UNITED STATES
DEPARTMENT OF COMMERCE
TECHNOLOGY ADMINISTRATION
NATIONAL INSTITUTE OF STANDARDS
AND TECHNOLOGY

NIST Technical Note 1354

The NIST 60-Millimeter Diameter Cylindrical Cavity Resonator: Performance Evaluation for Permittivity Measurements

Eric J. Vanzura
Richard G. Geyer
Michael D. Janezic

The NIST 60-Millimeter Diameter Cylindrical Cavity Resonator: Performance Evaluation for Permittivity Measurements

Eric J. Vanzura
Richard G. Geyer
Michael D. Janezic

Electromagnetic Fields Division
Electronics and Electrical Engineering Laboratory
National Institute of Standards and Technology
325 Broadway
Boulder, Colorado 80303-3328

August 1993

U.S. DEPARTMENT OF COMMERCE, Ronald H. Brown, Secretary
TECHNOLOGY ADMINISTRATION, Mary L. Good, Under Secretary for Technology
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY, Arati Prabhakar, Director

National Institute of Standards and Technology Technical Note
Natl. Inst. Stand. Technol., Tech. Note 1354, 236 pages (August 1993)
CODEN:NTNOEF

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1993

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington, DC 20402-9325

Contents

List of Figures

List of Tables

The NIST 60-millimeter Diameter Cylindrical Cavity Resonator: Performance Evaluation for Permittivity Measurements

Eric J. Vanzura
Richard G. Geyer
Michael D. Janezic

Antenna and Materials Metrology Group
Electromagnetic Fields Division

Abstract

Uncertainty estimates are developed for dielectric permittivity calculations made using the NIST 60-mm diameter cylindrical resonator. A mode-filtering helical waveguide makes up the cavity's cylindrical wall, which permits the generation of high-purity TE_{01p} resonant modes for high accuracy permittivity measurements. The cavity's length can be varied from 408 to 433 mm. Fixed-length and fixed-frequency measurements in the X-band frequency range are evaluated with particular emphasis on 10 GHz. Resonator theory and design, measurement tolerances, and software are included.

Keywords: cylindrical cavity, dielectric permittivity, measurement software, resonator theory and design, TE_{01} helical waveguide, uncertainty estimates, x-band 10 GHz microwave.

Chapter 1

Introduction

1.1 Definition of Permittivity

Permittivity is a physical quantity that describes the relation between electric field $\overline{E}$ in volts per meter and electric displacement $\overline{D}$ in coulombs per square meter. In a vacuum, this relation is given by

$$\overline{D} = \epsilon_0 \overline{E} \ , \tag{1.1}$$

where $\epsilon_0 = 8.854\,187\,82 \times 10^{-12}$ farads per meter is the permittivity of free space. In matter, electric displacement is affected by the material's electric susceptibility, χ_e^*, which is a measure of the material's polarizability. In an isotropic, linear media the polarization vector $\overline{P}$ is parallel and proportional to the electric field,

$$\overline{P} = \epsilon_0 \chi_e^* \overline{E} \ , \tag{1.2}$$

and the electric displacement vector then becomes

$$\begin{aligned} \overline{D} &= \epsilon_0 \overline{E} + \overline{P} \\ &= \epsilon_0 (1 + \chi_e^*) \overline{E} \\ &= \epsilon^* \overline{E} \ , \end{aligned} \tag{1.3}$$

where

$$\begin{aligned} \epsilon^* &= \epsilon_0 (1 + \chi_e^*) \ , \\ &= \epsilon_0 \epsilon_R^* \end{aligned} \tag{1.4}$$

is the permittivity of the material, and $\epsilon_R^* = (1 + \chi_e^*)$ is the material's relative permittivity or dielectric constant. In general, electric susceptibility has real (i.e., dispersive) and imaginary (i.e., absorptive) parts which makes relative permittivity a complex quantity. We define relative permittivity as

$$\begin{aligned} \epsilon_R^* &= 1 + \chi_e^* \\ &= (1 + \Re\{\chi_e^*\}) - j\Im\{\chi_e^*\} \\ &= \epsilon_R' - j\epsilon_R'' . \end{aligned} \tag{1.5}$$

For low-loss materials ϵ_R' is often loosely called the relative permittivity, with the distinction between ϵ_R^* and ϵ_R' implicitly understood. The term ϵ_R'' is called the dielectric loss factor.

If the induced conduction current is directly proportional to the electric field, the curl of the magnetic field $\overline{H}$ is

$$\begin{aligned} \nabla \times \overline{H} &= (j\omega\epsilon^* + \sigma)\overline{E} , \\ &= j\omega\epsilon_0\epsilon_R' \left(1 - j\frac{\epsilon_R''}{\epsilon_R'} - j\frac{\sigma}{\omega\epsilon_0\epsilon_R'}\right)\overline{E} , \\ &= j\omega\epsilon_0\epsilon_R'(1 - j\tan\delta)\overline{E} . \end{aligned} \tag{1.6}$$

This relation for loss tangent $\tan\delta$ now explicitly separates dielectric and conductive losses. In most discussions dielectric loss of nonconductive materials at microwave and millimeter wave frequencies is described by a frequency dependent conductivity of the material at the measurement frequency. In this case loss tangent $\epsilon_R'' = \sigma/\omega\epsilon_0$, and loss tangent becomes

$$\tan\delta = \frac{\epsilon_R''}{\epsilon_R'} = \frac{\sigma}{\omega\epsilon_0\epsilon_R'} . \tag{1.7}$$

1.2 Operation of Mode-Filtered Cylindrical Resonators

The cylindrical cavity resonator described in this Technical Note was built by the NIST Electromagnetic Properties of Materials program to accurately measure the complex permittivity of dielectric materials. The application of cavity resonator methods to measure the complex permittivity of dielectric samples is not new [1]. However, resonator methods are inherently accurate and to this day prove to be the most accurate measurement method available. NIST chose to build a cavity resonator to accurately characterize dielectric materials that in turn can be used as check standards to test the validity of other measurement methods. Special test services are performed regularly for those outside NIST, and, since its introduction, the NIST resonator has characterized a large number of materials both for internal and external use.

Cavity resonators can measure permittivity only at fixed-frequencies and over a narrow range of frequency. This is the case especially for the shorter, solid-walled cavities because higher-order modes occur very easily. Measurement of permittivity over a broader range of frequencies is partially solved by the use of helical waveguide cavities [2, 3]. The mode filtering capability of helical waveguides allows construction of cavity resonators that can support several highly pure modes in a limited frequency range.

The NIST resonator has a 60-mm diameter helical wire-wound cylindrical wall, with a variable length of approximately 408 to 433 mm. The cavity's helical winding allows only circumferential currents to flow, which permits propagation of only the TE_{0n} family of circular transmission line modes, where n describes the number of radial variations. Mode-filtering capability permits the use of a long cavity that would otherwise be highly overmoded. This allows permittivity measurements at several TE_{01} resonant frequencies throughout X-band (8.2 to 12.4 GHz) rather than at only one or two specific frequencies.

The well defined geometry of the NIST cylindrical resonator allows standing waves to occur at certain frequencies. Power enters through a coupling iris located in one of the cavity's endplates and excites the TE_{01} waveguide mode. The present coupling endplate connects to X-band waveguide feeds. The cylindrical TE_{01} waveguide mode has an easily calculable wavelength that depends on waveguide diameter and the permittivity of the medium inside the waveguide (7.1). Power propagates in this mode from the coupling endplate to the other endplate and reflects back. At certain frequencies the cavity length will be an integer number p of half-wavelengths, resulting in constructive interference. The integer p is called the axial mode number. At these frequencies a standing wave is set up, and a resonance condition occurs. For the NIST cavity, the first TE_{01} resonant mode occurs near 6.1 GHz, and the first TE_{02} resonant mode occurs near 11.2 GHz.

The NIST cavity resonator is ideally suited for accurate permittivity measurements because the resonator's quality factor Q is very high (60 000 to 80 000). Q is sometimes defined as the number of radians a wave propagates before the wave's power decays to e^{-1}, or approximately 37%, of its initial power. For a high-Q resonator this means that the excitation wave reflects off the cavity endplates and travels the length of the cavity several times before the wave's power is significantly reduced. For example, if $p = 25$ and the Q of the $TE_{01(25)}$ mode is 80 000, then an excitation wave travels the length of the cavity approximately $Q/(25\pi) \approx 1000$ times before decaying to e^{-1} of its initial power. If this high-Q cavity contains a sample, the excitation wave also passes through the material more than 1000 times. In effect this increases the effective thickness of the sample and makes cavity resonator measurements sensitive to small changes in ϵ'_R and tanδ.

1.3 Methods of Measurement

To determine the permittivity of a material, two measurement methods can be used, as shown graphically in Fig. 1.1. One method involves reducing the sample-loaded cavity's length so that it resonates at the same frequency as the empty cavity. The change in length is a measure of permittivity, and the change in Q is a measure of the sample's dielectric loss (Sec. 2.2.3). The second method holds the cavity length fixed. The change in resonant frequency determines ϵ'_R, and the change in Q determines dielectric loss.

The theory from which ϵ'_R is calculated assumes that the cavity and sample have zero loss (infinite Q), so field orthogonality can be assumed. In practice, this assumption causes little problem because of the cavity's low surface impedance and because dielectric samples are usually low loss. The cavity resonator is so sensitive, however, that it is possible to determine and make corrections to frequency measurements that improve ϵ'_R measurement accuracy. These corrections are discussed in Chapter 4. In a sense, the concept of having two methods to measure the real part of permittivity is artificial. All that is needed to calculate ϵ'_R are cavity length and diameter, sample thickness and axial mode number. The empty cavity needs to be measured only because we have to determine the cavity's dimensions, where cavity length and diameter are calculated from the frequencies and axial mode numbers p of the TE_{01p} resonance spectrum.

The fixed-frequency method is more accurate than the fixed-length method for calculating dielectric loss. Cavity Q is frequency dependent. Skin depth (resistive losses) of the cavity walls decrease and coupling port losses increase as frequency increases. Presently, the measured empty-cavity Q is used as the reference that determines the cavity's conductor and port losses. When we measure the Q of the empty cavity, we determine the cavity loss at that resonant frequency. When a sample is inserted the resonant frequency is lowered. If we use the Q of the empty cavity resonating at a different frequency as our reference, our loss tangent calculation will be in error. This is especially true when the sample is very low loss (tan$\delta < 0.0003$), because small differences in cavity Q become important. When sample loss is greater than tan$\delta > 0.0008$ the differences between empty-cavity losses and the cavity losses of the sample-loaded cavity are small enough that the fixed-length method can be used with nearly the same accuracy as the fixed-frequency method.

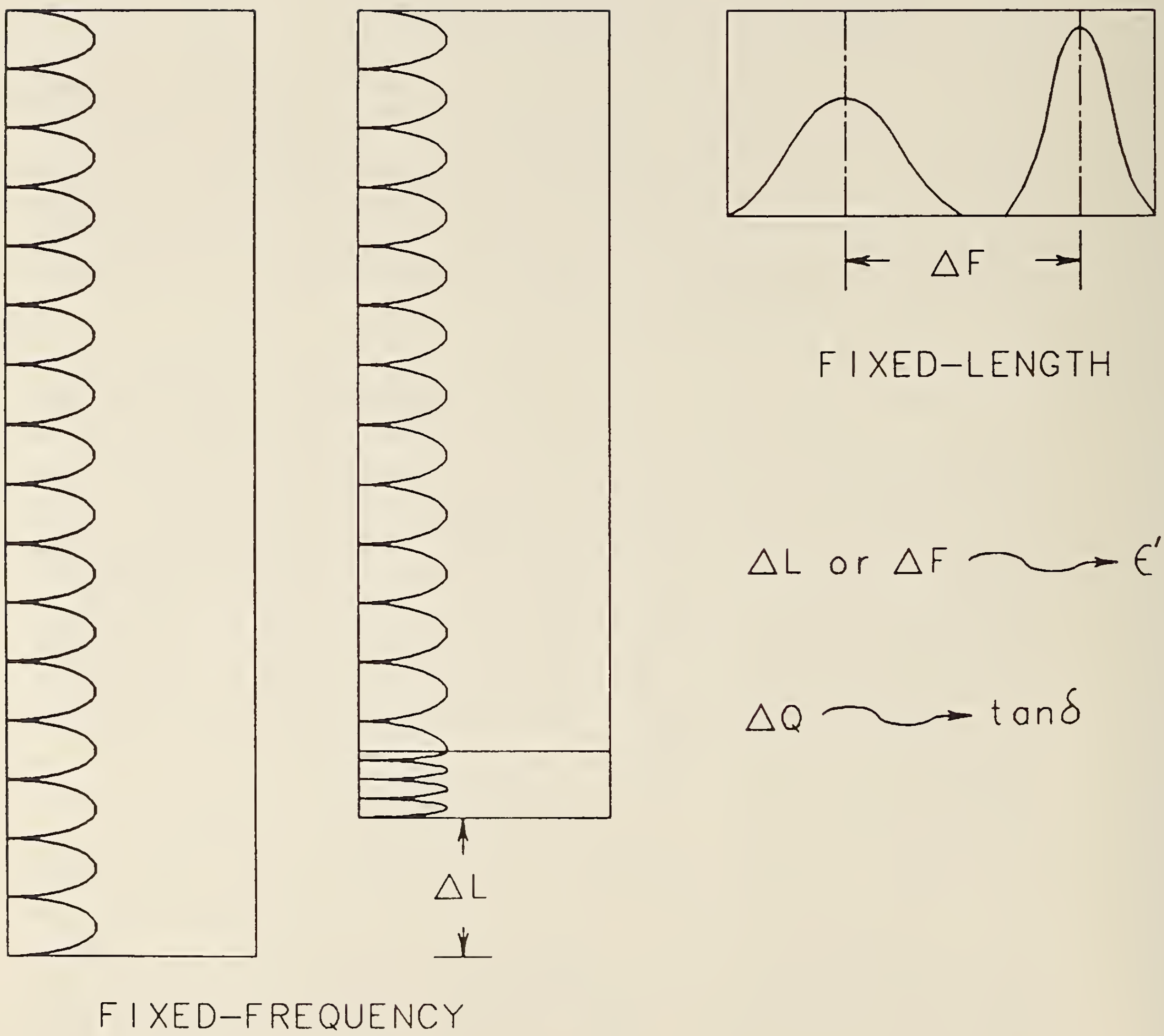

Figure 1.1: The fixed-frequency/variable-length and fixed-length/variable-frequency permittivity measurement methods rely on the empty-cavity dimensions and Q as a reference.

Chapter 2

Theory of Microwave Resonators

2.1 Introduction

Resonators create, filter, and select frequencies in oscillators, amplifiers, and tuners. Recent advances in the miniaturization of microwave circuits have spawned the development of low-loss, temperature-stable dielectric resonators. These resonators replace waveguide filters in many communication systems where microstrip and stripline resonators cannot be used due to their intrinsic high loss. Dielectric resonators are also very important fixtures for measuring the electrical properties of low-loss solids in the microwave region. An important resonator circuit at microwave frequencies is the closed cylindrical cavity operated in a resonant TE_{01p} mode. The magnetic field for this mode is sketched in Fig. 2.1. For a distant observer this mode appears as a magnetic dipole, and for this reason some authors call it a "magnetic dipole mode." The electric field lines are simply circles concentric with the axis of the cylinder (Fig. 2.2).

For a TE mode in a waveguide, the electric field is everywhere *transverse* to the propagation direction or to the axial z-direction of the guide. Coordinate systems for circular and rectangular waveguides are sketched in Figs. 2.3 and 2.4, respectively. For a TE_{mn} mode in a rectangular guide, m simply represents the number of antinodes occurring in the electric field E in the x-direction and n represents the number of antinodes in E in the y-direction. For a TE_{mnp} mode in a finite circular guide, m represents the number of antinodes occurring in E in the ϕ-direction, n the number of antinodes in the radial r-direction, and p the number in the z-direction. The same notation is used for transverse magnetic or TM_{mn} modes, with the m and n indices representing the antinodes occurring

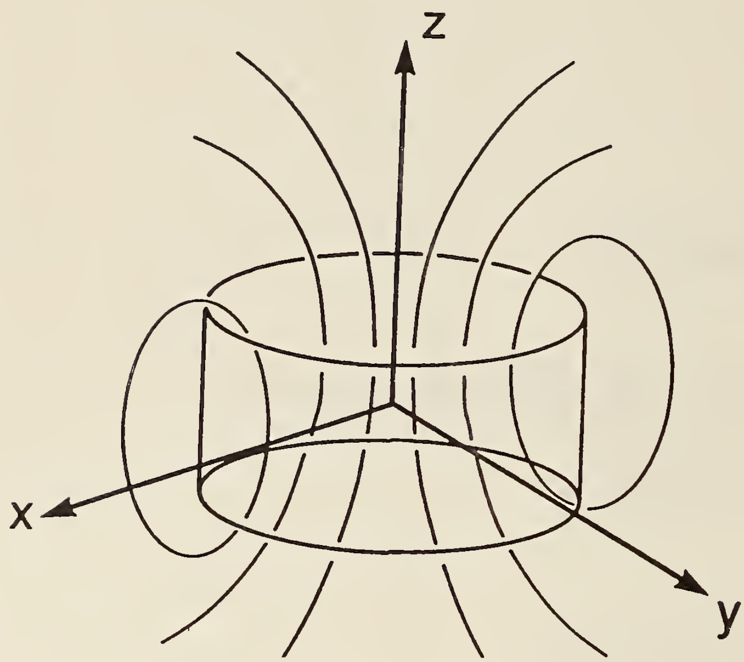

Figure 2.1: Magnetic field lines of the resonant mode TE_{01p} in an isolated dielectric resonator.

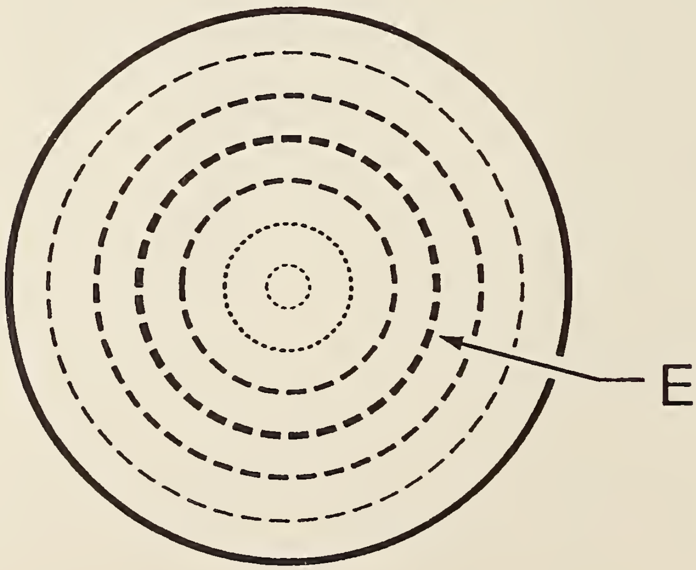

Figure 2.2: Electric field distribution in equatorial plane for TE_{01p} mode. Breadth of dashed lines is proportional to the electric field.

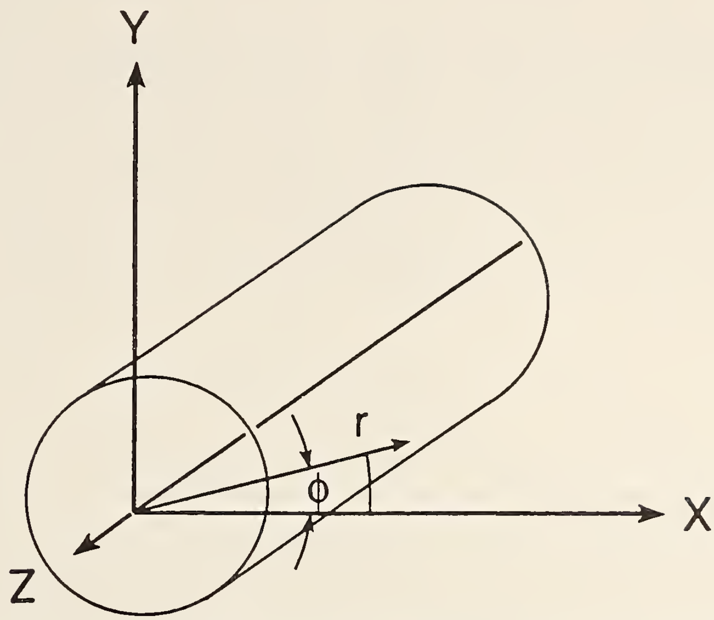

Figure 2.3: Cylindrical coordinate system for waveguides.

in the magnetic field H for a given coordinate direction. Generally, the choice of cavity to be used in dielectric metrology is influenced by the shape and size of the sample to be measured. Insertion and removal of the sample from the cavity should be convenient and not significantly alter the Q [4] of the air-filled cavity or increase the radiation from the cavity. Hollow circular cylindrical waveguide resonators, which are terminated by two short-circuit endplates, are the most commonly used for dielectric measurements of low-loss solids. The reasons for this are the relative ease of fabrication and the very high Q-factors and accompanying narrow bandwidths that can be obtained with this fixture in the microwave region [1]. If the cavity is made from helically wound waveguide, it also acts as a mode filter in that all other waveguide modes except the TE_{01p} mode [3] are greatly attenuated.

In general, microwave energy is coupled to the cavity through transmission line probes, as illustrated in Fig. 2.5. As in resonant transmission lines where resonance occurs at many frequencies, the hollow cylindrical waveguide has many resonance frequencies and associated field distributions or modes. The field of the mode with the lowest or dominant resonant frequency is termed the dominant mode of that fixture. Only enough energy is provided by the input excitation to match cavity losses. Once the internal fields at a resonant frequency are determined, an equivalent lumped parameter circuit of the fixture may be ascertained. In addition, the internal power dissipation, stored energies, and energy flow out of the cavity can be determined.

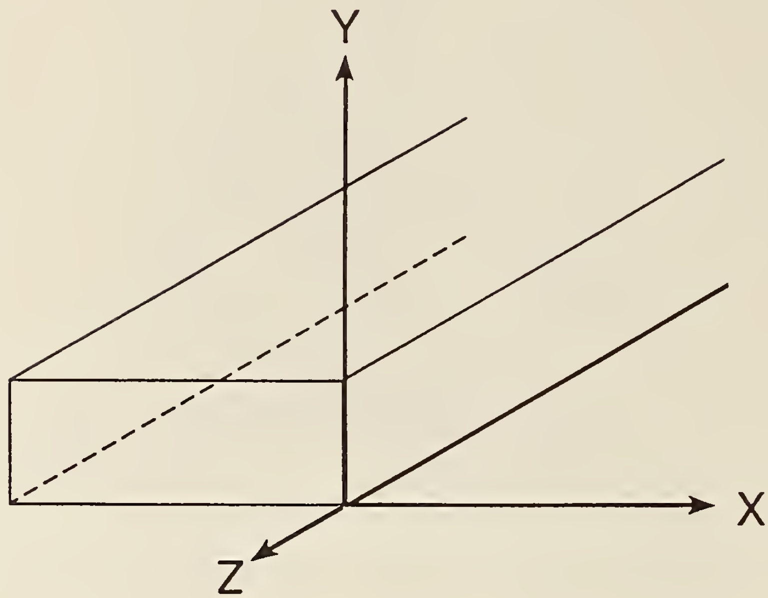

Figure 2.4: Rectangular coordinate system for waveguides.

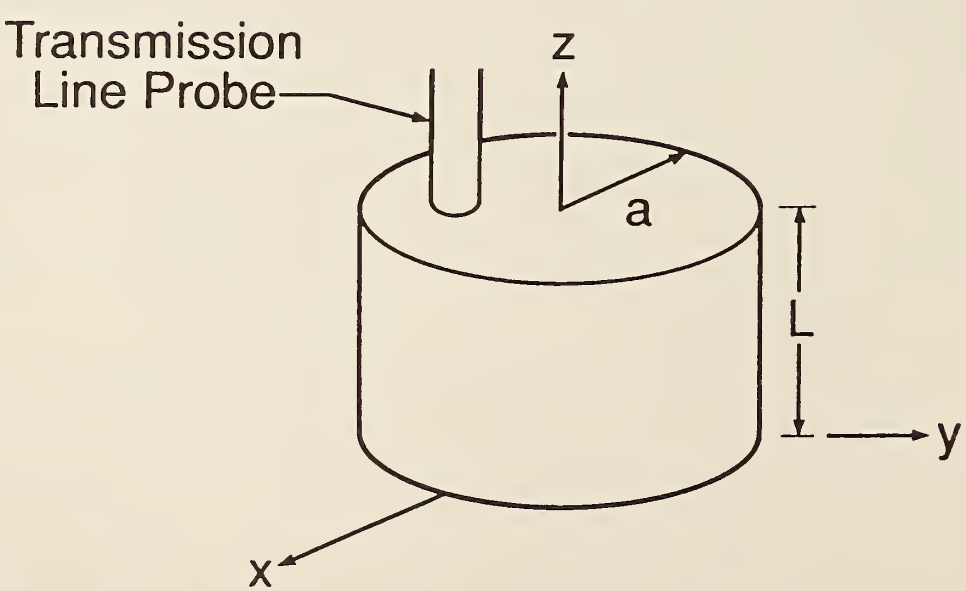

Figure 2.5: Circular cylindrical hollow cavity of length L and radius a.

As expected, the different microwave modes that can be set up in a hollow right circular cylinder depend on both the dimension of the cylinder and the microwave frequency. A hollow right circular cylindrical cavity is supportive of both TE- and TM-mode field structures. For steady-state $\exp(j\omega t)$ field time dependence, the equations for the axial components of the electric and magnetic field interior to a waveguide of arbitrary cross-section and filled with material of complex permittivity ϵ^* and complex permeability μ^* satisfy the homogeneous Helmholtz equation

$$\nabla^2 \begin{pmatrix} E_z \\ H_z \end{pmatrix} + k^2 \begin{pmatrix} E_z \\ H_z \end{pmatrix} = 0 , \tag{2.1}$$

where z is the direction of propagation and E_z, H_z are taken for TM or TE modes, respectively. The differential operator ∇^2 is the Laplacian, which can be expressed in any curvilinear coordinate system by evaluation of the metric. All other field components can be expressed in terms of E_z and H_z. In general, any field in a hollow waveguide, however complicated, may be represented by a combination of TE and TM modes. The wavenumbers

$$k^2 = k_c^2 - \gamma^2 = \omega^2 \epsilon^* \mu^* , \tag{2.2}$$

are the characteristic eigenvalues of (2.1). In other words, to each value of k^2 there will correspond a function E_z or H_z (for TM or TE modes) which is a *characteristic* function from which may be derived the other components of the fields. In general ϵ^* and μ^* in (2.2) can be complex dyadics, but for this discussion the material is assumed isotropic and each dyadic becomes a single complex value multiplied by the unit dyadic. As defined in (1.7), the conductivity of the medium is contained in the imaginary part of the dielectric constant, $\epsilon^* = \epsilon' - j\epsilon''$. For a plane-wave field $H_z(z) = H_z e^{-\gamma z}$ in which the electric field is entirely transverse (TE waves),

$$\gamma^2 = \left(\frac{2\pi}{\lambda}\right)^2 = k_c^2 - \omega^2 \epsilon^* \mu^* , \tag{2.3}$$

where the value of λ is the wavelength of a plane TE wave in the medium that fills the hollow pipe and k_c is the cutoff wavenumber of the TE mode. In order to have propagation down the pipe, γ^2 must be negative. The quantity k_c^2 is always real because, for a circular waveguide of radius a,

$$(k_c)_{mn} = \frac{t'_{mn}}{a} , \tag{2.4}$$

where t'_{mn} is the nth root of the first derivative of the Bessel function $J'_m(k_c a) = 0$ for a TE_{mn} mode. When γ^2 is positive or $\Re\left(\omega^2 \epsilon^* \mu^*\right) \leq k_c^2$ there is no propagation of energy through the waveguide. This condition is termed cutoff. The cutoff or critical frequency is given by

$$\omega_c^2 = \Re\left\{\frac{1}{\epsilon^* \mu^*}\right\} k_c^2 . \tag{2.5}$$

Substituting (2.4) into (2.5) gives the cutoff frequency for TE_{mn} modes in a circular waveguide:

$$(f_c)^2_{TE_{mn}} = \left(\frac{1}{2\pi}\right)^2 \Re\left\{\frac{1}{\epsilon^* \mu^*}\right\} \left(\frac{t'_{mn}}{a}\right)^2 , \tag{2.6}$$

or

$$(\lambda_c)_{TE_{mn}} = \frac{c}{(f_c)_{TE_{mn}}} = \frac{c \cdot 2\pi a \cdot \sqrt{\Re\left(\epsilon^* \mu^*\right)}}{t'_{mn}} , \tag{2.7}$$

where c is the velocity of light in the medium.

Similarly, for a TM_{mn} mode in circular waveguide,

$$(k_c)_{mn} = \frac{t_{mn}}{a} , \tag{2.8}$$

where t_{mn} is the nth root of $J_m(k_c a) = 0$.

As before, the cutoff wavelength for TM_{mn} modes is given by:

$$(\lambda_c)_{TM_{mn}} = \frac{c \cdot 2\pi a \cdot \sqrt{\Re\left(\epsilon^* \mu^*\right)}}{t_{mn}} . \tag{2.9}$$

If λ_g is the wavelength in the guide and k_g the guide wavenumber, then

$$\gamma = jk_g = j\frac{2\pi}{\lambda_g} , \tag{2.10}$$

or, from (2.3),

$$\left(\frac{2\pi}{\lambda_g}\right)^2 = \left(\frac{2\pi}{\lambda}\right)^2 - \left(\frac{2\pi}{\lambda_c}\right)^2 , \tag{2.11}$$

from which we derive the axial wavelength in the waveguide,

$$\lambda_g = \frac{\lambda}{\sqrt{1 - \left(\frac{\lambda}{\lambda_c}\right)^2}} , \tag{2.12}$$

where λ is the wavelength of a uniform plane wave in the medium:

$$\begin{aligned}\lambda &= \frac{\lambda_0}{\sqrt{\frac{\Re\{\epsilon^*\mu^*\}}{\epsilon_0\mu_0}}} \\ &\approx \frac{\lambda_0}{\sqrt{\frac{\epsilon'\mu'}{\epsilon_0\mu_0}}} \ .\end{aligned} \tag{2.13}$$

where λ_0 is the free-space wavelength and we assume that $\Re\{\epsilon^*\mu^*\} \approx \epsilon'\mu'$. Equation (2.12) then becomes

$$\lambda_g = \frac{\lambda_0}{\sqrt{\frac{\epsilon'\mu'}{\epsilon_0\mu_0} - \left(\frac{\lambda_0}{\lambda_c}\right)^2}} = \frac{\lambda_0}{\sqrt{\epsilon'_R\mu'_R - \left(\frac{\lambda_0}{\lambda_c}\right)^2}} \ . \tag{2.14}$$

Equation (2.14) shows that when the free-space wavelength is much less than the cutoff wavelength, that waves propagate in the guide with a guide wavelength normalized by the square root of the product of relative permittivity and relative permeability of the material filling the guide. Of course, in the empty cavity situation, this means that the guide wavelength is essentially that of air. Similarly, when wavelengths are much greater than the cutoff wavelength for a given mode, the guide wavelength becomes pure imaginary or, stated physically, *nonpropagating* evanescent modes are set up.

2.1.1 Fields in a Right Circular Cylindrical Cavity

For the problem at hand, we are dealing with a cylindrical structure having two metallic end plates. In other words, it is bounded in axial extent and the associated internal fields may have variations in *both* the transverse and axial directions. In the case of TE waves, the wave equation (2.1) for H_z expressed in cylindrical coordinates becomes

$$\frac{1}{r}\frac{\partial}{\partial r}\left[r\frac{\partial}{\partial r}H_z\right] + \frac{1}{r^2}\frac{\partial^2}{\partial\phi^2}H_z + \frac{\partial^2}{\partial z^2}H_z + k^2H_z = 0 \ , \tag{2.15}$$

where

$$k_c^2 = \gamma^2 + k^2 \ , \tag{2.16}$$

and from (1.7)

$$\begin{aligned}k^2 &= \omega^2\mu^*\epsilon^* = \omega^2\mu^*\left(\epsilon' - j\epsilon''\right) \\ &= \omega^2\mu^*\epsilon_0\epsilon'_R\left(1 - j\tan\delta\right) \ .\end{aligned} \tag{2.17}$$

The wavenumber is complex for lossy dielectric materials; that is,

$$k = k_{real} - jk_{imaginary} \, , \tag{2.18}$$

where it is easily shown that

$$k_{real}^2 = \frac{\omega^2 \mu \epsilon}{2} \left[\sqrt{\left(\frac{\sigma}{\omega \epsilon}\right)^2 + 1} + 1 \right] \, , \tag{2.19}$$

and

$$k_{imaginary}^2 = \frac{\omega^2 \mu \epsilon}{2} \left[\sqrt{\left(\frac{\sigma}{\omega \epsilon}\right)^2 + 1} - 1 \right] \, . \tag{2.20}$$

In (2.19) and (2.20) $\mu = \mu'_R \mu_0$ and $\epsilon = \epsilon'_R \epsilon_0$, where $\epsilon_0 \approx 8.854 \times 10^{-12}$ F/m and $\mu_0 = 4\pi \times 10^{-7}$ H/m. For lossless dielectrics, $\sigma = 0$ and $\epsilon''_R = 0$ with $k_{real} = \omega\sqrt{\mu\epsilon}$ and $k_{imaginary} = 0$.

