for spectral Blackman filter

      OPEN (UNIT=6, FILE='Output_junk')

      IF (INPUT.EQ.'1') THEN
          IREAD=1
          IWRITE=1
      ELSE
          IREAD=11
          OPEN (UNIT=11, FILE=INPUT)
          IWRITE=6
      END IF

      BELL=CHAR(7)
      PI=ACOS(-1.)
      CJ=(0.,1.)
      DR=PI/180
      RD=180./PI

      CALL DateTime(IDATE,ITIME)
      CALL FTIME (DBUFF)
      CALL SWIPE
      WRITE (1,'(3A1)') CHAR(10),CHAR(10),CHAR(10)
      WRITE (1,4)
      WRITE (1,'(//,20X,15A2)') DBUFF
      WRITE (6,5) (IDATE(I),I=1,3),(ITIME(I),I=1,3)
4     FORMAT (  20X,
     +     ' ********  PROGRAM NFFT  ******** ')
5     FORMAT (' **** PROGRAM NFFT ****    '1X,2(I2,'/'),I2,I4,2(':',I2))

      WRITE (1,97)
      WRITE (1,*) ' Default responses are shown in parentheses.  When'
     +            ,' a choice is'
      WRITE (1,*) '      displayed, the first response is the default.'
      WRITE (1,*) ' Defaults may be selected with the Return key.'
      WRITE (1,97)

99    FORMAT ( A80 )          ! For user inputs with CANS
98    FORMAT ( A )           ! For use with BELL
97    FORMAT ( /// )
```

```
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     Input the test data
c

      WRITE (1,98) BELL
      WRITE (IWRITE,*) ' 1. How many polarizations will be analyzed ? '
     +                    ,'  (1 or 2) '
      READ (IREAD,99) CANS
      NPOL=1
      IF (CANS .EQ. '2')  NPOL=2
      WRITE (IWRITE,*) ' NPOL = ',NPOL

      WRITE (IWRITE,*)
      IF (NPOL.EQ.2) THEN
         WRITE (IWRITE,*) ' 2a. For the parallel pole aperture data -'
      ELSE
         WRITE (IWRITE,*) ' 2. For the aperture data to be analyzed -'
      END IF
      CALL NAMFILE(2,0)
      COFILE=NAME
      WRITE (6,110) NAME
      IF (NPOL.EQ.2) THEN
         WRITE (IWRITE,*) ' 2b. For the cross pole aperture data -'
         CALL NAMFILE(3,0)
         XFILE=NAME
         WRITE (6,110) NAME
      END IF

      CALL HEADREAD(2,IRDAT)
      TEMP=TITLE
      COPOL=POL
      NX=INT(RSCAN(3))
      NY=INT(RSCAN(6))
      CANS=CAXIS

      IF (NPOL.EQ.2) THEN
         CALL HEADREAD(3,IRDAT)
         IF ( (NX.NE.RSCAN(3)) .OR. (NY.NE.RSCAN(6)) .OR.
     +        (CANS.NE.CAXIS) )  THEN
            WRITE (IWRITE,*) '** File mismatch - program aborted **'
            STOP
         END IF
         XPOL=POL
      END IF

      WRITE (6,112) TEMP
      IF (NPOL.EQ.2) WRITE (6,112) TITLE

      NX0=1
      NY0=1
      NX1=NX
      NY1=NY
```

```
      WRITE (IWRITE,*)
      WRITE (IWRITE,*) ' 3. Enter row numbers for starting, ending X:'
     +                 ,'(1,',NX,')'
      READ (IREAD,99) CANS
      IF (CANS .GT. ' ')  READ (CANS,*) NX0, NX1

      WRITE (IWRITE,*)
      WRITE (IWRITE,*) ' 4. Enter row numbers for starting, ending Y:'
     +                 ,'(',NY0,',',NY1,')'
      READ (IREAD,99) CANS
      IF (CANS .GT. ' ')  READ (CANS,*) NY0, NY1

      IXINC=1
      IYINC=1

      WRITE (IWRITE,*)
      WRITE (IWRITE,*) ' 5. Enter X thinning increment: (1)'
      READ (IREAD,99) CANS
      IF (CANS .GT. ' ')  READ (CANS,*) IXINC

      WRITE (IWRITE,*)
      WRITE (IWRITE,*) ' 6. Enter Y thinning increment: (1)'
      READ (IREAD,99) CANS
      IF (CANS .GT. ' ')  READ (CANS,*) IYINC

      WRITE (1,*) 'Data set to be analyzed: '
      WRITE (1,*) '  X points ',NX0,' through ',NX1,', every ',
     +                 IXINC,'th point;'
      WRITE (1,*) '  Y points ',NY0,' through ',NY1,', every ',
     +                 IYINC,'th point.'

      WRITE (6,*) 'Data set to be analyzed: '
      WRITE (6,*) '  X points ',NX0,' through ',NX1,', every ',
     +                 IXINC,'th point;'
      WRITE (6,*) '  Y points ',NY0,' through ',NY1,', every ',
     +                 IYINC,'th point.'

      Time0 = ElapsedTime()

      MX = 1 + (NX1-NX0)/IXINC
      MY = 1 + (NY1-NY0)/IYINC
      AMIN = 100.
      AMAX = -100.

      CALL ARRAY_FILL(DATA, NX0, NY0, MX, MY, IXINC, IYINC, 2, 1)
      IF (NPOL .EQ. 2) THEN
         CALL ARRAY_FILL(DATA2, NX0, NY0, MX, MY, IXINC, IYINC, 3, 2)
      END IF

      NX = MX
      NY = MY
      RSCAN(3) = NX
      RSCAN(6) = NY
      RSCAN(2) = RSCAN(2)*IXINC
      RSCAN(5) = RSCAN(5)*IYINC
```

```
  17 FORMAT(A4)
  110 FORMAT(A15)
  112 FORMAT(A80,/)

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
C     In this routine, the DATA arrays are one-dimensional.  They are
C     re-dimensioned in the subroutines for compactness of data storage.
C
C     In the subroutines, a row represents a scan of constant Y; a
C     column, a scan of constant X. (The first subscript represents
C     the row number.)  In other words, all data is stored in locally
C     packed form with the first subscript varying fastest.
C
C      NX      Number of pts. per row - the extent of the first index
C      NY      Number of rows - the extent of the second index
C      XINC    Spacing of data along the X axis (inches)
C      YINC    Spacing of data along the Y axis (inches)
C      FREQ    Frequency in GHz

      Time1 = ElapsedTime()

      FREQ = RSCAN(7)
      ALAM = 11.80283/FREQ        ! Wavelength
      AK0 = 2. * PI / ALAM        ! Wave Number
      XINC = RSCAN(2)
      IF (NX.EQ.1)  XINC = 2. * PI
      YINC = RSCAN(5)
      IF (NY.EQ.1)  YINC = 2. * PI

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
C    NORMALIZE NF DATA

      WRITE (1,*) ' Ready to normalize the aperture data.'
      WRITE (1,*)

C     First, get the feed through level for reference

      WRITE (IWRITE,*)
      WRITE (1,98) BELL
      WRITE (IWRITE,*) ' 7. Enter the reference amplitude and phase, '
     +                        ,' in dB and degrees.'
      WRITE (IWRITE,*) '    (Use the feedthrough values if available.'
      WRITE (IWRITE,*) '     Default is the maximum amplitude.) '
      READ (IREAD,99) CANS
      IF (CANS .GT. ' ')  READ (CANS,*) AMAX,PMAX
      A0 = CMPLX(AMAX,PMAX)

C     Next, translation in wave-number space

      AKX = 0.
      AKY = 0.
```

```
      WRITE (IWRITE,*)
      WRITE (IWRITE,*) ' 8. Enter normalized wave numbers (Kx,Ky) for '
      WRITE (IWRITE,*) '    the desired K-space translation:  (0.,0.) '
      READ (IREAD,99) CANS
      IF (CANS .GT. ' ')  READ (CANS,*) AKX,AKY

      WRITE (IWRITE,*) 'New pattern origin at ',AKX,AKY

      AKX = AKX*AKO
      AKY = AKY*AKO

      CALL NFNORM (DATA, NX, NY, AKX, AKY)
      IF (NPOL.EQ.2) CALL NFNORM (DATA2, NX, NY, AKX, AKY)

      WRITE (6,290)  AMAX, PMAX
290   FORMAT ( ' Near field normalization: ', F10.5, ' dB, ',
     +                                         F10.5, ' deg.' / )

C     Pad input for desired resolution enhancement

      CALL TESTP2(NX,ISXP2)
      CALL TESTP2(NY,ISYP2)
      CALL POWRT(NX,NXP2,ISXP2)
      CALL POWRT(NY,NYP2,ISYP2)

      REPEAT = .TRUE.
      DO WHILE (REPEAT)
         REPEAT = .FALSE.
         SNXRES = ALAM / (XINC*NXP2)
         SNYRES = ALAM / (YINC*NYP2)
         IF (SNXRES .GT. 1.) THEN
            SNXRES = 1.
            WRITE (IWRITE,*)
            WRITE (IWRITE,*) ' WARNING: X scan less than a wavelength'
     +                              ,'.  Potential error at '
            WRITE (IWRITE,*) '          resolution enhancement. '
            WRITE (IWRITE,*)
         END IF
         IF (SNYRES .GT. 1.) THEN
            SNYRES = 1.
            WRITE (IWRITE,*)
            WRITE (IWRITE,*) ' WARNING: Y scan less than a wavelength'
     +                              ,'.  Potential error at '
            WRITE (IWRITE,*) '          resolution enhancement. '
            WRITE (IWRITE,*)
         END IF
         ANXRES = ASIN(SNXRES) * RD
         ANYRES = ASIN(SNYRES) * RD

         WRITE (IWRITE,220) NXP2,SNXRES,ANXRES, NYP2,SNYRES,ANYRES

220      FORMAT ( //
     +'       Dimension     Resolution    Main-beam Angular Res.   ', /
     +'       ---------     ----------    ----------------------   ', /
     +'  X    ',I6,'        ',F8.4,'        ',F8.4,' deg. ',       /
```

```
     +'   Y    ',I6,'        ',F8.4,'         ',F8.4,' deg. '  // )

          WRITE (IWRITE,*)
          WRITE (1,98) BELL
          WRITE (IWRITE,*) ' 9a. Would you like increased resolution on'
          WRITE (IWRITE,*) '     the X-axis ? (N/Y)'
          READ (IREAD,99) CANS
          IF (CANS .EQ. 'Y' .OR. CANS .EQ. 'y')  THEN
             NXP2=NXP2*2
             IF (NXP2.GT.4096) NXP2 = NXP2/2
             REPEAT = .TRUE.
          END IF

          WRITE (IWRITE,*)
          WRITE (IWRITE,*) ' 9b. Would you like increased resolution on'
          WRITE (IWRITE,*) '     the Y-axis ? (N/Y)'
          READ (IREAD,99) CANS
          IF (CANS .EQ. 'Y' .OR. CANS .EQ. 'y')  THEN
             NYP2=NYP2*2
             IF (NYP2.GT.4096) NYP2 = NYP2/2
             REPEAT = .TRUE.
          END IF
      END DO

      IF (NX.NE.NXP2 .OR. NY.NE.NYP2)  THEN
          CALL EXPAND (DATA,NX,NY,NXP2,NYP2)
          IF (NPOL.EQ.2) CALL EXPAND (DATA2,NX,NY,NXP2,NYP2)
      END IF

      WRITE (6,*) ' Old Dimensions = ',NX,NY
      WRITE (6,*) ' New Dimensions = ',NXP2,NYP2

      NX = NXP2
      NY = NYP2

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     FFT Section, including resolution enhancement for a sector
c

      WRITE (1,*) ' Ready for the FFT section.'
      WRITE (1,*)

      WRITE (IWRITE,*)
      WRITE (1,98) BELL
      WRITE (IWRITE,*) ' 10. Does this data set contain independent ',
     +                        'column or row measurements? (N/Y)'
      READ (IREAD,99) CANS
      SINGLE = .FALSE.
      IF (CANS.EQ.'Y' .OR. CANS.EQ.'y')  SINGLE = .TRUE.