The method of separation of variables [5] yields a solution to (2.15) of the form

$$H_z(r, \phi, z) = R(r)\Phi(\phi)Z(z) \, . \tag{2.21}$$

Substitution of (2.21) into (2.15) and then division by $H_z(r, \phi, z)$ yields

$$\frac{1}{rR}\frac{d}{dr}\left[r\frac{d}{dr}R\right] + \frac{1}{r^2\Phi}\frac{d^2\Phi}{d\phi^2} + \frac{1}{Z}\frac{d^2Z}{dz^2} + k^2 = 0 \, . \tag{2.22}$$

The third term in (2.22) is explicitly independent of r and ϕ. It is also necessarily independent of z if (2.22) is to sum to zero for all (r, ϕ, z). Therefore,

$$\frac{1}{Z}\frac{d^2Z}{dz^2} = -\beta^2 \, , \tag{2.23}$$

where β is a constant. Substitution of (2.23) into (2.22) and multiplication by r^2 results in

$$r\frac{1}{R}\frac{d}{dr}\left[r\frac{d}{dr}R\right] + \frac{1}{\Phi}\frac{d^2\Phi}{d\phi^2} + \left(k^2 - \beta^2\right) r^2 = 0 \, . \tag{2.24}$$

The second term in (2.24) is a function of ϕ only, whereas the rest of the equation is a function of r only. By the same argument the azimuthal component of H_z obeys the relation,

$$\frac{1}{\Phi}\frac{d^2\Phi}{d\phi^2} = -m^2 \, , \tag{2.25}$$

where m is a constant. Substitution of (2.25) into (2.24) and multiplication throughout by $R(r)$ yields

$$r\frac{d}{dr}\left[r\frac{d}{dr}R\right] + \left[\left(k^2 - \beta^2\right) r^2 - m^2\right] R = 0 \, , \tag{2.26}$$

where β is the axial waveguide propagation constant and k is the medium wavenumber. Now $k^2 - \beta^2$ represents the square of the transverse radial wavenumber k_c, where

$$k_c = \sqrt{k^2 - \beta^2} \; , \tag{2.27}$$

so (2.26) can be written

$$r\frac{d}{dr}\left[r\frac{dR}{dr}\right] + \left[(k_c r)^2 - m^2\right] R = 0 \; . \tag{2.28}$$

The original Helmholtz equation (2.1) is now separated into three equations, each of which determines only one of the functions $R(r)$, $\Phi(\phi)$ or $Z(z)$. Equations (2.23) and (2.25), are harmonic equations, whose solutions are harmonic functions, or linear combinations of sines and cosines. Equation (2.28) is a Bessel equation of the mth order with independent solutions $J_m(k_c r)$ and $N_m(k_c r)$, where J_m and N_m represent Bessel and Newmann functions, respectively, of order m. Because $N_m(k_c r)$ is not finite at $r = 0$, the solution for $R(r)$ is

$$R(r) = J_m(k_c r) \; . \tag{2.29}$$

The choice of the constants β and m, as well as the solutions for (2.23) and (2.25), depends on the physical geometry of the fixture, conditions at the boundaries, and the type of field to be supported by the fixture. For nonevanescent modes in a right circular cylindrical cavity, k_c can take on only *characteristic* or certain discrete values that correspond to different modes of propagation. In the case of the cylindrical cavity, the constant m must be an integer if the solution for E_z is to be single-valued in ϕ (periodic). The radical that defines the transverse radial wavenumber k_c calls for some comment. The branch of the square root is usually chosen such that $k_c \to k$ as $|\beta^2| \to 0$ and $k_c \to |\beta|$ as $|k^2| \to 0$.

2.1.2 *TE* Modes

For *TE* mode structure ($E_z = 0$) we need solve for only H_z. All other field components interior to the cavity are derived from H_z by Maxwell's equations. The complete solution for $H_z(r, \phi, z)$ is

$$H_z(r, \phi, z) = J_m\left(k_c r\right)\left[A\cos m\phi + B\sin m\phi\right]\sin\beta z \; , \tag{2.30}$$

which satisfies the boundary conditions at the cavity's cylindrical walls $r = a$ and at the cavity endplate $z = 0$. This includes the case where the cavity is filled with dielectric materials such that the transverse wavenumber k_c is complex. A and B are constants in (2.30) which determine the phase of the azimuthal field

Table 2.1: Zeroes t'_{mn} of the first derivative of the Bessel function of first kind and order m.

m	n		
	1	2	3
0	3.8317	7.0156	10.1735
1	1.8412	5.3314	8.5363
2	3.0542	6.7061	9.9695
3	4.2012	8.0152	11.3459

orientation relative to the coupling port(s). We are currently considering only an empty cylindrical cavity with $k = \omega\sqrt{\mu_0\epsilon_0}$ and arbitrary azimuthal phase. In order to enforce the boundary condition on E_ϕ, derivable from H_z from Maxwell's equations, we must have

$$E_\phi(a, \phi, z) = 0 \ , \tag{2.31}$$

or

$$J'_m(k_c a) = 0 \ , \tag{2.32}$$

where J'_m represents the first derivative of the mth-order Bessel function. If the zeroes of J'_m are denoted by t'_{mn}, where $n = 1, 2, 3, \ldots$ represents the zero-crossing number, then k_c must be chosen to take only certain discrete values; that is,

$$k_c = \frac{t'_{mn}}{a} \ . \tag{2.33}$$

Table 2.1 gives some representative values for t'_{mn}.

The boundary condition on the azimuthal electric field at $z = L$ results in certain allowable values for the longitudinal propagation wavenumber:

$$\sin \beta L = 0 \ , \tag{2.34}$$

or

$$\beta = p\frac{\pi}{L} \ , \tag{2.35}$$

where $p = 1, 2, 3, \ldots$. The final result for the axial magnetic field of the TE mode within the cavity is:

$$H_z\left(TE_{mnp}\right) = J_m\left(\frac{t'_{mn}r}{a}\right)\left[A\cos m\phi + B\sin m\phi\right]\sin\left[\frac{p\pi z}{L}\right] \ , \tag{2.36}$$

where m, n, p are integers describing the TE mode. An identical analysis can be performed to derive the axial electric field for the TM case.

The characteristic equation used to find the resonant frequencies for TE modes in the empty cavity is given by Harrington [5] pp. 213–216 as,

$$f_0\left(TE_{mnp}\right) = \frac{c_{air}}{2\pi}\left[\left(\frac{t'_{mn}}{a}\right)^2 + \left(\frac{p\pi}{L}\right)^2\right]^{\frac{1}{2}}, \tag{2.37}$$

where c_{air} is the speed of light under testing conditions in air. When $f = f_0(TE_{mnp})$, we have a TE-mode solution to Maxwell's equations. From (2.2), the resonant frequency for the cylindrical cavity filled with material of complex permittivity ϵ^* and real permeability μ' is

$$f_0\left(TE_{mnp}\right) = \frac{1}{2\pi\sqrt{\mu'\epsilon'\left(\tan^2\delta + 1\right)^{\frac{1}{2}}}}\left[\left(\frac{t'_{mn}}{a}\right)^2 + \left(\frac{p\pi}{L}\right)^2\right]^{\frac{1}{2}}. \tag{2.38}$$

As expected, when the length or diameter of a cylindrical cavity increases, the resonant frequency decreases for any given mode.

The figure of merit for assessing the performance or quality of a cavity resonator is the quality factor Q which is a measure or energy stored in the fields inside the resonator compared to the energy loss or dissipation per cycle. The Q-factor is defined by

$$\begin{aligned} Q &= 2\pi\frac{maximum\ energy\ stored\ during\ a\ cycle}{average\ energy\ dissipated\ per\ cycle}, \\ &= \frac{2\pi W}{PT} = \frac{\omega_0 W}{P}, \end{aligned} \tag{2.39}$$

where W is stored energy, P is power dissipation, ω_0 is resonant radian frequency, and T is period $= \frac{2\pi}{\omega_0}$. The higher the axial TE-mode number for any given azimuthal and radial mode number, the greater the cavity quality factor, for any given cavity diameter or length. The quality factor Q will be discussed in more detail in connection with the evaluation of dielectric loss measurements of materials. Some examples of normalized cavity Q values are shown in Fig. 2.6 for some representative TM_{mnp} modes and in Fig. 2.7 for some TE_{0np} modes as a function of cavity diameter-to-length ratio. In Fig. 2.7, the optimal (highest) Q value is obtained for TE_{mnp} modes when the diameter of the cavity equals its length. Our 60-mm diameter resonator's length is roughly seven times longer than its diameter. Resonance Q-value is compromised somewhat by this length, but the cavity's helical windings yield better than 30-dB mode purity. The length of the resonator allows for the measurement of several TE_{01p} modes in the X-band frequency range.

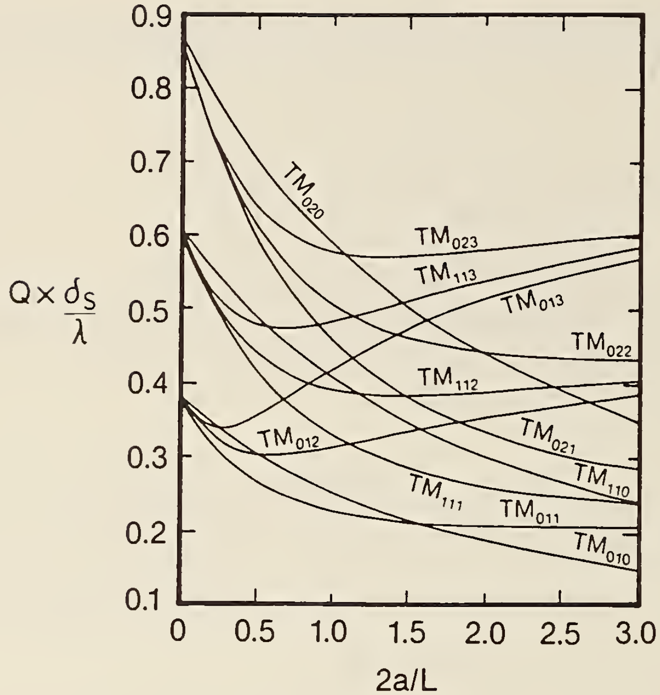

Figure 2.6: Normalized cavity Q versus diameter-to-length ratio for a right circular cylinder for TM_{mnp} modes. δ_s is the skin depth of the cavity wall material and λ is the free-space wavelength.

2.2 TE_{01p} Mode-Filter Cylindrical Cavity

One very useful cavity resonator for microwave dielectric property measurements on low-loss materials is constructed to filter all modes resulting from current other than that flowing circumferentially about the cavity wall. This yields a very high-Q cavity with a very pure TE-mode structure for precision electrical property measurements.

Consider, for example, the wall and endplate currents flowing in the cylindrical cavity for TE_{01p} and TM_{11p} modes as shown in Fig. 2.8. For the TM_{11p} mode there are both azimuthal and radial currents, whereas in the case of the TE_{01p} mode there are only azimuthal currents flowing in the wall and end plates. As Cook [2] notes, the presence of unwanted modes produces larger data scatter of loss tangent and permittivity with change in specimen length than when there is only one effective mode propagating, such as the TE_{01p}.

If the currents associated with the TM_{11p} mode can be interrupted while at the same time leaving those associated with the TE_{01p} mode unchanged, then signif-

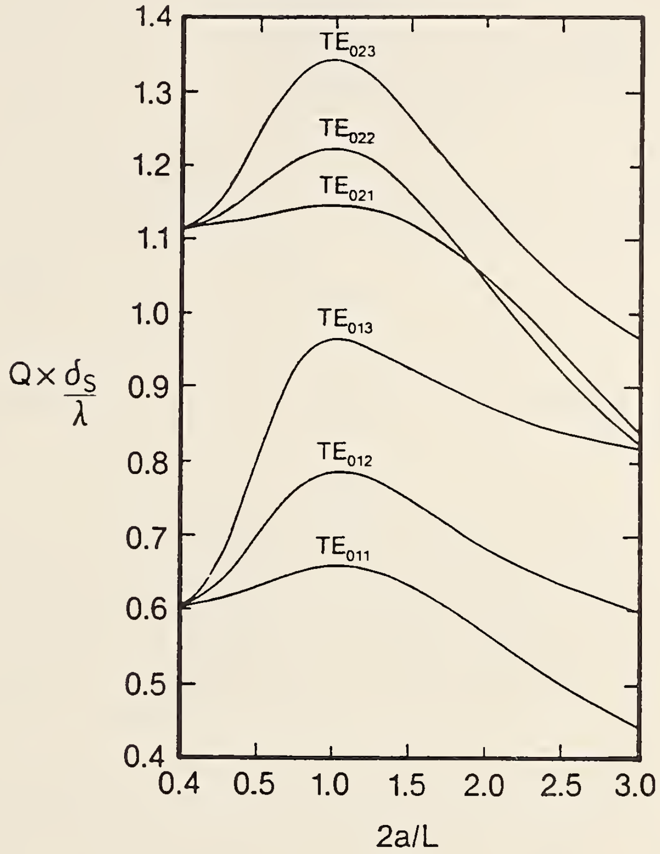

Figure 2.7: Normalized cavity Q versus diameter-to-length ratio for a right circular cylinder for TE_{0np} modes. δ_s is the skin depth of the cavity wall material and λ is the free-space wavelength.

icant attenuation of the TM_{11p} mode will take place. This can be accomplished by constructing a cavity wall in which the conductivity is discontinuous along the length of the cylinder but continuous around its circumference. This type of cavity becomes a mode filter. One approach to the construction of a mode-filter cavity is to construct a cylinder of annular copper rings electrically insulated from each other [2] to form a continuous cylinder. In this case, the currents flowing from ring to ring will be greatly impeded while those flowing around the rings are unimpeded.

Another approach is to make a helical waveguide by winding fine enameled wire into a precision cylindrical former (Fig. 2.9). This type of waveguide has been discussed in the literature by Morgan and Young [6]; Unger [7]; Young [8]; Cook

Figure 2.8: Currents flowing in (a) cylinder wall and (b) cavity end plates for TE_{01p} and TM_{11p} modes.

and Jones [9]; Cook [2]; Waldron [10] [11]; Waldron and Bowe [12]; Shimba [13]; Kazantsev, Kaznacheev and Meriakri [14]; Mikoshiba [15]; Piefke [16]; Waldron, Bowe, Wackrill and Wescott [17]; and Noda, Yamaguchi and Suzuki [18]. In a helical waveguide the modes whose wall currents follow the conducting helix possess attenuation constants which are essentially the same as for copper pipe. All other modes, however, have a high transmission loss. In other words, all modes other than the TE_{01p} have very large attenuation constants in a helical waveguide. The exact calculation of the attenuation constants for modes other than TE_{01p} depends on the helix pitch angle and the electrical properties of the thick lossy dielectric jacket surrounding the helix; the fields of those modes penetrate into the jacket and are attenuated, whereas that of the TE_{01p} mode does not and is therefore minimally attenuated. The helical waveguide, then, is the equivalent of an anisotropically conducting cylinder. Whether we use helically wound wire as opposed to annular copper rings is a question of convenience in manufacture.

Waldron [10], and Waldron and Bowe [12] have analyzed the azimuthal electric field dependence as a function of the radius of the helical winding, the complex permittivity of the jacket material, and the pitch angle of the helix. To a first-order approximation the TE_{01p} mode is independent of the pitch angle. Waldron and Bowe [12] also treat the effect of pitch for the TE_{01p} mode to the second order and find that the effect of a finite pitch does indeed cause a slight attenuation of the TE_{01p} mode. Cook [2] compares the attenuation of the TE_{01p} mode to other modes in a 50-mm diameter helix at 35 GHz; these results are given in Table 2.2. Note that the modes other than TE_{01p} have attenuation constants larger by several orders of magnitude than that for TE_{01p}. Piefke [16] demonstrates that

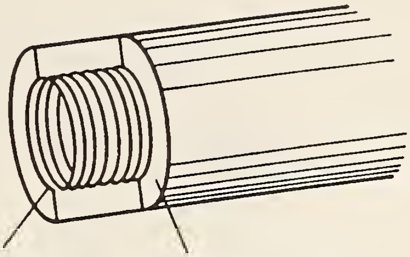

Figure 2.9: TE_{01p} mode filter consisting of helix waveguide.

Table 2.2: Typical mode attenuation in helical waveguide at 35 GHz (from Cook [2]).

Mode	Attenuation (dB/km)
TE_{01}	2–3
TE_{12}	4 000–5 000
TE_{11}	6 000–7 000
TE_{21}	10 000–12 000

attenuation for the TE_{12p} mode exceeds by a factor of 3×10^3 the attenuation of the same mode in a homogeneous circular copper cavity having the same dimensions. An electromagnetic field analysis of propagation interior to a helix of arbitrary radius and pitch that is loaded with a lossy dielectric sample is useful in the actual design of the mode-filter structure.

2.2.1 NIST 60-mm X-Band, Mode-Filtered Cavity Resonator

The 60-mm diameter X-band cavity constructed by NIST is mode filtered and is similar to those developed in the United Kingdom and Germany [2, 3]. A cutaway drawing of the fixture is shown in Fig. 2.10. The cylindrical walls of the cavity consist of helically wound wires (0.15 mm diameter) which allow only azimuthal current flow that, in turn, yields very pure TE_{01p} mode structure and empty cavity Q values of approximately 8×10^4 at 10 GHz. Non-TE_{01p} modes such as TE_{12p} and TE_{02p}, if excited, are attenuated by at least 30 dB, so high precision dielectric measurements can be made. Details of the cavity resonator design are given in Chapter 3, and drawings are given in Appendix C.

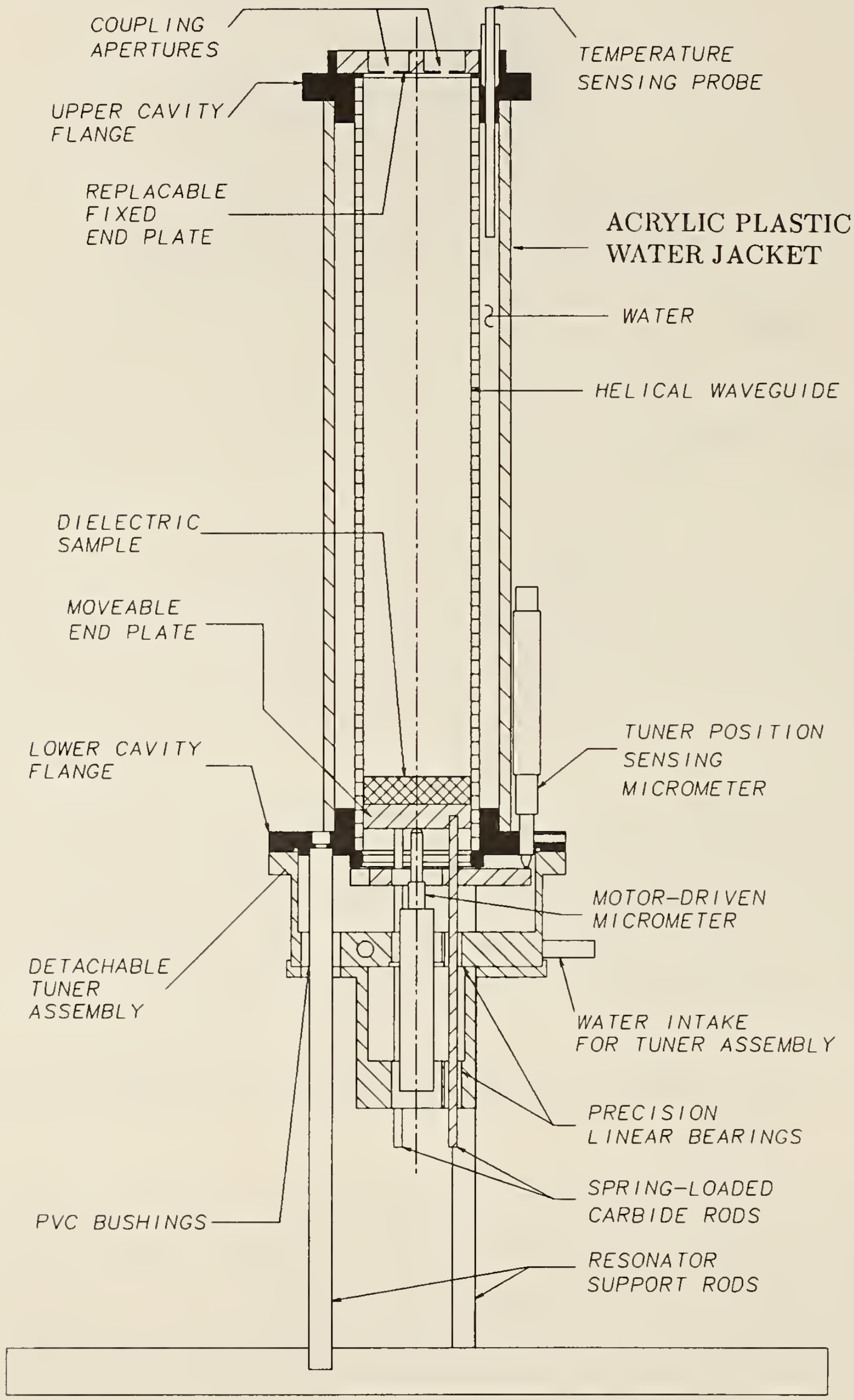

Figure 2.10: NIST mode-filtered X-band cavity.

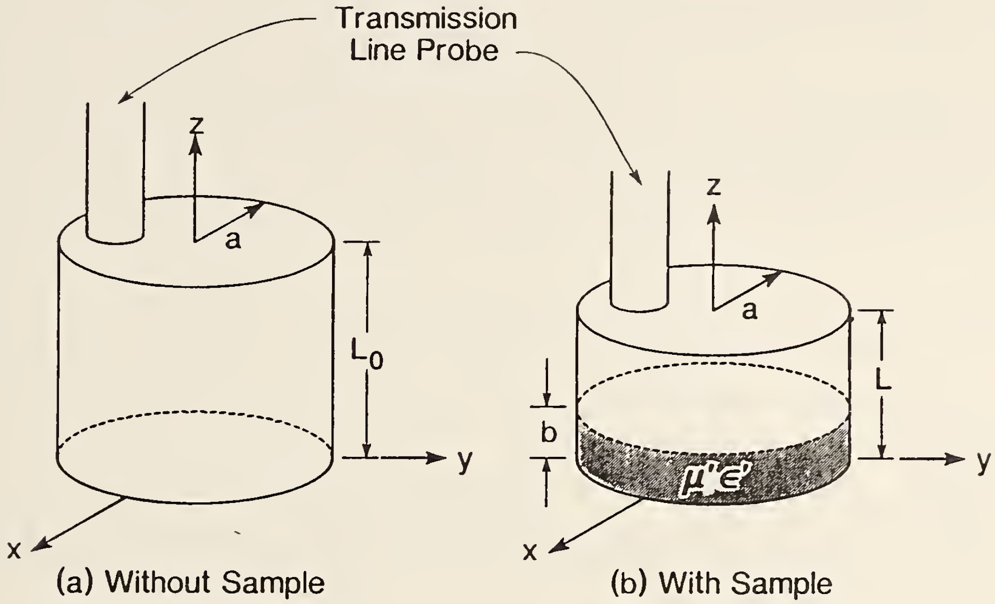

Figure 2.11: Schematic of fixed-frequency cavity resonator measurement technique.

The relative real effective permittivity (ϵ'_R) and loss tangent ($\tan\delta = \epsilon''/\epsilon'$) calculations can be made by noting the change in a TE_{01p} mode when the resonator contains the dielectric sample. Either a fixed-length or fixed-frequency technique can be used. The fixed-length technique uses the shift in resonant frequency and change in resonant bandwidth when the sample is inserted into the cavity to calculate ϵ'_R and $\tan\delta$. With the fixed-frequency technique, the cavity length is shortened so the cavity resonates at the empty-cavity frequency (Fig. 2.11). In this case, the changes in cavity length and resonance bandwidth are used to calculate ϵ'_R and $\tan\delta$. In order to measure the change in length, the change in the lower movable endplate's position relative to the cavity body must be measured. This measurement is made using another micrometer attached to the bottom cavity flange. We now address these calculations in more detail.

2.2.2 Evaluation of Real Effective Permittivity

The nonzero electromagnetic field components for the TE_{01p} mode interior to a right circular cylindrical cavity are derived from (2.36) with $m = 0$ and $n = 1$.

The axial magnetic field is

$$H_z = H_0 J_0\left(\frac{t'_{01}r}{a}\right)\sin\left(\frac{p\pi z}{L}\right) . \tag{2.40}$$

The two other nonzero field components are derivable from H_z using Maxwell's equations expanded in cylindrical coordinates:

$$H_r = \frac{1}{k_c^2}\frac{\partial^2 H_z}{\partial z \partial r} = \frac{pH_0\pi a}{t'_{01}L}J'_{01}\left(\frac{t'_{01}r}{a}\right)\cos\left(\frac{p\pi z}{L}\right) , \tag{2.41}$$

$$E_\phi = \frac{j\omega\mu}{k_c^2}\frac{\partial H_z}{\partial r} = \frac{j\omega\mu_0 H_0 a}{t'_{01}}J'_0\left(\frac{t'_{01}r}{a}\right)\sin\left(\frac{p\pi z}{L}\right) . \tag{2.42}$$

The steady-state time factor $\exp(j\omega t)$ is understood in (2.40)–(2.42). Application of boundary conditions at resonance for the tangential electric (E_ϕ) and tangential magnetic (H_r) fields on the surface of a homogeneous, linear, isotropic dielectric disk sample and for the tangential electric field on the cavity end plates yields the following transcendental equation in terms of the axial wavenumber β_1 in the sample under test:

$$\frac{\tan(\beta_1 b)}{\beta_1} = -\frac{\tan\left[\beta_0\left(L_r - b\right)\right]}{\beta_0} , \tag{2.43}$$

where

$$\beta_1 = \sqrt{\omega^2\mu'_R\epsilon'_R\mu_0\epsilon_0 - k_c^2} , \tag{2.44}$$

with $k_c = t'_{01}/a$ and where the axial propagation wavenumber β_0 in the air-filled portion of the cavity resonator is

$$\beta_0 = \sqrt{\left(\frac{\omega}{c_{air}}\right)^2 - k_c^2} . \tag{2.45}$$

Here we allow for change in the speed of light in air due to ambient temperature and humidity. From (2.42) we may determine the radial variation of the azimuthal electric field internal to the cavity, as shown in Fig. 2.2. For the fixed-frequency technique, L_r is simply L, whereas for the fixed-length technique $L_r = L_0$. The real effective permittivity is implicitly expressed in the axial propagation constant in the sample. The Newton-Raphson iterative method may be used (Sec. 5.2.7) to solve for β_1 in (2.43) from which the relative real effective permittivity is given by

$$\begin{aligned}\epsilon'_R &= \frac{\beta_1^2 + k_c^2}{\beta_0^2 + k_c^2} \\ &= \frac{\beta_1^2 + (t'_{01}/a)^2}{\beta_0^2 + (t'_{01}/a)^2} .\end{aligned} \tag{2.46}$$

Equation (2.43) is solved for either cavity length, keeping the resonant frequency constant, or resonant frequency, keeping the cavity length fixed.

2.2.3 Evaluation of Dielectric Loss and Quality Factor

Evaluation of the loss tangent $\tan\delta$ in terms of measured changes in the Q-factor of the cavity is more complicated. For any given mode the resonator bandwidth is proportional to dissipated power and therefore inversely proportional to Q-factor. Hence, high Q-factor resonators have narrow bandwidths. In this section, we discuss resonators with respect to the Q factor and its measurement for dielectric loss evaluation.

Useful insight into some properties of cavity resonators may be gained by examining the differential equation for a simple damped resonant circuit with a forcing input function $f(t)$ and and output function $v(t)$:

$$\frac{d^2v}{dt^2} + 2\sigma\frac{dv}{dt} + \omega_0^2 v = f(t)\ . \tag{2.47}$$

Equation (2.47) may be obtained by suitable manipulation of Maxwell's equations. When $\sigma = 0$, the homogeneous solution to (2.47) is simply

$$v(t) = C_1 \sin\omega_0 t + C_2 \cos\omega_0 t\ . \tag{2.48}$$

The case where $\sigma > 0$ always corresponds to a resonator with losses. If we Laplace transform (2.47) and solve for the transfer function of our resonator, we obtain

$$T(s) = V(s)/F(s) = \frac{1}{s^2 + 2\sigma s + \omega_0^2}\ , \tag{2.49}$$

or

$$T(s) = \frac{j}{2\omega_L}\left[\frac{1}{s+\sigma+j\omega_L} - \frac{1}{s+\sigma-j\omega_L}\right]\ , \tag{2.50}$$

where ω_L is the loaded natural resonant frequency (either with or without sample) given by

$$\omega_L = \sqrt{\omega_0^2 - \sigma^2}\ . \tag{2.51}$$

The presence of loss ($\sigma > 0$), however small, in the cavity results in a change in the resonant frequency. This is commonly called frequency pulling due to loss. Under zero-loss conditions ($\sigma = 0$) we get $\omega_L = \omega_0$. This indicates that uncompensated losses in the resonator can introduce a positive bias in ϵ'_R results.

The natural response of a simple resonator described by (2.47) is

$$v(t) = Ve^{-\sigma t}\sin\omega_L t\ . \tag{2.52}$$

The stored energy W is proportional to the average value of $v^2(t)$ over a period, which for small σ is given by

$$W = \frac{1}{2}V^2 e^{-2\sigma t}\ . \tag{2.53}$$

The average power P in the system is

$$P = -\frac{dW}{dt} = 2\sigma W \ , \tag{2.54}$$

so

$$\sigma = \frac{P}{2W} \ . \tag{2.55}$$

From (2.39) we may now write the Q-factor in terms of the resonant unloaded natural frequency and loss,

$$Q = \frac{\omega_0}{2\sigma} \ . \tag{2.56}$$

From (2.51), the loaded natural resonant frequency now becomes

$$\omega_L = \omega_0 \left[1 - \frac{1}{4Q^2}\right]^{\frac{1}{2}} \ . \tag{2.57}$$

Thus, the frequency pulling due to loss can be determined by the Q-factor. Substitution of (2.56) into (2.47) yields

$$\frac{d^2v}{dt^2} + \frac{\omega_0}{Q}\frac{dv}{dt} + \omega_0^2 v = f(t) \ , \tag{2.58}$$

from which we see that in the case of a perfect resonator ($Q \to \infty$), the first derivative term vanishes. For a practical resonator, however, Q is finite, and the second term of (2.58) must be kept. Kajfez [19] has also treated this resonance frequency correction for the case of a one-port cavity.

If the transfer function $T(s)$ is written in the frequency domain rather than as a Laplace transform, we set $s = j\omega$ and $\sigma = \omega_0/2Q$ from (2.56). Equation (2.49) then becomes

$$T(\omega) = V(j\omega)/F(j\omega) = \frac{1}{\omega_0^2 - \omega^2 + j\omega\omega_0/Q} \ . \tag{2.59}$$

The relevance of this discussion on the resonator transfer function in terms of the Q-factor of any resonator is now apparent. The denominator in (2.59) may be written as

$$\omega_0^2 - \omega^2 + \frac{j\omega\omega_0}{Q} = \frac{j\omega\omega_0}{Q}\left[1 + jQ\left(\frac{\omega}{\omega_0} - \frac{\omega_0}{\omega}\right)\right] \ . \tag{2.60}$$

The frequency dependence in (2.60) can be factored in the following way,

$$\frac{\omega}{\omega_0} - \frac{\omega_0}{\omega} = \left(\frac{\omega - \omega_0}{\omega_0}\right)\left(\frac{\omega_0}{\omega} + 1\right) \ . \tag{2.61}$$

For a high-Q resonator the measurement frequency is close to the resonance frequency, $\omega \approx \omega_0$, so $\omega_0/\omega + 1 \approx 2$. Thus,

$$\frac{\omega}{\omega_0} - \frac{\omega_0}{\omega} \approx 2\left(\frac{\omega - \omega_0}{\omega_0}\right) = 2\delta\omega \ , \tag{2.62}$$

where $\delta\omega$ is known as the frequency tuning parameter of the resonator. Substitution of (2.62) into (2.59) then gives the following relation for the cavity transfer function

$$T(\omega) = \frac{-jQ/\left(\omega\omega_0\right)}{1 + jQ2\left(\delta\omega\right)} \ , \tag{2.63}$$

where the magnitude of the transfer function versus frequency is a bell-shaped Lorentzian resonance line shape whose form is Q-dependent.

The half-power bandwidth of any resonator is defined as the frequency spread $\Delta\omega$, where $\Delta\omega$ is the difference in half-power frequencies ω_1 and ω_2, and with

$$|T(\omega)| = \frac{1}{\sqrt{2}}|T(\omega_0)| \tag{2.64}$$

for either $\omega = \omega_1$ or $\omega = \omega_2$. Equation (2.64) may be written as

$$\frac{Q/\left(\omega\omega_0\right)}{\sqrt{1 + 4Q^2\left(\delta\omega\right)^2}} = \frac{1}{\sqrt{2}}\frac{Q}{\omega_0^2} \ . \tag{2.65}$$

Equation (2.65) is only valid when $4Q^2(\delta\omega)^2 = 1$ (assuming $\omega \approx \omega_0$) or when

$$\delta\omega = \pm\frac{1}{2Q} \ . \tag{2.66}$$

The half-power frequencies must be symmetric about the resonant frequency for pure modal structure; that is,

$$\begin{aligned} \omega_1 &= \omega_0 - \frac{\omega_0}{2Q} \ , \\ \omega_2 &= \omega_0 + \frac{\omega_0}{2Q} \ . \end{aligned} \tag{2.67}$$

The bandwidth by definition is

$$\Delta\omega = |\omega_1 - \omega_2| = \frac{\omega_0}{Q} = 2\sigma \ . \tag{2.68}$$

Hence, the quality factor is given by

$$Q = \frac{\omega_0}{\Delta\omega} = \frac{f_0}{\Delta f_{1/2}} \ . \tag{2.69}$$

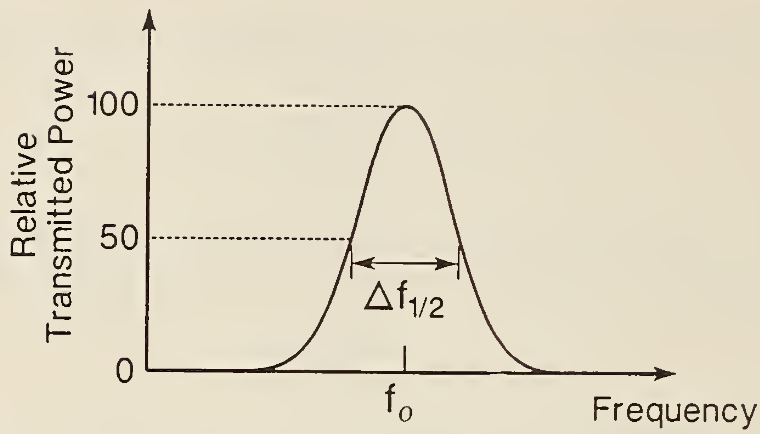

Figure 2.12: Cavity transmission curve defining loaded Q-factor, $Q = f_0/\Delta f_{1/2}$.