      Time2 = ElapsedTime()

      IF (SINGLE) THEN
         CALL SEPARATE(XINC,YINC,NPOL,NX,NY,DATA,DATA2,CAXIS)
```

```
      ELSE
          DA = XINC * YINC / (4. * PI**2)
          CALL FFT2 (1, NX, NY, DA, DATA)
          IF (NPOL.EQ.2) CALL FFT2 (1, NX, NY, DA, DATA2)
      END IF

C     For area factor in FFT (DA) see Kerns, 3.1-3, p. 87

      Time3 = ElapsedTime()

      SXINC = ALAM / (NX*XINC)      ! X increment for spectrum data
      IF (XINC .EQ. 0) SXINC=0
      SYINC = ALAM / (NY*YINC)      ! Y increment for spectrum data
      IF (YINC .EQ. 0) SYINC=0
      SX0 = -(NX/2)*SXINC
      SY0 = -(NY/2)*SYINC
      RSCAN(1) = SX0
      RSCAN(2) = SXINC
      RSCAN(3) = NX
      RSCAN(4) = SY0
      RSCAN(5) = SYINC
      RSCAN(6) = NY
      RSCAN(7) = -FREQ      ! Negative to indicate spectrum data

      WRITE (1,98) BELL
      WRITE (IWRITE,*)
      WRITE (IWRITE,*) ' 11a. Would you like to examine a sector of'
      WRITE (IWRITE,*) '     the data with greater resolution? (N/Y) '
      READ (IREAD,99) CANS
      IF (CANS.EQ.'Y' .OR. CANS.EQ.'y')  THEN
          SXL = -1
          SXU = 1
          WRITE (IWRITE,*) ' 11b. Enter the sector limits for Kx :',
     +                     '  (-1., 1.)'
          READ (IREAD,99) CANS
          IF (CANS .GT. ' ')  READ (CANS,*) SXL, SXU
          IF (SXL.GT.SXU) THEN
              SWAP = SXL
              SXL = SXU
              SXU = SWAP
          END IF
          IL = (SXL-SX0)/SXINC + 1
          RIU = (SXU-SX0)/SXINC + 1.
          IU = RIU
          IF (FLOAT(IU) .LT. RIU)   IU = IU+1

          SYL = -1
          SYU = 1
          WRITE (IWRITE,*) ' 11c. Enter the sector limits for Ky :',
     +                     '  (-1., 1.)'
          READ (IREAD,99) CANS
          IF (CANS .GT. ' ')  READ (CANS,*) SYL, SYU
          IF (SYL.GT.SYU) THEN
              SWAP = SYL
              SYL = SYU
```

```
         SYU = SYL
      END IF
      JL = (SYL-SYO)/SYINC + 1
      RJU = (SYU-SYO)/SYINC + 1.
      JU = RJU
      IF (FLOAT(JU) .LT. RJU)   JU = JU+1

      NXSECT = IU - IL + 1
      NYSECT = JU - JL + 1
      CALL TESTP2 (NXSECT,ISXP2)
      CALL TESTP2 (NYSECT,ISYP2)
      CALL POWRT (NXSECT,NXSECT,ISXP2)
      CALL POWRT (NYSECT,NYSECT,ISYP2)
      IF (NXSECT.GT.NX)  NXSECT = NX
      IF (NYSECT.GT.NY)  NYSECT = NY

      IF (NXSECT.GE.NX .AND. NYSECT.GE.NY) THEN
         WRITE (IWRITE,*) ' *WARNING: Sector size is the entire',
     +                    ' data set.  No resolution '
         WRITE (IWRITE,*) '           enhancement applied. '
      ELSE
         NXP = NXSECT                   ! Old sector size
         NYP = NYSECT                   !    (power of 2)
         IUP = IU + (NXP - IU + IL - 1)/2
         IF (IUP.GT.NX)  IUP = NX
         IF (IUP.LT.NXP)  IUP = NXP
         ILP = IUP - NXP + 1            ! Index of 1st sector point
         SXO = SXO + (ILP-1)*SXINC      ! Coord. "  "     "     "
         JUP = JU + (NYP - JU + JL - 1)/2
         IF (JUP.GT.NY)  JUP = NY
         IF (JUP.LT.NYP)  JUP = NYP
         JLP = JUP - NYP + 1            ! Index of 1st sector point
         SYO = SYO + (JLP-1)*SYINC      ! Coord. "  "     "     "

         DXSECT = SXINC
         DYSECT = SYINC
         XTENT = DXSECT*NXSECT
         YTENT = DYSECT*NYSECT

         REPEAT = .TRUE.
         DO WHILE (REPEAT)
            REPEAT = .FALSE.
            IF (DXSECT .GT. 1.) THEN
               DXSECT = 1.
               WRITE (IWRITE,*)
               WRITE (IWRITE,*) ' WARNING: Kx spacing > 1.  '
     +              ,'Potential error at sector enhancement.'
               WRITE (IWRITE,*)
            END IF
            IF (DYSECT .GT. 1.) THEN
               DYSECT = 1.
               WRITE (IWRITE,*)
               WRITE (IWRITE,*) ' WARNING: Ky spacing > 1.  '
     +              ,'Potential error at sector enhancement.'
               WRITE (IWRITE,*)
```

```
            END IF
            ADXS = ASIN(DXSECT)*RD
            ADYS = ASIN(DYSECT)*RD
            WRITE (IWRITE,220) NXSECT,DXSECT,ADXS,
     +                         NYSECT,DYSECT,ADYS

            WRITE (IWRITE,*)
            WRITE (IWRITE,*) ' 11d. Would you like increased ',
     +                'resolution on the X-axis ? (N/Y)'
            READ (IREAD,99) CANS
            IF (CANS .EQ. 'Y' .OR. CANS .EQ. 'y')  THEN
               NXSECT = NXSECT*2
               IF (NXSECT.GT.4096) NXSECT = NXSECT/2
               DXSECT = XTENT/NXSECT
               REPEAT = .TRUE.
            END IF

            WRITE (IWRITE,*)
            WRITE (IWRITE,*) ' 11e. Would you like increased ',
     +                'resolution on the Y-axis ? (N/Y)'
            READ (IREAD,99) CANS
            IF (CANS .EQ. 'Y' .OR. CANS .EQ. 'y')  THEN
               NYSECT = NYSECT*2
               IF (NYSECT.GT.4096) NYSECT = NYSECT/2
               DYSECT = YTENT/NYSECT
               REPEAT = .TRUE.
            END IF
         END DO

         DSA = SXINC * SYINC
         CALL BLOWUP (DATA,NX,NY,NXP,NYP,ILP,JLP,NXSECT,NYSECT,
     +                         ALAM,DSA)
         IF (NPOL.EQ.2)  CALL BLOWUP (DATA2,NX,NY,NXP,NYP,ILP,JLP,
     +                         NXSECT,NYSECT,ALAM,DSA)

         RSCAN(1) = SXO
         RSCAN(4) = SYO
         RSCAN(2) = SXINC * NXSECT / NXP
         RSCAN(5) = SYINC * NYSECT / NYP
         RSCAN(3) = NXSECT
         RSCAN(6) = NYSECT
         SXINC = RSCAN(2)
         SYINC = RSCAN(5)
         NX = RSCAN(3)
         NY = RSCAN(6)

         WRITE (6,*) ' Results of sector enhancement:'
         WRITE (6,*) ' Old Dimensions = ',NXP,NYP
         WRITE (6,*) ' New Dimensions = ',NXSECT,NYSECT