This is shown in Fig. 2.12. Knowledge of the Q-factor at any resonant frequency allows us to rapidly determine the resonator bandwidth and loaded natural resonant frequency ω_L.

When a resonant cavity is used as a dielectric measurement device in a microwave system, several different Q-factors can be defined. The first Q-factor accounts for internal cavity losses. It is defined as the unloaded Q-factor Q_0. Second, the external Q-factor Q_E accounts for external losses due to the coupling ports. These occur because the cavity resonator must be connected to external microwave circuitry in order to be usable. Finally, the loaded Q-factor Q_L is the measured Q-factor that includes both internal and external losses.

Consider the unloaded Q-factor that is due to losses in the cavity itself. For the unloaded Q,

$$Q_0 = \frac{\omega_0 W}{P_i} , \tag{2.70}$$

where P_i is the internal power dissipation. For cavity resonators, power dissipation P_C in the cavity conducting end plates and wall and power dissipation P_S in any dielectric sample in the cavity contribute to internal power dissipation. Hence,

$$\frac{1}{Q_0} = \frac{P_C + P_S}{\omega_0 W} = \frac{1}{Q_C} + \frac{1}{Q_S} , \tag{2.71}$$

where the cavity conductor quality factor is given by Q_C, and the dielectric quality factor by Q_S.

For linear, homogeneous dielectrics the dielectric quality factor Q_S is

$$Q_S = \frac{\omega_0 W}{P_S} = \frac{\omega_0 \epsilon_0 \epsilon'_R \int\int\int |E|^2 dV}{\sigma \int\int\int |E|^2 dV} \tag{2.72}$$

$$= \frac{\omega_0 \epsilon_0 \epsilon_R'}{\sigma} = \frac{1}{\tan\delta} .$$

The loss tangent is simply the reciprocal of Q_S. Clearly the lowest value of Q-factor in (2.71) dominates Q_0, the unloaded Q. It is therefore to our advantage to have a high-Q resonator for good measurement precision of low-loss dielectric materials.

In practice, a cavity must deliver power to an external load through the coupling ports. The power loss due to the presence of an external load in a cavity dielectric measurement system results in the external quality factor Q_E, which is defined by

$$Q_E = \frac{\omega_0 W}{P_E} . \tag{2.73}$$

The stored energy in the numerator is still the energy stored inside the cavity, but the power loss in the denominator is an external drain on the internal energy.

The loaded Q is the total Q for the system including power losses both internal and external to the cavity resonator. The loaded Q is

$$Q_L = \frac{\omega_0 W}{P_T} , \tag{2.74}$$

where P_T is the total power loss. That is,

$$P_T = P_i + P_E , \tag{2.75}$$

so

$$\frac{1}{Q_L} = \frac{1}{Q_E} + \frac{1}{Q_0} . \tag{2.76}$$

The loss tangent is then

$$\tan\delta = \frac{1}{Q_S} = \frac{1}{Q_L} - \frac{1}{Q_C} - \frac{1}{Q_E} , \tag{2.77}$$

or

$$\tan\delta = \frac{1}{Q_0} - \frac{1}{Q_C} , \tag{2.78}$$

with a sample inserted into the cavity. For the empty cavity the total loaded Q losses are just

$$\frac{1}{Q_L} = \frac{1}{Q_E} + \frac{1}{Q_C} . \tag{2.79}$$

The loaded Q-factor Q_L and the unloaded Q-factor Q_0 are related by the coupling coefficient $\kappa = P_E/P_i$. This relationship is given by Ginzton [20] and Sucher and Fox [21] as

$$Q_L = \frac{Q_0}{1+\kappa} . \tag{2.80}$$

For a two-port resonator like the NIST X-band cavity,

$$\kappa = \kappa_1 + \kappa_2 \ , \tag{2.81}$$

where κ_1 and κ_2 are the coupling coefficients for ports one and two. In terms of the unloaded cavity Q with a sample, we have

$$Q_0 = \frac{Q_C}{1 + Q_C \tan\delta} \ , \tag{2.82}$$

or, in terms of the loaded cavity Q with a sample under test,

$$(1+\kappa)\, Q_L = \frac{Q_C}{1 + Q_C \tan\delta} \ . \tag{2.83}$$

Thus, from

$$\tan\delta = \frac{Q_C - (1+\kappa)\, Q_L}{(1+\kappa)\, Q_L Q_C} \tag{2.84}$$

we can determine the loss tangent if we know the coupling coefficient κ, the loaded quality factor Q_L and the cavity conductor quality factor Q_C.

2.2.4 Port Coupling Coefficients for Determining Unloaded Cavity Q-Factor

Frequency domain techniques for cavity measurements can be a reflection method or transmission method. An automatic network analyzer (ANA) displays the magnitude and phase of the reflection coefficient S_{11} or transmission coefficient S_{21}. The reflection coefficient can be displayed on the network analyzer in polar coordinates. A Smith chart overlay for the polar reflection-coefficient display enables the impedance to be read directly. Swept-frequency measurements can also be performed and displayed with the network analyzer. Ginzton [20] considers various equivalent circuits for cavities with multiple resonant modes and shows that a resonant circuit with an external load has a coupling coefficient given by

$$\kappa = \frac{Q_0}{Q_L} - 1 = \frac{R_L}{R_C} \ , \tag{2.85}$$

where R_L and R_C are the internal cavity and external resistive losses, respectively. For $\kappa < 1$, the cavity is undercoupled to any external component. For $\kappa > 1$, the cavity and external component are overcoupled, and for $\kappa = 1$, the cavity is critically coupled. In order to determine the total empty cavity conductor loss Q_C, we need only measure Q_L (without sample) and Q_E. Q_C is then given by

$$Q_{c,empty} = Q_{0,empty} = \frac{Q_L Q_E}{Q_E - Q_L} \ . \tag{2.86}$$

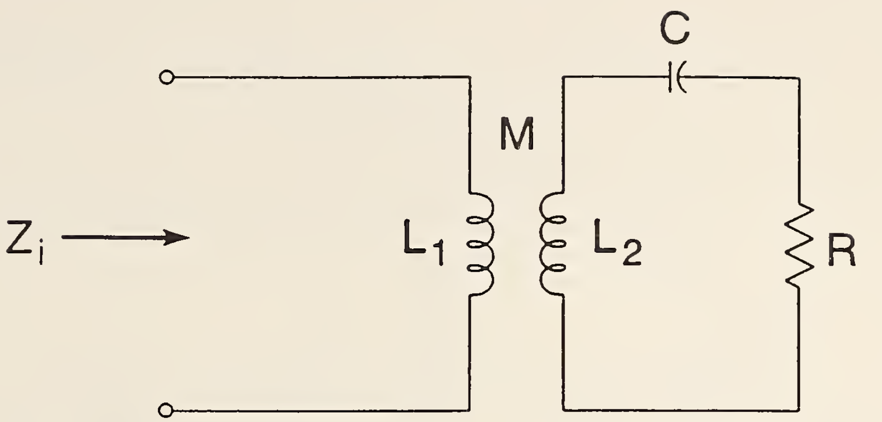

Figure 2.13: Equivalent resonator circuit for reflection Q measurement [18].

Consider now a reflection type of Q measurement with the equivalent circuit for a simple one-port cavity shown in Fig. 2.13. The input impedance is given by

$$Z_i = j\omega L_1 + \frac{(\omega M)^2/R}{1 + jQ_0\left(\frac{\omega}{\omega_0} - \frac{\omega_0}{\omega}\right)} . \tag{2.87}$$

The reactive component ωL_1 due to the coupling port is usually ignored or incorporated into a phase term for data-fitting purposes as described in Sec. 4.3.2. The factor $R_i = (\omega M)^2/R$ is defined as the induced input resistance. Using (2.62) we may rewrite the expression for input impedance as

$$Z_i = \frac{R_i}{1 + jQ_0 2\frac{\omega-\omega_0}{\omega_0}} = \frac{R_i}{1 + jQ_0 2\frac{\Delta\omega}{\omega_0}} . \tag{2.88}$$

The corresponding input reflection coefficient is

$$\Gamma_i = \frac{Z_i - Z_0}{Z_i + Z_0} , \tag{2.89}$$

where Z_0 is the characteristic impedance. When the cavity is detuned, we may allow $\Delta\omega \rightarrow \infty$ so that $Z_i \rightarrow 0$, and the input reflection coefficient becomes

$$\Gamma_i = \Gamma_D = -1 . \tag{2.90}$$

At frequencies close to the resonant frequency ω_0, the locus of the input reflection coefficient describes a circle (See Fig. 2.14). The complex number $\Gamma_i - \Gamma_D$ is given by

$$\Gamma_i - \Gamma_D = \frac{2Z_i/Z_0}{1 + jQ_0 2\frac{\Delta\omega}{\omega_0} + \frac{Z_i}{Z_0}} . \tag{2.91}$$

Now the ratio of the input impedance to the characteristic impedance Z_0 is

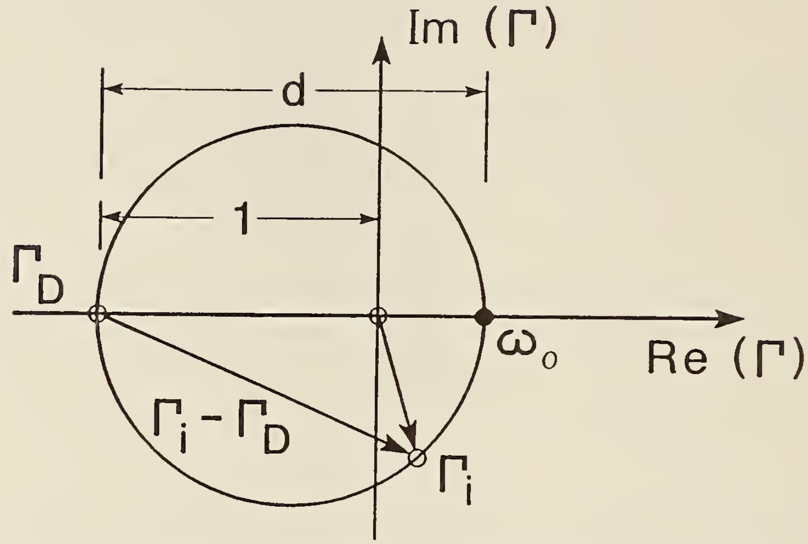

Figure 2.14: Input reflection coefficient versus ω.

defined to be the coupling coefficient; that is,

$$\kappa = \frac{Z_i}{Z_0} . \tag{2.92}$$

Furthermore, the loaded coupling coefficient is defined in (2.80) and (2.85) so that (2.91) may be rewritten

$$\begin{aligned} \Gamma_i - \Gamma_D &= \frac{2\kappa}{1+\kappa+j\,(1+\kappa)\,Q_L 2\frac{\Delta\omega}{\omega_0}} \\ &= \frac{2}{\frac{1}{\kappa}+1+j\left(\frac{1}{\kappa}+1\right)Q_L 2\frac{\Delta\omega}{\omega_0}} \\ &= \frac{2}{\left(\frac{1}{\kappa}+1\right)\left[1+jQ_L 2\frac{\Delta\omega}{\omega_0}\right]} . \end{aligned} \tag{2.93}$$

At resonance $\Delta\omega = 0$ and the circle $\Gamma_i - \Gamma_D$ intersects the real axis. From (2.93) the diameter of this circle is

$$d = |\Gamma_i - \Gamma_D|_{max} = \frac{2\kappa}{1+\kappa} . \tag{2.94}$$

Consequently, if we determine the diameter of the circle on the polar display of the network analyzer, we can compute the coupling coefficient from (2.94) to be

$$\kappa = \frac{d}{2-d} . \tag{2.95}$$

For two ports,

$$Q_L = \frac{Q_0}{1 + \kappa_1 + \kappa_2} , \qquad (2.96)$$

and κ_1 and κ_2 can be determined from separate measurements, as noted above. Then, since we know from transmission measurements that

$$Q_L = \frac{f_0}{\Delta f_{1/2}} , \qquad (2.97)$$

we can determine Q_0. Kajfez and Hwan [22] give similar expressions for a one-port cavity with slightly different coupling elements. This analysis has been extended by Estin and Janezic [23, 24] to the case of a two-port cavity.

2.2.5 Cavity Conductor Quality Factor

The cavity conductor quality factor Q_C is defined as

$$\frac{1}{Q_C} = \frac{P_C}{\omega_0 W} , \qquad (2.98)$$

where P_C is the power loss caused by the finite conductivity of the cavity walls and W is the total stored energy in the cavity. At resonance the total energy interior to the cavity is constant. The average values of the magnetic energy $W_m(t)$ and electric energy $W_e(t)$ are also equal. In other words, the energy alternates between being stored in the electric field and the magnetic field. Lastly, the total stored energy in the cavity is twice the average value; that is,

$$W(t) = 2W_e = 2W_m . \qquad (2.99)$$

Equation (2.99) implies that the input admittance at the cavity ports is real under resonance conditions. To find the total energy storage, the volume integral of either the electric or magnetic field energy over the volume of the resonator must be performed. The average stored electric energy in the empty cavity is

$$W_e = \frac{\epsilon'_{air}}{4} \int_{\text{cavity}} |E|^2 dV . \qquad (2.100)$$

For the TE_{01p} mode structure, the electric field E_ϕ is given by (2.42), so

$$W_e = \frac{\mu_0 \pi L}{8} H_0^2 a^2 \left\{ 1 + \left[\frac{p\pi a}{L t'_{01}} \right]^2 \right\} J_0^2(t'_{01}) . \qquad (2.101)$$

Equation (2.101) demonstrates that increased energy storage results as the length L of the cavity is increased, particularly for the higher axial mode numbers. This fact was taken into account in the design of the NIST 60-mm diameter X-band mode-filtered resonator.

In order to determine the cavity Q, the losses caused by the finite conductivity of the cavity walls must now be evaluated. For small losses the surface currents are essentially those associated with the loss-free field solutions in (2.40) through (2.42). The surface current is given by

$$\vec{J}_s = \vec{n} \times \vec{H} \ , \tag{2.102}$$

where $\vec{n}$ is a unit normal to the surface and directed into the cavity. The wave impedance Z_m of the metal wall of a uniform cavity is given by

$$Z_m = \sqrt{\frac{j\omega\mu}{\sigma + j\omega\epsilon'}} \ , \tag{2.103}$$

or

$$Z_m = e^{j\frac{\pi}{4}} \sqrt{\frac{\omega\mu}{\sigma}} \left[1 - j\frac{\omega\epsilon'}{2\sigma} - \frac{3}{8}\frac{\omega^2 (\epsilon')^2}{\sigma} + \cdots \right] \ . \tag{2.104}$$

For metals σ is usually greater than 10^7 S/m and ϵ' is of the order of 10^{-11} F/m. For ω equal to 10^{10} rad/s, $\omega\epsilon'/\sigma$ is only 10^{-8}; for ω equal to 10^{13}, $\omega\epsilon'/\sigma$ is only 10^{-5}. Hence,

$$Z_m \approx \sqrt{\frac{\omega\mu}{2\sigma}} (1 + j) \tag{2.105}$$

is a very good approximation to metal wave impedance at microwave frequencies and

$$R_m = \Re\left(Z_m\right) = \sqrt{\frac{\omega\mu}{2\sigma}} = \frac{1}{\delta_s \sigma} \ , \tag{2.106}$$

where

$$\delta_s = \sqrt{\frac{2}{\omega\mu\sigma}} \tag{2.107}$$

is the plane-wave skin depth of the metallic cavity wall (depth at which fields have decreased by a factor of e^{-1} from their surface values). The skin depth and relative loss of various metals at 10 GHz are shown in Table 2.3.

The total power loss in the walls of the cavity is given by

$$P_C = \frac{1}{2} \int_{\text{cavity walls}} \Re\left(Z_m\right) \vec{J}_s \cdot \vec{J}_s^* ds \ , \tag{2.108}$$

or, if the real part of the wave impedance is uniform over the wall surface,

$$P_C = \frac{R_m}{2} \int_{\text{cavity walls}} |H_{\tan}|^2 ds \ . \tag{2.109}$$

From (2.41) and (2.42) we may write the equivalent surface currents at the top, side, and bottom of our cavity. For TE_{01p} mode structure they are

$$\vec{J}_{s,\text{ top endplate}} = \frac{H_0 \pi a}{t'_{01} L} J'_0 \left(\frac{t'_{01} r}{a}\right) \cos(p\pi)\, \hat{u}_\phi \,, \tag{2.110}$$

$$\vec{J}_{s,side} = H_0 J_0 (t'_{01}) \sin\left(\frac{p\pi z}{L}\right) \hat{u}_\phi \,, \tag{2.111}$$

$$\vec{J}_{s,\text{ bottom endplate}} = \frac{H_0 \pi a}{t'_{01} L} J'_0 \left(\frac{t'_{01} r}{a}\right) \hat{u}_\phi \,, \tag{2.112}$$

where $\hat{u}_\phi$ is the unit vector in the azimuthal direction. The magnitudes of the surface currents at the top and bottom end plates are the same. If the real part of the wave impedance is the same for both the top and bottom endplates, the total top and bottom power dissipation $P_{t,b}$ is

$$P_{t,b} = R_m H_0^2 \left[\frac{p\pi a}{t'_{01} L}\right]^2 2\pi \int_0^a r \left\{ J'_0 \left(\frac{t'_{01} r}{a}\right) \right\}^2 dr \,. \tag{2.113}$$

Since

$$\int_0^a r \left\{ J'_0 \left(\frac{t'_{01} r}{a}\right) \right\}^2 dr = \frac{a^2}{2} J_0^2 (t'_{01}) \,, \tag{2.114}$$

we have

$$P_{t,b} = R_{m,ep} H_0^2 \pi a^2 \left(\frac{p\pi a}{t'_{01} L}\right)^2 J_0^2 (t'_{01}) \,, \tag{2.115}$$

where $R_{m,ep}$ is the surface resistance of the endplates.

Table 2.3: Skin depth and relative loss of various metals.

Metal	Conductivity, σ ($\times 10^7$ S/m)	Skin depth, δ_s, at 10 GHz ($\times 10^{-7}$ m)	Relative loss per meter
Ag	6.17	6.42	0.97
Cu	5.80	6.60	1.00
Au	4.10	7.85	1.19
Cr	3.84	8.11	1.23
Al	3.72	8.26	1.25
70-30 brass	1.57	12.7	1.92
P	0.9	17.0	2.5
solder	0.71	18.5	2.8

Similarly, the total side wall power loss can be determined as

$$P_{side} = R_{m,side} H_0^2 \pi a J_0^2 \left(t'_{01}\right) \frac{L}{2} , \tag{2.116}$$

so the total cavity conductor power dissipation is

$$P_C = P_{t,b} + P_{side} = H_0^2 \pi a J_0^2 \left(t'_{01}\right) \left[R_{m,side} \frac{L}{2} + R_{m,ep} a \left(\frac{p\pi a}{t'_{01} L} \right)^2 \right] . \tag{2.117}$$

The cavity quality factor Q_C for TE_{01p} mode structure is then

$$Q_C = \frac{\eta}{2} \frac{\left[\left(\frac{t'_{01}}{a}\right)^2 + \left(\frac{p\pi}{L}\right)^2 \right]^{\frac{3}{2}}}{\frac{R_{m,side}}{a} \left(\frac{t'_{01}}{a}\right)^2 + \frac{2R_{m,ep}}{L} \left(\frac{p\pi}{L}\right)^2} , \tag{2.118}$$

where

$$\eta = \sqrt{\mu_0 / \epsilon'_{air}} . \tag{2.119}$$

Clearly, if the surface resistance of the endplates is the same as that of the cylindrical wall ($R_{m,side} = R_{m,ep} = R_m$) of the cavity, then (2.118) reduces to

$$Q_C = \frac{\eta}{2R_m} \frac{\left[\left(\frac{t'_{01}}{a}\right)^2 + \left(\frac{p\pi}{L}\right)^2 \right]^{\frac{3}{2}}}{\left[\frac{1}{a} \left(\frac{t'_{01}}{a}\right)^2 + \frac{2}{L} \left(\frac{p\pi}{L}\right)^2 \right]} , \tag{2.120}$$

In general, for any TE_{mnp} mode structure in an empty right circular cylindrical cavity, we may in similar fashion determine

$$Q_C \left(TE_{mnp}\right) = \frac{\lambda_0}{2\pi\delta_s} \frac{\left[1 - \left(\frac{m}{t'_{mn}}\right)^2\right] \left[\left(t'_{mn}\right)^2 + \left(\frac{p\pi a}{L}\right)^2 \right]^{3/2}}{\left[\left(t'_{mn}\right)^2 + 2\frac{a}{L} \left(\frac{p\pi a}{L}\right)^2 + \left(1 - \frac{2a}{L}\right) \left(\frac{mp\pi a}{t'_{mn} L}\right)^2 \right]} \tag{2.121}$$

Equation (2.121) is valid when

$$R_{m,side} = R_{m,ep} = R_m , \tag{2.122}$$

for any transverse electric mode and reduces to (2.120) for $m = 0$, $n = 1$. In practice, experimentally determined values of Q (whether the cavity is empty or filled with a sample) are usually less than the theoretical value by an unpredictable amount. This discrepancy between measurement and theory usually is due to the effects of surface finish in the cavity interior. Surface roughness leads to an effective penetration depth for electromagnetic energy which is larger than that given by the plane-wave δ_s in (2.107). The effective penetration depth $\delta_{s,m}$ is that which is determined by the *measured* quality factor $Q_{C,m}$ of the (empty) air-filled cavity. That is, from (2.120) and (2.121)

$$\delta_{s,m} = \frac{c_{air}}{2\pi Q_{C,m} f} \frac{\left[\left(\frac{t'_{01}}{a}\right)^2 + \left(\frac{p\pi}{L}\right)^2\right]^{\frac{3}{2}}}{\left[\frac{1}{a}\left(\frac{t'_{01}}{a}\right)^2 + \frac{2}{L}\left(\frac{p\pi}{L}\right)^2\right]} . \tag{2.123}$$

As we see from (2.123), the measured $Q_{C,m}$ is usually less than the theoretical Q_C for any given resonant frequency f or axial mode number p. Thus, $\delta_{s,m}$ is generally greater than the theoretical δ_s. In the case of the mode-filtered cavity where the side wall is constructed with helically wound waveguide, the real part of the side wall wave impedance $R_{m,side}$ may not exactly equal the real part of the end plate wave impedance $R_{m,ep}$. Through the use of (2.123), the loss tangent evaluation can be corrected by considering the difference in cavity wall losses with and without sample insertion (see Cook [2]).

A somewhat simpler approach for calculating the cavity quality factor Q_C that avoids the calculation of wall currents is the use of an incremental frequency rule [25, 26]. The incremental frequency rule, valid only for rotationally symmetric modes (TE_{0np}), replaces the detailed surface integrations outlined above by a simple computation of increment in inductance per unit length when all metal walls are receded by a distance δ_s. The steps are as follows:

1. Compute the resonant frequency f_0 of an ideal cavity with perfectly conducting walls.

2. Move (perfectly conducting) walls into cavity by one skin depth δ_s. Compute change in resonant frequency for this reduced cavity (r.c.), $\Delta f_{0,\text{ r.c.}}$.

3. Compute the cavity quality factor from $Q_C = f_0/\Delta f_{0,\text{ r.c.}}$.

For the helically wound cavity, where $\delta_{s,side}$ may differ slightly from $\delta_{s,ep}$, we have

$$\Delta f_0 = -\frac{df_0}{d\delta} = -\frac{\partial f_0}{\partial a}\delta_{s,side} - \frac{\partial f_0}{\partial L} 2\delta_{s,ep} . \tag{2.124}$$

For the TE_{01p} mode,

$$f_0 = \frac{c_{air}}{2\pi}\sqrt{\left(\frac{t'_{01}}{a}\right)^2 + \left(\frac{p\pi}{L}\right)^2} , \tag{2.125}$$

so

$$Q_C = \frac{\eta}{2} \frac{\left[\left(\frac{t'_{01}}{a}\right)^2 + \left(\frac{p\pi}{L}\right)^2\right]^{3/2}}{\left[\frac{R_{m,side}}{a}\left(\frac{t'_{01}}{a}\right)^2 + 2\frac{R_{m,ep}}{L}\left(\frac{p\pi}{L}\right)^2\right]}, \tag{2.126}$$

or, for $R_{m,side} = R_{m,ep} = R_m$,

$$Q_C = \frac{\eta}{2R_m} \frac{\left[\left(\frac{t'_{01}}{a}\right)^2 + \left(\frac{p\pi}{L}\right)^2\right]^{3/2}}{\left[\frac{1}{a}\left(\frac{t'_{01}}{a}\right)^2 + \frac{2}{L}\left(\frac{p\pi}{L}\right)^2\right]}, \tag{2.127}$$

which is the same as (2.120). An alternate form of (2.126) is

$$Q_C = \frac{\lambda_0}{2\pi} \frac{\left[(t'_{01})^2 + \left(\frac{p\pi D}{2L}\right)^2\right]^{3/2}}{\delta_{s,side}\,(t'_{01})^2 + \frac{D}{L}\left(\frac{p\pi D}{2L}\right)^2 \delta_{s,ep}}, \tag{2.128}$$

where D is the cavity diameter and δ_s is the skin depth at f_0. Q_C is the unloaded quality factor of the air-filled cavity and that it decreases with increasing frequency since R_m is proportional to $\sqrt{f_0}$. Larger Q_C values are obtained by using higher values of p. The actual loaded Q for the empty cavity is simply related to the theoretical Q by the factor $(1 + \kappa_1 + \kappa_2)$, as discussed previously. If $\delta_{s,side} = \delta_{s,ep} = \delta_s$, then (2.128) may be written

$$Q_C = \frac{\lambda_0}{2\pi\delta_s} \frac{\left[(t'_{01})^2 + \left(\frac{p\pi D}{2L}\right)^2\right]^{3/2}}{(t'_{01})^2 + \frac{D}{L}\left(\frac{p\pi D}{2L}\right)^2}. \tag{2.129}$$

2.2.6 Separation of Cavity Endplate Losses from Side-Wall Losses in Helical Cavity

In some applications it is useful to separate the endplate cavity losses from the side-wall losses. This is particularly true if one or both of the end plates are physically changed. One reason to change the end plate(s) would be to obtain a surface resistance measurement of a conductor. One approach to resolving this problem would be to measure the cavity quality factor Q_C with two rotationally symmetric modes at resonant frequencies which differ by 10 MHz, say, where the tuning factors are not asymmetrically distributed about the associated, resonant frequencies and at the same resonant length. Those modes must be excited and be measurable. From (2.121) we have

$$Q_C\left(TE_{0np}\right) = \frac{\lambda_0}{2\pi} \frac{\left[\left(t'_{0n}\right)^2 + \left(\frac{p\pi a}{L}\right)^2\right]^{3/2}}{\delta_{s,side}\left(t'_{0n}\right)^2 + \frac{2a}{L}\left(\frac{p\pi a}{L}\right)^2 \delta_{s,ep}}, \tag{2.130}$$

and

$$Q_C\left(TE_{0lq}\right) = \frac{\lambda_0}{2\pi} \frac{\left[\left(t'_{0l}\right)^2 + \left(\frac{q\pi a}{L}\right)^2\right]^{3/2}}{\delta_{s,side}\left(t'_{0l}\right)^2 + \frac{2a}{L}\left(\frac{q\pi a}{L}\right)^2 \delta_{s,ep}}, \tag{2.131}$$

where $n \neq l$, $p \neq q$. With these distinct modes there is only an azimuthal electric field, so it is not necessary to consider any capacitive loading at the end plate gaps. If these modes are 10 MHz apart, errors in the skin depth losses at X-band should be less than 3%. Solving (2.130) and (2.131) for $\delta_{s,side}$ and subtracting, we obtain the following relation for the skin depth loss of the end plates for $p \neq q$:

$$\delta_{s,ep} = \frac{\lambda_0 L^3}{4\pi^3 a^2 (p-q)} \left\{ \frac{A \cdot B}{Q_C\left(TE_{0np}\right)} - \frac{C \cdot D}{Q_C\left(TE_{0lq}\right)} \right\}, \tag{2.132}$$

where

$$A = 1 + \left(\frac{p\pi a}{t'_{0n}}\right)^2, \tag{2.133}$$

$$B = \sqrt{\left(t'_{0n}\right)^2 + \left(\frac{p\pi a}{L}\right)^2}, \tag{2.134}$$

$$C = 1 + \left(\frac{q\pi a}{t'_{0l}}\right)^2, \tag{2.135}$$

$$D = \sqrt{\left(t'_{0l}\right)^2 + \left(\frac{q\pi a}{L}\right)^2}. \tag{2.136}$$

Chapter 3

Cavity Resonator Design

This chapter describes the design of the NIST helically wire-wound cylindrical cavity resonator. The resonator is designed to give the best dimensional stability and repeatability possible. Detailed machine drawings are given in Appendix C. The cylindrically shaped cavity is nominally 60.0 mm in diameter. The resonator length varies from 408.5 mm to 433.3 mm. The inner wall of the cylindrical waveguide consists of two insulated copper wires helically wound, side-by-side. Flanges connect to the top and bottom of the cylindrical waveguide. The top flange of the resonator holds a replaceable coupling endplate. The bottom flange connects to a retractable tuner assembly. This tuner assembly contains an optically flat endplate on three hardened steel rods that each run along a pair of instrument-grade linear ball bearings. This lower endplate moves up and down inside the cavity by a motor-driven micrometer. A capacitive sensing micrometer detect resonator length by pressing against a yoke that ties the three travelling rods together.

The entire tuner assembly disconnects and lowers away from the cavity flange. In this way, disk-shaped samples can be placed onto the tuner endplate, then raised up into the cavity. Surrounding the cylindrical waveguide is a water jacket for temperature control. The water jacket is sealed at the top and bottom flanges with O-rings. The base of the sliding tuner assembly is also temperature controlled. Water circulates from a water bath to the bottom flange then into the water jacket surrounding the helical windings and back out to the water bath. Temperature is nominally controlled to within $\pm 0.1°C$.

3.1 Helically Wound Circular Waveguide

The cylindrical waveguide section is 430.8 mm long. The cylindrical wall of the cavity resonator is made from two insulated, helically wound copper wires. The windings consist of two 0.165-mm diameter (0.0065 in) wires wound side-by-side. The wires are embedded in an epoxy and surrounded by a 2-mm thick fiberglass sheath. This sheath is then epoxied into a steel pipe. To prepare the waveguide section for the resonator, the waveguide's steel outer wall was made concentric with the inner wall by slipping a cylindrical mandrel into the waveguide and turning the outer wall of the waveguide in a lathe. Shrink tubing was then placed over the waveguide to act as a seal from the water jacket. Finally, the ends of the waveguide were cut to give a good cylindrical geometry.

The helical waveguide allows currents to flow in the circumferential direction and prevents currents from flowing in the axial direction. As such, the waveguide acts as a mode-filter that allows TE_{01} modes to propagate, while attenuating other modes. There are over 1170 possible TE and TM modes below 18 GHz for a cylindrical cavity measuring 60 mm in diameter and 430 mm in length. The mode-filtering characteristics of this resonator allows us to readily identify the TE_{01} modes and alleviates problems caused by mode-degeneracy. As we will see in Chapter 4, TE_{02} and TE_{12} modes are weakly excited and can cause interference.

3.2 Coupling Flange

The resonator's top flange slides onto and butts up against the end of the helix waveguide. Figures C.4 through C.8 display this piece. The part that fits against the waveguide has a 60-mm diameter hole. The flange is machined to fit very concentric and perpendicular to the longitudinal axis of the cylindrical waveguide. Opposite the recessed side that butts against the helical waveguide is another recessed surface which accommodates the coupling endplate. Proper machining of these two surfaces assures good cylindrical resonator geometry. A threaded hole is placed in the flange to accommodate a temperature probe. The temperature probe slides vertically into the water cavity jacket, through the coupling flange. The knurled temperature-probe bolt shown in Fig. C.19 presses a small O-ring against the temperature probe and the coupling flange to seal off the water jacket. Three other threaded holes placed in the outer radius of the flange hold steel rods that tie to both the top and bottom flanges, which hold the entire cavity together.

3.3 Apertured Coupling Endplate

The apertured coupling endplate which slides into the coupling flange is shown in Fig. C.18. The coupling endplate is tightened to the coupling flange with a pressure ring, shown in Fig. C.19. The inner side of the coupling endplate is polished to optical flatness. The outer side of the coupling plate contains precise waveguide flange-bolt holes and two short (10.795 mm) precision waveguide sections which continue from the flange-bolt holes down to the coupling slits. The two rectangular slits couple energy into and out of the cavity. The coupling slits are located to optimally couple with the TE_{01} mode. This is done by locating the slits at the point where the Bessel function J_0' is a maximum for the first zero.