      END IF
   END IF


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

```
C
C     PROBE CORRECTION & OUTPUT CONVERSION
C

      WRITE (1,*) ' Ready for probe correction section.'
      WRITE (1,*)

      WRITE (1,98) BELL
      WRITE (IWRITE,*)
      WRITE (IWRITE,*) ' 12. What direction is the first polarization?'
      WRITE (IWRITE,*) '     Enter angle (degrees) from Y-axis toward'
      WRITE (IWRITE,*) '     minus X:   (0.) '
      READ (IREAD,99) CANS
      IF (CANS .EQ. ' ')  THEN
         POLY=0.
      ELSE
         READ (CANS,*) POLY
      END IF

      WRITE (IWRITE,*) ' First polarization at ',POLY,' degrees.'

      WRITE (IWRITE,*)
      WRITE (IWRITE,*) ' 13a. Should a probe correction be used? (N/Y)'
      READ (IREAD,99) CANS
      ICORR=-1
      IF (CANS.EQ.'Y' .OR. CANS.EQ.'y')  THEN
         WRITE (IWRITE,*) '13b. Empirical or Theoretical?  (E/T)'
         READ (IREAD,99) CANS
         ICORR=1
         IF (CANS.EQ.'T' .OR. CANS.EQ.'t') ICORR=0

         IPRBR = -1
         WRITE (IWRITE,*) ' 13c. Enter the probe rotation -'
         WRITE (IWRITE,*) '           1   for X into Y, or '
         WRITE (IWRITE,*) '          -1   for Y into X :    (-1) '
         READ (IREAD,99) CANS
         IF (CANS .EQ. '1')  IPRBR=1
         WRITE (IWRITE,*) ' Second polarization at ',
     +                POLY + IPRBR*90,' degrees.'
      END IF

      IF (ICORR.EQ.0) THEN
         A = ALAM/1.6
         B = A/2
         WRITE (IWRITE,*) ' 13d. Enter the probe dimensions in inches.'
         WRITE (IWRITE,*) '      Enter large, small dimensions: ',
     +                    '(',A,',',B,')'
         READ (IREAD,99) CANS
         IF (CANS .GT. ' ')   READ (CANS,*) A,B

         WRITE (IWRITE,*)
         IF (ICORR .EQ. 0) THEN
            WRITE(IWRITE,*)'Correcting for probe size ',A,' x ',B,' "'
C        ELSE
C           WRITE(IWRITE,*)'Gain calc. for probe size ',A,' x ',B,' "'
```

```
        END IF
    END IF

    IF (ICORR.GT.0) THEN
        WRITE (IWRITE,*)
        WRITE (IWRITE,*) ' 13d. For the probe pattern (1st pole) -'
        CALL NAMFILE(8,0)
        WRITE (6,110) NAME
        CALL HEADREAD(8,IRDAT)
        WRITE (6,112) TITLE
        IF ( (NX.NE.RSCAN(3)) .OR. (NY.NE.RSCAN(6)) ) THEN
            WRITE (IWRITE,*) '** File mismatch - program aborted **'
            STOP
        END IF

        WRITE (IWRITE,*)
        WRITE (IWRITE,*) ' 13e. For the probe pattern (2nd pole) -'
        CALL NAMFILE(9,0)
        WRITE (6,110) NAME
        CALL HEADREAD(9,IRDAT)
        WRITE (6,112) TITLE
        IF ( (NX.NE.RSCAN(3)) .OR. (NY.NE.RSCAN(6)) ) THEN
            WRITE (IWRITE,*) '** File mismatch - program aborted **'
            STOP
        END IF
    END IF

    WRITE (IWRITE,*)
    WRITE (IWRITE,*) ' 14a. Specify the type of output data desired:'
    WRITE (IWRITE,*)
    WRITE (IWRITE,*) '      To output the far-field pattern --    '
    WRITE (IWRITE,*) '      Enter  "Y"  for an azimuth/elevation '
    WRITE (IWRITE,*) '                  system (conical about the'
    WRITE (IWRITE,*) '                  Y-axis) rotated about the '
    WRITE (IWRITE,*) '                  Z axis by a specified angle'
    WRITE (IWRITE,*) '      Enter  "H"  for a Huygens system rotated'
    WRITE (IWRITE,*) '                  by a specified angle, '
    WRITE (IWRITE,*) '      Enter  "Z"  for a theta/phi system '
    WRITE (IWRITE,*) '                  system (conical about the '
    WRITE (IWRITE,*) '                  Z-axis) rotated about the '
    WRITE (IWRITE,*) '                  Z axis by a specified angle'
    WRITE (IWRITE,*)
    WRITE (IWRITE,*) '      Or -- '
    WRITE (IWRITE,*) '      Enter  "A"  for a physical translation '
    WRITE (IWRITE,*) '                  of the planar aperture data,'
    WRITE (IWRITE,*) '      or Return to output the transverse '
    WRITE (IWRITE,*) '                  spectrum data'
    READ (IREAD,99) CANS
    NPOUT=0
    NTRANS=0
    IF (CANS.EQ.'Y' .OR. CANS.EQ.'y')  NPOUT=1
    IF (CANS.EQ.'H' .OR. CANS.EQ.'h')  NPOUT=2
    IF (CANS.EQ.'Z' .OR. CANS.EQ.'z')  NPOUT=3
    IF (CANS.EQ.'A' .OR. CANS.EQ.'a')  NTRANS=1
```

```
      IF (NPOL.EQ.1) THEN
         WRITE (IWRITE,*) ' 14b. Would you like to output both ',
     +                     'polarizations? (N/Y)'
         READ (IREAD,99) CANS
         IF (CANS.EQ.'y' .OR. CANS.EQ.'Y')  NPOL=0
         WRITE (IWRITE,*) ' Output ',2-NPOL,' polarizations.'
      END IF

      IF (NTRANS .NE. 0) THEN
         TX=0.
         TY=0.
         TZ=0.
         WRITE (IWRITE,*) ' 14c. Enter translation vector components'
         WRITE (IWRITE,*) '      in inches  (X, Y, Z) :    (0.,0.,0.)'
         READ (IREAD,99) CANS
         IF (CANS .GT. ' ')  READ (CANS,*) TX,TY,TZ

         FILTER=0.
         WRITE (IWRITE,*) ' 14d. Enter low-pass filter radius in '
         WRITE (IWRITE,*) '      normalized wave-number units '
         WRITE (IWRITE,*) '      (Return for no filter)'
         READ (IREAD,99) CANS
         IF (CANS .GT. ' ')  READ (CANS,*) FILTER

         WRITE (IWRITE,*) ' Data origin translated to (',TX,TY,TZ,')'
         WRITE (IWRITE,*) ' Filter applied at Kt = ',FILTER
      END IF

      POLOUT = POLY

      IF (NPOUT.NE.0) THEN
         WRITE (IWRITE,*)
         WRITE (IWRITE,*) ' 14c. What direction is the desired output '
     +                   ,'polarization?  Enter '
         WRITE (IWRITE,*) '      angle (degrees) from Y-axis toward ',
     +                    'minus X:  (',POLY,')'
         READ (IREAD,99) CANS
         IF (CANS .GT. ' ')  READ (CANS,*) POLOUT
         WRITE (IWRITE,*)
         WRITE (IWRITE,*) ' Output pole referenced to ',POLOUT,
     +                 ' degrees. '
         WRITE (IWRITE,*)
      END IF

      POLY = POLY*DR          ! Convert to radians
      POLOUT = POLOUT*DR

      IF (SINGLE) THEN
         IF (CAXIS.EQ.'R') THEN
            RSCAN(4) = 0.
            RSCAN(5) = 0.
         ELSE
            RSCAN(1) = 0.
            RSCAN(2) = 0.
         END IF
```

```
      END IF

13    FORMAT(A)

      WRITE (IWRITE,*) '15. Do you want to apply a ',
     +                 'Blackman filter(N/Y)?'
      READ (IREAD,99) CANS
      IBM=0
C     IF (CANS.EQ.'Y' .OR. CANS.EQ.'y') IBM=1
      IF (CANS.EQ.'Y' .OR. CANS.EQ.'y') THEN
         CFILT=' '
         CTIT=' '
         CTIT2=' '
         IBM=1
C        WRITE(IWRITE,*) '15a. Enter output form for filter, S for '
C        WRITE(IWRITE,*) 'space domain, W for wave number, B for '
C        WRITE(IWRITE,*) 'both, CR for none. '
C        READ(IREAD,13) CFILT
C        IF (CFILT .EQ. 'S' .OR. CFILT .EQ. 'B' .OR.
C    +    CFILT .EQ. 's' .OR. CFILT .EQ. 'b') THEN
C           WRITE(IWRITE,*) '15b. Give name for spatial filter ',
C    +                            'output file.'
C           CALL NAMFILE(13,1)
C           FNAME=NAME
C           WRITE(IWRITE,*) '15c. Default title is ',TITLE
C           WRITE(IWRITE,*) 'Enter alternate title(CR to default)'
C           READ(IREAD,99) CTIT
C        END IF
C        IF (CFILT .EQ. 'W' .OR. CFILT .EQ. 'B' .OR.
C    +    CFILT .EQ. 'w' .OR. CFILT .EQ. 'b') THEN
C           WRITE(IWRITE,*) '15b. Give name for wave # filter ',
C    +                            'output file.'
C           CALL NAMFILE(14,1)
C           FNAME2=NAME
C           WRITE(IWRITE,*) '15c. Default title is ',TITLE
C           WRITE(IWRITE,*) 'Enter alternate title(CR to default)'
C           READ(IREAD,99) CTIT2
C        END IF
      END IF

      IF (NPOL.NE.1) THEN
         CALL PCORR (DATA, NX, NY, DATA2, NX, NY, ICORR, IPRBR,
     +                        NPOL, NPOUT, POLY, POLOUT)
         IF (IBM.EQ.1) CALL BLACKMAN (NPOL, ALAM, NX, NY, BFILT,
     +        DATA, NX, NY, DATA2,CTIT,CTIT2,CFILT,FNAME,FNAME2)
      ELSE
         CALL PCORR (DATA, NX, NY, DUMMY, 1, 1, ICORR, IPRBR,
     +                        NPOL, NPOUT, POLY, POLOUT)
         IF (IBM.EQ.1) CALL BLACKMAN (NPOL, ALAM, NX, NY, BFILT,
     +            DATA, 1, 1, DUMMY,CTIT,CTIT2,CFILT,FNAME,FNAME2)
      END IF


      IF (NPOUT .EQ. 1) THEN
```

```
      IF (POLOUT.EQ.0) THEN
         COPOL = 'El.'
         XPOL = 'Az.'
      ELSE
         WRITE (COPOL,'(F4.0,'' El'')') POLOUT*RD
                          ! "Elevation" pole relative to Y-axis
                          !   rotated by angle POLOUT
         WRITE (XPOL,'(F4.0,'' Az'')') POLOUT*RD
                          ! "Azimuth" pole relative to Y-axis
                          !   rotated by angle POLOUT
      END IF
   ELSE IF (NPOUT .EQ. 2) THEN
      WRITE (COPOL,'(F4.0,'' HyA'')') POLOUT*RD
                          ! Huygens pole "A" relative to Y-axis
                          !   rotated by angle POLOUT
      WRITE (XPOL,'(F4.0,'' HyB'')') POLOUT*RD
                          ! Huygens pole "B" relative to Y-axis
                          !   rotated by angle POLOUT
   ELSE IF (NPOUT .EQ. 1) THEN
      IF (POLOUT.EQ.0) THEN
         COPOL = 'Theta'
         XPOL = 'Phi'
      ELSE
         WRITE (COPOL,'(F4.0,'' Th.'')') POLOUT*RD
                          ! "Theta" pole relative to Z-axis
                          !   rotated by angle POLOUT
         WRITE (XPOL,'(F4.0,'' Phi'')') POLOUT*RD
                          ! "Phi" pole relative to Z-axis
                          !   rotated by angle POLOUT
      END IF
   ELSE IF (ICORR.EQ.0 .AND. NPOUT.EQ.0) THEN
      COPOL = 'Ver. (Y)'
      XPOL = 'Hor. (X)'
   END IF


   TR = ABS(TX) + ABS(TY) + ABS(TZ) + ABS(FILTER)

   IF (TR .NE. 0.) THEN

      IF (SINGLE) THEN

         CALL SEPTRANS(XINC,YINC,NPOL,NX,NY,DATA,DATA2,CAXIS)
      ELSE
         CALL TRANSLATE (DATA,NX,NY,TX,TY,TZ,FILTER)
         IF (NPOL.EQ.2) CALL TRANSLATE (DATA2,NX,NY,TX,TY,TZ,
     +                                   FILTER)
         DSA = SXINC * SYINC*AK0**2
         CALL FFT2 (-1, NX, NY, DSA, DATA)
         IF (NPOL.EQ.2) CALL FFT2 (-1, NX, NY, DSA, DATA2)
      END IF

      XINC = ALAM / (NX*SXINC)
      IF (SXINC .EQ. 0) XINC=0
      YINC = ALAM / (NY*SYINC)
```

```
            IF (SYINC .EQ. 0) YINC=0
            RSCAN(7) = FREQ
            RSCAN(1) = -(NX/2) * XINC
            RSCAN(2) = XINC
            RSCAN(4) = -(NY/2) * YINC
            RSCAN(5) = YINC
         END IF

         IF (RSCAN(2) .EQ. 0) RSCAN(2)=RSCAN(5)     !IF 0, ARBITRARILY SET
         IF (RSCAN(5) .EQ. 0) RSCAN(5)=RSCAN(2)     !  INC1=INC2


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     OUTPUT DATA
C

      WRITE (1,*) ' Ready to output spectrum data files.'
      WRITE (1,*)

      CALL CONVERT (DATA, NX, NY)
      IF (NPOL.NE.1) CALL CONVERT (DATA2, NX, NY)

      POLOUT = POLOUT * RD

      IF (NPOL.EQ.0) THEN
         XFILE = 'for 2nd pole '
         TITLE = ' Second output polarization.'
      END IF

      WRITE (1,98) BELL
      WRITE (IWRITE,*)
201   FORMAT (///,5X, 'Ready to output results from file', 4A)
      IF (NPOL.EQ.1) THEN
         WRITE (IWRITE,201) ' ',COFILE
      ELSE
         WRITE (IWRITE,201) 's ',COFILE,' and ',XFILE
      END IF
      IF (NTRANS.EQ.1) THEN
         IF (NPOL.EQ.1) THEN
202         FORMAT (/' 16. This file contains data translated by (',
     +              3F7.2,')' )
            WRITE (IWRITE,202) TX,TY,TZ
            WRITE(IWRITE,*) 'Enter data file name:'
            READ(IREAD,98) COFILE
         ELSE
203         FORMAT (/' 16',A,'. The ',A,' file contains ',A,'-pole '
     +                    ,'translated data.')
            WRITE (IWRITE,203) 'a', 'first', COPOL
            WRITE(IWRITE,*) 'Enter data file name:'
            READ(IREAD,98) COFILE
            WRITE (IWRITE,203) 'b', 'second', XPOL
            WRITE(IWRITE,*) 'Enter data file name:'
            READ(IREAD,98) XFILE
         END IF
```

```
      ELSE IF (NPOUT.EQ.0) THEN
204       FORMAT (/A, ' file contains ',A,' polarized spectrum data. ')
          IF (NPOL.EQ.1) THEN
              WRITE (IWRITE,204) ' 16. This', COPOL
              WRITE(IWRITE,*) 'Enter data file name:'
              READ(IREAD,98) COFILE
          ELSE
              WRITE (IWRITE,204) ' 16a. The first', COPOL
              WRITE(IWRITE,*) 'Enter data file name:'
              READ(IREAD,98) COFILE