Presently, the cavity's coupling endplate is equipped with X-band flanges and is designed for nominal minus 30 dB resonance transmission at 10 GHz. The slits measure 5.080 by 2.286 mm (0.2 by 0.09 in). At lower frequencies, the apertures couple less energy into the cavity because of their reduced size relative to the waveguide wavelength. Conversely, energy couples more strongly into the resonator at higher frequencies. Weak coupling into the cavity is most ideal in order to maximize cavity Q, and to minimize Q-value and frequency corrections. Coupling could be made weaker by reducing the aperture sizes, but this would make the transmission response at lower frequencies too weak to be measured by a network analyzer.

3.4 Tuner Flange

The tuner flange shown in Figs. C.9, C.10, C.11, C.12 and C.13 connects the helical waveguide and water jacket to the tuner assembly. The helix waveguide fits into a recess in the flange similar to the coupling flange. Two O-rings in the flange press against the waveguide and the acrylic plastic tube to seal the water jacket. Three threaded holes attach to tie rods that pull the coupling flange and tuner flange against the helix waveguide. The entire cavity is supported from the tuner flange by three 12.7-mm ($\frac{1}{2}$-in) diameter rods that rest in recessed holes. A vertical hole near the edge accommodates the measurement micrometer, which extends down to press against a measurement yoke connected to the moveable tuner endplate. The micrometer is fastened to the tuner flange by a lateral screw. There are three U-shaped recesses at the edge of the flange. These are for the tuner assembly latches. A protruding ring in the tuner flange lines up with the tuner assembly, and assures good alignment. Outside of this ring is a recession in the tuner flange to catch dust and other contaminants.

This improves length repeatability when the tuner base is lowered and then reattached. Finally, a sample centering ring is machined into the tuner flange. This ring has a chamfered corner that helps to center a poorly aligned sample when it is raised up into the cavity, which prevents damage to the tuner.

3.5 Tuning Endplate Assembly

The tuner assembly is designed to position the endplate inside the cavity with utmost precision. This is accomplished by using a motor-driven micrometer to move the endplate assembly up and down combined with a 0.1-μm precise measurement probe connected to the cavity's tuner flange. The measurement probe pushes against a yoke extending out from the endplate's guide shafts. The resonance frequencies can be tuned by varying the effective length of the cavity.

3.5.1 Sliding Base

The base which holds all the parts of the tuner assembly is shown in Figs. C.3, C.14, C.15, C.16 and C.17. This base has inserted PVC bushings (Fig. C.21) which allow the base to easily slide up and down on the cavity-stand rods. Three instrument grade linear ball bearings are press fit into this tuner base. The tuner endplate guide shaft rods run along these bearings. The three latches which connect to the tuner flange swing up from the edge of the base. Each latch has a torque nut (Fig. C.21) that allows exact pressure to be placed on the latch. The tuner base does not make contact everywhere around the flange. Instead, raised surfaces on either side of each latch help ensure length repeatability by preventing contamination. The micrometer which drives the tuner-endplate is fastened to the tuner base with a replaceable cup shown in Fig. C.21. This cup is machined to fit the particular drive micrometer, and if a different micrometer is used, one can design a different cup. The micrometer cup fits into a hole with a recessed lip in the center of the tuner base. Finally, for temperature control, the tuner base has a channel machined into it which allows water to circulate.

3.5.2 Sliding Base Extension

An extension to the sliding base shown in Figs. C.3 and C.22 contains three more press fit linear ball bearings, which help the alignment of the tuner endplate guide shafts. The extension butts against the sliding base and is attached by knurled nuts that screw onto the sliding PVC bushings.

3.5.3 Tuner Endplate Assembly

The tuner endplate assembly consists of an optically polished endplate, three hardened-steel guide shafts and a micrometer reference yoke. These and other parts for the tuner base assembly are shown in Figs. C.20, C.21, C.22 and C.23. A flat hardened-steel plug, press-fit into the center of the endplate rests against the drive micrometer. The guide shafts are attached to the tuner endplate with set screws. The guide shafts are stabilized linear bearings in the tuner base and base extension, and by the micrometer reference yoke. This yoke has fingers which extend radially outwards. The measurement micrometer attached to the tuner flange presses against one of these fingers to give a differential length measurement. The three guide shafts extend further through the tuner base's linear ball bearings, and then through the linear ball bearings in the tuner-base extension. Between the tuner base and the tuner-base extension, the guide shafts are spring loaded. The combination of yoke, linear ball bearings and spring loading helps assure alignment stability of the tuner endplate.

Chapter 4

Resonator Evaluation

We have quantified the cavity resonator's dimensional and electrical performance to ensure accurate calculations of permittivity and loss factor in dielectric samples. The uncertainty in resonator parameters such as cavity diameter and length, resonance frequency, and Q have been determined in order to calculate overall measurement uncertainty.

Knowledge of the behavior and performance of the resonator allowed us to develop several possible corrections that enhance measurement accuracy. These corrections, which adjust for resonator losses and geometric deviations are included in this chapter. Even without corrections, the NIST resonator's permittivity measurement yield excellent agreement with previously published results, as demonstrated by the 1723 glass results given in Sec. 8.1 [27].

Resonator characterization also allows us to determine when the resonator's performance has degraded. In this respect control charts have been developed to help track the resonator's behavior.

The early portions of this chapter describe the results of cavity dimension experiments. In particular precision in the diameter and length is shown, and the tuner-endplate assembly's travel accuracy and attach repeatability are evaluated. We then discuss the effects on cavity dimension calculation caused by coupling aperture perturbations and resistive wall losses. Later portions deal with the evaluation of resonance quality factor Q. Methods for determining Q and their associated uncertainties are discussed. Loss mechanisms are then separated into parts in order to deduce sample loss.

4.1 Dimensional Characterization

This section presents results from experiments that determined the accuracy for resonator diameter and length. In a geometric sense, the resonator diameter is fixed, and the length changes as the tuning endplate moves up and down inside the cavity. In an electrical sense this is only partly true. The electrical skin depth penetration varies with frequency. Presently, no work has been performed to study the effective penetration depths into the helical wall and endplates. Cavity dimensions are determined from resonance spectra, and therefore should be considered the "electrical dimensions" of the resonator.

The change in position of the tuning endplate is measured with a sensing micrometer. The accuracy of this sensing micrometer is verified. The assembly that holds the tuning endplate can detach from the resonator. We investigated both the accuracy and repeatability of the length measurement assembly and tuner assembly, and the effective electrical diameter and length of the resonator chamber under varying conditions.

4.1.1 Effective Electrical Length and Diameter

The resonator's effective electrical diameter and length are found through linear regression from the distribution of the resonant-mode spectrum as described in Sec. 5.2.4. For example, if we know the resonant frequencies and axial mode numbers p of at least three TE_{01p} modes, we can estimate resonator length and diameter. If we use more than three modes in calculating length and diameter, their precision estimates decrease. The frequencies of the resonant modes can be very accurately determined to ±500 Hz, which allows accurate determination of the resonant cavity's dimensions. However, as we will show, various physical phenomena can pull resonant frequencies up or down, which results in different estimates of the cavity dimensions, depending on which modes are used in the linear regression.

Figure 4.1 shows typical length versus diameter results which demonstrate the frequency dependence of cavity dimensions calculation. Cavity length and diameter were calculated from subsets of the 7–14 GHz TE_{01p} resonance spectrum. Each subset consisted of five adjacent resonant frequencies, and began with the lower frequencies and moved upward ($TE_{01(10-14)}$, $TE_{01(14-18)}$, ..., $TE_{01(33-37)}$). Cavity length and diameter calculated for each subset are slightly different, and there is a systematic trend in which calculated diameter decreases and length increases as the higher frequency modes are used to calculate cavity dimensions. Also, the calculated cavity dimensions crosses back onto itself before the increas-

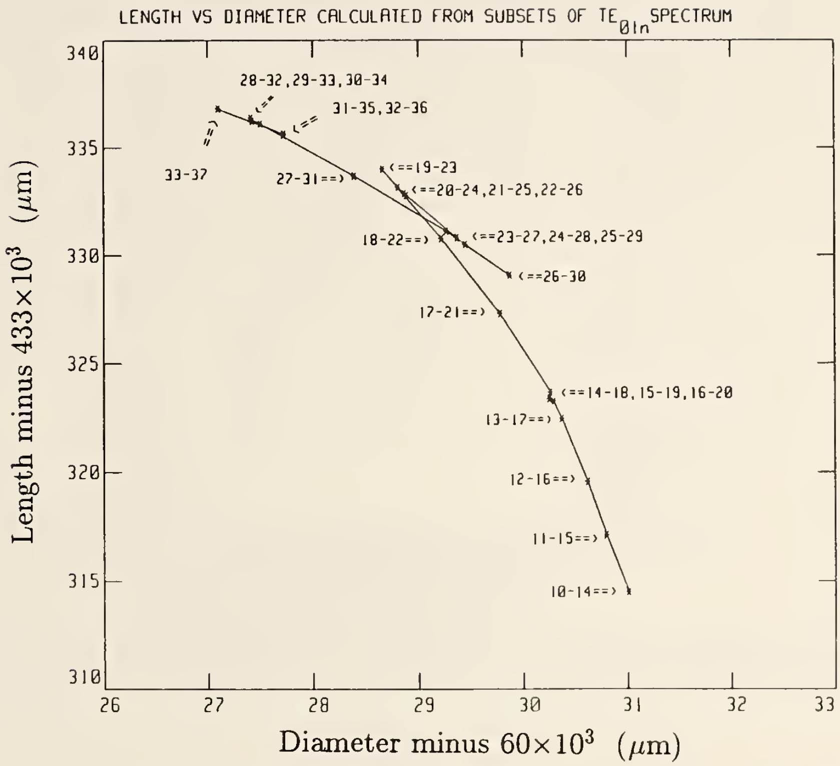

Figure 4.1: Cavity length versus diameter calculated from subsets of the 7–14 GHz resonance spectrum, $TE_{01(10)}$–$TE_{01(37)}$. Subsets consist of five adjacent modes ($TE_{01(10-14)}$, $TE_{01(14-18)}$, ..., $TE_{01(33-37)}$).

ing length/decreasing diameter trend continues. This is most likely to be caused by mechanisms that are not smoothly varying with frequency, mode interference for example. The systematic behavior demonstrated in Fig. 4.1 indicates that one or more frequency-dependent mechanisms are affecting our cavity dimension calculation. We will discuss in Sec. 4.1.4 and Sec. 4.1.5 possible causes for this behavior, and Sec. 4.1.4 demonstrates how coupling-iris perturbations reduce cavity dimension uncertainties.

By looking at the range of length and diameter values calculated from the entire 7 to 14 GHz resonance spectrum, we conclude from Fig. 4.1 that the uncertainty in our length determination is less than or equal to $\Delta L = \pm 0.012$ mm, and that diameter can be determined within $\Delta D = \pm 0.003$ mm. These numbers also correspond to approximately 1.5 standard deviations estimated by least-squares regression of all X-band TE_{01p} frequencies. Chapter 9 will demonstrate that these estimated length and diameter uncertainties are significant contributors to the total permittivity uncertainty. The length and diameter uncertainties

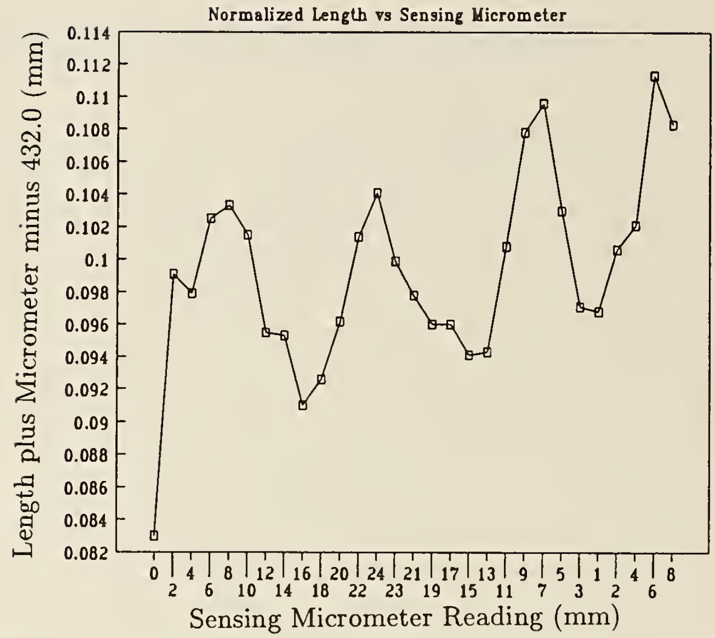

Figure 4.2: Difference between micrometer-measured length and length calculated from X-band resonance spectra.

can be reduced if we limit our measurements to a narrower frequency range. For example, the range of cavity dimensions calculated from subsets of the $TE_{01(20)}$ to $TE_{01(29)}$ modes (approximately 9 to 11 GHz) is about one third smaller than the range of values calculated for 7 to 14 GHz subsets. This fact allows us to calculate cavity length and diameter from the $TE_{01(20)}$ to $TE_{01(29)}$ modes that gives $\Delta L = \pm 0.004$ mm and $\Delta D = \pm 0.001$ mm.

4.1.2 Sensing Micrometer Accuracy

When the fixed-frequency method is employed to measure permittivity, the tuner endplate must be adjusted to shorten the cavity when the sample is in place. The change in cavity length determines ϵ'_R, and the sensing-micrometer measurement accuracy is therefore critical. To assess the accuracy of the sensing micrometer, we measured the TE_{01} X-band resonance spectra at different cavity lengths in 1 mm increments over the tuner endplate's 24-mm range of travel. Cavity dimensions were calculated from the X-band resonance spectra at these different cavity lengths. Figure 4.2 shows the difference between the length measured by the micrometer and the resonator length calculated from the resonant frequencies. Over the entire 24-mm endplate travel, the sensing micrometer reading and the calculated cavity length agree to within ± 0.011 mm.

Figure 4.3 shows how the calculated cavity diameter changes by approximately

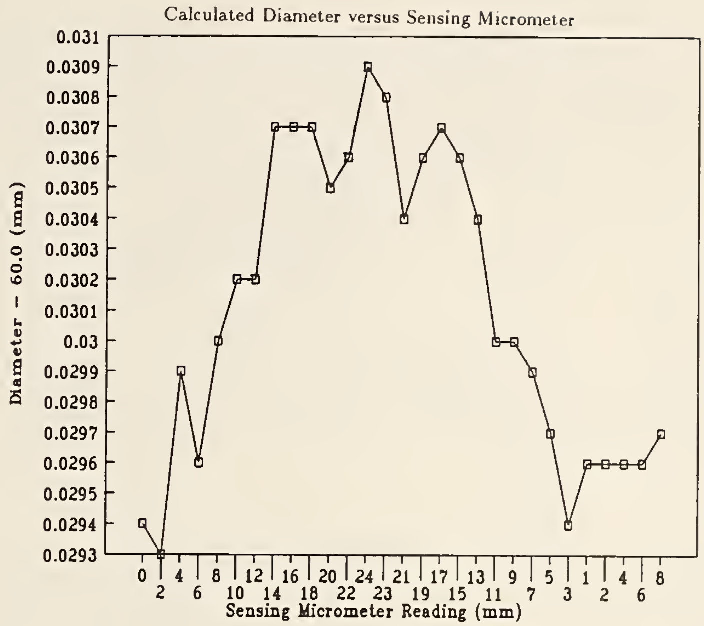

Figure 4.3: Calculated diameter versus change in length as measured by the sensing micrometer. A diameter uncertainty of ±0.002 mm is found from the range of results.

±0.002 mm as the cavity length changed by 24 mm. Because length changes when the fixed- frequency method is used, this result determines the diameter uncertainty that should be used in calculating permittivity uncertainty. Because frequency changes when the fixed-length method is used, the same diameter uncertainty of ±0.002 mm was found from the range of diameter results calculated from TE_{01p} subsets shown in Fig. 4.1.

A similar experiment which was performed two years earlier indicates that some deterioration of the length-measurement system has taken place. In this experiment cavity dimensions were calculated for seven different cavity lengths from five-frequency subsets of the 7–14 GHz resonance spectra, as explained for Fig. 4.1. The sensing micrometer was used to change the cavity length, and variations in calculated cavity length give an indication of the length-measurement accuracy. Figure 4.4 shows how cavity dimensions calculated from different parts of the resonance spectrum vary as the cavity length is changed. The difference between dimension results calculated from identical subsets shows a length variation less than ±0.004 mm and a diameter variation of less than ±0.001 mm.

The resonator has been opened and closed several hundred times during those two years, and the sensing-micrometer end created a very small pit in the aluminum yoke that is part of the tuner-endplate assembly. If the micrometer had not been settled into the pit during the 0-mm measurements, the anomalous 0-

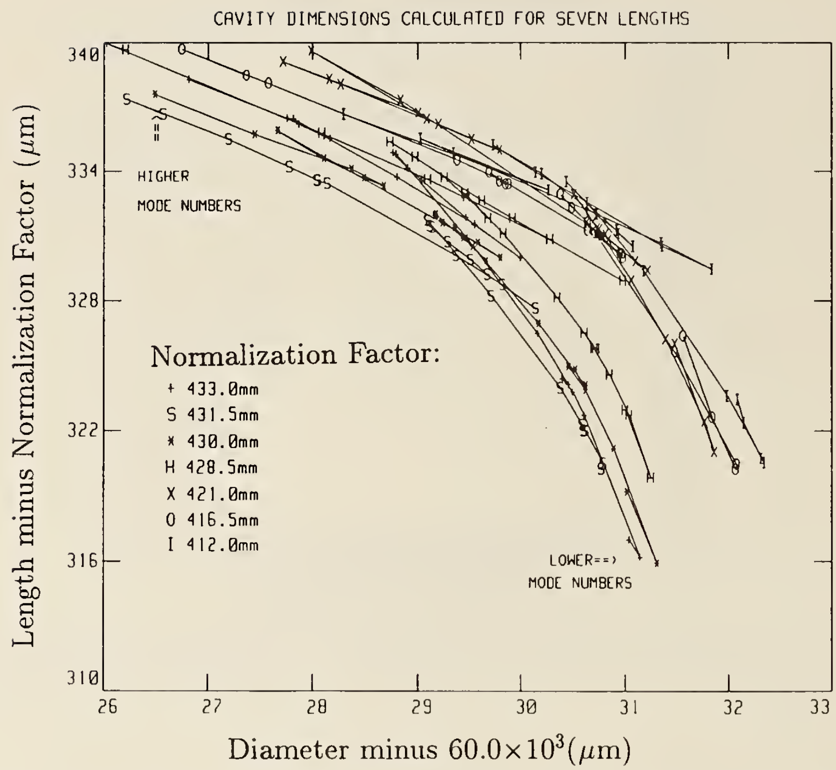

Figure 4.4: Length versus diameter calculated from five-frequency subsets for seven different cavity lengths. Displacement of curves from one another indicates sensing micrometer accuracy.

mm result shown in Fig. 4.2 could occur. A temporary solution to this problem has been to rotate the endplate so that another one of the yoke's three fingers, shown in Fig. C.23, presses against the sensing micrometer. A permanent solution to this problem is to press a hardened steel plug into the aluminum yoke to serve as the reference surface for the micrometer. Meanwhile the sensing micrometer accuracy is given to be ±0.011 mm. We shall see in Chapter 9 that this contributor to length uncertainty is a major contributor to permittivity uncertainty.

4.1.3 Tuner Assembly Attachment Repeatability

To determine whether the small pit in the tuner-endplate yoke affects the tuner assembly attachment repeatability, an experiment was performed in which the tuner assembly was detached and then reattached to the resonator 23 times. Each time the tuner assembly was detached, the tuner endplate was lowered,

Table 4.1: Repeatability of tuner assembly connection.

Mean micrometer reading (mm)			
0.000 00	-0.001 00	-0.000 10	
0.000 00	-0.000 90	-0.000 80	0.001 22
0.001 15	0.000 20	0.000 72	0.000 25
0.001 07	0.001 17	0.001 60	0.000 70
0.000 90	0.001 35	0.000 40	-0.000 95
0.000 17	0.001 43	-0.000 22	-0.001 25
Average ± standard error: 0.000 31 ± 0.000 87			

then raised until the $TE_{01(25)}$ mode resonated as close to 10.63 GHz as possible. The micrometer reading was then recorded. Results are given in Table 4.1. The standard error of the 23 micrometer readings is 0.000 9 mm, and the 95% certainty interval is ±0.001 8 mm. This demonstrates excellent tuner assembly attachment repeatability in which the cavity length can be repeated to within 2 μm. This means that the pit in the tuner-endplate yoke does not affect length repeatability when the length is held fixed. Only when the length varies do we see the discrepancy in length measurements. These results are in agreement with an experiment performed two years previously in which the cavity's tuner base was reattached three times. Cavity dimensions were calculated from the five-mode subsets described for Fig. 4.1 and results are shown in Fig. 4.5. For cavity dimensions calculated from identical subsets, length is repeatable to within ±0.002 mm and diameter is repeatable to within ±0.0003 mm.

4.1.4 Iris Perturbation

The coupling irises cause two perturbing effects on cavity dimension calculations. First, power loss through the coupling irises shifts the resonance frequencies downward. Second, field distortion near the coupling irises also causes resonance frequencies to shift downward. Both of these perturbations are frequency dependent and increase in magnitude as frequency increases. We quantified both of these perturbations under typical resonator operating conditions.

Power loss through irises

The radiated power lost from the cavity through the coupling irises is analogous to an added resistance in the coupling loop of a resonant circuit, the result

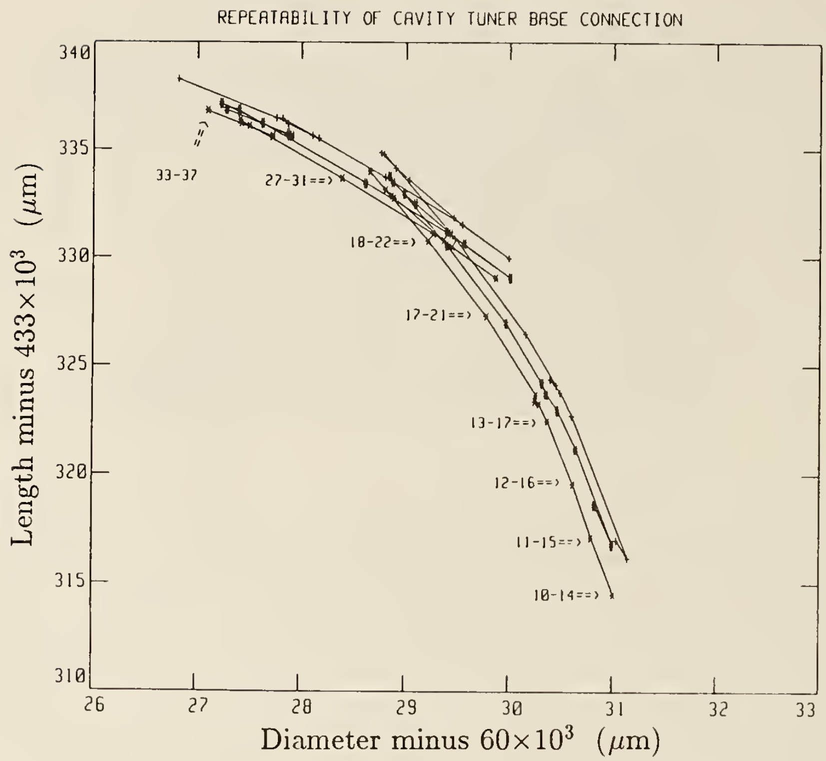

Figure 4.5: Length versus diameter results showing cavity dimension repeatability for three tuner-base detach/reattachments. Cavity dimensions calculated from subsets of five adjacent TE_{01p} modes. Also note the repeatability of the crossing pattern that occurs for the middle subsets.

being that the resistance lowers the resonant frequency and Q. Gallop and Radcliffe [28] show that the insertion loss I and unloaded Q_0 can be used to estimate the frequency shift due to power lost through the coupling apertures. The frequency shift due to coupling aperture loss is given by

$$\frac{\Delta f}{f_0} = -\frac{\sqrt{I}}{4Q_0} \; . \tag{4.1}$$

At 10.63 GHz, the insertion loss is minus 25.4 dB and the unloaded Q_0 is approximately 84 000. This results in a frequency shift of approximately minus 1.7 kHz. The effect of this perturbation is much less than the field-distortion perturbation described next. We have made no attempt to make this correction to the measured resonance frequencies. Nonetheless, the generalized method for correction is to measure the insertion loss at several frequencies in the resonator's frequency range, calculate the theoretical frequency shift for each resonance frequency, and plot a chart of frequency shift versus resonance frequency. Then, for any given resonance frequency, we may refer to the chart to determine the appropriate frequency shift to be applied.

Field distortion

The electromagnetic field bulges into the coupling irises and distorts the ideal cylindrical field pattern inside the resonator. The resulting field distortion increases the effective volume of the cavity, and therefore lowers the resonant frequencies. Because the distortion occurs at the coupling endplate, we expect the cavity length calculation to be affected. Perturbation theory given by Slater [29] and Waldron [10] can be used to estimate the frequency shift due to field distortion at the coupling irises. In general, perturbation theory can be applied when the change in volume is much less than the total volume ($\Delta V \ll V$). The frequency shift due to field distortion near the coupling irises can be written as

$$\frac{\Delta f}{f} = \frac{\int_{\Delta V} (\mu H^2 - \epsilon E^2)\, dV}{\int_V (\mu H^2 + \epsilon E^2)\, dV} , \tag{4.2}$$

where in this case $\mu = \mu_0$, $\epsilon = \epsilon_0$. E and H are the electric and magnetic fields inside the cavity. The integral in the numerator is over the volume added to the cavity by the coupling irises, and the integral in the denominator is over the entire cavity volume. The volume ΔV added to the cavity by the coupling irises is difficult to estimate because the depth of field penetration in the coupling iris is frequency dependent. Use of (4.2) to estimate the frequency shift will be prone to errors in most cases. Therefore we used an experimental approach to estimate the magnitude of the frequency shift due to the coupling irises. We plugged one iris and measured the frequency shift of the TE_{01p} modes at the other port. Results for both coupling irises are given in Fig. 4.6. The frequency shift due to field distortion into the coupling irises increases as frequency increases because field penetration into the irises increases as wavelength decreases.

Iris perturbation corrections

The tendency for calculated length to increase and diameter to decrease as frequency increases as seen in Figs. 4.1, 4.4, and 4.5 can be explained by field pattern distortion near the coupling irises and to a lesser degree, by radiated power loss through the coupling irises. The coupling irises are located in a cavity endplate, and the field distortion in the vicinity of the coupling endplate will make the calculated resonator length larger than its actual physical length. The magnitude of the perturbation increases as frequency increases, so we expect to see an increase in calculated cavity length at higher frequencies.

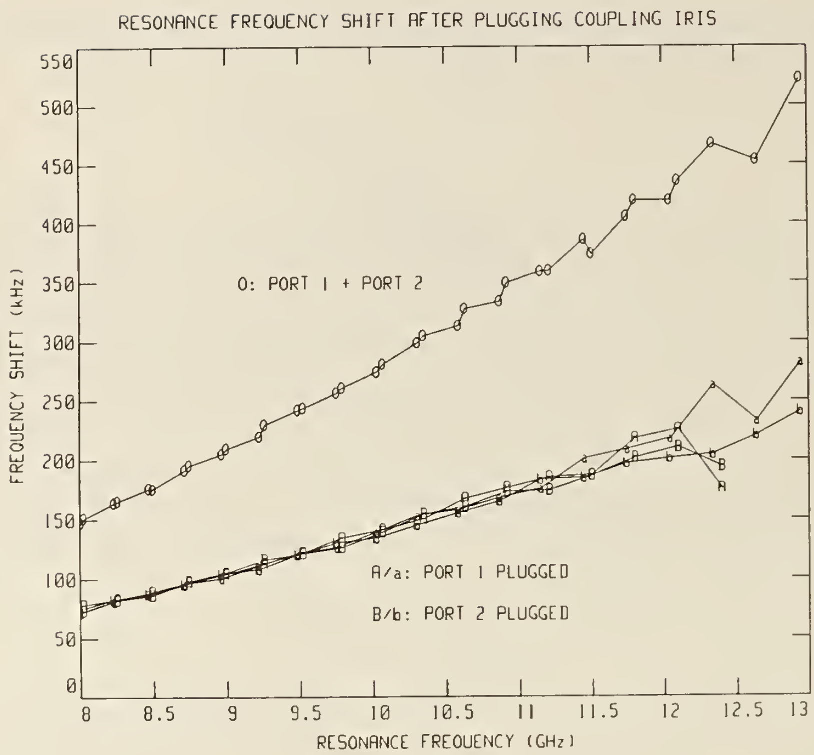

Figure 4.6: Measured resonant frequency shifts when coupling irises are plugged relative to measured resonant frequencies when both ports are unplugged.

Correction for the iris perturbation can be made by adding the frequency shifts given in Fig. 4.6 to the measured resonant frequencies. Figure 4.7 demonstrates the result of this correction. For a given length, the cavity's length and diameter are calculated from subsets of five adjacent modes from the 8–13 GHz TE_{01p} resonance spectrum. Estimated length and diameter uncertainties that come from the linear regression are also shown. The uncorrected cavity dimensions vary with frequency by more than the precision estimates, indicative of a systematic error. When the iris-perturbation frequency correction is applied cavity length decreases and diameter increases slightly, as expected. Most importantly, when the correction is applied, the systematic relation between length and diameter is reduced, and the scatter of cavity dimension results is in general agreement with precision estimates. This finding is important for demonstrating that the frequency dependence of the cavity dimension calculation can be largely eliminated by applying the frequency shift corrections from Fig. 4.6. Without the iris-perturbation frequency correction, cavity dimensions would have to be recalculated at each frequency, and the data reduction process would be slowed down. With the frequency correction, Fig. 4.7 shows that the range of calculated

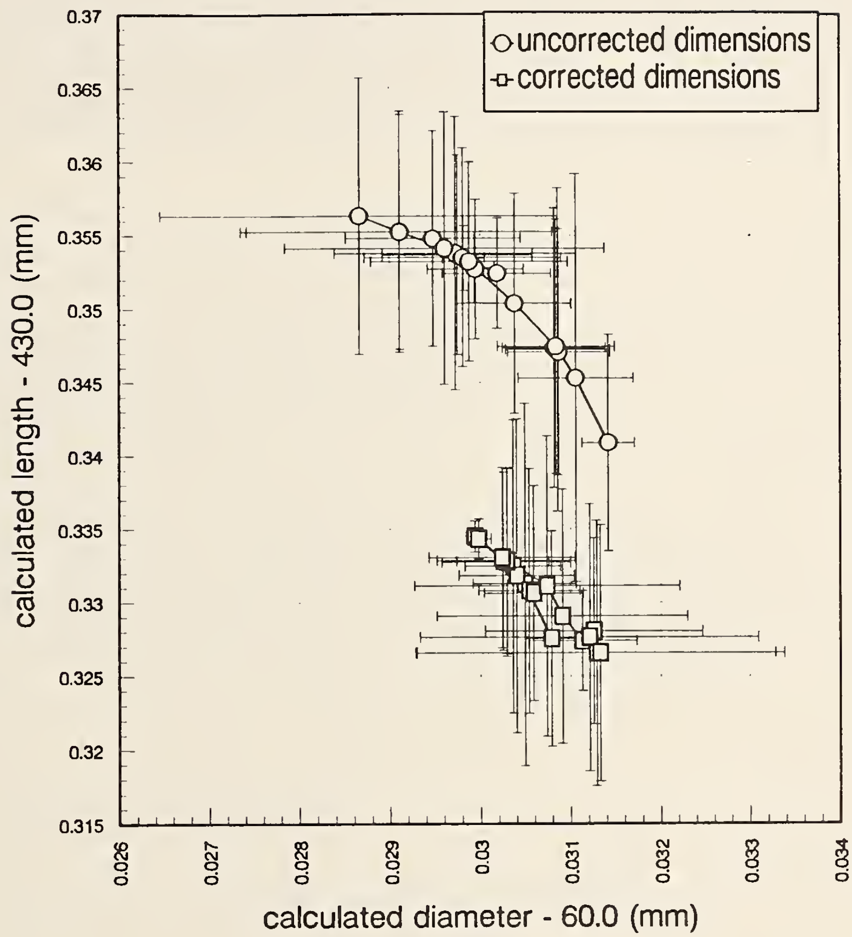

Figure 4.7: Cavity length versus diameter calculated from an uncorrected resonant frequency spectrum, and from the same frequency spectrum corrected by the measured iris perturbation shown in Fig. 4.6. Cavity dimensions are calculated from subsets of five adjacent resonant frequencies. Length and diameter uncertainty estimates for each calculation are also shown.

cavity length values improves from approximately ±0.009 mm to ±0.005 mm, and the calculated cavity diameter range slightly improves from approximately ±0.0013 mm to ±0.0010 mm.

4.1.5 Resistive Loss Perturbation

Slater p. 70 [29], as well as many other authors[10][30][31] apply perturbation theory to find the first-order frequency correction due to radiation skin-depth penetration into the cavity walls (a form of resistive loss). Skin-depth penetration makes the effective cavity length and diameter larger by approximately one skin depth at all cavity walls [25, 26]. This effect is analogous to resistive damping R in an LRC circuit that lowers the resonant frequency by approximately one-half of the half-power bandwidth,

$$\Delta f = \frac{f_0}{2Q_0} . \tag{4.3}$$

Therefore, the proper correction is to add this amount to the measured resonant frequency. By using corrected resonant frequencies, we calculate the physical dimensions of the cavity. Without the correction we calculate the effective electrical dimensions of the cavity.

Use of the cavity's physical dimensions is important so that the same simple correction can be made when the dielectric sample is measured. While the physical dimensions remain constant, the electrical dimensions of the cavity are frequency dependent, and change because of coupling iris perturbations and wall losses. Additionally, when power is absorbed by the sample the resonant frequency is further reduced and the half-power bandwidth is broadened. By correcting the empty-cavity resonant frequencies to calculate the cavity's physical dimensions, we can apply the same frequency-shift correction given in (4.3) to the sample-loaded cavity to calculate ϵ'_R. If this correction were not applied, the frequency shift caused by the sample in the cavity would be slightly too large because the sample's power absorption had further pulled the frequency down. This increased frequency shift biases permittivity results making them somewhat high, on the order of 0.5%, depending on the sample permittivity and thickness. This correction becomes especially necessary when measuring medium-loss, low-permittivity samples, because the magnitude of the correction becomes more significant relative to the frequency shift caused by the sample.

4.1.6 Summary of Diameter and Length Uncertainties

When the fixed-length method is used to measure permittivity, length uncertainty is primarily affected by frequency-dependent perturbations caused by the coupling irises. Figure 4.7 shows the ± 0.009 mm length uncertainty can be improved to ± 0.005 mm when the iris perturbation correction is applied.

Table 4.2: Overall cavity dimension uncertainties.

Parameter uncertainty	Measurement method	With corrections	Without corrections
Δ Length (mm)	Fixed length	0.0054	0.0092
Δ Length (mm)	Fixed frequency	0.0054	0.0112
Δ Diameter (mm)	Either	0.0010	0.0013

If the fixed-frequency method is applied, length uncertainty depends on micrometer accuracy, which is approximately ±0.011 mm over the micrometer's entire 24-mm travel range. The sensing micrometer's accuracy was previously shown to be approximately ±0.005 mm. The resulting change indicates that some wear occurred in the length measurement assembly.

The attachment repeatability of the tuner assembly should also be figured into the overall length uncertainty when either method is used. The attachment repeatability is less than ±0.002 mm. This repeatability is taken to be independent of the frequency- and length-dependent accuracy estimates given above, and the overall length uncertainty is calculated as the root sum squares of dimensional calculation accuracy and tuner assembly repeatability, as shown in Table 4.2.

4.2 Evaluation of Q

Resonance quality factor is proportional to the energy stored in the resonating system W divided by the average power loss of the total system

$$Q = \frac{\omega_0 W}{P} \; . \tag{4.4}$$

Power is consumed by several mechanisms including conductor losses P_C (endplates and helical wall), dielectric sample losses P_S and coupling losses P_E. Conductive and dielectric losses are considered internal that define the unloaded Q_0 of the resonator. Losses caused by the iris coupling ports are considered external losses. Each one of these loss mechanisms has a characteristic Q. The sum of the reciprocals of all Q values "loaded" Q_L value:

$$\frac{1}{Q_L} = \frac{P_C + P_S + \sum_i P_{Ei}}{\omega_0 W}$$

$$
\begin{aligned}
&= \frac{1}{Q_S} + \frac{1}{Q_C} + \sum_i \frac{1}{Q_{Ei}} \\
&= \frac{1}{Q_0} + \sum_i \frac{1}{Q_{Ei}} ,
\end{aligned}
\tag{4.5}
$$

where the sum is over all coupling ports. Loaded Q_L is the quantity measured.

To calculate dielectric loss of a sample, we take the measured "loaded" Q_L and remove conductor and aperture losses to find Q_S. The aperture coupling losses are defined by Q_{Ei} (see (2.71), p. 28, and Sec. 2.2.5, p. 33). A common approach for finding Q_S is to subtract the empty-cavity Q_L from the value of Q_L of the cavity with sample. For our cavity, this approach is usually sufficient if we measure samples with loss tangents greater than 0.001. When measuring very low-loss samples, however, slight differences in cavity wall and port losses between the empty cavity and the cavity with sample can lead to errors that are of the same order as the sample's loss tangent. For example, Fig. 4.8 shows how Q_L increases as frequency increases up to approximately 11.7 GHz where the TE_{021} mode interferes with the TE_{01p} modes we wish to observe. When a sample is placed in the cavity, the resonant frequency will be lower unless the cavity is shortened. Figure 4.8 shows that if the resonant frequency lowers, Q_L will also lower even if the sample were completely lossless. The scatter in Q_L measured at or near the same frequency for different cavity lengths is smaller than the change in Q_L when frequency shifts by a typical 200–300 MHz. Because Q_L remains more constant with changing length than with changing frequency, we can understand that errors in loss tangent calculation are minimized if the cavity with sample is shortened so that it resonates at the same frequency as the empty cavity.

4.2.1 Port Losses - Coupling Factor (κ)

If the variable-frequency measurement method is used the largest correction to the measured Q_L is for power lost through the coupling ports. The Q_{E1} and Q_{E2} of the coupling ports can be determined from the coupling factors (κ_1, κ_2), so that the internal Q_0 can be determined from (4.5). We simultaneously determine κ_1 and κ_2 from measured scattering parameter data obtained from a network analyzer. The reflected scattering parameters, S_{11} and S_{22}, are measured. The locus of resonance S-parameter data is a circle in the complex reflection plane, and the radius of this "Q-circle" depends on the coupling factors. If we measure the S_{11} and S_{22} resonance responses, κ_1 and κ_2 can be solved simultaneously

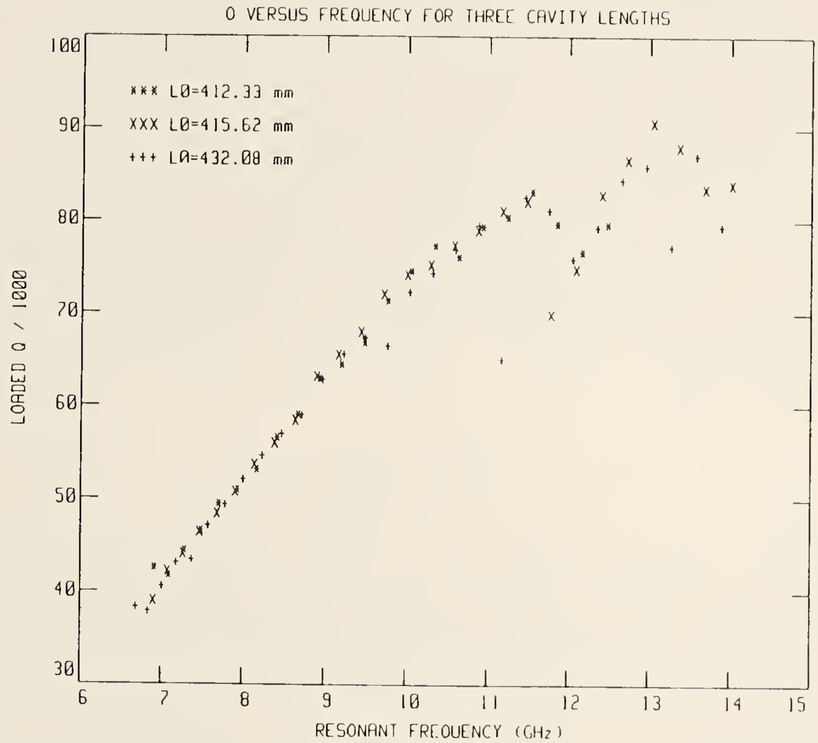

Figure 4.8: Measured Q_L versus frequency for three cavity lengths. Data show increasing Q and mode interference above 11 GHz.

from the equation [23]

$$\kappa_i = \frac{r_i}{1 - r_1 - r_2} \quad i = 1 \text{ or } 2 \ , \tag{4.6}$$

where r_i are the radii of the Q-circles in the complex reflection plane. From (2.96), the coupling factors and Q are related by

$$\begin{aligned} \frac{1}{Q_L} &= \frac{1}{Q_0}(1 + \kappa_1 + \kappa_2) \ , \\ &= \frac{1}{Q_0} + \frac{1}{Q_{E1}} + \frac{1}{Q_{E2}} \ . \end{aligned} \tag{4.7}$$

4.2.2 Q Corrections: Fixed-Length Method

If we employ the fixed-length method, the resonant frequency decreases when the sample is inserted. When frequency decreases, coupling port losses decrease and internal conductor losses increase slightly. To compensate for the coupling port losses, the coupling Q_E must be determined at both the empty-cavity and the cavity-with-sample resonant frequencies, then the internal (unloaded) Q_0 can be separated from the measured (loaded) Q_L, as shown in (4.5). This correction for changes in coupling port losses improves loss factor accuracy by approximately 15%. When frequency changes, internal conductor losses change slightly, and a correction to Q_0 can be made to obtain the best accuracy possible. To correct for changes in internal conductor loss, the skin depths at both resonant frequencies must be known, and the change in Q_0 has to be calculated from the volume and surface integrals that comprise the numerator and denominator of (4.4).

Power absorbed by the resonator's conductive walls is proportional to skin depth δ_s given in (2.107) as

$$\delta_s = \sqrt{\frac{2}{\omega\sigma\mu}} \,. \tag{4.8}$$

Equation (4.8) shows that a one-percent decrease in frequency leads to a half-percent increase in skin depth. This leads to a half-percent increase in conductor loss and, from (4.5), a half-percent decrease in Q_0. For example, with the fixed-length method, frequency decreases in our resonator by approximately 300 MHz when a half-wavelength thick sample is inserted. At 10 GHz this would lead to a 1.5% increase in conductor loss when the sample is in place. To correct for this change, the empty-cavity Q_0 should be *decreased* by 1.5% in order to approximate the conductor loss in the sample-loaded cavity.

4.2.3 Q corrections: Fixed-Frequency Method

Fewer corrections to Q are needed if the fixed- frequency method is used to measure a dielectric material. No corrections to the coupling Q_E are needed because the iris coupling factors do not change. At a fixed frequency, skin depth does not change and conductor loss in the resonator's air-filled section remains constant. A resonator containing a dielectric sample must be shortened in order to resonate at the empty-cavity frequency. The shortened length of the cavity leads to less conductor loss. This decrease in resonator wall loss is determined from surface integral equations given in Chapter 2. The calculation is time consuming and results in only a minor correction to Q_0 on the order of 0.5–2%. Moreover, the uncertainties in the endplate and helical wall skin depths leads to significant uncertainty in the correction. For this reason corrections to Q because of reduced wall losses becomes undesirable when several samples or measurement frequencies are involved.

Table 4.3: Stability of Q_0 and coupling factors. Precision is given as the standard error of 10 measurements.

Measurement Numbers		Mean $Q_0 \pm$ precision	Mean Coupling Factor $(\kappa) \pm$ precision	Micrometer Reading (mm)
1–10	port 1	$84\,633 \pm 33$	$0.028\,457 \pm 0.000\,012$	0.018 08
	port 2	$84\,650 \pm 44$	$0.028\,110 \pm 0.000\,010$	
11–20	port 1	$84\,676 \pm 42$	$0.028\,4601 \pm 0.000\,019$	0.018 10
	port 2	$84\,648 \pm 63$	$0.028\,111 \pm 0.000\,013$	

4.2.4 Stability of Q and Coupling Factors

To perform a precise measurement of a sample's loss factor, the resonator's Q and coupling factors must remain stable throughout the time required to make a measurement, usually less than one hour. In this experiment the stability of the coupling factors and resonance Q was observed for a 4-h period to confirm that these parameters were not drifting by more than 4 parts in 10 000.

The network analyzer was calibrated with a TRL two-port calibration. The tuner endplate was adjusted until the $TE_{01(25)}$ mode resonated at 10.630 GHz. Waveguide port extensions were manually adjusted until the resonance peak passed through the real axis in the reflection plane. The resonance response was then measured once every 10 min for a total of 10 measurements. Without detaching the tuner assembly from the cavity, the tuner endplate was then lowered and raised back up until the cavity resonated at the same frequency. The sensing-micrometer reading was recorded, and port extensions were again adjusted and recorded. The resonance S-parameters were then measured another 10 times at 10-min intervals without adjustments. Coupling factors and unloaded Q_0 were calculated from the zero-order equations given in (4.6) and (4.7). Results from this measurement are given in Table 4.3 and indicate an excellent stability of both coupling factors and the unloaded Q_0.

4.2.5 Repeatability of Q and Coupling Factors

In the same experiment that tested the tuner assembly attachment repeatability described in Sec. 4.1.3, Q_0, κ_1 and κ_2 were also calculated from S_{11} and S_{22} scattering parameter data. The network analyzer was calibrated with one-port response calibrations on both port 1 and port 2 in which the detuned short of the coupling apertures was the calibration standard. The resonator tuner assembly

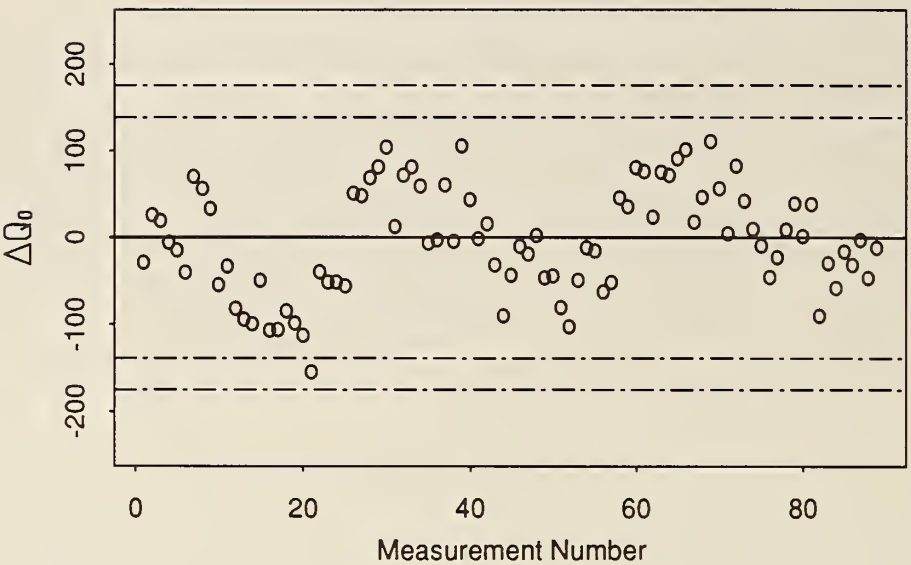

Figure 4.9: Values of $Q_0 \approx 84\,700$ calculated from scattering parameters. The cavity was opened and closed every three measurements for the first nine measurements and every four measurements thereafter. A new calibration was performed after measurements 9, 25, 41, 57 and 73. Horizontal hatched lines show 95% and 99% certainty bounds.

was re-attached and the $TE_{01(25)}$ mode was tuned to 10.630 GHz. A set of four consecutive measurements was taken to test stability, then the resonator tuner assembly was detached and reattached. The open/close measurement was repeated four times for each calibration, and after the fourth set of measurements, the network analyzer was recalibrated. An exception to this procedure was made in the first nine measurements, during which three consecutive measurements were made before the tuner assembly was opened, and the tuner assembly was detached and reattached three times, rather than four, before recalibration. There were a total of 89 measurements, 23 opening and closings of the tuner assembly, and 6 different calibrations.Results from this experiment are given in Figs. 4.9–4.12.

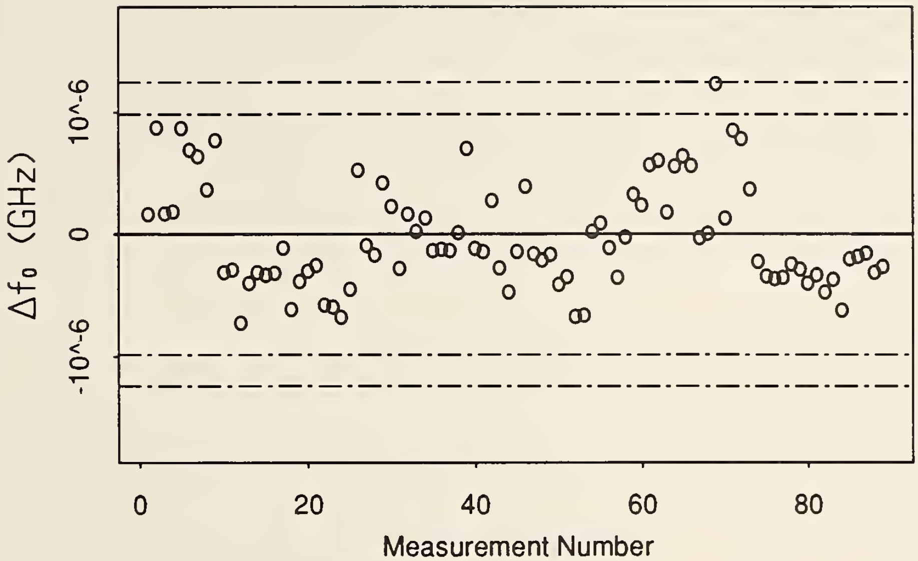

Figure 4.10: Calculated values of f_0. The cavity was opened and closed every three measurements for the first nine measurements and every four measurements thereafter. A new calibration was performed after measurements 9, 25, 41, 57 and 73. Horizontal hatched lines show 95% and 99% certainty bounds.

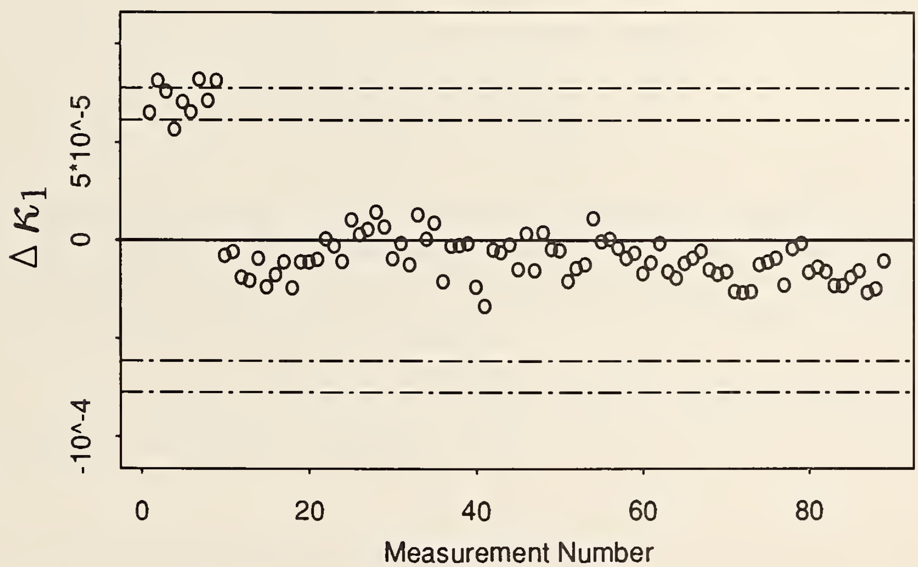

Figure 4.11: Calculated values of κ_1. The cavity was opened and closed every three measurements for the first nine measurements and every four measurements thereafter. A new calibration was performed after measurements 9, 25, 41, 57 and 73. Horizontal hatched lines show 95% and 99% certainty bounds.

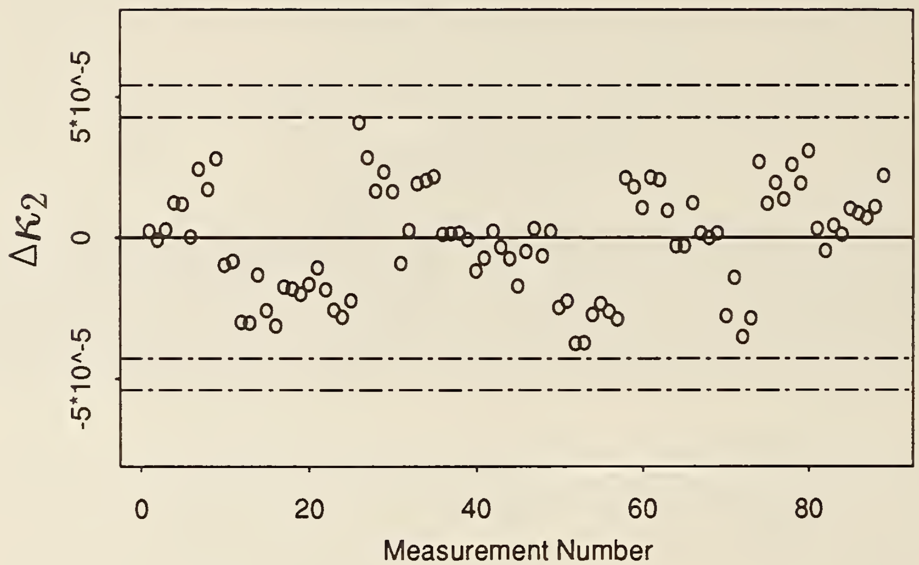

Figure 4.12: Calculated values of κ_2. The cavity was opened and closed every three measurements for the first nine measurements and every four measurements thereafter. A new calibration was performed after measurements 9, 25, 41, 57 and 73. Horizontal hatched lines show 95% and 99% certainty bounds.

Figures 4.9, 4.10, 4.11 and 4.12 show the estimated parameters for Q_0, f_0, κ_1 and κ_2, respectively. The mean value of Q_0 is very stable within the scope of a single calibration and changes slightly from calibration to calibration. The standard error for Q_0 over all measurements is ±62.2. The standard error for f_0 repeatability is ±395 Hz. The discrepancy of κ_1 in the first nine measurements from all the other κ_1 results cannot be explained at this time. To the best of our knowledge, no physical changes were made during the measurements. Most importantly, however, Q_0 was not significantly altered by the change in κ_1.

4.3 Systematic Uncertainty

The excellent results from the Q stability and repeatability experiments of Secs. 4.2.4 and 4.2.5 indicate that the loss tangent of a sample should be measurable to high accuracy. Unfortunately, systematic errors reduce the accuracy of our loss tangent measurements.

In general, systematic errors are difficult to detect because the instrument appears to work properly. If detected, systematic errors are usually very difficult to quantify and correct. This section describes some systematic errors inherent in the NIST 60-mm cavity resonator and offers some discussion on how to avoid permittivity and loss tangent measurement errors. These errors arise from:

1. interfering resonant modes,
2. simplifying assumptions used in the cavity theory,
3. the resonance model chosen to fit measured scattering parameter data,
4. imperfect resonator geometry and
5. limitations of the network analyzer.

There may be other systematic errors, and their detection must rely on future comparisons with other instruments.

4.3.1 Simplifying Assumptions

Permittivity (ϵ'_R) Model

The model to calculate ϵ'_R assumes that the cavity and sample are lossless. As discussed in Sec. 4.1, loss lowers resonant frequency, which causes the cavity's electrical dimensions to be larger than the physical dimensions. The corrections that can be made for this systematic error due to limitations in the ϵ'_R model have already been discussed.

Loss Tangent (tanδ) Model

The loss tangent model we use assumes we know the Q' of a resonator that contains a lossless sample with the same permittivity as the real sample. Typically the empty-cavity Q is used instead of the Q' of the cavity with a "lossless" sample, which leads to errors in tanδ calculation because the empty-cavity Q is slightly different than Q'. For example, the fixed-length method leads to errors in Q because of the frequency-dependent behavior of the coupling ports and skin-depth. In the case of the fixed-frequency method, Q differs slightly from Q' because the length of the cavity with sample is shorter than the empty cavity.

4.3.2 Resonance Model

To find resonance parameters such as f_0, Q_0, κ_1 and κ_2 a nonlinear least-squares fitting routine [32] is used to fit scattering parameter data to a resonance equation. The resonance equation is derived from the two-port, lumped-element equivalent circuit [33] shown in Fig. 4.13. This represents a more accurate and detailed representation of the NIST cavity than that given in Fig. 2.13. The input impedance at the port 1 coupling port is given by

$$\frac{Z_{aa1}(\omega)}{Z_0} = r_1 + X_1 + \frac{\kappa_1}{1 + jQ_0\left(\frac{(\omega-\omega_0)(\omega+\omega_0)}{\omega_0\omega}\right) + \frac{\kappa_2}{1+r_2+j\omega X_2}} , \tag{4.9}$$

where the normalized parameters

$$r_i = \frac{R_i}{Z_0} \quad \text{and} \quad X_i = \frac{L_i}{Z_0} , \quad i = 1, 2 \tag{4.10}$$

characterize the self-impedances of the coupling ports; the coupling factors

$$\kappa_i = \frac{\omega^2 M_i^2}{Z_0 R_s} , \quad i = 1, 2 \tag{4.11}$$

are functions of the mutual inductances M_i. The resonant frequency, ω_0 and the decoupled "unloaded" quality factor Q_0 are defined by

$$\omega_0 = \frac{1}{LC} \quad \text{and} \quad Q_0 = \frac{\omega_0 L}{R_s} . \tag{4.12}$$

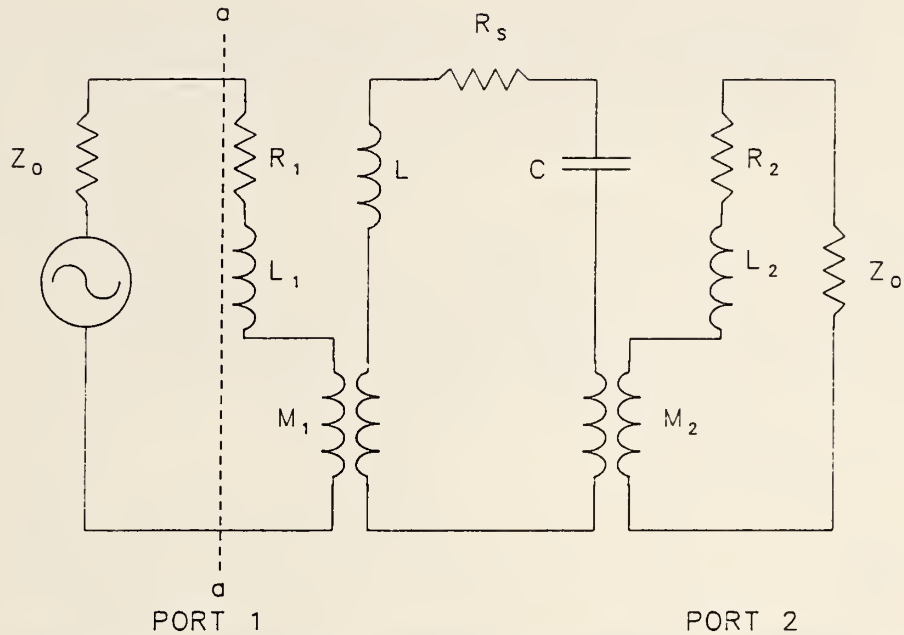

Figure 4.13: Lumped-parameter equivalent resonant-circuit showing coupling port impedances and reference plane a-a.

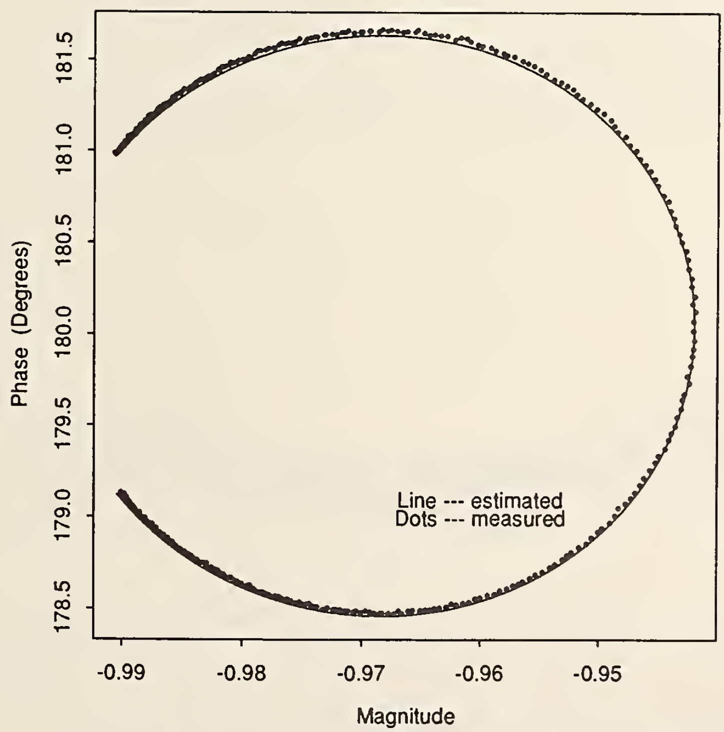

Figure 4.14: Measured scattering parameters (dots) and locus predicted from nonlinear least-squares fit to (4.9).

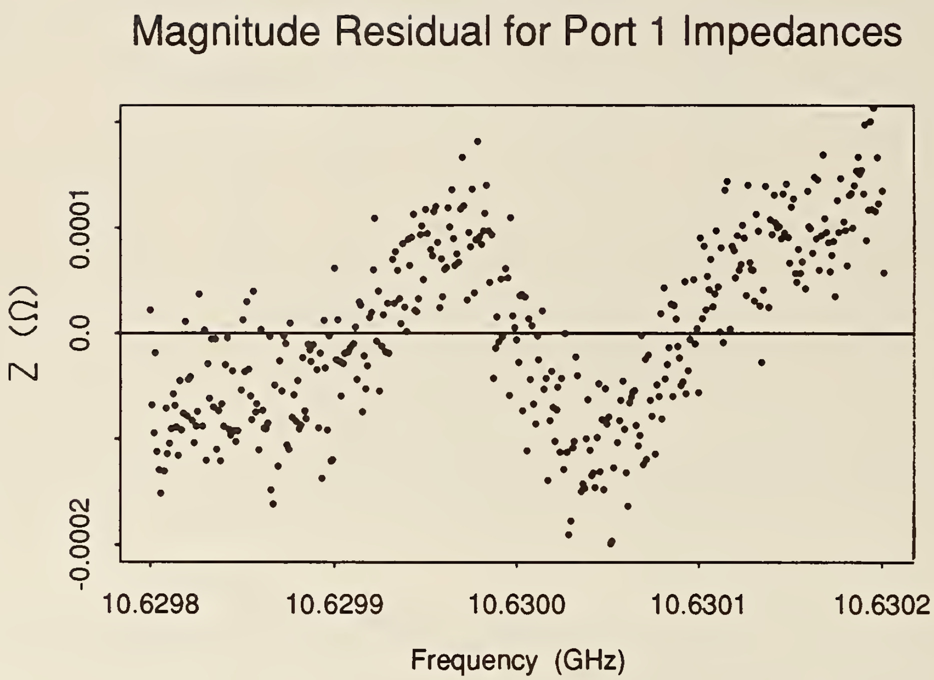

Figure 4.15: Magnitude of residuals to circuit model fit.

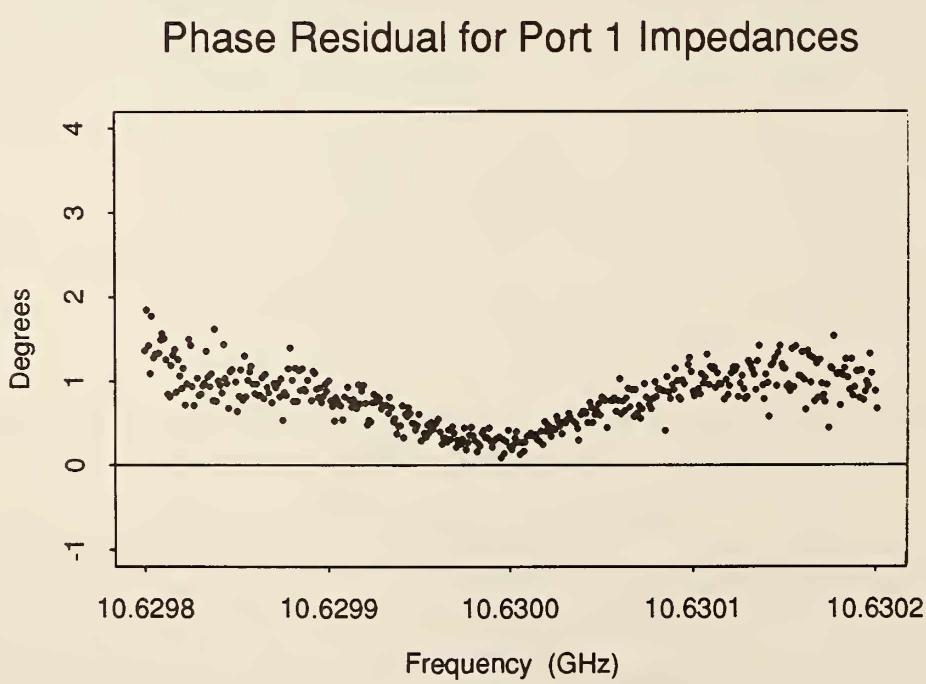

Figure 4.16: Phase of residuals to circuit model fit.