              WRITE (IWRITE,204) ' 16b. The second', XPOL
              WRITE(IWRITE,*) 'Enter data file name:'
              READ(IREAD,98) XFILE
          END IF
      ELSE
205       FORMAT (/' 16',A,' file contains pattern data which is ',A,
     +                   ' polarized',/,'      relatve to the Y-axis'
     +                   ,' rotated ',I4,' degrees. ')
          IF (NPOL.EQ.1) THEN
              WRITE (IWRITE,205) '. This', 'elevation', POLOUT
              WRITE(IWRITE,*) 'Enter data file name:'
              READ(IREAD,98) COFILE
          ELSE
              WRITE (IWRITE,205) 'a. The first', 'elevation', POLOUT
              WRITE(IWRITE,*) 'Enter data file name:'
              READ(IREAD,98) COFILE
              WRITE (IWRITE,205) 'b. The second', 'azimuth', POLOUT
              WRITE(IWRITE,*) 'Enter data file name:'
              READ(IREAD,98) XFILE
          END IF
      END IF
      WRITE (6,110) COFILE, ' ', XFILE

206   FORMAT (//' 17',A,' The default title for file ',A,' is:', //,A80
     +       ,//'      Enter a new title, or RETURN to default: '/ )

      IF (NPOL.EQ.1) THEN
          WRITE (IWRITE,206) '.', COFILE, TEMP
          READ (IREAD,99) CANS
          IF (CANS .GT. ' ')  TEMP = CANS
          WRITE (6,112) TEMP
      ELSE
          WRITE (IWRITE,206) 'a.', COFILE, TEMP
          READ (IREAD,99) CANS
          IF (CANS .GT. ' ') TEMP = CANS
          WRITE (6,112) TEMP
          WRITE (IWRITE,206) 'b.', XFILE, TITLE
          READ (IREAD,99) CANS
          IF (CANS .GT. ' ')  TITLE = CANS
          WRITE (6,112) TITLE
      END IF

      WRITE(IWRITE,*) '17. New data file dimensions are (',NY,' x ',NX,
     +                                                            ')'
      WRITE(IWRITE,*) 'Would you like to change the file dimensions',
```

```
     +                 '(Y/N)?'
      READ(IREAD,99) CANS
      IF (CANS .EQ. 'Y' .OR. CANS .EQ. 'y') THEN
          CALL GRIDSET(4096,0,ISTARTX,ISTARTY,MX,MY,NXO,NYO,IXINC,IYINC)
          RSCAN(1)=RSCAN(1) + (ISTARTX-1)*RSCAN(2)
          RSCAN(2)=RSCAN(2)*IXINC
          RSCAN(3)=MX
          RSCAN(4)=RSCAN(4) + (ISTARTY-1)*RSCAN(5)
          RSCAN(5)=RSCAN(5)*IYINC
          RSCAN(6)=MY
      ELSE
          ISTARTX=1
          ISTARTY=1
          NXO=NX
          NYO=NY
          IXINC=1
          IYINC=1
      END IF

      CALL XYZOPEN(COFILE,4,1)      !OPEN FILE FOR 1ST POL
      IF (NPOL .EQ. 2) THEN
          CALL XYZOPEN(XFILE,5,1)   !OPEN FILE FOR 2ND POL
      END IF

      Time4 = ElapsedTime()

      CALL DateTime (IDATE,ITIME)

      IF (NPOL.NE.1) THEN
          NAME = XFILE
          POL = XPOL
          CALL ARRAY_DUMP (DATA2,NX,NY,NXO,NYO,IXINC,IYINC,
     +                             ISTARTX,ISTARTY,5)
          CALL HEADWRITE (5,IRDAT)
          WRITE(1,*) 'MAXIMUM FOR CROSS-POL FILE IS',AMAX
      END IF

      NAME = COFILE
      POL = COPOL
      TITLE = TEMP
      CALL ARRAY_DUMP (DATA,NX,NY,NXO,NYO,IXINC,IYINC,
     +                             ISTARTX,ISTARTY,4)
      CALL HEADWRITE (4,IRDAT)
      WRITE(1,*) 'MAXIMUM FOR COPOL FILE IS',AMAX

      Time5 = ElapsedTime()

      WRITE (6,*)
      WRITE (6,*) ' Time to input data:  ',TIME1-TIME0,' ms'
      WRITE (6,*) ' Condition for FFT:   ',TIME2-TIME1,' ms'
      WRITE (6,*) ' Perform FFT:         ',TIME3-TIME2,' ms'
      WRITE (6,*) ' Output data:         ',TIME5-TIME4,' ms'
      WRITE (6,*)
      WRITE (6,*)  ' *** NORMAL TERMINATION ***'
```

```
      WRITE(1,98) BELL
      WRITE (1,*)  ' *** NORMAL TERMINATION ***'

      END
```

```
$CDS ON
$EMA /BUFFER/
      SUBROUTINE ARRAY_DUMP(CBUF,NX,NY,NXO,NYO,IXINC,IYINC,
     +                              ISTARTX,ISTARTY,IUNIT)

C     LAST REVISED:   8/5/88

      CHARACTER CSCAN*80,CAXIS*1,NAME*15,POL*8
      COMPLEX CBUF(NX,NY)
      COMMON /PARAM/ RSCAN(7),CAXIS,POL,CSCAN,NAME,IDATE(3),ITIME(3)
      COMMON /BUFFER/ ABUF(4096),PBUF(4096),IBUF
      COMMON /MINMAX/ AMIN,AMAX,PMIN,PMAX,MAXY,MAXX
      EMA CBUF

C     SUBOUTINE TO WRITE AMP, PHASE TO DISK FILE

      AMIN = 100.
      AMAX = -100.
      PMIN = 180.
      PMAX = -180.
      MAXY=0
      MAXX=0

      IBUF=0

      IF (CAXIS.EQ.'Y') THEN
          IROW=1
          DO J=ISTARTX,NXO,IXINC
              IPT=1
              DO I=ISTARTY,NYO,IYINC
                  ABUF(IPT) = REAL(CBUF(J,I))
                  PBUF(IPT) = AIMAG(CBUF(J,I))
                  IPT=IPT+1
              END DO
              CALL WRITE_DATA (IUNIT,IROW,2,2,ABUF,PBUF,IBUF,AMIN,AMAX,
     +                                             PMIN,PMAX,MAXY,MAXX)
              IROW=IROW+1
          END DO
      ELSE
          IROW=1
          DO J=ISTARTY,NYO,IYINC
              IPT=1
              DO I=ISTARTX,NXO,IXINC
                  ABUF(IPT) = REAL (CBUF(I,J))
                  PBUF(IPT) = AIMAG(CBUF(I,J))
                  IPT=IPT+1
              END DO
              CALL WRITE_DATA (IUNIT,IROW,2,2,ABUF,PBUF,IBUF,AMIN,AMAX,
     +                                              PMIN,PMAX,MAXY,MAXX)
              IROW=IROW+1
          END DO
      END IF

      RETURN
      END
```

```
$CDS ON
$EMA /BUFFER/
      SUBROUTINE ARRAY_FILL(CBUF,NXO,NYO,MX,MY,IXINC,IYINC,IUNIT,IPOL)

C     LAST REVISED:   8/5/88

      CHARACTER CAXIS*1,NAME*15,POL*8 ,CSCAN*80
      COMPLEX CBUF(MX,MY)
      COMMON /PARAM/ RSCAN(7),CAXIS,POL,CSCAN,NAME,IDATE(3),ITIME(3)
      COMMON /BUFFER/ ABUF(4096),PBUF(4096),IBUF
      COMMON /MINMAX/ AMIN,AMAX,PMIN,PMAX,MAXY,MAXX
      EMA CBUF

C     ARRAY_FILL fills the data array in memory from the data file
C     on disk.

      NX=INT(RSCAN(3))
      NY=INT(RSCAN(6))

      IF (CAXIS.EQ.'Y') THEN
          DO I=1,MX
              IROW = NXO + (I-1)*IXINC
              CALL READ_DATA(IUNIT,IROW,2,2,ABUF,PBUF,IBUF) !READ FROM FILE
              DO J=1,MY
                  JN = NYO + (J-1)*IYINC
                  IF (IPOL.EQ.1) THEN
                      IF (ABUF(JN) .GT. AMAX) THEN
                          AMAX = ABUF(JN)
                          PMAX = PBUF(JN)
                          MAXX = I
                          MAXY = J
                      END IF
                  END IF
                  IF (ABUF(JN) .LT. AMIN)  AMIN=ABUF(JN)
                  CBUF(I,J)=CMPLX(ABUF(JN),PBUF(JN))
              END DO
          END DO
      ELSE
          DO J=1,MY
              IROW = NYO + (J-1)*IYINC
              CALL READ_DATA(IUNIT,IROW,2,2,ABUF,PBUF,IBUF) !READ FROM FILE
              DO I=1,MX
                  IN = NXO + (I-1)*IXINC
                  IF (IPOL.EQ.1) THEN
                      IF (ABUF(IN) .GT. AMAX) THEN
                          AMAX = ABUF(IN)
                          PMAX = PBUF(IN)
                          MAXX = I
                          MAXY = J
                      END IF
                  END IF
                  IF (ABUF(IN) .LT. AMIN)  AMIN=ABUF(IN)
                  CBUF(I,J)=CMPLX(ABUF(IN),PBUF(IN))
              END DO
          END DO
```

```
END IF

RETURN
END
```

```
$CDS ON
      SUBROUTINE BLACKMAN(NPOL,ALAM,NX,NY,BFILT,DATA,NX2,NY2,DATA2,
     +                CTIT,CTIT2,CANS,FNAME,FNAME2)

      CHARACTER CANS*1,C1*1,CAXIS*1,POL*8,NAME*15,CSCAN*80
      CHARACTER FNAME*15,FNAME2*15,CTIT*80,CTIT2*80,TEMP*80
      COMPLEX CJ,TVAR,BFILT(NX,NY),DATA(NX,NY),DATA2(NX2,NY2)
      EMA BFILT,DATA,DATA2
      COMMON /PARAM/RSCAN(7),CAXIS,POL,CSCAN,NAME,IDATE(3),ITIME(3)
      COMMON /USER/IWRITE,IREAD

C     SUBROUTINE TO APPLY BLACKMAN FILTER

      CJ=(0.,1)
      PI=ACOS(-1.)
      RD=180./PI
      DR=PI/180.
      NUM=NX*NY
      DELX=RSCAN(2)
      DELY=RSCAN(5)

      XINC=ALAM/(DELX*NX)
      YINC=ALAM/(DELY*NY)
      XMIN=-NX*XINC/2
      YMIN=-NY*YINC/2

      KXLPB=PI/XINC
      KYLPB=PI/YINC
      TX=3*PI/KXLPB
      TY=3*PI/KYLPB

      AKXMIN=-ALAM/(2*XINC)
      AKYMIN=-ALAM/(2*YINC)
      AKXDEL=ALAM/(NX*XINC)
      AKYDEL=ALAM/(NY*YINC)

      HBMAX=0.
      DO I=1,NY
          Y=YMIN+(I-1)*YINC
          ARGY=PI*Y/TY
          DO J=1,NX
              X=XMIN+(J-1)*XINC
              IF(ABS(X) .GT. TX .OR. ABS(Y) .GT. TY) THEN
                  HBXY=0.
              ELSE
                  ARGX=PI*X/TX
                  HBXY=0.42+.5*COS(ARGX) + 0.08*COS(2*ARGX)
                  HBXY=HBXY*(0.42+.5*COS(ARGY) + 0.08*COS(2*ARGY))
                  HBMAX=AMAX1(HBMAX,HBXY)
              END IF
              BFILT(J,I)=CMPLX(HBXY,0.0)
          END DO
     'D DO

         *1,NY
```

```
          DO J=1,NX
              BFILT(J,I)=BFILT(J,I)/HBMAX
          END DO
      END DO

      DO I=1,NY
          DO J=1,NX
              TVAR=BFILT(J,I)
              CALL POLAR(TVAR,RE,AI)
              IF (RE .LE. 0) THEN
                  RE=-99.
              ELSE
                  RE=20*ALOG10(RE)
              END IF
              BFILT(J,I)=CMPLX(RE,AI*RD)
          END DO
      END DO

      IF (CANS .EQ.'S' .OR. CANS .EQ. 'B') THEN
          TEMP=CSCAN
          IF (CTIT .GT. ' ') CSCAN=CTIT
          CALL DUMP_FILTER(BFILT,NX,NY,13,FNAME)
          CSCAN=TEMP
      END IF

      DO I=1,NY
          DO J=1,NX
              RE=REAL(BFILT(J,I))
              RE=10.**(RE/20.)
              AI=AIMAG(BFILT(J,I))
              BFILT(J,I)=RE* CEXP(CJ*AI*DR)
          END DO
      END DO

      CALL FFT2(1,NX,NY,1.,BFILT)
      HBMAX=0.

      DO I=1,NY
          DO J=1,NX
              HBMAX=AMAX1(CABS(BFILT(J,I)),HBMAX)
          END DO
      END DO

      DO I=1,NY
          AKY=AKYMIN + (I-1)* AKYDEL
          DO J=1,NX
              AKX=AKXMIN + (J-1)* AKXDEL
              BFILT(J,I)=BFILT(J,I)/HBMAX
              IF (CABS(BFILT(J,I)) .LT. 0.03162) THEN
                  DATA(J,I)=CMPLX(0.0,0.0)
                  IF (NPOL .NE. 1) DATA2(J,I)=CMPLX(0.0,0.0)
              ELSE
                  DATA(J,I)=DATA(J,I)/BFILT(J,I)
                  IF (NPOL .NE. 1) DATA2(J,I)=DATA2(J,I)/BFILT(J,I)
              END IF
```

```
          IF ((AKX*AKO)**2+(AKY*AKO)**2 .GT. KYLPB**2+KXLPB**2) THEN
             DATA(J,I)=CMPLX(0.0,0.0)
             IF (NPOL .NE. 1) DATA2(J,I)=CMPLX(0.0,0.0)
          END IF
          TVAR=BFILT(J,I)
          CALL POLAR(TVAR,RE,AI)
          IF (RE .LE. 0) THEN
             RE=-99.
          ELSE
             RE=20*ALOG10(RE)
          END IF
          BFILT(J,I)=CMPLX(RE,AI*RD)
       END DO
    END DO

    IF (CANS .EQ.'W' .OR. CANS .EQ. 'B') THEN
       TEMP=CSCAN
       IF (CTIT2 .GT. ' ') CSCAN=CTIT2
       CALL DUMP_FILTER(BFILT,NX,NY,14,FNAME2)
       CSCAN=TEMP
    END IF

    RETURN
    END
```

```
$cds on

      SUBROUTINE BLOWUP (DATA,NX,NY,NXP,NYP,ILP,JLP,NXSECT,NYSECT,
     +                        ALAM,DSA)

C     LAST REVISED:   7 OCT 86

C     Replaces sector of data at beginning of array, transforms to
C     space domain and zero-fills to increase resolution, then
C     transforms back to wave-number domain.

      EMA DATA
      COMPLEX DATA(NX*NY)
      INTEGER*4 K,KP

C     NX,NY       Dimensions of original array
C     NXP,NYP     Dimensions of sector before resolution enhancement
C     NXSECT,
C       NYSECT    Dimensions of sector after enhancement
C     ILP,JLP     Starting indices of sector within original array
C     ALAM        Wavelength
C     DSA         Area factor for FFT from spectrum to aperture
C                 (In general, DSA = DSX * DSY = Kx/Ko * Ky/Ko )
C     DA          Area factor for FFT from aperture to spectrum
C                 (In general, DA = DX * DY / 4*PI**2)

C     Download sector:

      DO J=1,NYP
          DO I=1,NXP
              KP = (J-1)*NXP + I
              K = (JLP+J-2)*NX + ILP+I-1
              DATA(KP) = DATA(K)
          END DO
      END DO

C     Zero-fill in space domain:

      CALL FFT2 (-1, NXP, NYP, DSA, DATA)
C     (New) DX = ALAM / (NXP*DSX)
C     (New) DY = ALAM / (NYP*DSY)
      CALL EXPAND (DATA, NXP, NYP, NXSECT, NYSECT)

C     Return to wave-number domain:

      DA = ALAM**2 / (NXP*NYP*DSA) / (4. * PI**2)
      CALL FFT2 (1, NXSECT, NYSECT, DA, DATA)
C     (New) DSX = ALAM / (NXSECT*DX)
C     (New) DSY = ALAM / (NYSECT*DY)

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE CONVERT (DATA, NX, NY)