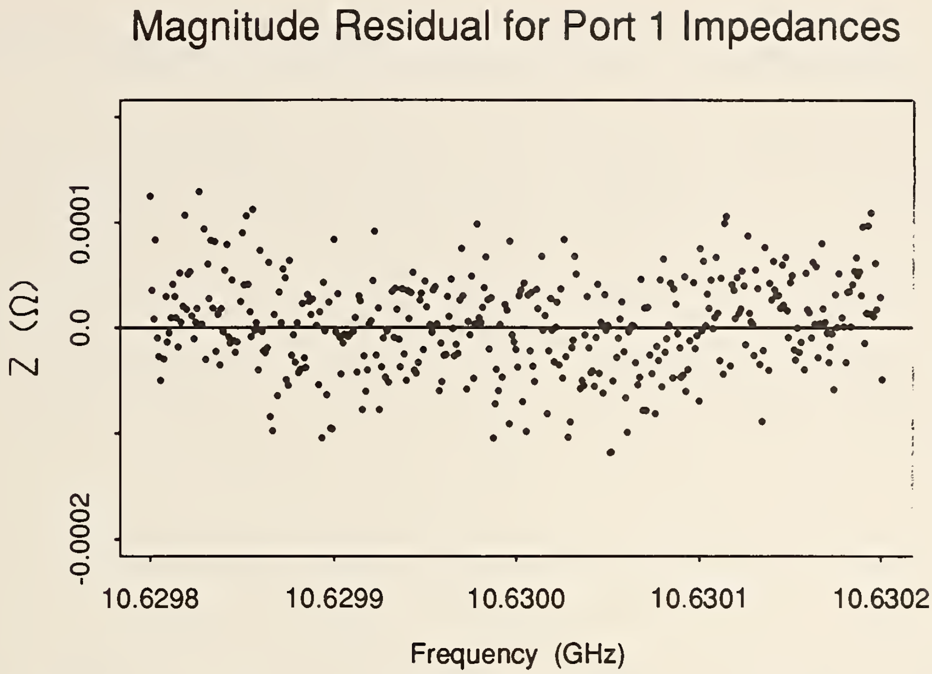

Figure 4.17: Magnitude of residuals to circuit model fit when an $e^{j\theta}$ term is included in the circuit model to compensate for network analyzer phase drift.

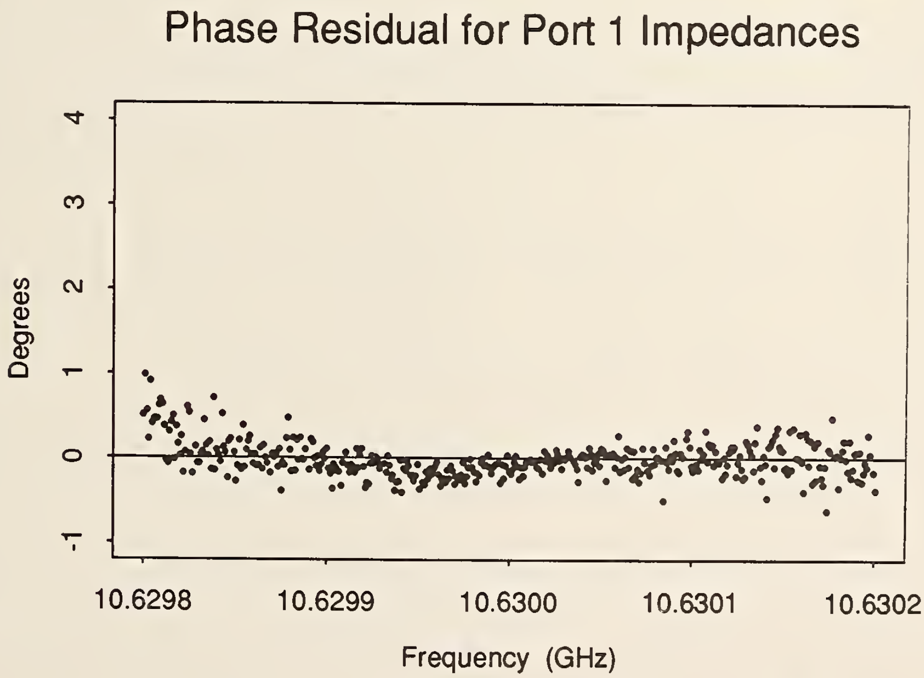

Figure 4.18: Phase of residuals to circuit model fit when an $e^{j\theta}$ term is included in the circuit model to compensate for network analyzer phase drift.

Table 4.4: Parameter estimates for data subsets. Q_L values are calculated from Q_0, κ_1 and κ_2.

Data used	κ_1	κ_2	Q_0	Q_L	f_0 (GHz)
All points	0.028 47	0.028 15	84 232	79 719	10.630 001 462
$i_{f_0} \pm 20$ points	0.028 57	0.028 24	83 988	79 473	10.630 001 971
$i_{f_0} \pm 3$ dB points	0.028 51	0.028 18	84 199	79 682	10.630 001 508
$i_{f_0} \pm 100$ points	0.028 51	0.028 19	84 049	79 539	10.630 001 676
First half of data	0.028 17	0.027 82	83 856	79 410	10.630 001 685
Second half of data	0.028 70	0.028 31	85 042	80 456	10.630 001 882

Figure 4.14 shows typical results when we fit the magnitude and phase of the resonance equation to measured S-parameters. Close examination of Fig. 4.14 reveals a systematic pattern in the residuals to the fit. A residual is defined as the distance between a measured datum and its predicted value. Figures 4.15 and 4.16 more clearly show the magnitude and phase of the residuals. If our model perfectly described the measured data, the scatter in the residuals would be random. This systematic pattern points to a discrepancy between the lumped-parameter circuit model and the measured resonance data. Isolating the cause of this discrepancy is difficult because the precision is so good. Either the circuit model or the S-parameter data could be exhibiting their limitations. We suspect that phase drift in the network analyzer's calibration is the cause of this residual error. This is so because when (4.9) is multiplied by a variable $e^{j\theta}$ term, the residuals to the fit become random, as shown Figs. 4.17 and 4.18.

The potential error due to the systematic pattern in the residuals has been estimated [34]. Nonlinear regression was performed on an S-parameter data set to estimate the parameters in (4.9). The full data set consisted of 401 evenly spaced frequencies, with f_0 at the center frequency. We compared parameter estimates from five subsets of the data to the estimates obtained using all 401 frequencies. The first three data sets were centered about f_0. The last two data sets consisted of the first and second halves of the data. Results are shown in Table 4.4. Results show that the systematic pattern has little effect on the final parameter estimates ($\approx 0.3\%$) when the regression uses data evenly distributed about f_0. However, when the frequencies are not evenly distributed about f_0, Q-value and coupling factor estimates can differ by 2% or more.

4.3.3 Interfering Non-TE_{01} Modes

The resonator's helical waveguide wall acts as an excellent mode suppressor, but other modes such as the TE_{02} and TE_{12} still exist. Although these modes are excited at much lower power than the dominant TE_{01} modes (nominally minus 30 dB), their existence still can cause mode interference. If a non-TE_{01} mode causes interference, no corrections can be made. Instead, we must try to resonate the cavity at either a different frequency or a different length in order to adequately separate the TE_{01} mode from the undesired mode. Mode interference changes the resonant frequency and Q of a TE_{01} resonance, which introduces significant permittivity measurement errors. An interfering mode makes the TE_{01} resonance response slightly asymmetric. This asymmetry is revealed when resonance scattering-parameter data are fit to characteristic resonance equations. Even slight distortions to the resonance shape can be seen in a plot of the residuals to the fit [34]. The computer program `MEAS_RES` described in Chapter 5 calculates resonance parameters from S-parameter data. To aid in detecting mode interference, `MEAS_RES` gives residual plots and prints out the average residual variance (χ). Presently, the threshold S-parameter residual variance at which mode interference becomes suspect is on the order of $\chi = 2.3 \times 10^{-3}$.

In the rare instances when two modes resonate at exactly the same frequency, mode interference may not be detected by distortion of the resonance response. When this situation occurs, the iris coupling coefficients κ_1 and κ_2 change from their expected values.

4.3.4 Imperfect Geometry and Wall Resistivity

As given in (4.4), resonance quality factor is defined as the ratio of the energy contained in a resonating system divided by its energy dissipation rate. The energy contained in the cavity depends on its cylindrical geometry. If the resonator's geometry is poor, the cavity Q will be reduced. Also, if the conductivity of the resonator walls is different than what we expect, resistive wall losses will make the cavity Q change from the expected value. To get some idea of the cavity's "mode-efficiency," we can compare the theoretical Q_0 for an ideal cylindrical copper cavity to the Q_0 of our resonator. Disparities between the two Q values can be attributed to a combination of imperfect geometry and resonator conductivity different from copper. Table 4.5 gives the percent discrepancy from an ideal copper cavity with a bulk conductivity of 5.80×10^7 S/m. Four different frequencies were tested, and two different endplates were used. Values in square brackets were found using an older endplate, and the unbracketed values were

Table 4.5: Mode efficiencies ($Q_{\text{measured}}/Q_{\text{theoretical}}$) × 100 for four frequencies. Square brackets: older endplate, unbracketed values: newer endplate.

Axial mode no.	Frequency (GHz) 10.05	10.34	10.63	10.92
20			78.46	
21	80.49		78.27	
22		79.97	78.70	
23	[78.13]	79.42 [76.77]		
24			78.84 [75.68]	76.86
25			[75.50]	
26				[74.93]

found using a newer endplate.

The measured Q-value is higher by approximately 2000 (86 000 vs. 84 000) with the new endplate, which indicates that the surface conductivity of the older endplate is approximately 15% less than that of the newer endplate [23]. This comes as no surprise because the older endplate has become scratched from use and the cavity Q has steadily degraded over the course of three years. The fact that the cavity resonator does not yield Q values expected for a copper cavity is no surprise. The helical windings that compose the cylindrical resonator wall are made of enamel-insulated copper wires embedded in an adhesive binder. Power loss in the binder and insulation will increase the effective surface resistivity of the cavity wall, thereby lowering the Q from the ideal copper-cavity value. Also, the helical walls are not perfectly smooth, which would allow less energy storage in the resonator.

Whether the cavity Q is lower than that of a copper cavity because of geometrical imperfections or increased surface resistivity does not matter. It is important only that the cavity Q remain constant. We have seen a steady decrease in cavity Q since the resonator was built and now have shown that a new endplate increases the Q to levels found previously. The resonator still has the old endplate installed and is functioning properly. However, the scratches on the endplate can lead to systematic errors. For example, the scratches create an air gap between the endplate and the sample, which will very slightly increase the calculated permittivity of thinner samples. When the resonator is refurbished, as discussed in Chapter 10, the tuner-endplate will be replaced.

4.3.5 Resonance Fitting - S_{11}/S_{22} Versus S_{21}/S_{12} Measurements

The MEAS_RES computer program obtains data from a network analyzer and applies a Levenburg-Marquart least-squares fitting routine to estimate $f_0, Q_0, \kappa_1, \kappa_2$ etc. These resonance parameters can be estimated from either S_{11}, S_{22}, S_{21} or S_{12} data. The transmitted signal of a resonance response is well out of the network analyzer's noise floor, and Q estimates calculated from S_{21}-only data generally yield precision estimates of approximately ± 50. Q precision estimates from combined S_{11}, S_{22} fits are on the order of ± 300 because the resonance response appears as a small dip in a high reflection.

Although precision estimates from least-squares fits to S_{21} data are better than combined S_{11}, S_{22} precision estimates, S_{11}, S_{22} measurements deliver better overall repeatability because one-port calibrations require less cable flexing. One-port calibrations result in better phase and amplitude stability than two port calibrations. Additionally, one-port calibrations take about half the time to perform as two-port calibrations. These considerations make combined S_{11}, S_{22} measurements preferable to through S_{21} or S_{12} measurements.

Chapter 5

Software

This chapter describes our computer programs and routines as applied to the resonator method of permittivity measurement. The programs are modular and certain routines are used in several places. All programs are written in BASIC. Notation denoting the actual variable used in computation are used throughout this section. The two primary programs, MEAS_RES and CAVITYPROG are listed in Appendices D and E.

The MEAS_RES program utilizes some subprograms many times in different places. `Zoom_on_peak` adjusts the network analyzer center frequency and span. `Fit_sparms` is a nonlinear Levenberg-Marquart fitting routine that finds resonance parameters. The two subprograms which actually send commands and receive data from the network analyzer are named `Set_nwa` and `Read_nwa`. These subprograms interpret mnemonics sent to them by higher level routines, then send the properly formatted commands to the network analyzer.

5.1 Instrument Control - MEAS_RES

The program MEAS_RES controls the network analyzer and automatically acquires resonance parameters of one or several resonances. Any of the four S-parameters measured by the network analyzer may be used to calculate resonance parameters. The operator controls the program's flow by pressing softkeys.

5.1.1 `Measuresweep` Subprogram

If the resonance frequencies are unknown, the computer slowly steps through a frequency range and searches for resonances. Once a resonance is found, the computer zooms in on the resonance, acquires the S-parameter data, and fits the data to the appropriate Lorentzian resonance equation (reflected or transmitted resonance response).

5.1.2 `Measurelist` Subprogram

This subroutine steps through a list of frequencies and finds the resonance parameters of any resonances found. Resonance frequencies are typed into data statements before the program is run. The data statements are located at the line labelled `Resonance_data`. The computer reads the frequency list from the data statements when the subroutine is invoked. Once the resonance parameters are found, the computer sets the network analyzer to the next frequency and repeats the process.

5.1.3 `Measuretrigger` Subprogram

If the operator wants full control of both the network analyzer and computer, this method is useful. The computer waits until the operator presses the trigger softkey, obtains the S-parameters of interest and calculates the resonance parameters. This is useful when the fixed-frequency, variable-length method is employed. With this method, the operator has to move the tuner-endplate to achieve the desired frequency.

5.1.4 `Measurelistrig` Subprogram

This routine sets the network analyzer to a predefined frequency given in a data-statement list and then waits until you are ready to measure the resonance. When you have made the necessary adjustments to the resonator, press the trigger softkey to measure the resonance. When the resonance parameters have been found the computer sets the network analyzer to the next frequency in the frequency list. This capability is useful when the network analyzer is to be calibrated at each resonance, or when the operator wants to adjust the resonance frequency of the cavity.

5.1.5 Measuredumdata Subprogram

This subroutine is useful for program development and to test the effects of noise inserted into ideal data. The driver contained in this subroutine can be modified to use different resonance-fitting models. S-parameters and frequency list data are loaded from disk.

5.1.6 Measurediskdata Subprogram

This routine loads previously measured S-parameter data from disk and calculates the resonance parameters. Sometimes this is useful if the network analyzer is calibrated in `frequency list` mode. Several frequencies can be measured at once, the data stored on disk and then the measurement can be repeated. By changing the `Start_index` and `Stop_index`, the operator can select the proper resonance from the S-parameter data set.

5.1.7 Fit_sparms Subprogram - Resonance Fitting

In both the variable-length and the variable-frequency sections of CAVITYPROG, and throughout MEAS_RES the `Fit_sparms` subprogram calls a CSUB that calculates resonance parameters and their uncertainties from measured scattering parameter resonance data. `Fit_sparms` is the CSUB driver that sets up the S-parameters, iterates the CSUB to parameter-convergence, and prints and plots the results.

Presently there are three Lorentzian resonance equations to which the scattering parameters can be fitted. Two are transmission equations, and the third is a reflection equation. Each describes a circle in the complex reflection plane. The two transmission-response equations describe a circle near the origin in the complex impedance plane. The third is a reflection-response equation that describes a circle near $|\Gamma| = 1$ on the impedance plane. The variable `Sss` selects which equation to fit to the data.

When `Sss = 1`, a simple resonance transmission expression given in many textbooks [35] is applied:

$$S_t(f) = \frac{-f^2 A \exp(j\theta)}{(f^2 - f_0^2) - j\frac{f_0 f}{Q_L}} \, . \tag{5.1}$$

This is a Lorentzian line-shape equation with an arbitrary phase and magnitude term. The loaded Q_L depends only on the distribution of data around the circle. The scattering parameters trace out the circle faster with respect to frequency as resonance Q increases. Unfortunately, (5.1) does not decouple Q_0 from Q_L. To do so, the S-parameters' magnitudes must be accurately measured. Shulten [36] gives both reflected and transmitted resonance line-shape equations that express Q_0 separately as follows:

$$S_r(f) = \frac{\frac{Q_L}{Q_0} + 2jQ_L\frac{f-f_0}{f_0}}{1 + 2jQ_L\frac{f-f_0}{f_0}} A\exp(j\theta) \quad \text{and} \tag{5.2}$$

$$S_t(f) = \frac{1 - \frac{Q_L}{Q_0}}{1 + 2jQ_L\frac{f-f_0}{f}} A\exp(j\theta)\ . \tag{5.3}$$

In the `Fit_sparms` subprogram, the reflected S-parameter expression (5.2) corresponds to `Sss = 2` and the transmission equation (5.3) is used when `Sss = 3`.

To accurately determine Q_0, the amplitude factor A must be very close to 1. The arbitrary phase term compensates for phase delays due to residual calibration error and drift. Equations (5.2) and (5.3) express Q_0 separately from Q_L, which allows us to calculate the coupling port Q-values (Q_E). For an empty cavity,

$$\frac{1}{Q_L} = \frac{1}{Q_0} + \frac{1}{Q_{E1}} + \frac{1}{Q_{E2}}\ . \tag{5.4}$$

5.1.8 Levenberg-Marquart CSUB

To find the resonance parameters from measured S-parameter data, we implement a Levenberg-Marquart nonlinear least-squares fit technique [37]. The user supplies the constitutive equation (5.1), (5.2) or (5.3) and its derivatives with respect to the unknown parameters. The derivatives are used to provide a set of linear equations which are used to increment the approximated values of the unknown parameters in the correct direction. The CSUB performs one iteration that computes the derivatives and increments the estimated values of the unknown parameters. `Fit_sparms` iterates the CSUB until convergence is reached. Once the parameter values are determined, `Fit_sparms` makes a final iteration to compute the covariance matrix. The square root of the diagonal elements of the covariance matrix provide the estimated standard errors in the fitted parameters.

This implementation of the Levenberg-Marquart method uses source code written in FORTRAN. Since BASIC and FORTRAN are inherently incompatible because of integer, string and array operations, the creation of the CSUB requires some restructuring of I/O operations and array pointers [38], [39].

5.2 Data Analysis - CAVITYPROG

This section describes the CAVITYPROG computer program. The program contains all the utilities needed to fit resonance data, calculate permittivity and loss tangent and perform error analysis. Either the variable-length or variable-frequency technique may be applied. Structured programming methods have been used.

CAVITYPROG is designed to be interactive and facilitates user interaction through the use of menus and softkeys. Various parameters such as cavity and sample dimensions, and resonance frequencies can be changed through the use of softkeys. Descriptions of the softkey functions are located at the bottom of the computer display.

Because of the complexity of incorporating error-correction routines into CAVITYPROG, be aware of the need to enter corrected data into the calculations. Specifically, the unloaded Q_0 must be used to properly calculate loss tangent, and resonant frequency must be corrected to reflect the downward frequency shift due to finite cavity losses. These corrections are discussed more thoroughly in Sec. 4.2.

CAVITYPROG uses softkeys to direct program flow. The various utilities are accessed by the operator pressing softkeys to move up and down through the program's branches. For example, when the program is run the computer first displays the main menu and a set of softkeys. The main menu displays:

1. Laboratory temperature, air pressure and relative humidity
2. Speed of light in the laboratory
3. Sample description and thickness
4. Cavity diameter and length

The operator may use the main-menu softkeys to call specific utilities. Each one of these utilities contains its own menu and softkeys which allow the user change parameters or call other utilities. The utilities directly accessible from the main menu include:

1. Calculation of permittivity by the fixed-frequency technique
2. Calculation of permittivity by the fixed-length technique
3. Calculation of speed of light from environmental parameters

4. Calculation of resonator diameter and length from the empty resonator's TE_{01} resonance spectrum

5. Calculation of all TE and TM modes for a given cavity

6. Calculation of the TE_{01} frequency spectrum for either an empty or a dielectric-loaded cavity

7. Calculation of resonance frequency and Q from S-parameters loaded from disk

Sections 7.5.2 and 7.5.3 give typical step-by-step procedures for performing fixed-frequency and fixed-length permittivity measurements.

When CAVITYPROG is first run, verify that all parameters have their proper values. For instance, the speed of light in the cavity should be recalculated from the current environmental conditions before proceeding with any other calculation. Also, check the cavity's length and diameter, and if necessary, recalculate them.

Permittivity calculations rely on the user to provide a first guess of ϵ'_R from which a convergence routine finds the proper value. To aid the user in making this guess two utilities are provided. One utility calculates the frequencies of all TE and TM modes for the cavity. The other calculates the TE_{01} modes for either an empty or dielectric-loaded cavity. These routines are also useful during the data-acquisition process in helping to locate desired modes.

5.2.1 `Calc_c` Subprogram - Speed of Light

The `Calc_c` subprogram calculates the speed of light and its uncertainty from the resonator temperature, barometric pressure, and relative humidity. The speed of light in the laboratory should be the first calculation performed when running CAVITYPROG. Nearly all other calculations depend on the speed of light. `Calc_c` displays a menu showing the values of temperature, pressure and relative humidity. Their values and uncertainties may be changed using softkeys. The speed of light and its uncertainty is recalculated and displayed each time one of the values is changed.

Calculation of speed of light

We implement the following equations to calculate the speed of light, given by Liebe [40]:

$$c_{air} = \frac{c_0}{n} \; , \tag{5.5}$$

where

$$n = n' - jn'' = 1 + N_t \; , \tag{5.6}$$

$$\begin{aligned} N_t &= (n-1) \cdot 10^6 \text{ ppm} \; , \\ &= N_0 + N'(f) - jN''(f) \; , \end{aligned} \tag{5.7}$$

$$\begin{aligned} N_0 &= N_1 + N_2 \; , \\ &= 2.588p\theta + (41.63\theta + 2.39)e\theta \; . \end{aligned} \tag{5.8}$$

The dispersive term in (5.7), $N'(f) - jN''(f)$, is neglected. In the frequency range 8–27 GHz at our typical laboratory conditions, the dry-air refractivity $N_0 \cong 269.03$, whereas the dispersive term affects the total refractivity N_t in the fifth digit. N_1 and N_2 are the nondispersive refractivity contributions of dry air and water vapor. The terms for the partial pressures for dry air and water vapor, p and e, are calculated from the barometric pressure P, relative humidity U, and temperature T in degrees Celsius as follows:

$$p = P - e \; , \tag{5.9}$$

$$\theta = 300/(T + 273.15) \; , \tag{5.10}$$

$$U = (e/e_s)100 \leq 100 \; , \tag{5.11}$$

$$e = 2.4076 \cdot 10^8 U\theta^4 \exp(-22.64\theta) \; . \tag{5.12}$$

The saturated water vapor pressure e_s is found by setting $U = 100$ in (5.12).

5.2.2 `Var_length` Subprogram

The `Var_length` subprogram calculates complex permittivity using the variable-length, fixed-frequency technique. The variable-length technique is generally considered to be more accurate than the variable-frequency technique. Frequency dependent perturbations such as coupling factors and skin depth remain constant if the frequency is fixed.

When running the variable-length section of CAVITYPROG, the computer displays a menu containing the values of all the necessary parameters. The user may change the sample description, thickness and thickness uncertainty. The resonance frequency is required to be a single frequency for both the empty and dielectric-loaded conditions. The resonance frequency is very sensitive to changes in cavity length, and a high accuracy micrometer must be used to make this constraint experimentally achievable. The micrometer readings and unloaded Q-values along with their uncertainties may be entered for the empty and loaded resonances. If desired, the user can call a fitting routine (`Fit_sparms`) which calculates the loaded and unloaded Q-values, center frequency and their uncertainties from measured resonance S-parameter data. A description of this routine is found in Secs. 5.1.7 and 5.1.8.

When all the necessary parameters have their proper values, the user then presses the "Calc Epsilon" softkey to invoke the subroutine `Calc_eps_len`. The equations to calculate complex permittivity are nearly identical for both the variable-frequency and the variable-length techniques, so both techniques use the same subprograms. A flag `Do_eps_freq` informs the subprograms which technique to use. The subroutine `Calc_eps_len` sets `Do_eps_freq` equal to zero, then calls the subprograms `Calc_eps_re`, `Calc_tand` and `Calc_eps_im` which calculate permittivity and loss tangent. After ϵ_R' and $\tan\delta$ are calculated, pressing the "Calc Errors" softkey calls the subroutine `Calc_errs_len` which again sets `Do_eps_freq` equal to zero and calls the subprogram `Calc_errors`. The computer then displays an error table showing the error contributions due to the uncertainty of each variable while holding all other variables fixed.

5.2.3 `Var_freq` Subprogram

The variable-frequency section of CAVITYPROG closely resembles the variable-length section except that the micrometer readings are forced to a single value and the resonance frequencies for the empty and loaded cavity are allowed to be different. The computer displays a menu identical to the variable-length menu containing all the parameters and their uncertainties needed for permittivity calculation. Use the softkeys to set the empty and dielectric-loaded cavity dimensions and Q-values (or bandwidths), sample thickness, and initial guess for ϵ_R'. Like the variable-length technique, the empty or dielectric-loaded cavity Q-value, resonance frequency and their uncertainties can be calculated by using the "Fit S parms" softkey. When you have set the necessary variables to their proper values, the subroutine `Calc_eps_freq` is run by pressing the "Calc Epsilon" softkey. `Calc_eps_freq` sets the flag `Do_eps_freq` equal to one and calls the subprograms `Calc_eps_re`, `Calc_tand` and `Calc_eps_im`. After calculating permittivity, uncertainties can be calculated by pressing the "Calc Errors" softkey.

To enhance accuracy, correct the resonance frequency and resonator Q_L. Energy absorption by the cavity and sample pull the resonant frequency downward and will affect permittivity results. To compensate, the resonant frequency of the cavity with sample should be increased by one-half the 3-dB resonance bandwidth, and the cavity dimensions should be decreased to remove the perturbation due to skin-depth penetration. These corrections are especially important for lossy samples. To properly calculate loss tangent, the frequency-dependent coupling factors should be removed from Q_L. By using Q_0 rather than Q_L, you can remove the effect of frequency-dependent coupling factors. For utmost loss-tangent accuracy, you should also compensate the empty-cavity Q_0 to reflect the change in frequency-dependent skin-depth losses.

5.2.4 `Cavity_dimens` Subprogram

The `Cavity_dimens` subprogram allows the operator to enter the cavity's diameter and length from the keyboard or to calculate cylindrical cavity dimensions. The `Cavity_dimens` subprogram is a driver for the `Calc_dimens` subprogram that calculates cavity dimensions from the measured TE_{01n} frequency spectrum. Because the cavity dimensions affect many other calculations, this subprogram can be called only from the main menu. To calculate the cavity dimensions from the TE_{01} spectrum, the operator first must find at least three TE_{01} frequencies for a given cavity length. The operator enters the axial mode number and frequency for each of these modes, then presses the "Find Dimens" softkey. The cavity diameter and length uncertainties are calculated from the variance/covariance matrix returned from the regression operation. If the operator wants to calculate the resonator's dimensions with skin-depth included, the measured resonance frequencies are entered directly. If the cavity dimensions without skin-depth are desired, the operator must enter the resonant frequencies plus one-half the unloaded resonance Q_0 bandwidth. For example:

$$f_{\text{compensated}} = f_{\text{measured}} \left[1 + \frac{1}{2Q_0} \right] . \tag{5.13}$$

This does not include frequency pulling due to coupling port perturbations or coupling port losses. The need for and method of frequency compensation for these effects is under investigation and is described in Sec. 4.1.

Calculation of Cavity Dimensions

The resonant frequency for a TE_{mnp} mode is given in a linearizable form by

$$f_{mnp}^2 = B(1) + B(2)p^2 , \tag{5.14}$$

where

$$B(1) = \left(\frac{ct'_{mn}}{\pi D}\right)^2 , B(2) = \left(\frac{c}{2L}\right)^2 . \tag{5.15}$$

The speed of light in the resonator is c; the length and diameter of the resonator are L and D, and $t'_{mn} = t'_{01} \cong 3.831\,705\,970\,2$.

By measuring at least three TE_{01} modes, we can use the mode numbers and frequencies to perform a least- squares estimate to find the unknowns $B(1)$ and $B(2)$. Let

$$\bar{Y} = \begin{bmatrix} f_1^2 \\ \vdots \\ f_N^2 \end{bmatrix} , \bar{X} = \begin{bmatrix} 1 & p_1^2 \\ \vdots & \vdots \\ 1 & p_N^2 \end{bmatrix} , \bar{B} = \begin{bmatrix} B(1) \\ B(2) \end{bmatrix} . \tag{5.16}$$

The vector $\bar{B}$ is found by

$$\bar{B} = \left(\bar{X}^T \bar{X}\right)^{-1} \bar{X}^T \bar{Y} . \tag{5.17}$$

The estimated variance-covariance matrix is given by

$$\bar{V}(\bar{B}) = \left(\bar{X}^T \bar{X}\right)^{-1} s^2 , \tag{5.18}$$

where s^2 is given by the residual sum of squares divided by the number of degrees of freedom

$$s^2 \cong \frac{\bar{Y}^T \bar{Y} - \bar{B}^T \bar{X}^T \bar{Y}}{N - 2} . \tag{5.19}$$

The diagonal elements of $\bar{V}(\bar{B})$ yield the square of the standard uncertainties of $\bar{B}$. The noncovariant uncertainties in the fit parameters are:

$$\Delta B(1) = \sqrt{\bar{V}_{11}(\bar{B})} \text{ and } \Delta B(2) = \sqrt{\bar{V}_{22}(\bar{B})} . \tag{5.20}$$

95% uncertainty bounds for the fit parameters are given by the program by multiplying the standard uncertainties by the 95%-area value of a t-distribution with $(N - 2)$ degrees of freedom. An analysis of this problem is given in Draper and Smith [41].

These uncertainties are appropriate for specifying the possible ranges for an individual parameter without regard to the value of the other parameter. Be aware that the values $B(1)$ and $B(2)$ are covariant. In a two-dimensional ($B(1)$ versus $B(2)$) space, the uncertainties given by the covariance matrix diagonals describe a rectangular, noncovariant region of possible values of $B(1)$ and $B(2)$. The variance/covariance matrix defines the elliptically shaped (63%) standard uncertainty contour interval in a $B(1)$ versus $B(2)$ plane. The rectangle described by the noncovariant uncertainties bounds most of the covariant, elliptical contour interval. The areas inside the rectangular area and outside the elliptical area are regions where the probability of simultaneous values of $B(1)$ and $B(2)$ are highly unlikely because of the covariant relationship between the two fit parameters. For example, if $B(1)$ increases it is most likely that $B(2)$ will decrease. This means that the 95% noncovariant rectangle will be a good approximation of the covariant elliptical confidence interval for most regions. Areas outside of the elliptic locus but within the prescribed rectangle are accounted for, but have low probability of occuring.

5.2.5 `Calc_modes` Subprogram

The `Calc_modes` subprogram can calculate all TE and TM modes which can exist in a cylindrical cavity up to the $TE_{8,20,40}$ mode and the $TM_{8,20,39}$ mode. This subprogram is often useful in helping to locate cavity modes during an experiment, and can help verify cavity performance. In X-band the non-TE_{01} modes of greatest concern are TE_{12} and TE_{02}. Before pressing the "Calc Modes" softkey, the operator must set the cavity dimensions to the desired values. After the operator presses the Calc Modes softkey, the computer displays the cavity dimensions and the start and stop frequencies for the calculation. The operator sets the start and stop frequencies, then proceeds with the calculations by pressing the "OK Begin" softkey.

Calculation of TE and TM modes

TM resonance frequencies are calculated from the equation:

$$f_r(TM_{mnp}) = \frac{c_{air}}{2\pi}\sqrt{\left(\frac{t_{mn}}{a}\right)^2 + \left(\frac{p\pi}{L}\right)^2} \,. \tag{5.21}$$

The TE resonance frequencies are calculated from the equation:

$$f_r(TE_{mnp}) = \frac{c_{air}}{2\pi}\sqrt{\left(\frac{t'_{mn}}{a}\right)^2 + \left(\frac{p\pi}{L}\right)^2} \,, \tag{5.22}$$

where

t_{mn} is the n-th root of the equation $J_m(x) = 0$,

t'_{mn} is the n-th root of the equation $J'_m(x) = 0$,

m, n, and p are integer mode numbers,

a is the cavity radius in m,

L is the cavity length in m and

c_{air} is the speed of light in the laboratory in m/s.

t_{mn} and t'_{mn} are in the form of look-up tables in data statements found in the subprogram `Modefreqs2`. These equations are from the book by Jackson [42].

5.2.6 `Calc_freqs` Subprogram

This subprogram calculates the TE_{01} resonance frequencies for either an empty or dielectric-loaded cylindrical cavity.

The calculations for the empty cavity finds all TE_{mn} resonance frequencies from the given start frequency up to axial mode number 50, or up to the maximum frequency of interest. On the other hand, the calculations for a dielectric-loaded cavity find only one TE_{01} frequency at a time. This is because an iterative technique is needed to match boundary conditions. The operator sets the theoretical parameters for the cavity dimensions, sample thickness, and permittivity and then enters into the computer an approximate value for the resonance frequency. The computer then iterates on the frequency using Newton's method until resonance boundary conditions are satisfied.

5.2.7 `Calc_eps_re` Subprogram - Permittivity Calculation

The subprogram `Calc_eps_re` is used by both the variable-frequency and the variable-length sections of CAVITYPROG to calculate ϵ'_R. It is also used by the subprogram `Calc_errors`. The same code is used for the variable-frequency and variable-length technique because only the sample thickness, dielectric-loaded resonance frequency and cavity dimensions are required to calculate ϵ'_R.

The initial guess for ϵ'_R must be near the actual value for the calculation to converge to the proper value. This is usually a simple matter because we usually know the approximate permittivity. Sometimes it may be necessary to measure a very thin sample, which assures us that we are on the first root no matter how high the permittivity may be. A way to check whether the correct value of permittivity has been found is to calculate ϵ'_R for several adjacent TE_{01n} modes. If the results agree with one another, then the proper permittivity has been found.

Iterative Method for ϵ'_R

To find ϵ'_R, the proper value of the axial wavenumber inside the resonator sample (β_1) must be obtained from the transcendental equation (see Sec. 2.2.2)

$$f(\beta_1) = 0 = \frac{\tan\beta_1 b}{\beta_1} + \frac{\tan\beta_0 Y}{\beta_0}\ , \tag{5.23}$$

where b is the sample length (thickness); $Y = L - b$ is the length of the air-filled portion of the resonator, and β_0 is the axial wavenumber in the air-filled portion of the resonator given in by (2.45)

$$\begin{aligned}\beta_0 &= \sqrt{\omega^2\mu\epsilon_{air} - k_c^2} \\ &= \sqrt{\left(\frac{\omega}{c_{air}}\right)^2 - k_c^2}\ ,\end{aligned} \tag{5.24}$$

with $k_c = t'_{01}/a$.

Every term in (5.23) is known except for β_1. The equation has many roots and cannot be solved directly. However, if we know on which root the solution lies, Newton's iterative method may be applied to find the proper solution. For this case, Newton's method takes the form

$$\beta_{1_{n+1}} = \beta_{1_n} - \frac{f(\beta_{1_n})}{f'(\beta_{1_n})}\ . \tag{5.25}$$

The derivative of (5.23) is

$$f'(\beta_1) = \frac{\frac{\beta_1 b}{\cos^2(\beta_1 b)} - \tan(\beta_1 b)}{\beta_1^2}\ . \tag{5.26}$$

We begin with a guess for ϵ'_R and compute the starting value for β_1. We repeatedly apply (5.25) to converge to the proper value of β_1. Convergence is deemed appropriate when

$$\frac{f(\beta_1)}{f'(\beta_1)} < 10^{-4}\ . \tag{5.27}$$

5.2.8 `Calc_tand` Subprogram

The calculation of loss tangent is performed using equations given in Cook [2]. The `Calc_tand` subprogram is used by both the variable- frequency technique and the variable-length technique. Dielectric loss tangent is given by

$$\tan\delta = \frac{p\,(2b - s) + (1/\epsilon'_R)\,[2\,(L_r - b) - q]}{p\,(2b - s)} \cdot \left(\frac{1}{Q_d} - \frac{1}{Q'}\right)\ , \tag{5.28}$$

where Q_d is the Q for a cavity filled with sample, and Q' is the theoretical Q for a cavity filled with a "lossless" sample of the same permittivity as the lossy sample. L_r is the total resonator length and b is the sample thickness, with

$$p = \frac{\sin^2 \beta_0(L_r - b)}{\sin^2 \beta_1 b}, \tag{5.29}$$

$$q = \frac{\sin 2\beta_0 (L_r - b)}{\beta_0}, \text{ and} \tag{5.30}$$

$$s = \frac{\sin 2\beta_1 b}{\beta_1} \tag{5.31}$$

The wavenumber β_0 is given by (2.45) and β_1 by (2.44). As stated in Cook, the theoretical Q is determined by measurement of the air-filled cavity so that the effective skin depth d is determined from

$$d = \frac{a\,(\beta_0^2 + k_c^2)}{Q_0\,[k_c^2 + (2a/L_0)\,\beta_0^2]}, \tag{5.32}$$

where L_0 is the length, and Q_0 is the unloaded Q of the air-filled cavity. From this calculation of the effective skin-depth, the theoretical value of Q' is determined from

$$Q' = \frac{A + (1/\epsilon'_R)B}{\left\{\frac{d}{a(\beta_1^2 + k_c^2)}\right\}\{k^2\,[A + B] + 2a\,(p\beta_1^2 + \beta_0^2)\}}, \tag{5.33}$$

with

$$A = p(2b - s), \tag{5.34}$$

$$B = 2(L_r - b) - q. \tag{5.35}$$

5.3 Other Software

A computer program named GRAPH_DATA was developed by the NIST Electromagnetic Fields Division. Most of the figures and graphs produced for this report were made by GRAPH_DATA. Also, many of the subprograms found in GRAPH_DATA are incorporated into MEAS_RES and CAVITYPROG to aid in disk I/O and graphing. Some of these subprograms are `Enterfilename`, `Enter_id`, `Select_disk`, `Save_file`, `Enter_file`, `Menu_scroll`, `Plot_all`, `Pack_data`, `Setup_hp7475a` and `Setup_laserjet`.

Chapter 6

Laboratory Environment and Equipment

6.1 Laboratory Environment

Atmospheric temperature, pressure and relative humidity affect the speed of light in the laboratory, and to calculate the permittivity of a material relative to a vacuum, we must use the proper value for the speed of light in various equations. Our laboratory maintains 40% relative humidity to within ±2%. Temperature is maintained at a nominal 23°C, but variations can be more than ±3°C. Temperature variations can change the cavity resonator's dimensions, and Chapter 9 shows that uncertainty in speed of light can contribute a sizable uncertainty to permittivity calculations.

6.2 Laboratory Equipment

6.2.1 Network Analyzer

The automatic network analyzer (ANA) is used to obtain scattering parameter data for the cavity resonator. These scattering parameters are used to find resonance parameters such as resonant frequency, resonance bandwidth and coupling factor. Certain cavity measurements do not require ANA calibration. For example, the resonant frequency is found at the frequency where the resonance response is a maximum. Also, the half-power bandwidth can be accurately measured with an uncalibrated ANA because the response of the network analyzer is very linear. Measured Q can be accurately calculated from the half-power bandwidth obtained from uncalibrated ANA data because the ANA has a very linear response. To measure unloaded Q and coupling factors, the network analyzer must be calibrated.

6.2.2 Computers

The electromagnetic properties of materials (EPM) program currently has an HP 9000 series 300 computer that runs the instrument control and data analysis programs described in Chapter 5. This computer runs the HP BASIC (RMB) operating system. Additionally, we have an 80486-based computer that runs TransEra HTBasic that emulates the RMB environment. This computer and other DOS based computers are used for running DOS based programs (word processing, typesetting, spreadsheets, graphics).

Mention of specific products does not constitute a product endorsement by NIST. Product makes and models are mentioned to give the reader an understanding of the equipment currently in use by the EPM program.

6.2.3 Dimensional Measurement

Endplate Sensing Micrometer

The sensing micrometer that measures the cavity's change in length can be detached and connected to a flat granite surface in order to measure sample dimensions.

High-Precision Micrometer

The endplate sensing micrometer compares well with a hand-held micrometer with a 2.5 μm (0.0001 in) precision and gives excellent agreement with the Fowler measurement micrometer.

Gage Blocks

To check the accuracy of our micrometers, we have a set of A+ rated gage blocks.

6.3 ANA Waveguide Calibration

We calibrate the network analyzer to obtain the most accurate measurements of resonance S-parameters. Calibration removes the effects of mismatches, losses and phase delays which occur before the measurement reference plane. The resonator, as discussed in Chapter 3, is excited by coupling energy through small apertures that are connected to a waveguide feed system. We wish to define the measurement reference planes at the resonator's coupling ports. The precise rectangular X-band waveguide connects to waveguide-to-coaxial cable adapters, which in turn connect to the network analyzer. In uncalibrated mode the reference plane is undefined. Moreover, in uncalibrated mode the phase and magnitude at our waveguide measurement ports varies with frequency in a nonlinear way. Even for narrowband responses, such as for the empty cavity, nonlinear phase dispersion may affect the Q-value measurement. Measurements of wider resonance-width responses, such as our glass measurements (see Sec. 8.1), can show significant phase and amplitude distortion due to the nonlinear response characteristics of the cavity's feed system.

6.3.1 X-band TRL Calibration Kit

To calibrate the network analyzer, the Through-Reflect- Line (TRL) method is employed. This calibration technique is easy to implement and has minimal hardware requirements. The calibration kit requires only three standards which, in this instance, are very easy to obtain. The first is a "through" connection, in which the two waveguide flanges of both measurement ports are connected together. The second is a "reflect," which is a simple waveguide short that is connected to both ports, one at a time. The third standard, the "line," is a 1-cm long electroformed section of X-band waveguide. Table 6.1 gives the definitions of the standards and class assignments of our X-band TRL calibration kit.

Optimum Line Length

The difference in phase delay between the "through" and the "line" must be between (20 and 160°)$\pm$n $\times$180° over the desired frequency range. The exact length of the waveguide line is arbitrary, as long as this criteria is met. In fact, the network analyzer needs an approximate value for the line delay only so it can determine the above value of n. The optimum phase delay for the TRL method is 90° The optimal line length yields as close to a 90° phase shift over our frequency range of interest. Phase delay of a rectangular waveguide section

Table 6.1: Definitions and class assignments of an X-band TRL calibration kit.

Standard		Offset			Freq. (GHz)		Type	Name
No.	Class	Length (mm)	Delay (ps)	Z_0	Min	Max		
9	Load	0	0	1	6.5554	16.0	W/G	XLOAD
13	Delay	0	0	1	6.5554	16.0	W/G	ADAP
14	Thru	0	0	1	6.5554	16.0	W/G	THRU
15	Line	9.977	33.38	1	6.5554	16.0	W/G	LINE1
16	Line	25.387	84.681	1	6.5554	16.0	W/G	LINE2
18	Reflect	0	0	1	6.5554	16.0	W/G	SHORT

is given by

$$\phi_g = \frac{2\pi\ell}{\lambda_g} , \tag{6.1}$$

where the line length is ℓ and the guide wavelength is

$$\lambda_g = \frac{\lambda}{\sqrt{1 - (\lambda/\lambda_c)^2}} . \tag{6.2}$$

For the TE_{10} mode, the cutoff wavelength λ_c is simply two times the width of the guide.

Because of the nonlinear phase shift in rectangular waveguide, we cannot use a simple center-frequency argument to determine the optimum line length. A straightforward way to find the optimum length goes as follows:

1. Calculate the line length ℓ_1 for a 20° delay at the lowest frequency of interest.

2. Calculate the line length (ℓ_2) for a 160° delay at the highest frequency of interest.

3. If $\ell_1 < \ell_2$, choose a line length anywhere between ℓ_1 and ℓ_2.

4. If $\ell_1 > \ell_2$, either reduce the frequency range of interest, or split it into two separate frequency ranges. Each frequency range then needs a separate line with its own optimum length.

Minimum Frequency

In specifying the minimum frequency of each standard, the network analyzer needs to be given the exact waveguide cutoff frequency under the conditions of operation. The waveguide is air-filled, and the cutoff frequency depends on the speed of light. To perform a calibration of the utmost accuracy, the calibration kit definitions must reflect this fact. The cutoff frequency for the TE_{10} mode is

$$f_c = \frac{c}{2A}, \tag{6.3}$$

where A is the waveguide width and the speed of light c is determined from the equations given in Sec. 5.2.1.

Offset Impedance

The line impedance of rectangular waveguide varies nonlinearly with frequency. Unlike coaxial systems a single impedance cannot be specified. To calibrate the network analyzer properly, we must set the network analyzer system impedance equal to the line impedance of the calibration standards. (This is true only if one wishes the apparent line impedance of the ports to be the same as the defined line impedance of the standards.) If we define the system and line impedances equal to 1 Ω, calibrated impedance data will be normalized to 1 Ω. Calculation of the true impedance of a device then involves simply multiplying the calibrated impedance data by the waveguide line impedance at that frequency.

6.3.2 Calibration Procedure

Network analyzer calibration mathematically removes the effects of losses and phase delays prior to the measurement ports. This permits us to define 0 dB, 0° phase reference planes at the measurement ports. S-parameter data are then measured relative to these reference planes. We define our reference planes at the cavity's coupling ports. During resonator measurements, both waveguide H-bends curve downward to feed energy to the resonator. During the calibration process, the configuration of the waveguide H-bends is necessarily different. One of the H-bends must be turned 180° This allows the two measurement ports to be connected together. To avoid misalignment and repeatability problems when the H-bend is turned over, alignment holes are placed in all flanges. This includes the coupling endplate and coaxial-to-waveguide adapters as well as the H-bends.

1. Set the network analyzer system impedance to 1 Ω.

2. Load the X-band TRL calibration kit definitions into the network analyzer.

3. Disconnect the waveguide feed system from the resonator, and turn one H-bend 180°

4. Support the waveguides so that the measurement planes are aligned. Bend the cables as little as possible. Do not twist.

5. Connect the measurement planes together.

6. Enter the TRL calibration menu by pressing the appropriate keys.

7. Press "thru." Wait for the network analyzer to finish.

8. Disconnect the through connection and attach the line. Press "line."

9. If more than one line is needed for the ANA's frequency range, connect the lines and measure as is appropriate.

10. Disconnect the standard line.

11. Connect the short to one measurement port. Measure.

12. Connect the short to the other measurement port. Measure.

13. Save the calibration in one of the network analyzers calibration registers.

6.3.3 Waveguide Port Extensions

When the network analyzer is calibrated we often wish to use port extensions to translate the calibration reference plane to the cavity's coupling apertures. If port extensions are applied the following steps should be taken.

1. Set the type of port extensions to "waveguide" and set the waveguide cutoff frequency to that calculated for a vacuum.

2. Set the relative velocity factor to the number obtained when the speed of light in the laboratory is divided by the speed of light in a vacuum. In Boulder, Colorado, this number is approximately 0.9997 as calculated from equations by Liebe [40].

3. When a resonance is being measured by the network analyzer, set the display to show phase and adjust the port extensions until the phase of the peak response (the resonant frequency) passes through zero.

Chapter 7

Sample Specifications and Measurement Procedures

7.1 Sample Specifications

This section gives specifications for resonator samples, and describes how to calculate the proper sample thickness. Because resonance conditions inside a resonator are sensitive to a sample's geometry, accurate permittivity and loss tangent measurements require stringent sample specifications. A sample must have very flat and parallel faces. Tolerances on the sample diameter are less stringent. Because the electric field near the cavity wall is small, the air gap between the sample and the wall causes minimal perturbation for nonmagnetic materials. The sample diameter is allowed to be comfortably smaller (0.2 mm) than the lowest cavity diameter which permits easy movement of the sample up and down in the cavity.

1. Thickness: $\pm 5\,\mu$m ($\pm$0.0002 in), maximum.

2. Flatness: $\pm 0.01\,\mu$m, nominal, depending on lapping capability.

3. Parallelism: $\pm 2.5\,\mu$m, ($\pm$0.0001 in).

4. Diameter: 59.7 ± 0.03 mm, (2.350 ± 0.001 in).

5. Edge breaks: Sharp to 0.5 mm, (0.02 in).

7.2 Optimum Sample Thickness

Error analyses presented in this report show a minimum measurement uncertainty when the sample thickness is an integer multiple of one-half wavelength. The axial wavenumber of a TE_{01} mode in a dielectric-loaded cylindrical waveguide is given in (2.44) as

$$\beta_1^2 = \omega^2 \epsilon_0 \mu_0 \epsilon'_R \mu'_R - \frac{t'^2_{01}}{a^2} \,. \tag{7.1}$$

The permittivity and permeability of free space are ϵ_0 and μ_0. The relative permittivity and permeability of the sample material are ϵ'_R and μ'_R. The cavity radius is a. The term $t'_{01} \cong 3.831\,705\,970\,2$ is the first nonzero root of the derivative of the zero-order Bessel function $J'_0(x) = 0$.

Using (7.1), we calculate the axial guide wavelength inside the material $\lambda_{g\epsilon} = \beta_1/2\pi$ at a desired frequency. Any integral multiple of $\lambda_{g\epsilon}/2$ yields minimum uncertainty due to sample thickness uncertainty.

7.3 Grinding and Lapping

A typical procedure for grinding a resonator sample is given below.

1. Cut out rough sample with a diamond saw. Proceed slowly to avoid cracking or stressing the sample.
2. Grind diameter to specification in a rotating grinder.
3. Coarse surface grind a face to within 0.05 mm (0.002 in) of final specification.
4. Fine surface grind to 0.01 mm (0.0005 in) of specification.
5. Polish one side to flatness specification.
6. Flip sample over and mount on optically flat tool.
7. Polish second side to final specifications.

Flatness should be checked with a laser interferometer.

Parallelism is checked with a dial indicator. If thickness differences over the surface are small, an autocollimator may be used. On optically clear parts, the overall flatness and parallelism can be checked by observation of interference fringes caused by front and rear surface reflections.

7.4 Sample Cleaning

A sample should be free of contaminants before it is measured in a cavity. Grinding can leave embedded particles and lubricants on the sample surface. Handling the sample can leave oils. Samples should be handled with lint-free gloves and cleaned with 99% pure isopropyl alcohol before being placed in the resonator.

Also, contaminants can be removed by ultrasonic cleaning. This is not a recommended practice because of possible porosity or hydrophilic nature of the material to be measured. To prevent the sample from being damaged by the ultrasonic cleaner, gauze is wrapped around the sample. After cleaning, the sample is rinsed off with distilled water and allowed to dry. A desiccator may be used if the sample is porous or hydrophilic. After cleaning and drying the sample, observe its condition under a microscope.

7.5 Measurement Procedures

7.5.1 Port Coupling Factors and Unloaded Q_0

The port coupling factors κ_i and unloaded resonance Q_0 are found from this experiment. Port coupling factors are calculated from reflection S-parameters (S_{11} and S_{22}). Transmitted power can be calculated from port coupling factors and compared to measured values.

1. Perform two-port TRL calibration.
2. Detune cavity by completely lowering bottom endplate from cavity.
3. Look at phase of S_{11} and S_{22}. Adjust port extensions until phases equal zero at resonant frequency.
4. Run software which reads S-parameters.
5. Least-squares fit S-parameters S_{11} and S_{22} to circle to find the center and radius of circle.
   (a) x-offset plus radius should equal one. The difference of this quantity from one is a measure of the resistive loss of the coupling port. If the number is very near one, network analyzer uncertainties will predominate. The x-offset plus radius is near 0.997 ± 0.005, which indicates that the correction for r_i is indeed small (no greater than the order of 0.001).

   (b) y offset shows how close zero phase adjustment is. The small y offset is a combination of uncorrected reactance of the coupling port, phase delay and ANA uncertainties.
   (c) coupling factor κ is a function of the Q-circle radii given by

$$\kappa_i = \frac{r_i}{1 - r_1 - r_2} , \tag{7.2}$$

   as discussed in Sec. 4.2.1

6. Calculate theoretical transmission coefficient from κ_1 and κ_2.

$$| T(\omega) |^2 = | S_{21} |^2 = \frac{4\kappa_1\kappa_2}{(1 + \kappa_1 + \kappa_2)^2} .$$

7. Transform transmission S-parameters, S_{21} and S_{12},
8. Perform linear least-squares fit to transformed parameters.
   (a) Slope is a function of loaded Q_L
9. Calculate unloaded Q_0 from loaded Q_L and coupling factors κ_1, κ_2.
10. Calculate transmission coefficient from (loaded or unloaded) Q and coupling factors. Compare to measured values.

7.5.2 Fixed-Frequency Method of Permittivity Measurement

To measure dielectric loss, the fixed-frequency method is more accurate than the fixed-length method because the effects of frequency-dependent variables are minimized. When the frequency is fixed, conductor skin-depth and iris coupling coefficients remain constant. The following procedure outlines how to use the program `MEAS_RES` to measure several resonances on an uncalibrated network analyzer. This allows for permittivity calculation across the entire X-band frequency range.

1. Turn on water-bath circulator and network analyzer at least an hour before starting the experiment.

2. Open and close the empty cavity at the tuner-base. Tighten the three tuner-base nuts with a fixed-torque wrench. This assures the tuner base is properly affixed. Lower the tuner-endplate about 0.1 mm and raise it again. This will assure that the sensing micrometer is properly seated. Zero the sensing micrometer.

3. Load MEAS_RES into the computer. RUN. Press "Sweep nwa." A menu will appear that gives the start, stop and step frequencies for the sweep along with other information about data routing. Change the sweep parameters as desired and press "OK/DONE."

4. Enter a description of the experiment. For example, give information that it is the empty cavity, give date and time and why you are making the measurement. As many lines may be entered as desired. When finished, ENTER a null string to begin the sweep.

5. Control the network analyzer by computer to find resonances in the specified frequency range, and print the computed values of resonant frequency and Q_L.

6. Open the resonator when the program finishes. Clean the sample. With an edge of the sample on an edge of the tuner-endplate, slide the sample laterally onto the endplate. This minimizes the amount of air between the sample and the endplate. Make sure the sample is centered. Clear samples allow one to see an interference pattern to emerge.

7. Close the tuner-base and tighten the nuts with the torque wrench. Lower the endplate at least 0.1 mm and raise it up until the measurement micrometer is once again zeroed.

8. Run MEAS_RES again, but this time press the "Measure on trigger" softkey.

9. Adjust the center frequency of the network analyzer to the first resonant frequency on the empty-cavity frequency list. Move the endplate up so that the TE_{01n} mode once again resonates at the empty cavity frequency. Begin with a large frequency span and scale. As you raise the endplate to obtain the resonant frequency, reduce the frequency span and scale until the resonance display is very fine (for low-loss materials the frequency span should be about 200 kHz and the scale should be 0.2 dB). With this configuration it is easier to make the upper and lower frequency responses symmetric about the center frequency than to try to center the peak frequency. If you suspect that there may be more than one axial half-wavelength inside the sample (the sample is thick or has high permittivity), you may have to move the micrometer up so that more than one resonant mode passes the network analyzer center frequency. If you know the approximate permittivity of the sample, you can use the Calc_freqs section of CAVITYPROG to estimate the cavity length in order to have the TE_{01n} mode re-resonate. An alternative is to use the fixed-length method to calculate permittivity. With the fixed-length method you can guess at the axial mode number until the proper one is found.

10. Press the "Zoom Peak S21" softkey once the resonant frequency is centered at the desired frequency. Mark down the micrometer reading. The computer will set up the proper span, obtain the data and calculate the resonance parameters.

11. Set up the next center frequency while the analyze calculates resonance parameters. The desired resonance can be either higher or lower than the desired center frequency. If the resonance is at a higher frequency, make sure to lower the endplate at least 0.1 mm past the necessary point, and then raise the endplate up so that the resonance is at the desired frequency. This helps to eliminate any drive-micrometer backlash.

12. Repeat the above steps for all desired frequencies. Be aware that the micrometer travel is less than 1 in, and you may not have enough travel to obtain the needed frequency.

13. Obtain both empty and with-sample resonance spectra. Load and run the `CAVITYPROG` program.

14. Calculate speed of light from barometric pressure, relative humidity and temperature.

15. Calculate the empty-cavity dimensions from the empty-cavity X-band resonance spectrum. Discount interfered modes. If $\chi^2 \geq 3\times10^{-6}$ the resonance is most likely being interfered by a nearby non-TE_{01} mode.

16. Select the fixed-frequency softkey.

17. Calculate ϵ'_R and $\tan\delta$ from the resonant frequency, the micrometer reading and the empty and with-sample resonance Qs.

18. Compute the measurement uncertainties for ϵ'_R and $\tan\delta$ by pressing the "Calc Uncert" softkey. A table of uncertainties due to the uncertainties in individual parameters will be shown. These uncertainties can be printed by pressing the "Print Uncert" softkey.

7.5.3 Fixed-Length Method of Permittivity Measurement

If measurements are taken with the network analyzer uncalibrated, the procedure for the fixed-length method is much simpler than the fixed-frequency method. The two methods yield comparable measurement accuracy for ϵ'_R, but to obtain accurate loss measurements with the fixed-length method, the network analyzer must be calibrated in order to determine Q_0. The following is a fixed-length measurement procedure that uses an uncalibrated network analyzer.

1. Open the network analyzer at the tuner base, and clean the endplate with a lint-free cloth and pure isopropyl alcohol. Close the endplate and tighten the nuts with a fixed-torque wrench.

2. Zero the endplate by lowering then raising it. Lower the endplate at least 0.1 mm and raise it back up to take out any drive micrometer backlash and to seat the sensing micrometer. Zero the sensing micrometer.

3. Run the `MEAS_RES` program, and press "Sweep nwa." Setup the start, stop and step frequencies and the minimum signal level and signal-to-noise as desired and press "DONE/OK" to begin the measurement sweep. The computer will control the network analyzer and find all resonances in the specified frequency span. If the flag `Printer_on` is true, results will be printed out at the printer specified by `Print_addr`.

4. Open the resonator and place the sample onto the endplate. Place the edge of the sample onto the edge of the endplate and slide sideways onto the endplate. This minimizes any air gaps between the sample and the endplate that may occur.

5. Close the tuner assembly and tighten the nuts with the torque wrench. Lower the endplate at least 0.1 mm and raise it up again so the sensing micrometer reads zero. The resonator length is now the same as the empty-cavity length.

6. Run `MEAS_RES` again to measure the resonance spectrum and print out measurement parameters.

7. Run the `CAVITYPROG` program. Calculate the speed of light from barometric pressure, relative humidity and temperature. Calculate cavity dimensions from the empty-cavity TE_{01} resonance spectrum. From the main menu press "Fixed length," and the fixed-length method menu will appear on the computer screen.

8. Calculate permittivity for each resonant mode by entering the empty-cavity resonant frequency and Q_L, the sample-loaded resonant frequency and Q_L and all associated uncertainties. The uncertainty in Q_L for the uncalibrated fixed-length technique is 8%. The cavity length and diameter uncertainties are approximately 0.011 mm and 0.002 mm, respectively,

as described in Sec. 4.1. After entering in all necessary parameters press the "Calc Epsilon" softkey, then press the "Calc Uncert" softkey. Uncertainties are not recomputed until you press this softkey. An uncertainty table will appear on the screen, and these results can be printed out by pressing the "Print Uncert" softkey. When you return to the fixed-length menu the uncertainties given for ϵ'_R and tanδ are the root-sum-square of all uncertainties given in the uncertainty table.

9. Repeat the above step for each mode. If you have not matched the sample-loaded mode with the correct empty-cavity mode, permittivity (ϵ'_R) results will still be correct, but loss tangent will vary with frequency.

Chapter 8

Permittivity Measurements

This chapter presents results of permittivity ϵ'_R and loss tangent $\tan\delta$ measurements made in the 60-mm diameter resonator. The first section compares results of measurements made on a reference glass material that was previously characterized by the National Bureau of Standards (now NIST). The second section gives results of measurements made on candidate reference materials. NIST is evaluating these materials for external distribution.

8.1 Check Standard Measurements

Since 1974 the National Bureau of Standards and then NIST have issued an alumino-silicate glass as a dielectric reference material. This glass was subjected to international intercomparision [43]. Three resonator samples made from the existing glass stock have been measured, and our results show excellent agreement with accepted values. For all three samples, the fixed-frequency method was used. Figure 8.1 shows permittivity ϵ'_R and Fig. 8.2 shows loss tangent $\tan\delta$ results from several measurements. The error bar shown in each plot shows the internationally agreed upon value at 9.2 GHz and its associated uncertainty.

No error corrections have been applied to these results. Coupling factors and resistive losses remain more-or-less constant when the fixed-frequency method is used. Frequency-shift corrections that affect ϵ'_R tend to cancel out, and loaded Q's (Q_L) were used to calculate loss tangent. (Boron and OH^- impurities are responsible for the decreasing permittivity and increasing loss tangent as frequency increases.)

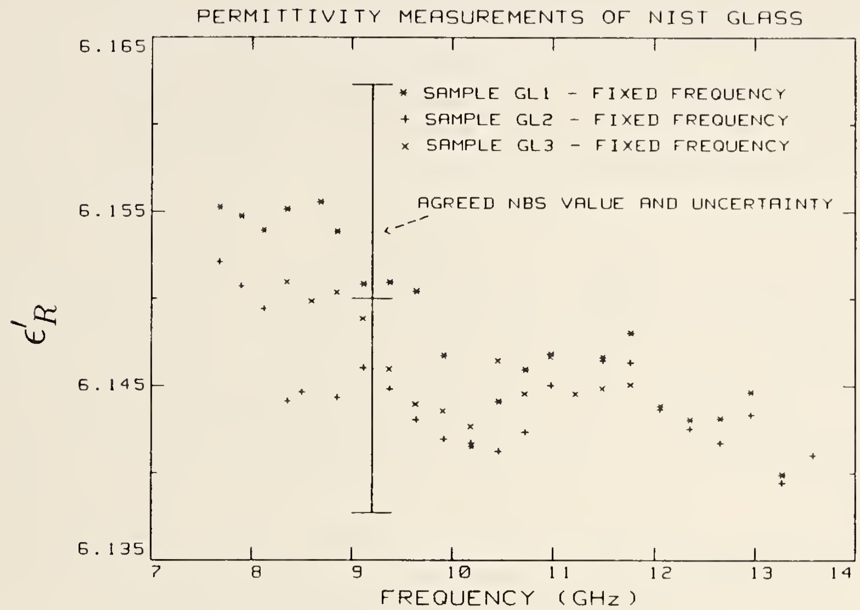

Figure 8.1: Uncorrected permittivity ϵ'_R of three glass samples measured in the NIST resonator. Both fixed-frequency, and fixed-length results compare favorably with previous results.

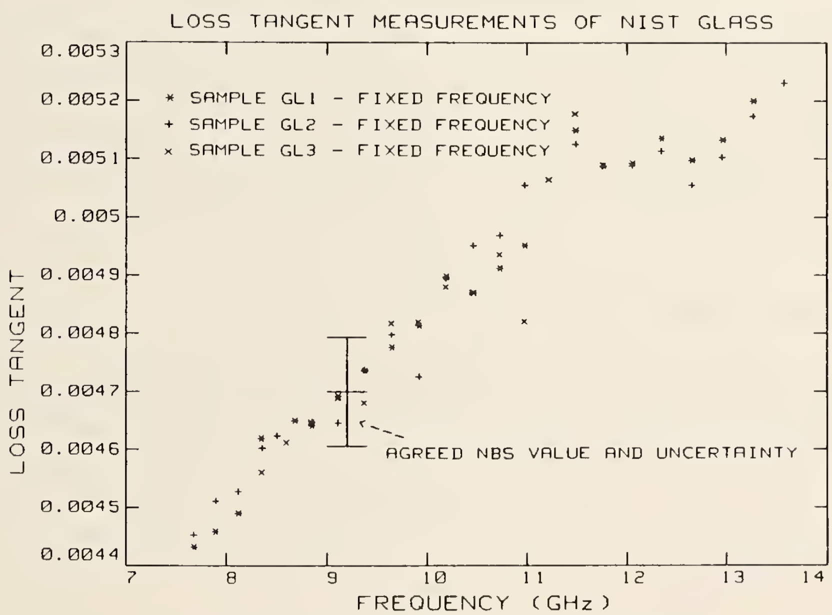

Figure 8.2: Uncorrected loss tangent $\tan\delta$ measurements of three NIST glass samples compare favorably with previous results.

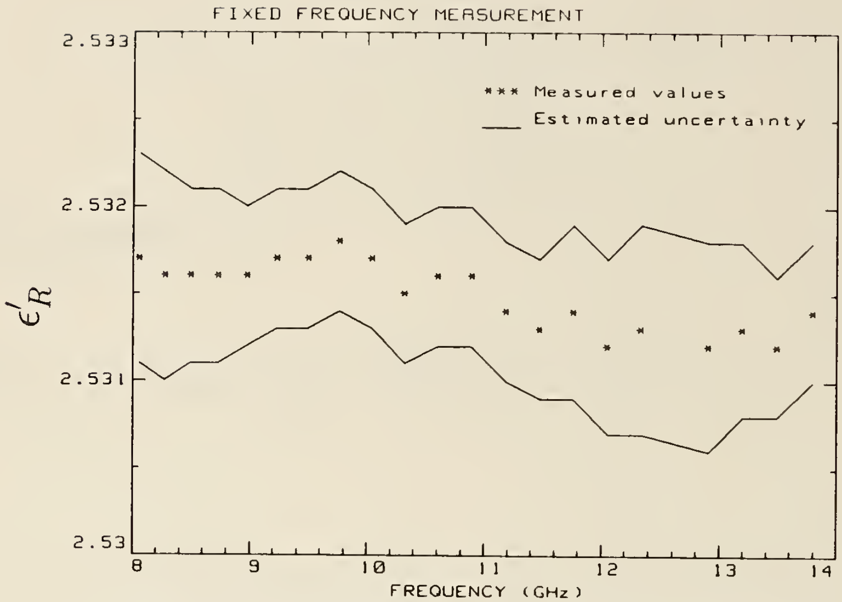

Figure 8.3: Permittivity ϵ'_R of cross-linked polystyrene.

8.2 Candidate Reference Materials

We selected three low-loss materials for evaluation and qualification as standard reference materials [4]. These materials and their nominal permittivities are: cross-linked polystyrene (2.53), fused silica (3.83) and alumina (10.0). Some criteria for selection include isotropy, homogeneity, purity and stability with frequency and temperature. After further evaluation, NIST plans to issue one or several of these materials to industry and governmental agencies.

8.2.1 Cross-linked Polystyrene

Two 19 × 60.9 × 60.9 cm (0.75 × 24 × 24 in) blocks of cross-linked polystyrene are under evaluation by NIST. Figures 8.3 and 8.4 present ϵ'_R and tanδ data from a fixed-frequency measurement.

8.2.2 Fused Silica

Three samples of recently manufactured fused silica were measured by NIST. Samples 1 and 2, from different manufacturers, are made of the highest-purity synthetic fused silica available. Sample 3 is a commercial-grade fused quartz.

Figure 8.5 gives ϵ'_R results for these three samples. Samples 1 and 2 yield nearly identical results, while sample 3 has a noticeably lower permittivity. Figure. 8.6 gives loss tangent results from these same three samples. All three samples have a noticeably different loss, and, surprisingly, the fused quartz sample has lower loss than the fused silica. We normally expect impurities to cause greater loss. The discontinuity in the results near 11.7 GHz are due to mode interference by TE_{02p} modes.

These results indicate that the permittivity and loss of high-purity, high homogeneity fused silica are consistent from manufacturer to manufacturer. However there are repeatable differences in permittivity and loss between the two fused silica samples. The fused-quartz sample has a noticeably lower permittivity and loss than the fused-silica samples. This could be because the fused-quartz sample is less dense and has more seed crystals and inclusions, or because of impurities. Also, the fixed-frequency and fixed-length results for sample 2 are nearly identical.

Uncertainty estimates for sample number 1 are given in Figs. 8.7 and 8.8.