C     LAST REVISED:   11 OCT 1986

C     Converts the complex array DATA passed in rectangular form to
C     polar form, with the phase in degrees and the amplitude in dB
C     with a floor of -200 dB.

      COMPLEX DATA(NX,NY)
      EMA DATA

      PI = ACOS(-1.)
      RD = 180 / PI

      DO J=1,NY
         DO I=1,NX

            X = REAL(DATA(I,J))
            Y = AIMAG(DATA(I,J))

            PHASE = ATAN2(Y,X) * RD     !PHASE IN DEGREES

            AMP = SQRT(X**2+Y**2)
            IF (AMP .LE. 1.E-10) THEN
               AMP=-200.
            ELSE
               AMP=20*ALOG10(AMP)       ! AMP IN dB
            END IF

            DATA(I,J) = CMPLX (AMP,PHASE)

         END DO
      END DO

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE CORREC(R01X1,R01Y1,R01X2,R01Y2,S10X,S10Y,D1,D2)

C     LAST REVISED:   6 OCT 86

C     Performs probe correction for two polarization measurement in
C     X,Y coordinates.

      COMPLEX R01X1,R01X2,R01Y1,R01Y2,S10X,S10Y,D1,D2,DEL
      DEL = R01X1 * R01Y2 - R01Y1 • R01X2
      S10X = (D1 • R01Y2 -D2 * R01Y1) / DEL
      S10Y = (D2 * R01X1 - D1 * R01X2) / DEL
      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!     SUBROUTINE DATETIME          Last Revised:  6/01/88     !
!                                                             !
!     This routine gets the current date and time from the system  !
!     clock and returns them in two integer arrays as follows:     !
!                                                             !
!             IDATE(1) = 2-digit year code                    !
!             IDATE(2) = month code (1-12)                    !
!             IDATE(3) = day (1-31)                           !
!             ITIME(1) = hours (0-23)                         !
!             ITIME(2) = minutes                              !
!             ITIME(3) = seconds                              !
!                                                             !
!                                                             !
!     Subroutines called:                                     !
!        None                                                 !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE DATETIME (IDATE, ITIME)


      INTEGER IDATE(3), ITIME(3), ITIME11(5), IYEAR, IBUFF(15)
      CHARACTER  FBUFF*30, MONTH*4
      EQUIVALENCE (FBUFF,IBUFF)

      CALL EXEC (11,ITIME11,IYEAR)       ! Numerical time
      CALL FTIME (IBUFF)                  ! Formatted time

      IDATE(1) = IYEAR - 1900
      ITIME(1) = ITIME11(4)
      ITIME(2) = ITIME11(3)
      ITIME(3) = ITIME11(2)

      READ (FBUFF,90)  IDATE(3), MONTH

90    FORMAT (16X, I2, 2X, A4)

      IF (MONTH .EQ. 'JAN.')  IDATE(2) = 1
      IF (MONTH .EQ. 'FEB.')  IDATE(2) = 2
      IF (MONTH .EQ. 'MAR.')  IDATE(2) = 3
      IF (MONTH .EQ. 'APR.')  IDATE(2) = 4
      IF (MONTH .EQ. 'MAY ')  IDATE(2) = 5
      IF (MONTH .EQ. 'JUNE')  IDATE(2) = 6
      IF (MONTH .EQ. 'JULY')  IDATE(2) = 7
      IF (MONTH .EQ. 'AUG.')  IDATE(2) = 8
      IF (MONTH .EQ. 'SEPT')  IDATE(2) = 9
      IF (MONTH .EQ. 'OCT.')  IDATE(2) = 10
      IF (MONTH .EQ. 'NOV.')  IDATE(2) = 11
      IF (MONTH .EQ. 'DEC.')  IDATE(2) = 12

      RETURN
```

```
      END
```

```
$CDS ON
$EMA /BUFFER/
      SUBROUTINE DUMP_FILTER(BFILT,NX,NY,IUNIT,FNAME)
      CHARACTER CAXIS*1,CSCAN*80,NAME*15,POL*8,CTEMP*1,C1*1,FNAME*15
      COMPLEX BFILT(NX,NY)
      EMA BFILT
      COMMON /PARAM/RSCAN(7),CAXIS,POL,CSCAN,NAME,IDATE(3),ITIME(3)
      COMMON /BUFFER/ABUF(4096),PBUF(4096),IBUF
      COMMON /MINMAX/AMIN,AMAX,PMIN,PMAX,MAXROW,MAXCOL
      COMMON /USER/IWRITE,IREAD


C     SUBROUTINE TO DUMP FILTER OUT TO FILE

      CTEMP=CAXIS
      CAXIS='R'                 !STORE BY ROWS
      NAME=FNAME

      AMIN=100.
      AMAX=-100.
      PMIN=180.             !INITIAL VALUES
      PMAX=-180.

      CALL DATETIME(IDATE,ITIME)

      DO I=1,NY
          DO J=1,NX
              ABUF(J)=REAL(BFILT(J,I))          !AMPLITUDE
              PBUF(J)=AIMAG(BFILT(J,I))          !PHASE
          END DO
          IROW=I
          CALL WRITE_DATA(IUNIT,IROW,2,2)
      END DO

      CALL HEADWRITE(IUNIT,2)
      CLOSE(UNIT=IUNIT,IOSTAT=IERR)
      IF (IERR .GT. 0) THEN
          WRITE(1,*) 'ERROR ON CLOSING FILE'
      END IF

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE EEU(U,ETE)

C     LAST REVISED:   6 OCT 86

C   Theoretical probe pattern in E-plane (F2 in memo).

      COMMON /WVGE/ A,B,K
      REAL K
      COMPLEX ETE,ARGC

      IF (U*U .GT. 1) THEN
          ETE = (0.,0.)
      ELSE
          ARG = K * U * B / 2
          ETE = SQRT(SINX(ARG))
          ARGC = CSQRT(CMPLX(1.0 - U * U,0.0))
          ARGC = -K * B * 0.25 * (1.0 - ARGC)
          ETE = ETE * CEXP(ARGC)
      END IF

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE EHU(U,ETH)

C     LAST REVISED:   6 OCT 86

C     Theoretical probe pattern in H-plane (F1 in memo)

      COMMON /WVGE/ A,B,K
      REAL K
      COMPLEX ETH

      PI = 3.141592654
      IF (U*U .GE. 1.) THEN
          ETH = (0.,0.)
      ELSE
          ARG1 = K * U * A / 2.
          ARG2 = K * U * A / PI
          ARG2 = 1.0 - ARG2 * ARG2
          IF (ABS(ARG2) .LE. .0001) THEN
              ETH = PI / 4.
          ELSE
              ETH = COS(ARG1) / ARG2
          END IF
      END IF

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE EXPAND(DATA,MX,MY,NX,NY)

C     LAST REVISED:   6 OCT 86

C     EXPAND moves the old data array ( DATA(MX,MY) ) into the
C     center of a larger array ( DATA(NX,NY) ) and zeros the extra
C     elements (0.,0.).

      EMA DATA
      COMPLEX DATA(NX,NY),TEMP
      INTEGER*4 K,II,JJ,I,J,IO,JO

      MX1 = (NX-MX+1)/2
      MY1 = (NY-MY+1)/2
      MX2 = MX1 + MX
      MY2 = MY1 + MY

      DO J=NY,1,-1
          JO = J-MY1                        ! J COORD. IN OLD ARRAY
          DO I=NX,1,-1
              IF (J.LE.MY1 .OR. J.GT.MY2) THEN
                  DATA(I,J) = (0.,0.)
              ELSE IF (I.LE.MX1 .OR. I.GT.MX2) THEN
                  DATA(I,J) = (0.,0.)
              ELSE
                  IO = I - MX1              ! I COORD. IN OLD ARRAY
                  K = (JO-1)*MX + IO        ! ABSOLUTE (1-DIM.) POSITION
                  JJ = (K-1)/NX + 1         ! OLD ELEMENT POSITON IN
                  II = K - (JJ-1)*NX        !      NEW ARRAY
                  DATA(I,J) = DATA(II,JJ)
              END IF
          END DO
      END DO

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE FFT2 (ISN, NX, NY, DA, DATA)

C     LAST REVISED:   6 OCT 86

C     Routine to calculate the Fast Fourier Transform or the
C     inverse FFT of an input two-dimensional, complex array
C     (DATA).  Returns result in the same array.
C
C     NX and NY are the dimensions of the array DATA and must
C     be non-negative integer powers of 2.
C
C     ISN is the control variable equal to +1 or -1.
C     (ISN is the sign of the exponent.)
C
C     DA is an area correction factor.
C
C     The origins of both input and output coordinate systems are
C     located at the (NX/2+1,NY/2+1) point of the array.
C
      EMA  DATA
      COMPLEX DATA(NX,NY),T1,T2
      REAL PI2,SO,CO,SI,CI,SN,CS,SOISN
      COMMON /USER/ IWRITE,IREAD

C     IF(IABS(ISN).NE.1)GO TO 24

C     WRITE(1,*) 'DA= ',DA

      PI2=2.*ACOS(-1.)

      IX=-1
      M=0
      DO WHILE (NX .GT. M)
          IX=IX+1
          M=2**IX
          IF (NX .LT. M) THEN
              WRITE (IWRITE,*) ' FFT ERROR: NX must be a power of 2.'
              STOP
          END IF
      END DO

      IY=-1
      M=0
      DO WHILE (NY .GT. M)
          IY=IY+1
          M=2**IY
          IF (NY .LT. M) THEN
              WRITE (IWRITE,*) ' FFT ERROR: NY must be a power of 2.'
              STOP
          END IF
      END DO

      NX2=NX/2
      NY2=NY/2
```

```
DO I=1,NX2,1
    I1=I+NX2
    DO J=1,NY,1
        T1=DATA(I,J)
        DATA(I,J)=DATA(I1,J)
        DATA(I1,J)=T1
    END DO
END DO

DO J=1,NY2,1
    J1=J+NY2
    DO I=1,NX,1
        T2=DATA(I,J)
        DATA(I,J)=DATA(I,J1)
        DATA(I,J1)=T2
    END DO
END DO

NXBIT=16-IX
NX1=NX-2
DO I=1,NX1,1
    IFLIP=0
    DO J=NXBIT,15,1
        N=NXBIT-J
        N=N+15
        IFLIP=2*IFLIP+IAND(ISHFTC(I,N+1,16),1)
    END DO
    IF(I.GT.IFLIP) THEN
        I1=I+1
        I2=IFLIP+1
        DO J=1,NY,1
            T1=DATA(I2,J)
            DATA(I2,J)=DATA(I1,J)
            DATA(I1,J)=T1
        END DO
    END IF
END DO

NYBIT=16-IY
NY1=NY-2
DO J=1,NY1,1
    JFLIP=0
    DO I=NYBIT,15,1
        M=NYBIT-I
        M=M+15
        JFLIP=2*JFLIP+IAND(ISHFTC(J,M+1,16),1)
    END DO
    IF(J.GT.JFLIP) THEN
        J1=J+1
        J2=JFLIP+1
        DO I=1,NX,1
            T2=DATA(I,J2)
            DATA(I,J2)=DATA(I,J1)
            DATA(I,J1)=T2
```

```
            END DO
        END IF
    END DO

    DO I=1,IX,1
        NEL=2**I
        NEL2=NEL/2
        NSET=NX/NEL
        SI=SIN(PI2/NEL)
        CI=COS(PI2/NEL)
        DO K=1,NSET,1
            INCR=(K-1)*NEL
            SO=0.0
            CO=1.0
            DO L=1,NEL2,1
                I1=L+INCR
                I2=I1+NEL2
                DO J=1,NY,1
                    T1=DATA(I1,J)
                    SOISN=SO*(FLOAT(ISN))
                    T2=DATA(I2,J)*CMPLX(CO,SOISN)
                    DATA(I1,J)=T1+T2
                    DATA(I2,J)=T1-T2
                END DO
                SN=SO*CI+CO*SI
                CS=CO*CI-SO*SI
                CO=CS
                SO=SN
            END DO
        END DO
    END DO

    DO J=1,IY,1
        NEL=2**J
        NEL2=NEL/2
        NSET=NY/NEL
        SI=SIN(PI2/NEL)
        CI=COS(PI2/NEL)
        DO K=1,NSET,1
            INCR=(K-1)*NEL
            SO=0.0
            CO=1.0
            DO L=1,NEL2,1
                J1=L+INCR
                J2=J1+NEL2
                DO I=1,NX,1
                    T1=DATA(I,J1)
                    SOISN=SO*(FLOAT(ISN))
                    T2=DATA(I,J2)*CMPLX(CO,SOISN)
                    DATA(I,J1)=T1+T2
                    DATA(I,J2)=T1-T2
                END DO
                SN=SO*CI+CO*SI
                CS=CO*CI-SO*SI
                CO=CS
```

```
                  SO=SN
               END DO
            END DO
         END DO

         DO I=1,NX2,1
            I1=I+NX2
            DO J=1,NY,1
               T1=DATA(I,J)
               DATA(I,J)=DATA(I1,J)
               DATA(I1,J)=T1
            END DO
         END DO

         DO J=1,NY2,1
            J1=J+NY2
            DO I=1,NX,1
               T2=DATA(I,J)
               DATA(I,J)=DATA(I,J1)
               DATA(I,J1)=T2
            END DO
         END DO

         IF (DA .NE. 1.) THEN
            DO J=1,NY
               DO I=1,NX
                  DATA(I,J)=DATA(I,J)*DA
               END DO
            END DO
         END IF

         RETURN

C  24 CONTINUE
C        WRITE(6,*)  ' ISN IS NOT +1 OR -1 IN FFT2 '
C        RETURN

         END
```

```
$CDS ON
      REAL FUNCTION GOWAVGD ()

      COMMON /WVGE/ A,B,AKO

C     INITIALIZATION

      EPS = .001
      STANDARD = .01
      PI = ACOS(-1.)
      ALAM = 2. * PI / AKO
      BETAO = SQRT(1. - (PI / (AKO * A))**2)
      G01=0.
      N = 62

C     REPEAT

 10   N = N * 2
      DELTHETA = PI / (N - 1)
      G02 = G01
      AMAXTHE = 7 * PI / 12
      D = 0.
      I = 0

 20   I = I + 1
      THETA = (I - 1) * DELTHETA
      IF (THETA .LE. AMAXTHE) THEN
          DD1 = (1. + BETAO) * AKO * B / 2. * SIN(THETA)
          DN1A = 1. + BETAO * COS(THETA)
          DN1B = SIN(AKO * B / 2. * SIN(THETA))
          DN1 = DN1A * DN1B
          D1 = (DN1 / DD1)**2
          IF (ABS(SIN(THETA) - (ALAM / (2. * A))) .LT. EPS) THEN
              D2 = (PI * COS(THETA) / 2)**2
          ELSE
              DD2 = (PI / 2.)**2 - (AKO * A / 2 * SIN(THETA))**2
              DN2 = COS(THETA) * COS(AKO * A / 2. * SIN(THETA))
              D2 = ((PI / 2.)**2 * (DN2 / DD2))**2
          ENDIF
      END IF

 30   D = (D1 + D2) * SIN(THETA) * DELTHETA + D
      IF ((THETA + DELTHETA) .LT. (PI - EPS)) GOTO 20
      G01 = 4 / D
      DIFF = ABS(G02 - G01) / G01
      IF ((DIFF .GT. STANDARD) .AND. (N .LE. 1000)) GOTO 10
      IF (N .GT. 1000) THEN
          WRITE (1,*) 'WARNING:  Probe gain fails to converge.'
      ENDIF
      GOWAVGD = G01

      RETURN

      END
```

```
$cds on

      SUBROUTINE GETPAT (J, NY, AMP1X, PHASE1Y, AMP2X, PHASE2Y)

      DIMENSION AMP1X(4096), PHASE1Y(4096), AMP2X(4096), PHASE2Y(4096)

      EMA  AMP1X,AMP2X,PHASE1Y,PHASE2Y

      DTOR = ACOS(-1.)/180.    !  degrees to radians

      CALL READ_DATA (8, J, 2, 2, AMP1, PHASE1, JUNK)
      CALL READ_DATA (9, J, 2, 2, AMP2, PHASE2, JUNK)

      DO J=1,NY
          AMP = 10.**(AMP1X(J)/20.)
          PHASE = PHASE1Y(J)*DTOR
          AMP1X(J) = AMP*COS(PHASE)
          PHASE1Y(J) = AMP*SIN(PHASE)

          AMP = 10.**(AMP2X(J)/20.)
          PHASE = PHASE2Y(J)*DTOR
          AMP2X(J) = AMP*COS(PHASE)
          PHASE2Y(J) = AMP*SIN(PHASE)
      END DO

      RETURN

      END
```

```
$CDS ON
      SUBROUTINE GRIDSET(MAXPTS,ITIT,ISTARTX,ISTARTY,MX,MY,NX,NY,
     +                             IXINC,IYINC)
      CHARACTER CAXIS*1,POL*8,CSCAN*80,NAME*15,CSTEP*10,TEMP*80
      COMMON /PARAM/RSCAN(7),CAXIS,POL,CSCAN,NAME,IDATE(3),ITIME(3)
      COMMON /USER/IWRITE,IREAD

C     LAST UPDATED: 4/2/87

C     SUBROUTINE TO PROMPT USER FOR OPTIONS TO DETERMINE GRID OF DATA
C     TO BE USED FOR PLOTTING OR LISTING.
C
C     MAXPTS---SUPPLIED BY CALLING ROUTINE. DETERMINES MAXIMUM NO. OF
C              PTS TO BE PLOTTED OR LISTED
C     ITIT---SUPPLIED BY CALLING ROUTINE. DETERMINES IF USER IS PROMPTED
C            FOR TITLE

C     ALL FOLLOWING VALUES ARE RETURNED BY GRIDSET
C
C     ISTARTX---STARTING X PT TO BE PLOTTED
C     ISTARTY---STARTING Y PT TO BE PLOTTED
C     MX--- THE NUMBER OF X PTS TO BE PLOTTED
C     MY--- THE NUMBER OF Y PTS TO BE PLOTTED
C     NX--- THE LAST X PT TO BE PLOTTED
C     NY--- THE LAST Y PT TO BE PLOTTED
C     IXINC---THE X THINNING INCREMENT
C     IYINC---THE Y THINNING INCREMENT

      NX=RSCAN(3)
      NY=RSCAN(6)

      WRITE(IWRITE,*)
     +   'ENTER CARR. RET. TO DEFAULT THE FOLLOWING QUESTIONS'
      WRITE(IWRITE,*)

      IF (ITIT .EQ. 1) THEN
          WRITE(IWRITE,*) 'THE CURRENT TITLE IS:'
          WRITE(IWRITE,*) CSCAN
          WRITE(IWRITE,*) 'ENTER THE TITLE YOU WOULD LIKE TO PRINT'
          READ(IREAD,10) TEMP
          IF (TEMP .GT. ' ') CSCAN=TEMP
      END IF

3     WRITE(IWRITE,*)
     +     'ENTER X AXIS STARTING,ENDING PT. TO BE PLOTTED(1,',NX,')'
      READ(IREAD,10) CSTEP
      IF (CSTEP .GT. ' ') THEN
          READ(CSTEP,*) ISTARTX,IENDX
          IF (ISTARTX .LT. 1 .OR. ISTARTX .GT. IENDX) GOTO 3
          IF (IENDX .GT. NX) GOTO 3
      ELSE
          ISTARTX=1
          IENDX=NX
      END IF
```

```
4     WRITE(IWRITE,*)
     +   'ENTER Y AXIS STARTING,ENDING PT. TO BE PLOTTED(1,',NY,')'
      READ(IREAD,10) CSTEP
      IF (CSTEP .GT. ' ') THEN
          READ(CSTEP,*) ISTARTY,IENDY
          IF (ISTARTY .LT. 1 .OR. ISTARTY .GT. IENDY) GOTO 4
          IF (IENDY .GT. NY) GOTO 4
      ELSE
          ISTARTY=1
          IENDY=NY
      END IF

      XSTEP=(IENDX-ISTARTX+1)/FLOAT(MAXPTS)
      YSTEP=(IENDY-ISTARTY+1)/FLOAT(MAXPTS)

      IF (XSTEP .LE. 1) THEN
          IXINC=1
      ELSE IF (XSTEP .NE.  INT(XSTEP)) THEN
          IXINC=INT(XSTEP+1.)
      ELSE
          IXINC=INT(XSTEP)
      END IF

      IF (YSTEP .LE. 1) THEN
          IYINC=1
      ELSE IF (YSTEP .NE.  INT(YSTEP)) THEN
          IYINC=INT(YSTEP+1.)
      ELSE
          IYINC=INT(YSTEP)
      END IF

16    WRITE(IWRITE,*)
     +   'ENTER X AXIS THINNING INCREMENT(INTEGER .GE. ',IXINC,')'
      READ(IREAD,10) CSTEP
      IF (CSTEP .GT. ' ') THEN
          READ(CSTEP,*) IX
          IF (IX .LT. IXINC) GOTO 16
          IXINC=IX
      END IF

18    WRITE(IWRITE,*)
     +     'ENTER Y AXIS THINNING INCREMENT(INTEGER .GE. ',IYINC,')'
      READ(IREAD,10) CSTEP
      IF (CSTEP .GT. ' ') THEN
          READ(CSTEP,*) IY
          IF (IY .LT. IYINC) GOTO 18
          IYINC=IY
      END IF

      MX=1 + (IENDX-ISTARTX)/IXINC      !# OF X PTS
      MY=1 + (IENDY-ISTARTY)/IYINC      !# OF Y PTS

      NX=ISTARTX+(MX-1)*IXINC           !LAST X PT
      NY=ISTARTY+(MY-1)*IYINC           !LAST Y PT
```

```
10    FORMAT(A)

      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!     SUBROUTINE HEADER          Last Revised:  6/03/88       !
!                                                             !
!     Entry points:                                           !
!         HEADREAD                                            !
!         HEADWRITE                                           !
!                                                             !
!     This routine reads or writes the header record of a data !
!         file depending on which entry point is used.        !
!             IUNIT - Unit number of the data file.           !
!             IRDAT - Indicates whether amplitude and/or      !
!                     phase information is stored in the file. !
!                                                             !
!                                                             !
!     Subroutines called:                                     !
!         None                                                !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE HEADER

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                  IDATE(3), ITIME(3), NPOL
      COMMON /MINMAX/  AMIN, AMAX, PMIN, PMAX, MAXY, MAXX
      COMMON /USER/  IWRITE, IREAD

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15


      ENTRY HEADWRITE (IUNIT,IRDAT)     ! To write the header record

      INQUIRE(UNIT=IUNIT,IOSTAT=IERR,ERR=17,RECL=IRECLB)  !GET RECORD LENGTH
      NDUM=(IRECLB-168)/2        !NUMBER OF DUMMY VAR. TO WRITE OUT
      WRITE(UNIT=IUNIT,IOSTAT=IERR,ERR=17,REC=1) RSCAN,CAXIS,POL,CSCAN,
     +        NAME,IDATE,ITIME,AMIN,AMAX,PMIN,PMAX,MAXY,MAXX,IRDAT,
     +                NPOL,(IDUM,I=1,NDUM)

17    IF (IERR .GT. 0) THEN
         WRITE(IWRITE,*) 'ERROR ',IERR,' WRITING HEADER'
         PAUSE
      END IF

      RETURN

C

      ENTRY HEADREAD(IUNIT,IRDAT)           ! To read the header record

      READ (UNIT=IUNIT,IOSTAT=IERR,ERR=27,REC=1) RSCAN,CAXIS,POL,CSCAN,
     +       NAME,IDATE,ITIME,AMIN,AMAX,PMIN,PMAX,MAXY,MAXX,IRDAT,
     +       NPOL

27    IF (IERR .GT. 0) THEN
```

```
      WRITE(IWRITE,*) 'ERROR ',IERR,' READING HEADER'
   END IF

   RETURN


   END
```

```
$CDS ON
!--------------------------------------------------------------!
!                                                              !
!    SUBROUTINE NAMFILE          Last Revised:  6/03/88        !
!                                                              !
!    This subroutine opens a datafile for subsequent reads or  !
!        writes.  IUNIT is the unit number to be associated with !
!        the file.  ISTATUS is the status of the file:         !
!            ISTATUS = 0  -  New file                          !
!                    = 1  -  Old file                          !
!                    = 2  -  Status unknown                    !
!            DDIR is the data directory, if other than         !
!                    ::XYZFILES                                !
!                                                              !
!        LGBUF is a library subroutine to enlarge I/O buffer size. !
!        NOTE:  the buffer array LBUF must not be in EMA under any !
!            circumstances.                                    !
!        NOTE:  if CDS is used, then either the common block   !
!            /RECBUFF/ must be declared in the main program and !
!            this subroutine, or the call to LGBUF must be made !
!            in the main program (in which case /RECBUFF/ is not !
!            required.)                                        !
!                                                              !
!                                                              !
!    Subroutines called:                                       !
!        DATETIME                                              !
!                                                              !
!--------------------------------------------------------------!

C     SUBROUTINE NAMFILE (IUNIT, ISTATUS, DDIR)
      SUBROUTINE NAMFILE (IUNIT, ISTATUS)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                   IDATE(3), ITIME(3), NPOL
      COMMON /RECBUFF/  LBUF(8200)
      COMMON /USER/  IWRITE, IREAD

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15
      CHARACTER DDIR*16, INFILE*30, STAT*7


C     NP = PCOUNT()           ! Number of parameters passed
C     IF (NP .LT. 3)  DDIR = '/XYZFILES '

C     ID = INDEX (DDIR, ' ') - 1     ! Length of string
C     IF (ID .LE. 0) ID=16

5     WRITE (IWRITE,*) 'Enter data file name:'
      READ (IREAD,20) NAME
20    FORMAT(A)
C     INFILE = DDIR(1:ID)// '/' // NAME
      INFILE = NAME//'::XYZFILES'

      IF (ISTATUS .EQ. 0) STAT='OLD    '
      IF (ISTATUS .EQ. 1) STAT='NEW    '
```

```
      IF (ISTATUS .EQ. 2) STAT='UNKNOWN'

      IF (STAT .EQ. 'NEW') THEN
          NPTS=RSCAN(6)
          IF (CAXIS .EQ. 'X') NPTS=RSCAN(3)
          IRECLB=(NPTS*4)+2  !RECORD LENGTH(BYTES)--AMP OR PHASE AND STATUS
          IF (IRECLB .LT. 180) IRECLB=180  !INSURE ENOUGH ROOM FOR HEADER REC.
          CALL DATETIME (IDATE,ITIME)
      ELSE
          INQUIRE(FILE=INFILE,IOSTAT=IERR,ERR=65,RECL=IRECLB) !READ RECORD LTH
      END IF


      OPEN(UNIT=IUNIT,FILE=INFILE,ACCESS='DIRECT',FORM='UNFORMATTED',
     +    RECL=IRECLB,IOSTAT=IERR,ERR=65,STATUS=STAT)


65    IF (IERR .GT. 0) THEN
          WRITE(IWRITE,*) 'ERROR ',IERR,' ON OPENING FILE'
          GOTO 5
      ELSE
          CALL LGBUF (LBUF,IRECLB/2)       !ENLARGE I/O BUFFER TO #BYTES/2
      END IF

      RETURN

      END
```

```
$CDS ON
      SUBROUTINE NFNORM (DATA,NX,NY,AKX,AKY)

C     LAST REVISED:   6 OCT 86

      CHARACTER CSCAN*80,CAXIS*1,POL*8,NAME*15
      COMMON /PARAM/ RSCAN(7),CAXIS,POL,CSCAN,NAME,IDATE(3),ITIME(3)
      COMMON /MINMAX/ AMIN,AMAX,PMIN,PMAX,MAXY,MAXX
      COMPLEX DATA(NX,NY),CJ
      EMA DATA

      CJ = (0.,1.)
      DR = ACOS(-1.) / 180.
      XINC = RSCAN(2)
      YINC = RSCAN(5)
      MX = NX/2 + 1
      MY = NY/2 + 1

C     NORMALIZE AND CONVERT TO RECTANGULAR FORM

      DO J=1,NY
          PY = (J-MY)*YINC*AKY
          DO I=1,NX
              PX = (I-MX)*XINC*AKX
              AMP=10.**( (REAL(DATA(I,J))-AMAX) /20.)
              PHS= (AIMAG(DATA(I,J))-PMAX) *DR
              DATA(I,J)=AMP*CEXP(CJ* (PHS + (PX+PY) ))
          END DO
      END DO

      RETURN

      END
```

```
$CDS ON
      FUNCTION PCALC (GAM, SX, SY, S10X, S10Y)

C     LAST REVISED:   14 Mar 86

C     Incremental calculation used to accumulate total power sum.

      COMPLEX  S10X,S10Y,B1,B2

      PCALC = 0.

      IF (GAM.EQ.0)  THEN
         PCALC = CABS(S10X)**2+CABS(S10Y)**2
      ELSE IF (GAM .LT. 0.9999)  THEN
         SZ=SQRT(1-GAM)

C        B1,B2 are b-sub-q (m,k), scalar spectral density functions(p. 55)
C        Kerns 1.2-1.5a, p. 57
         B1 = (SX*S10X+SY*S10Y)
         B2 = (-SY*S10X+SX*S10Y)

C        Kerns 1.4-9, p. 65
         PCALC = CABS(B1)**2/SZ + CABS(B2)**2*SZ
      END IF

      RETURN

      END
```

```
$cds on

      SUBROUTINE PCORR (DATA,NX,NY,DATA2,NX2,NY2,ICORR,IPRBR,NPOL,
     +                  NPOUT,POLIN,POLOUT)

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     Subroutine to do probe correction, accumulate power sum, and     C
C         convert to desired output polarization.                      C
C                                                                      C
C     Last Revised:     10 Aug 88                                      C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      COMMON /USER/ IWRITE,IREAD
      COMMON /WVGE/ A,B,AKO
      COMMON /PARAM/ RSCAN(7),CAXIS,POL,CSCAN,NAME,IDATE(3),ITIME(3)
      COMPLEX D1,D2
      COMPLEX DATA(NX,NY),DATA2(NX2,NY2),F1A,F1B,F2A,F2B,FA,FB
      COMPLEX SD1X,SD1Y,SD2X,SD2Y,SD1XR,SD1YR,SD2XR,SD2YR,S10X,S10Y
      DIMENSION PPAT1X(4096),PPAT1Y(4096),PPAT2X(4096),PPAT2Y(4096)
      CHARACTER CAXIS*1,POL*8,CSCAN*80,NAME*15
      EMA DATA,DATA2,PPAT1X,PPAT1Y,PPAT2X,PPAT2Y

C     Z0 is characteristic impedence of transmission line to probe.

      Z0 = 50                       ! 50 Ohms
      PI = ACOS(-1.)
      ALAM = 2. * PI / AKO
      SX0 = RSCAN(1)
      SY0 = RSCAN(4)
      SXINC = RSCAN(2)
      SYINC = RSCAN(5)
      POWER = 0.
      CPOLI = COS(POLIN)
      SPOLI = SIN(POLIN)
      CPOLO = COS(POLOUT)
      SPOLO = SIN(POLOUT)



C         GMAX is the probe gain on axis.

C         ALAMGP2 = 1. / (1 - (ALAM / (2 * A))**2)
C         IF (ALAMGP2 .GT. 0) THEN
C             ZPRIME = SQRT(ALAMGP2)
C             PTRANS = 4. * ZPRIME / (1 + ZPRIME)**2
C         ELSE
C             ZPRIME = 1.
C             WRITE (IWRITE,*) 'WARNING:  Probe dimensions too small ',
C    +                         'in subroutine PCORR.'
C             PTRANS = 1.
C         END IF
C         GMAX = PTRANS * 32 * A * B / (PI * ALAM**2)
```

```
      IF (ICORR.EQ.0) THEN
          GMAX=GOWAVGD()
          SMAX =.0164*ALAM*SQRT(GMAX)
      ELSE
          SMAX=1
      END IF

C         SMAX is the probe spectrum peak as defined by Kerns 1.6-19
C         and 1.6-21a, page 76-77.
C
C         SMAX = SQRT(GMAX)*(4*PI*AKO**2*377/Z0)**-0.5
C
C         (Where Z0 is transmission line impedance to the probe - 50 ohms)


C                        For gain relative to available power, use the factor
C                            SQRT ( 4 * PI * Z0*AKO**2 / 377 )
C                        (See Kerns, 1.6-6, p. 74)

      GAINFAC=SQRT(4.*PI*AKO**2*Z0/377)

      IF (ICORR .GE. 0)  THEN

C         Probe correction (Polarizations are A and B) :

          DO J=1,NY
              SY=SY0+(J-1)*SYINC
              IF (ICORR.GT.0) CALL GETPAT (J, NY, PPAT1X, PPAT1Y,
     +                                        PPAT2X, PPAT2Y)

              DO I=1,NX
                  SX=SX0+(I-1)*SXINC

                  D1 = DATA(I,J)
                  IF (NPOL.EQ.2) THEN
                      D2=DATA2(I,J)
                  ELSE
                      D2=(0.,0.)
                  END IF

                  GAM = SX*SX + SY*SY
                  IF (GAM.GE..9999)  THEN
                      D1 = (0.,0.)
                      D2 = (0.,0.)
                  ELSE
                      UA = -CPOLI*SX + SPOLI*SY  ! Aperture position relative
                      VA = SPOLI*SX + CPOLI*SY   !     to probe orientation.
                      UB = -VA * IPRBR           ! Ditto, after probe rotation
                      VB = UA * IPRBR            !

                      IF (ICORR.EQ.0) THEN
                          CALL EHU(UA,F1A)         ! Theoretical probe pattern
                          CALL EEU(VA,F2A)         !     for principal planes
                          CALL EHU(UB,F1B)         !
                          CALL EEU(VB,F2B)         !
```

```
C     An electric source spectrum is assumed. Huygens must be converted
C     before using

                          FA = F1A*F2A*SMAX
                          FB = F1B*F2B*SMAX

                          SD1X = FA*SPOLI
                          SD1Y = FA*CPOLI
                          SD2X = -FB*CPOLI*IPRBR
                          SD2Y = FB*SPOLI*IPRBR
                      ELSE
                          SD1X = PPAT1X (I)
                          SD1Y = PPAT1Y (I)
                          SD2X = PPAT2X (I)
                          SD2Y = PPAT2Y (I)
                      END IF

C                     Convert transmit probe spectra to receive spectra :
                      CALL S10T01(SD1X,SD1Y,SX,SY,SD1XR,SD1YR)
                      CALL S10T01(SD2X,SD2Y,SX,SY,SD2XR,SD2YR)

C                     Probe correction :
                      CALL CORREC(SD1XR,SD1YR,SD2XR,SD2YR,S10X,S10Y,
     +                            D1,D2)
                      D1 = S10Y
                      D2 = S10X

C                     Accumulate total power sum :
                      IF (ICORR.EQ.0)
     +                    POWER = POWER + PCALC(GAM,SX,SY,S10X,S10Y)

C                     Convert to output polarization :
C

                      IF (NPOUT.EQ.1) CALL XYTYCON(SX,SY,S10X,S10Y,SPOLO,
     +                                            CPOLO,D1,D2)
                      IF (NPOUT.EQ.2) CALL XYTHUY(SX,SY,S10X,S10Y,
     +                                            SPOLO,CPOLO,D1,D2)
                      IF (NPOUT.EQ.3) CALL XYTZCON(SX,SY,S10X,S10Y,SPOLO,
     +                                            CPOLO,D1,D2)
                      IF (NPOUT.NE.0) THEN
                          D1 = D1 * GAINFAC
                          D2 = D2 * GAINFAC
                      END IF
                  END IF


                  DATA(I,J) = D1
                  IF (NPOL.NE.1)  DATA2(I,J) = D2

              END DO
          END DO

      ELSE
```

```
C        No probe correction :

         DO J=1,NY
             SY=SY0+(J-1)*SYINC
             DO I=1,NX
                 SX=SX0+(I-1)*SXINC

                 D1 = DATA(I,J)/SMAX
                 IF (NPOL.EQ.2) THEN
                     D2 = DATA2(I,J)/SMAX
                 ELSE
                     D2 = (0.,0.)
                 END IF

                 GAM = SX*SX + SY*SY
                 IF (GAM.GE..9999) THEN
                     D1 = (0.,0.)
                     D2 = (0.,0.)
                 ELSE
C                    Notice that D1 is Y-component if no rotation
                     S10X =(-D2*CPOLI*IPRBR + D1*SPOLI)
                     S10Y =(D2*SPOLI*IPRBR + D1*CPOLI)

                     D1=S10Y
                     D2=S10X

                     IF (NPOUT.EQ.1)  CALL XYTYCON(SX,SY,S10X,S10Y,
     +                                     SPOLO,CPOLO,D1,D2)
                     IF (NPOUT.EQ.2)  CALL XYTHUY(SX,SY,S10X,S10Y,
     +                                     SPOLO,CPOLO,D1,D2)
                     IF (NPOUT.EQ.3)  CALL XYTZCON(SX,SY,S10X,S10Y,
     +                                     SPOLO,CPOLO,D1,D2)
                     IF (NPOUT.NE.0) THEN
                         D1 = D1 * GAINFAC
                         D2 = D2 * GAINFAC
                     END IF

                     POWER = POWER + PCALC(GAM,SX,SY,S10X,S10Y)
                 END IF

                 DATA(I,J) = D1
                 IF (NPOL.NE.1)  DATA2(I,J) = D2

             END DO
         END DO

     END IF

C    DELK=AK0**2*SYINC*SXINC
C    POWER=POWER*DELK/(240.*PI)
C    POW=20.*ALOG10(POWER)
C    WRITE(1,*) 'TOTAL RADIATED POWER IS ',POW

     RETURN
```

```
      END
```

```
$CDS ON
      SUBROUTINE POLAR (DATA, AMP, PHA)

C     LAST REVISED:   9 OCT 86

      COMPLEX DATA

      X = REAL(DATA)
      Y = AIMAG(DATA)

      AMP = SQRT(X**2+Y**2)
      PHA = ATAN2(Y,X)

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE POWRT(N,NP2,NADD)

C     LAST REVISED:   6 OCT 86

      NP2=ALOG(FLOAT(N))/0.69314718+0.001
      NP2=NP2+NADD
      NP2=2**NP2
      RETURN
      END
```

```
$CDS ON
!-------------------------------------------------------------!
!                                                             !
!    SUBROUTINE READWRITE        Last Revised:  6/04/88       !
!                                                             !
!    Entry points:                                            !
!        READ_DATA                                            !
!        WRITE_DATA                                           !
!                                                             !
!    Depending on which entry point is used, this routine reads  !
!        a row of data from, or writes a row of data to, a data  !
!        file.                                                !
!            IUNIT - Unit number of data file                 !
!            IROW - Number of the row or column to be transferred  !
!            IRDAT = 0  - only amplitude is recorded          !
!                  = 1  - only phase is recorded              !
!                  = 2  - amplitude and phase are recorded    !
!            IDATA = 0 - only amplitude information is transferred !
!                  = 1 - only phase information is transferred  !
!                  = 2 - both amplitude and phase are transferred  !
!                                                             !
!                                                             !
!    Subroutines called:                                      !
!        None                                                 !
!                                                             !
!-------------------------------------------------------------!

      SUBROUTINE READWRITE

      EMA ABUF(4096), PBUF(4096)

      COMMON /PARAM/  RSCAN(7), CAXIS, POL, CSCAN, NAME,
     +                IDATE(3), ITIME(3), NPOL
      COMMON /USER/  IWRITE, IREAD

      CHARACTER CAXIS*1, POL*8, CSCAN*80, NAME*15

C

      ENTRY READ_DATA (IUNIT, IROW, IRDAT, IDATA, ABUF, PBUF, IBUF)

      IF (CAXIS .EQ. 'X' ) THEN       !DATA COLLECTED ALONG X AXIS
         NPTS=RSCAN(3)        !# X PTS
      ELSE                                 !DATA COLLECTED ALONG Y AXIS
         NPTS=RSCAN(6)        !# Y PTS
      END IF


C  Section for reading data from a file

      IF (IRDAT .NE. 2) THEN      !ONLY AMP OR PHASE STORED
         IF (IDATA .NE. IRDAT) WRITE(IWRITE,*) 'WARNING----',
     +                   'DATA REQUESTED WAS NOT RECORDED'
         IREC=1+IROW              !RECORD #
         IF (IDATA .EQ. 0) READ(UNIT=IUNIT,IOSTAT=IERR,ERR=99,REC=
```

```
     +                              IREC) (ABUF(M),M=1,NPTS),IBUF
         IF (IDATA .EQ. 1) READ(UNIT=IUNIT,IOSTAT=IERR,ERR=99,REC=
     +                              IREC) (PBUF(M),M=1,NPTS),IBUF
      ELSE                             !AMPLITUDE AND PHASE STORED
         IREC=2+2*(IROW-1)          !RECORD #
         IF (IDATA .NE. 1) READ(UNIT=IUNIT,IOSTAT=IERR,ERR=99,REC=IREC)
     +                                         (ABUF(M),M=1,NPTS),IBUF
         IF (IDATA .NE. 0) READ(UNIT=IUNIT,IOSTAT=IERR,ERR=99,REC=IREC+
     +                                        1) (PBUF(M),M=1,NPTS),IBUF
      END IF

      RETURN

C

      ENTRY WRITE_DATA (IUNIT, IROW, IRDAT, IDATA, ABUF, PBUF, IBUF,
     +                       AMIN, AMAX, PMIN, PMAX, MAXY, MAXX)


      IF (CAXIS .EQ. 'X' ) THEN       !DATA COLLECTED ALONG X AXIS
          NPTS=RSCAN(3)       !# X PTS
      ELSE                                  !DATA COLLECTED ALONG Y AXIS
          NPTS=RSCAN(6)       !# Y PTS
      END IF

C  Section to determine maximum and minimum amplitudes and phases

      IF (IROW .EQ. 1) THEN
          AMIN=100.
          AMAX=-100.
          PMIN=180.          !INITIALIZE THE MAX AND MINS
          PMAX=-180.
      END IF

      DO I=1,NPTS
          IF(ABUF(I) .GT. AMAX) THEN
              AMAX=ABUF(I)             !AMPLITUDE MAX
              IF (CAXIS .EQ. 'X' ) THEN
                  MAXY=IROW
                  MAXX=I                         !SECTION TO DETERMINE
              ELSE                                 !MAX AND MINS
                  MAXY=I
                  MAXX=IROW
              END IF
          END IF
          IF (ABUF(I) .LT. AMIN) AMIN=ABUF(I)    !AMP MIN
          IF (PBUF(I) .GT. PMAX) PMAX=PBUF(I)    !PHASE MAX
          IF (PBUF(I) .LT. PMIN) PMIN=PBUF(I)    !PHASE MIN
      END DO


C  Section for writing data to a file

      IF (IRDAT .NE. 2) THEN       !ONLY AMP OR PHASE STORED
          IREC=1+IROW              !RECORD #
```

```
          IF (IRDAT .EQ. 0) WRITE(UNIT=IUNIT,IOSTAT=IERR,ERR=98,REC=
     +                       IREC) (ABUF(M),M=1,NPTS),IBUF
          IF (IRDAT .EQ. 1) WRITE(UNIT=IUNIT,IOSTAT=IERR,ERR=98,REC=
     +                       IREC) (PBUF(M),M=1,NPTS),IBUF
      ELSE                          !AMPLITUDE AND PHASE STORED
          IREC=2+2*(IROW-1)       !RECORD #
          IF (IDATA .NE. 1) WRITE(UNIT=IUNIT,IOSTAT=IERR,ERR=98,REC=
     +                         IREC)  (ABUF(M),M=1,NPTS),IBUF
          IF (IDATA .NE. 0) WRITE(UNIT=IUNIT,IOSTAT=IERR,ERR=98,REC=
     +                       IREC+1) (PBUF(M),M=1,NPTS),IBUF
      END IF

      RETURN


C  Section for error messages

98    WRITE (IWRITE,*) 'ERROR ',IERR,' WRITING ROW ',IROW,' TO FILE ',
     +                                                       NAME
      RETURN

99    WRITE (IWRITE,*) 'ERROR ',IERR,' READING ROW ',IROW,' FROM FILE ',
     +                                                       NAME
      RETURN

      END
```

```
$CDS ON
      SUBROUTINE S10T01(S10X,S10Y,UX,UY,S01X,S01Y)

C     LAST REVISED:   6 OCT 86

C     Uses reciprocity to convert S10 to S01 in Cartesian coordinates
C     for a direction Ux, Uy.  Note that S01X and S01Y are at -K and
C     S10X and S10Y are at K.

      COMPLEX S10X,S10Y,S01X,S01Y,GAM,ET1,ET2,A(2,2)
      REAL KTSQ

      KTSQ = UX * UX + UY *UY
      IF (KTSQ .EQ. 0.) THEN
          A(1,1) = (1.,0.)
          A(1,2) = (0.,0.)
          A(2,1) = (0.,0.)
          A(2,2) = (1.,0.)
      ELSE
          GAM = CSQRT(CMPLX(1.0 - KTSQ,0.0))
          ET1 = 1.0 / GAM
          ET2 = GAM
          A(1,1) = (ET1 * UX*UX + ET2*UY*UY) / KTSQ
          A(1,2) = (ET1 - ET2) * UX *UY / KTSQ
          A(2,1) = A(1,2)
          A(2,2) = (ET1 * UY * UY + ET2 * UX * UX) / KTSQ
      END IF

      S01X = A(1,1) * S10X + A(1,2) * S10Y
      S01Y = A(2,1) * S10X + A(2,2) * S10Y

      RETURN
      END
```

```
$cds on

      SUBROUTINE SEPARATE(XINC,YINC,NPOL,NX,NY,DATA,DATA2,CAXIS)

      COMPLEX DATA(NX,NY),DATA2(NX,NY),SDATA(4096)
      CHARACTER CAXIS*1
      EMA DATA,DATA2,SDATA

      PI=ACOS(-1.)

      IF (CAXIS.EQ.'R') THEN
          DA = XINC / 2. / PI
          DO J=1,NY
              CALL FFT2 (1,NX,1,DA,DATA(1,J))
              IF (NPOL.EQ.2) CALL FFT2 (1,NX,1,DA,DATA2(1,J))
          END DO
      ELSE
          DA = YINC / 2./ PI
          DO I=1,NX
              DO J=1,NY
                  SDATA(J)=DATA(I,J)
              END DO
              CALL FFT2 (1, 1, NY, DA, SDATA)
C             CALL FFT2 (1,NY,1, DA, SDATA)
              DO J=1,NY
                  DATA(I,J) = SDATA(J)
              END DO
          END DO
          IF (NPOL. EQ. 2) THEN
              DO I=1,NX
                  DO J=1,NY
                      SDATA(J) = DATA2(I,J)
                  END DO
                  CALL FFT2 (1, 1, NY, DA, SDATA)
                  DO J=1,NY
                      DATA2(I,J) = SDATA(J)
                  END DO
              END DO
          END IF
      END IF

      RETURN
      END
```

```
$cds on

      SUBROUTINE SEPTRANS(XINC,YINC,NPOL,NX,NY,DATA,DATA2,CAXIS)

      COMPLEX DATA(NX,NY),DATA2(NX,NY),SDATA(4096)
      EMA DATA,DATA2,SDATA
      CHARACTER  CAXIS*1
      COMMON /WVGE/A,B,AKO
      COMMON  /TRANS/TX,TY,TZ,FILTER,SXINC,SYINC

      PI=ACOS(-1.)

      IF (CAXIS.EQ.'R') THEN
          DO J=1,NY
              CALL TRANSLATE (DATA(1,J), NX, 1, TX, TY, TZ,FILTER)
              DA=SXINC*AKO
              CALL FFT2 (-1, NX, 1, DA, DATA(1,J))
              IF (NPOL.EQ.2) THEN
                  CALL TRANSLATE (DATA2(1,J),NX,1,TX,TY,TZ,FILTER)
                  CALL FFT2 (-1, NX, 1, DA, DATA2(1,J))
              END IF
          END DO
      ELSE
          DO I=1,NX
              DO J=1,NY
                  SDATA(J) = DATA(I,J)
              END DO
              CALL TRANSLATE (SDATA, 1, NY, TX, TY, TZ, FILTER)
              DA=SYINC*AKO
              CALL FFT2 (-1, 1, NY, DA, SDATA)
              DO J=1,NY
                  DATA(I,J) = SDATA(J)
              END DO
          END DO
          IF (NPOL. EQ. 2) THEN
              DO I=1,NX
                  DO J=1,NY
                      SDATA(J) = DATA2(I,J)
                  END DO
                  CALL TRANSLATE (SDATA, 1, NY, TX, TY, TZ,FILTER)
                  CALL FFT2 (-1, 1, NY, DA, SDATA)
                  DO J=1,NY
                      DATA2(I,J)= SDATA(J)
                  END DO
               END DO
          END IF
      END IF

      RETURN
      END
```

```
$CDS ON
      FUNCTION SINX(X)

C     LAST REVISED:   6 OCT 86

      IF (ABS(X).GE.1.E-06) THEN
          SINX=SIN(X)/X
      ELSE
          SINX=1.-X*X/6
      END IF

      RETURN
      END
```

```
$CDS ON
!--------------------------------------------------------------------!
!                                                                    !
!     SUBROUTINE SWIPE              Last Revised:   5/19/88          !
!                                                                    !
!     This subroutine clears the terminal display.                   !
!                                                                    !
!     Subroutines called:                                            !
!         None                                                       !
!                                                                    !
!--------------------------------------------------------------------!

      SUBROUTINE SWIPE


      CHARACTER*4 A,G,U


      A=CHAR(27)//'H'//CHAR(27)//'J'      !Clear Alpha display
      G=CHAR(27)//'*da'                   !Clear Graphics display
      U=CHAR(27)//'&j@'                   !Clear User Keys display

      WRITE(1,5) A,G,U

5     FORMAT (3A4)

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE TESTP2(N,ISP2)

C     LAST REVISED:   6 OCT 86

C       TESTS N FOR POWER OF TWO.  IF N IS A POWER OF TWO,
C       ISP2=0; IF NOT, ISP2=1.
C
      XTRY=ALOG(FLOAT(N))/0.69314718
      XDEL=XTRY-INT(XTRY+.001)
      ISP2=0
      IF(ABS(XDEL) .GT. 1.E-5) ISP2=1
      RETURN
      END
```

```
$cds on

      SUBROUTINE TRANSLATE (DATA, NX, NY, X, Y, Z, FILTER)

C     LAST REVISED:   6 OCT 86

C     Performs a translation of the data set in physical space using
C     the vector  R = (X,Y,Z).   The  data set domain is assumed to
C     be K-space and the multiplier is  exp(-j K . R) .
C
C     An ideal low-pass filter can also be applied.  The FILTER
C     parameter is a radius (in normalized wave-number units).
C     Data points beyond this distance from the wave-number origin
C     are zeroed.  A value of FILTER=0. implies no filtering.

      COMPLEX DATA(NX,NY), CFACT, CJ
      CHARACTER CAXIS*1,POL*8,CSCAN*80,NAME*15
      EMA DATA

      COMMON /PARAM/ RSCAN(7),CAXIS,POL,CSCAN,NAME,IDATE(3),ITIME(3)
      COMMON /WVGE/ A,B,AKO

      CJ = (0.,1.)
      XKINC = RSCAN(2)*AKO           ! X axis spacing
      YKINC = RSCAN(5)*AKO           ! Y axis spacing
      XKO = RSCAN(1)*AKO             ! Initial X-axis point
      YKO = RSCAN(4)*AKO             ! Initial Y-axis point

      IF (FILTER.EQ.0.)  FILTER = 1. ! Same as no filter
      R2 = (FILTER*AKO)**2           ! Filter radius

      DO J=1,NY
          YK = YKO + (J-1)*YKINC
          YK2 = YK**2
          DO I=1,NX
              XK = XKO + (I-1)*XKINC
              XK2 = XK**2
              IF ((XK2+YK2) .GT. R2) THEN
                  DATA(I,J) = (0.,0.)
              ELSE
                  ZK2 = AKO**2 - XK2 - YK2
                  CFACT = CEXP( -CJ * (X*XK+Y*YK) )
                  IF (ZK2 .GT. 0.) THEN
                      ZK = -SQRT(ZK2)
                      CFACT = CFACT * CEXP(-CJ * Z*ZK)
                  ELSE IF (ZK2 .LT. 0.) THEN
                      ZK = -SQRT(-ZK2)
                      CFACT = CFACT * EXP(Z*ZK)
                  END IF

                  DATA(I,J) = DATA(I,J) * CFACT
              END IF
          END DO
      END DO
```

```
      RETURN
      END
```

```
$CDS ON
      SUBROUTINE XYTHUY(UX,UY,SX,SY,SK,CK,  SA,SB)

C     LAST REVISED:   6 OCT 86

C     CONVERTS X,Y COMPONENTS OF TRANSFORMED SPECTRUM TO HUYGENS
C     COMPONENTS IN ORTHOGONAL DIRECTIONS A AND B.

      COMPLEX SX,SY,SZ,SA,SB

      ST = SQRT(UX**2 + UY**2)
      CT = SQRT(1.-ST**2)
      SZ = -(UX*SX + UY*SY)/CT

      IF (ST .LT. .0001) THEN
          HBX = CK
          HBY = SK
          HBZ = 0.
          HAX = -SK
          HAY = CK
          HAZ = 0.
      ELSE
          CP = UX/ST
          SP = UY/ST
          CPB = CK*CP + SK*SP
          SPB = -SK*CP + CK*SP
          CPA = -SPB
          SPA = CPB

C         This is Huygens unit polarization pattern for X electric field.

          HX = SPB**2 + CPB**2*CT
          HY = SPB*CPB*(CT-1.)
          HZ = -CPB*ST
          HBX = CK*HX - SK*HY
          HBY = SK*HX + CK*HY
          HBZ = HZ
          HX = SPA**2 + CPA**2*CT
          HY = SPA*CPA*(CT-1.)
          HZ = -CPA*ST
          HAX = -SK*HX - CK*HX
          HAY = CK*HX - SK*HY
          HAZ = HZ
      ENDIF

      SA = SX*HAX + SY*HAY +SZ*HAZ
      SB = SX*HBX + SY*HBY +SZ*HBZ

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE XYTYCON (UX,UY,SX,SY,SPOL,CPOL,SEL,SAZ)

C     LAST REVISED:   13 MAY 88

C     Converts X,Y components of transformed spectrum (Sx, Sy) to azimuth,
C     elevation components (conical about Y-axis) including a possible
C     rotation about the Z-axis by angle POLOUT, where
C                      CPOL = COS(POLOUT)
C                      SPOL = SIN(POLOUT)
C
C     Components are computed for a direction Ux,Uy.

      COMPLEX SX,SY,SAZ,SEL,SZ,GAM,CB,SA,CA,CSQRT

      GAM = CSQRT(CMPLX(1.-UX*UX-UY*UY,0.0))
      SZ = -(UX * SX + UY * SY) / GAM
      SB = UY                               ! SIN EL
      CB = CSQRT(CMPLX(1. - SB*SB,0.0))     ! COS EL
      SA = UX / CB                          ! SIN AZ
      CA = GAM/CB                           ! COS AZ
      SEL = ((CPOL * (-SB * SA) + SPOL * CB) * SX +
     +       (SPOL * SB * SA + CPOL * CB) * SY +
     +       (-SB * CA) * SZ) * GAM
      SAZ = ((CA * CPOL * SX) - (CA * SPOL * SY) - (SA * SZ)) * GAM

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE XYTZCON (UX,UY,SX,SY,SPOL,CPOL,S1OTH,S1OPH)

C     LAST REVISED:    13 MAY 88

C     Converts X,Y components of transformed spectrum (Sx, Sy) to spherical
C     components (theta, phi - conical about Z-axis) including a possible
C     rotation about the Z-axis by angle POLOUT, where
C
C                      CPOL = COS(POLOUT)
C                      SPOL = SIN(POLOUT)
C

      COMPLEX SX,SY,S1OTH,S1OPH,SZ,GAM,CTH,STH,CPH,SPH

      GAM = CSQRT(CMPLX(1.-UX*UX-UY*UY,0.0))
      SZ = -(UX * SX + UY * SY) / GAM
      CTH = GAM                               ! COS THETA
      STH = CSQRT(1. - GAM*GAM)               ! SIN THETA
      SPH = UY / STH                          ! SIN PHI
      CPH = UX / STH                          ! COS PHI
      S1OTH = CTH*(CPH*CPOL-SPH*SPOL)*SX + CTH*(SPH*CPOL+CPH*SPOL)*SY
     +            -STH*SZ
      S1OPH = (CPH*CPOL-SPH*SPOL)*SY - (SPH*CPOL+CPH*SPOL)*SX

      RETURN
      END
```

```
$CDS ON
      SUBROUTINE XYZOPEN(FNAME,IUNIT,ISTATUS)


C     LAST REVISED: 4/2/87

      CHARACTER CAXIS*1,POL*8,CSCAN*80,NAME*15,INFILE*25,STAT*7,FNAME*15
      COMMON /RECBUFF/LBUF(8200)
      COMMON /PARAM/RSCAN(7),CAXIS,POL,CSCAN,NAME,IDATE(3),ITIME(3)
      COMMON /USER/IWRITE,IREAD

C     XYZOPEN opens a datafile.
C     LGBUF is a library subroutine to enlarge I/O buffer size. NOTE:
C     the buffer array LBUF must not be in EMA Under any circumstances.
C     NOTE: if CDS is used, then either the call to LGBUF must be made in
C     the main program(in this case common block RECBUFF is not required),
C     or common block RECBUFF must be declared in the main program and
C     this subroutine. If CDS is not used then the call can be made from
C     this subroutine without using common block RECBUFF.

      NAME=FNAME
      GOTO 77

5     WRITE(IWRITE,*) 'Enter data file name:'
      READ (IREAD,20) NAME
20    FORMAT(A)
77    INFILE=NAME//'::XYZFILES'

      IF (ISTATUS .EQ. 0) STAT='OLD    '
      IF (ISTATUS .EQ. 1) STAT='NEW    '
      IF (ISTATUS .EQ. 2) STAT='UNKNOWN'

      IF (STAT .EQ. 'NEW') THEN
          NPTS=RSCAN(6)
          IF (CAXIS .EQ. 'X') NPTS=RSCAN(3)
          IRECLB=(NPTS*4)+2  !RECORD LENGTH(BYTES)--AMP OR PHASE AND STATUS
          IF (IRECLB .LT. 180) IRECLB=180  !INSURE ENOUGH ROOM FOR HEADER REC.
          CALL DATETIME(IDATE,ITIME)
      ELSE
          INQUIRE(FILE=INFILE,IOSTAT=IERR,ERR=65,RECL=IRECLB) !READ RECORD LTH
      END IF


      OPEN(UNIT=IUNIT,FILE=INFILE,ACCESS='DIRECT',FORM='UNFORMATTED',
     +    RECL=IRECLB,IOSTAT=IERR,STATUS=STAT)


10    FORMAT('ERROR ON OPENING FILE ',A15)

65    IF (IERR .GT. 0) THEN
          WRITE(IWRITE,10) NAME
          GOTO 5
      ELSE
          CALL LGBUF(LBUF,IRECLB/2)      !ENLARGE I/O BUFFER TO #BYTES/2
      END IF
```

```
      RETURN

      END
```