8.2.3 Alumina

Eleven samples of 99.9% pure Al_2O_3 are under evaluation. Samples 8–11 were made separately from the first seven samples, and permittivity results for samples 1–7 are given in Figs. 8.9 and 8.10. A demonstrable difference between samples from the same batch can be seen. This is not due to measurement variations because results for Sample 6 are repeatable. Although further investigation is needed, the higher permittivity results at the lower frequencies are most likely due to our having used an effective cavity length that was too long for those frequencies (Sec. 4.1). Loss tangent for this high-purity alumina is the lowest of any material we have measured. Also, the fixed-frequency method gives stable and repeatable loss tangent results for most of the frequency range, while the fixed-length method varies much more. This is due to the limitations in the loss tangent model which assumes an equivalence between the empty cavity and a cavity containing a hypothetical lossless sample of the same permittivity as the real sample (Sec. 4.2). The negative loss tangent values near 11.5 GHz are due to the interfering TE_{021} mode. Uncertainty estimates for samples with $\epsilon'_R \approx 10$ are given in Chapter 9, so no uncertainty results are given here.

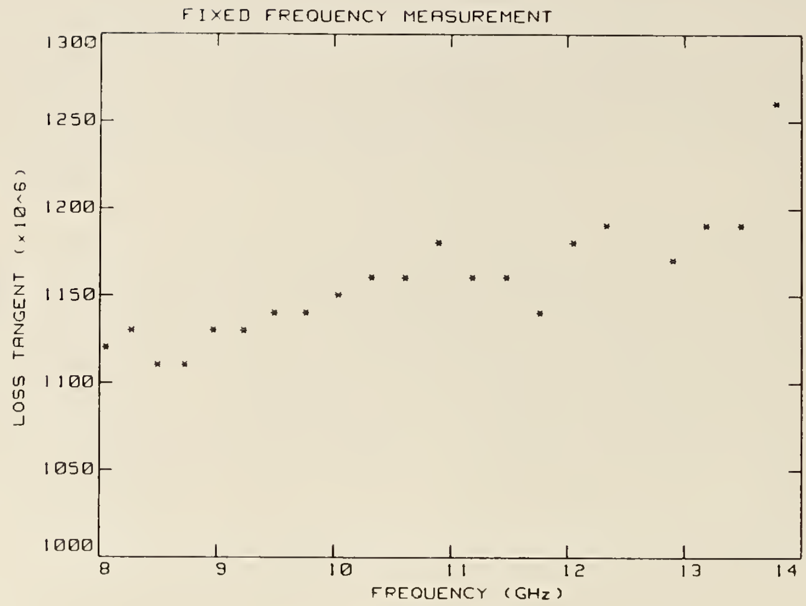

Figure 8.4: Loss tangent tanδ of cross-linked polystyrene.

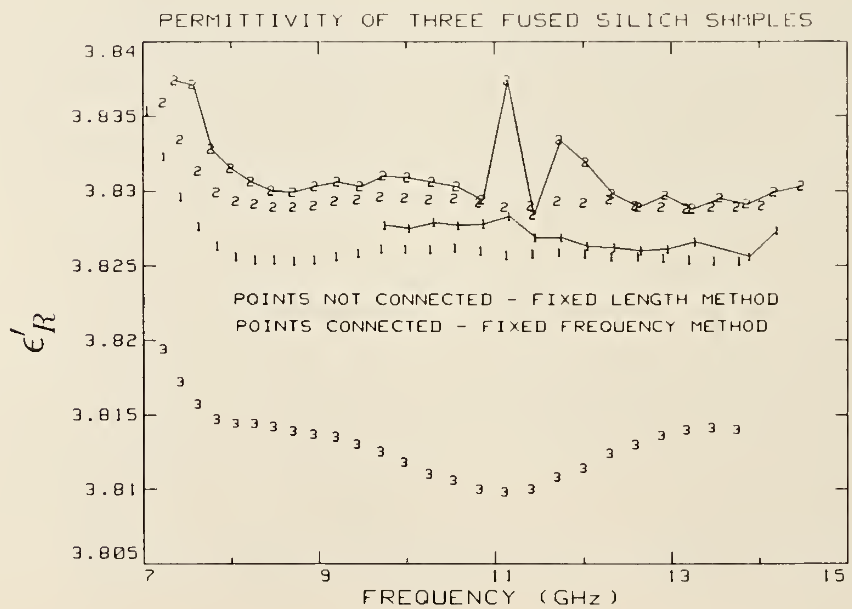

Figure 8.5: Permittivity ϵ'_R of two fused silica samples (1 and 2) and one fused quartz sample (3).

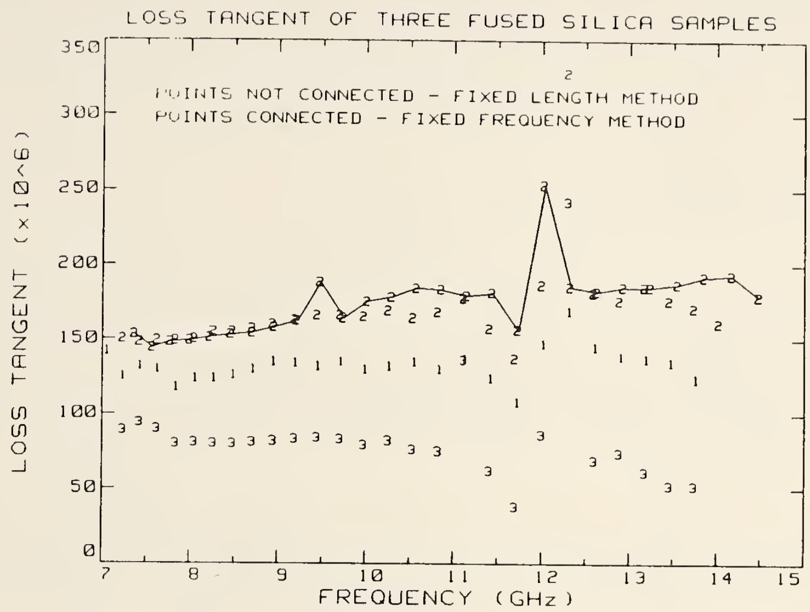

Figure 8.6: Loss tangent tanδ of two fused silica samples (1 and 2) and one fused quartz sample (3). Note effect of interference by a TE_{02} mode between 11 and 13 GHz.

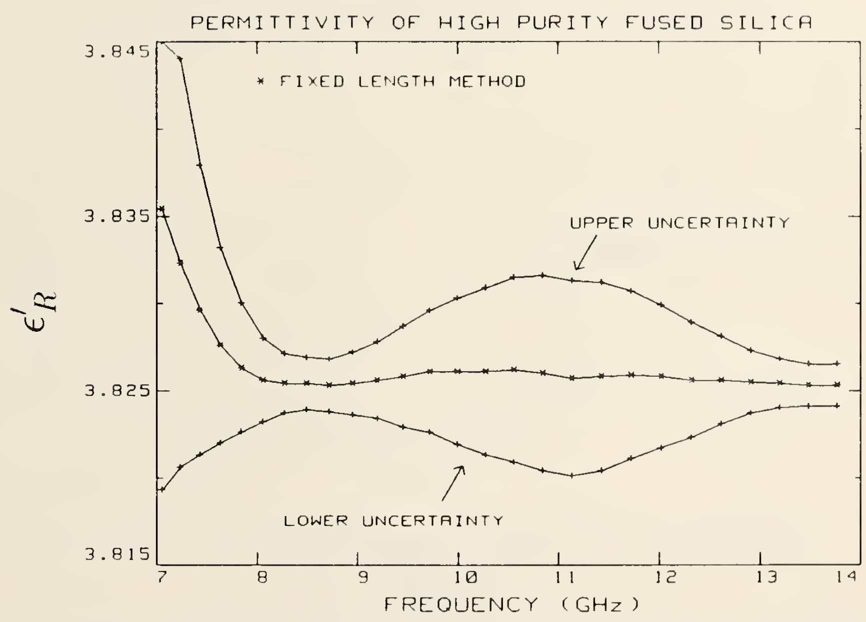

Figure 8.7: Permittivity ϵ'_R and estimated uncertainty of fused silica sample # 1.

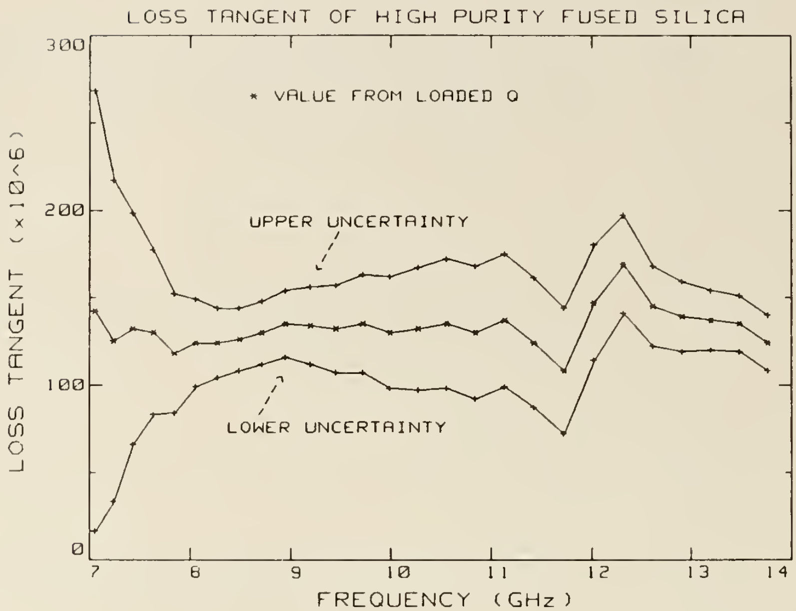

Figure 8.8: Loss tangent tanδ and estimated uncertainty of fused silica sample # 1.

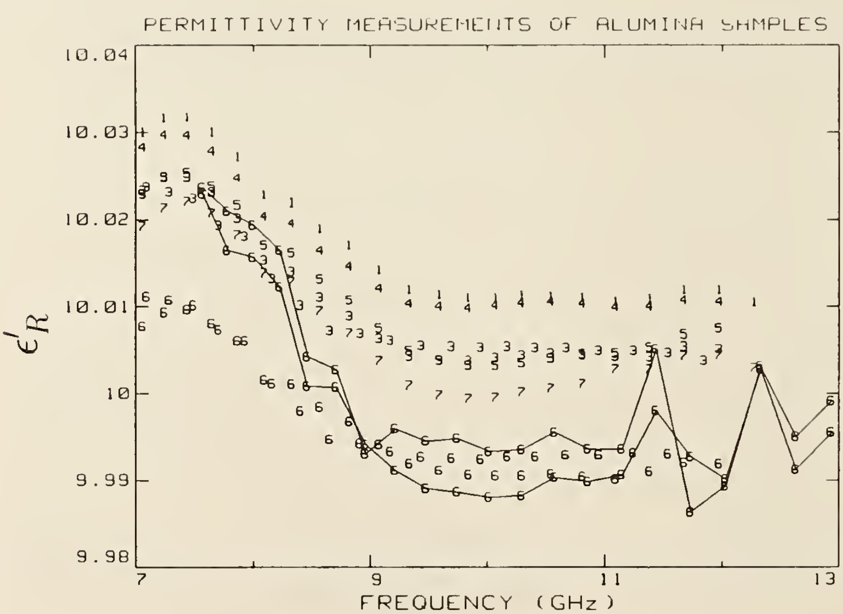

Figure 8.9: Measured permittivity ϵ'_R of several alumina samples from the same batch.

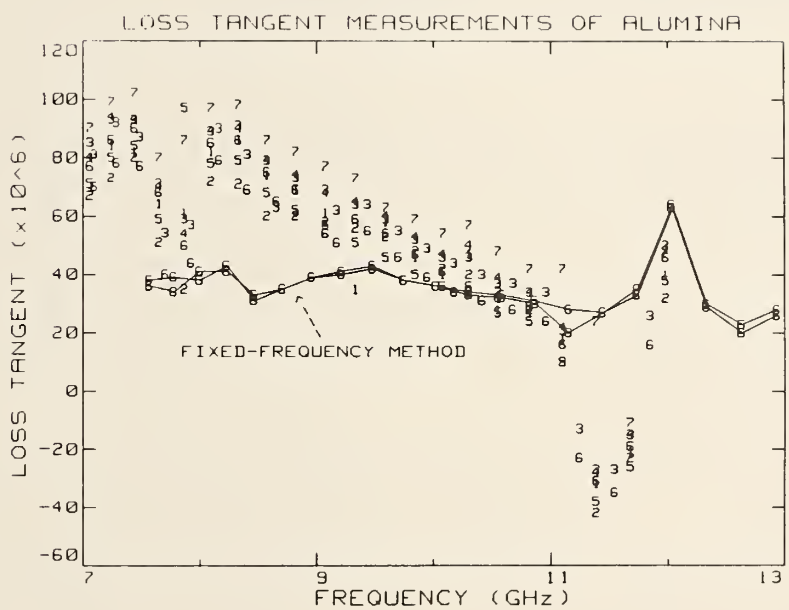

Figure 8.10: Measured loss tangent $\tan\delta$ of several alumina samples from the same batch.

Chapter 9

Uncertainty Analysis

This chapter describes how uncertainty estimates are obtained from the computer program CAVITYPROG. Typical uncertainty estimates are given for a 10-mm thick sample with a relative permittivity value of $10 - j0.001$. CAVITYPROG generates uncertainty tables for ϵ'_R and $\tan\delta$ due to individual parameter uncertainties. The uncertainties in cavity diameter and length, resonant frequency, micrometer reading, and resonance bandwidth were found in Chapter 4 and are summarized in Table 9.1.

The ϵ'_R and $\tan\delta$ results for the samples presented in Chapter 8 show estimated total uncertainty. This chapter presents estimated ϵ'_R and $\tan\delta$ uncertainties due to uncertainties in individual parameters. Because there can be any number of combinations of sample permittivity, loss, thickness, cavity length and resonant frequency, for the sake of brevity, we present results for a hypothetical sample with a relative permittivity of $10 - j0.001$. Uncertainty estimates are given for several frequencies in the X-band range. The uncertainty estimates are similar

Table 9.1: Estimated parameter uncertainties for the NIST resonator.

Cavity diameter	a	$\pm 2\,\mu$m
Cavity length	L_r	$\pm 9\,\mu$m
Micrometer reading	ΔL_r	$\pm 1\,\mu$m
Sample thickness	b	$\pm 3\,\mu$m
Resonance frequency	F_{sample}	± 1000 Hz
Speed of light	c_{air}	± 1000 m/s
Empty cavity Q	Q_0	$\pm 3\%$
With-sample Q	Q_s	$\pm 3\%$

for other permittivities, and must be evaluated on a case by case basis. For example, the measurement uncertainties for a thin sample with low dielectric constant can be much larger than one might expect (on the order of 20%), whereas a thicker sample or one with higher permittivity would have much smaller estimated measurement uncertainty.

9.1 Permittivity Uncertainty ($\Delta\epsilon'_R$)

To find uncertainties in ϵ'_R, `CAVITYPROG` calculates the change in ϵ'_R when one parameter is changed by its given uncertainty, while all other parameters are held fixed. This is repeated for each parameter to find the estimated uncertainty $\Delta\epsilon'_R$ due to individual parameters. `CAVITYPROG` then finds worst-case total uncertainty by summing the individual uncertainties. This variational method is equivalent to taking the total differential of ϵ'_R with respect to all parameters.

Figure 9.1 shows the estimated uncertainty $\Delta\epsilon'_R$ due to the individual measured parameter uncertainties given in Table 9.1.

Cavity diameter and length uncertainties are the greatest sources of uncertainty for ϵ'_R permittivity calculation. As shown previously in Fig. 4.3, the calculated cavity diameter changed less than 0.002 mm when the tuner endplate varied the cavity length by 24 mm. This result is corroborated at all frequencies in X-band as shown in Fig. 4.7 of Sec. 4.1.1 where the cavity dimensions were calculated from different subsets of the 7–14 GHz mode spectrum.

When we measure samples using the fixed-frequency method, the cavity length can be shortened by its maximum 24 mm range. From dimensional evaluation experiments described in Chapter 4, the uncertainty in diameter is less than ± 0.002 mm and the uncertainty in length is no greater than ± 0.009 mm. The overall uncertainty in cavity length is taken as the root-sum-square of the cavity length and the micrometer reading. These uncertainties are primarily due to endplate-travel accuracy (Sec. 4.1.2) and the agreement of calculated cavity dimensions from subsets of the 7–14 GHz resonance spectra (Sec. 4.1.1).

Sample thickness uncertainty is another significant error source that tends to have a maximum contribution at the frequencies at which the other error sources are minimized, as shown in Fig. 9.1. If the sample is thin < 2mm, this estimated error can become very significant especially if the sample surfaces are rough.

The speed of light in Boulder, Colorado is approximately $c_{air} = 2.997\,09 \times 10^8$ m/s at 10 GHz, which corresponds to a relative velocity factor of 0.99972. At standard temperature and pressure (23°C, 1.013×10^5 Pa (1013 mbar)) and 50% relative humidity the speed of light is approximately $2.996\,95 \times 10^8$ m/s with

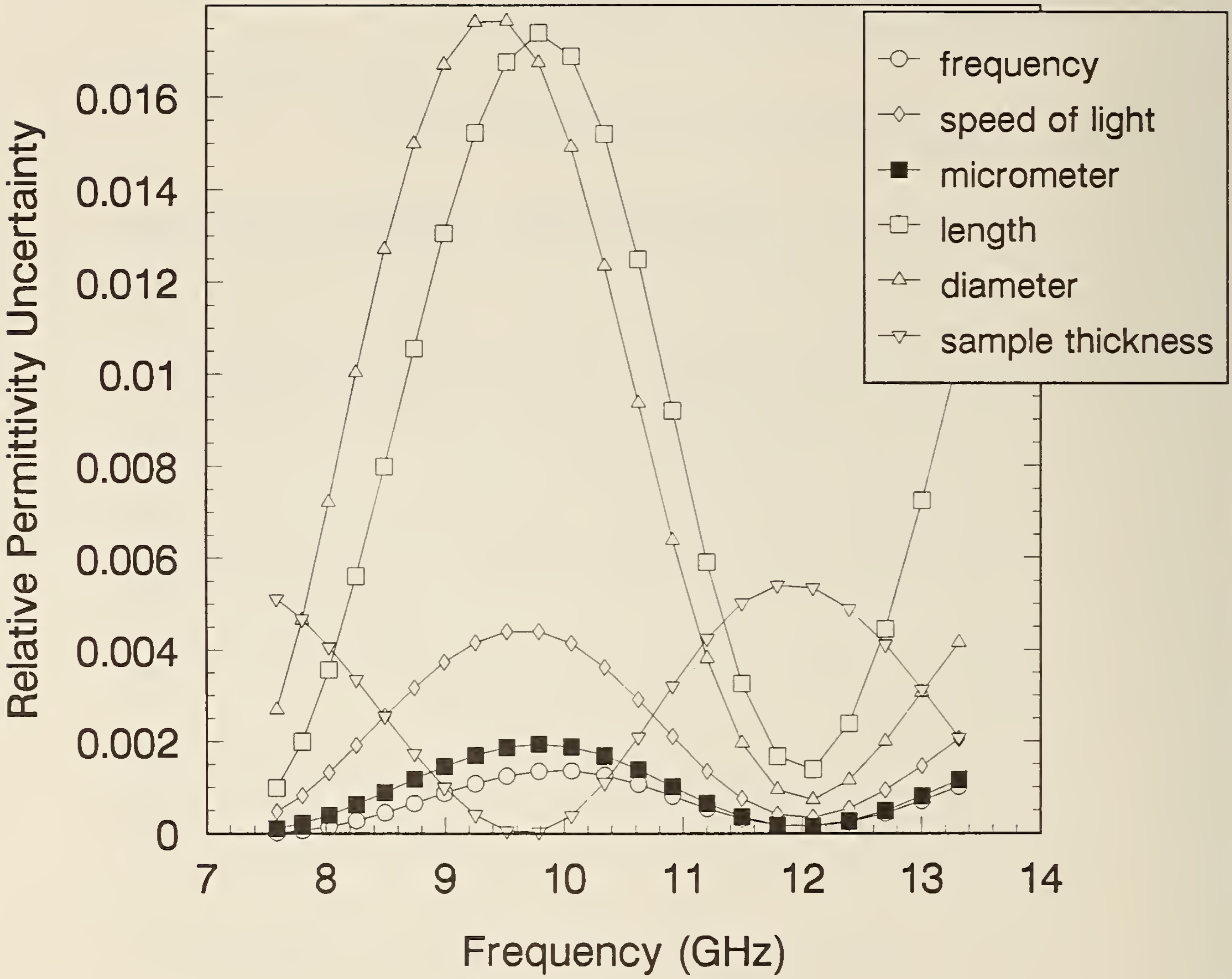

Figure 9.1: Permittivity uncertainty ($\Delta\epsilon'_R$) versus frequency due to the uncertainties in various measured parameters for a 10-mm thick sample with $\epsilon'_R = 10 - j0.001$.

relative velocity factor 0.99967. In the calculations for ϵ'_R, erroneously setting the speed of light in the air-filled portion of the cavity to the speed of light in a vacuum gives a significant upward bias to ϵ'_R results. If we use the proper value for the speed of light is calculations, the residual uncertainty in c_{air} is approximately ±300 m/s. To accommodate for miscellaneous atmospheric disturbances we have specified the uncertainty in the speed of light to be approximately three times greater ($\approx \pm1000$ m/s). Figure 9.1 shows that this uncertainty in the speed of light can indeed contribute an appreciable error to ϵ'_R.

Resonant frequency can be very accurately determined to within a few hundred hertz. However, if the resonant frequency is set and observed for one hour the resonant frequency drifts no more than 1 kHz due to changing cavity length caused by settling of the endplate drive micrometer or because of slight changes in cavity temperature. The reading of the sensing micrometer changes less than 0.001 mm to reflect this change in length. Because this problem occurs when the resonator is left alone for a period of time that is approximately the same as the time the computer takes to measure the X-band resonance spectrum, we use a 1 kHz uncertainty for the empty and sample-filled resonant frequencies. Fig. 9.1 shows the uncertainty in ϵ'_R due to a resonant frequency uncertainty of ±1 kHz. There are cases in which the Q of a resonance can be low (≈ 2000) because of a sample's extreme thickness or lossiness. In this case, resonant frequency and Q can be difficult to define because of asymmetries that may occur in the resonance response. When this is the case, one must use special precautions in defining the uncertainties in resonance frequency and Q because systematic errors having to do with the resonator's frequency dependent characteristics may be occuring.

9.2 Loss Tangent Uncertainty

Loss tangent uncertainty is calculated from the total differential of the loss tangent equation given by Cook [2]. We have verified that this method is identical to the variational method used in calculating $\Delta\epsilon'_R$ in which the uncertainties are added to individual parameters, one at a time, to find the change in $\tan\delta$. The resulting individual estimates for $\Delta\tan\delta$ are added together to get a worst-case uncertainty estimate. The total differential method can be used to estimate $\Delta\tan\delta$ because we have an explicit equation for $\tan\delta$. The equation is given by

$$\tan\delta = \frac{p(2b-s) + (1/\epsilon'_R)[2(L_r - b) - q]}{p(2b-s)} \left(\frac{1}{Q_d} - \frac{1}{Q'}\right) , \qquad (9.1)$$

where

$$p = \frac{\sin^2\beta_0(L_r - b)}{\sin^2\beta_1 b}, \tag{9.2}$$

$$q = \frac{\sin 2\beta_0(L_r - b)}{\beta_0}, \tag{9.3}$$

$$s = \frac{\sin 2\beta_1 b}{\beta_1}. \tag{9.4}$$

The total differential is found from

$$\begin{aligned}
\partial\tan\delta = \partial L_r &\left\{\frac{\partial\tan\delta}{\partial L_r} + \frac{\partial\tan\delta}{\partial p}\frac{\partial p}{\partial L_r} + \frac{\partial\tan\delta}{\partial q}\frac{\partial q}{\partial L_r}\right\} \\
+\partial b &\left\{\frac{\partial\tan\delta}{\partial b} + \frac{\partial\tan\delta}{\partial p}\frac{\partial p}{\partial b} + \frac{\partial\tan\delta}{\partial q}\frac{\partial q}{\partial b} + \frac{\partial\tan\delta}{\partial s}\frac{\partial s}{\partial b}\right\} \\
+\partial\epsilon'_R &\left\{\frac{\partial\tan\delta}{\partial\epsilon'_R} + \frac{\partial\tan\delta}{\partial p}\frac{\partial p}{\partial\beta_\epsilon}\frac{\partial\beta_\epsilon}{\partial\epsilon'_R} + \frac{\partial\tan\delta}{\partial s}\frac{\partial s}{\partial\beta_\epsilon}\frac{\partial\beta_\epsilon}{\partial\epsilon'_R}\right\} \\
+\partial Q_{\text{sample}} &\left\{\frac{\partial\tan\delta}{\partial Q_{\text{sample}}}\right\} \\
+\partial Q_{\text{empty}} &\left\{\frac{\partial\tan\delta}{\partial Q_{\text{empty}}}\right\} \\
+\partial c_{air} &\left\{\frac{\partial\tan\delta}{\partial p}\frac{\partial p}{\partial\beta_0}\frac{\partial\beta_0}{\partial c_{air}} + \frac{\partial\tan\delta}{\partial q}\frac{\partial q}{\partial\beta_0}\frac{\partial\beta_0}{\partial c_{air}}\right\} \\
+\partial a &\left\{\frac{\partial\tan\delta}{\partial p}\frac{\partial p}{\partial\beta_0}\frac{\partial\beta_0}{\partial a} + \frac{\partial\tan\delta}{\partial p}\frac{\partial p}{\partial\beta_\epsilon}\frac{\partial\beta_\epsilon}{\partial a} + \frac{\partial\tan\delta}{\partial q}\frac{\partial q}{\partial\beta_0}\frac{\partial\beta_0}{\partial a} + \frac{\partial\tan\delta}{\partial s}\frac{\partial s}{\partial\beta_\epsilon}\frac{\partial\beta_\epsilon}{\partial a}\right\} \\
+\partial f &\left\{\frac{\partial\tan\delta}{\partial p}\frac{\partial p}{\partial\beta_0}\frac{\partial\beta_0}{\partial f} + \frac{\partial\tan\delta}{\partial p}\frac{\partial p}{\partial\beta_\epsilon}\frac{\partial\beta_\epsilon}{\partial f} + \frac{\partial\tan\delta}{\partial q}\frac{\partial q}{\partial\beta_0}\frac{\partial\beta_0}{\partial f} + \frac{\partial\tan\delta}{\partial s}\frac{\partial s}{\partial\beta_\epsilon}\frac{\partial\beta_\epsilon}{\partial f}\right\}.
\end{aligned} \tag{9.5}$$

Uncertainty is found by replacing the partials ∂L_r, ∂b, $\partial\epsilon'_R$, $\partial Q_{\text{sample}}$, $\partial Q_{\text{empty}}$, ∂c_{air}, ∂a and ∂f with their respective estimated uncertainties ΔL_r, Δb, $\Delta\epsilon'_R$, ΔQ_{sample}, ΔQ_{empty}, Δc_{air}, Δa and Δf. The partial derivatives are given by

$$\frac{\partial\tan\delta}{\partial L_r} = \frac{2\left(Q_{\text{empty}} - Q_{\text{sample}}\right)}{\epsilon'_R p Q_{\text{empty}} Q_{\text{sample}}(2b - s)}, \tag{9.6}$$

$$\frac{\partial\tan\delta}{\partial b} = \frac{-2\left(Q_{\text{empty}} - Q_{\text{sample}}\right)(2L_r - q - s)}{\epsilon'_R p Q_{\text{empty}} Q_{\text{sample}}(2b - s)^2}, \tag{9.7}$$

$$\frac{\partial\tan\delta}{\partial\epsilon'_R} = \frac{(Q_{\text{empty}} - Q_{\text{sample}})(2b - 2L_r + q)}{(\epsilon'_R)^2 p Q_{\text{empty}} Q_{\text{sample}}(2b - s)}, \tag{9.8}$$

$$\frac{\partial \tan\delta}{\partial Q_{\text{sample}}} = -\frac{2b(\epsilon'_R p - 1) - \epsilon'_R ps + 2L_r - q}{\epsilon'_R p Q^2_{\text{sample}}(2b - s)}, \tag{9.9}$$

$$\frac{\partial \tan\delta}{\partial Q_{\text{empty}}} = \frac{2b(\epsilon'_R p - 1) - \epsilon'_R ps + 2L_r - q}{\epsilon'_R p Q^2_{\text{empty}}(2b - s)}. \tag{9.10}$$

The partial derivatives of p, q and s are given by

$$\frac{\partial p}{\partial \beta_0} = \frac{2(b - L_r)\sin(\beta_0(b - L_r))\cos(\beta_0(b - L_r))}{\sin^2(b\beta_\epsilon)}, \tag{9.11}$$

$$\frac{\partial p}{\partial \beta_\epsilon} = -\frac{2b\cos(b\beta_\epsilon)\sin^2(\beta_0(b - L_r))}{\sin^3(b\beta_\epsilon)}, \tag{9.12}$$

$$\frac{\partial p}{\partial b} = \frac{2\beta_0\sin(\beta_0(b - L_r))\cos(\beta_0(b - L_r))}{\sin^2(b\beta_\epsilon)} - \frac{2\beta_\epsilon\cos(b\beta_\epsilon)\sin^2(\beta_0(b - L_r))}{\sin^3(b\beta_\epsilon)}, \tag{9.13}$$

$$\frac{\partial p}{\partial L_r} = -\frac{2\beta_0\sin(\beta_0(b - L_r))\cos(\beta_0(b - L_r))}{\sin^2(b\beta_\epsilon)}, \tag{9.14}$$

$$\frac{\partial q}{\partial \beta_0} = \frac{\sin(2\beta_0(b - L_r))}{\beta_0^2} - \frac{2(b - L_r)\cos(2\beta_0(b - L_r))}{\beta_0}, \tag{9.15}$$

$$\frac{\partial q}{\partial b} = -2\cos(2\beta_0(b - L_r)), \tag{9.16}$$

$$\frac{\partial q}{\partial L_r} = 2\cos(2\beta_0(b - L_r)) \text{ and} \tag{9.17}$$

$$\frac{\partial s}{\partial \beta_\epsilon} = \frac{2b\cos(2b\beta_\epsilon)}{\beta_\epsilon} - \frac{2b\beta_\epsilon}{\beta_\epsilon^2}, \tag{9.18}$$

$$\frac{\partial s}{\partial b} = 2\cos 2b\beta_\epsilon . \tag{9.19}$$

The the wavenumbers by the guide wavelength defined as

$$\beta = \sqrt{(2\pi f)^2\mu\epsilon - \left(\frac{t'_{01}}{a}\right)^2}, \tag{9.20}$$

$$\beta_0 = \tfrac{2\pi}{\lambda_g} \sqrt{\left(\frac{2\pi f}{c_{air}}\right)^2 - \left(\frac{t'_{01}}{a}\right)^2}, \tag{9.21}$$

$$\beta_\epsilon = \tfrac{2\pi}{\lambda_{g\epsilon}} \sqrt{\left(\frac{2\pi f}{c_0}\right)^2 \epsilon'_R - \left(\frac{t'_{01}}{a}\right)^2}, \tag{9.22}$$

where $\epsilon'' \ll \epsilon'$ so that $\epsilon^* \approx \epsilon' - j\epsilon''$. The uncertainties in the wavenumbers are then found from the total differentials

$$\Delta\beta_0 = \frac{\partial\beta_0}{\partial c_{air}}\Delta c_{air} + \frac{\partial\beta_0}{\partial a}\Delta a + \frac{\partial\beta_0}{\partial f}\Delta f \text{ and} \tag{9.23}$$

$$\Delta\beta_\epsilon = \frac{\partial\beta_\epsilon}{\partial\epsilon'_R}\Delta c_{air} + \frac{\partial\beta_0}{\partial a}\Delta a + \frac{\partial\beta_0}{\partial f}\Delta f , \tag{9.24}$$

where the partial derivatives are

$$\frac{\partial\beta_\epsilon}{\partial f} = \frac{4\pi^2 f\epsilon'_R}{c_0^2}\frac{1}{\sqrt{\left(\frac{2\pi f}{c_0}\right)^2\epsilon'_R - \left(\frac{t'_{01}}{a}\right)^2}} , \tag{9.25}$$

$$\frac{\partial\beta_\epsilon}{\partial a} = \frac{(t'_{01})^2}{a^3}\frac{1}{\sqrt{\left(\frac{2\pi f}{c_0}\right)^2\epsilon'_R - \left(\frac{t'_{01}}{a}\right)^2}} , \tag{9.26}$$

$$\frac{\partial\beta_\epsilon}{\partial\epsilon'_R} = \frac{(2\pi f)^2}{2c_0^2}\frac{1}{\sqrt{\left(\frac{2\pi f}{c_0}\right)^2\epsilon'_R - \left(\frac{t'_{01}}{a}\right)^2}} , \tag{9.27}$$

and

$$\frac{\partial\beta_0}{\partial f} = \frac{4\pi^2 f}{c_{air}^2}\frac{1}{\sqrt{\left(\frac{2\pi f}{c_{air}}\right)^2 - \left(\frac{t'_{01}}{a}\right)^2}} , \tag{9.28}$$

$$\frac{\partial\beta_0}{\partial a} = \frac{(t'_{01})^2}{a^3}\frac{1}{\sqrt{\left(\frac{2\pi f}{c_{air}}\right)^2 - \left(\frac{t'_{01}}{a}\right)^2}} , \tag{9.29}$$

$$\frac{\partial\beta_0}{\partial c_{air}} = -\frac{(2\pi f)^2}{c_{air}^3}\frac{1}{\sqrt{\left(\frac{2\pi f}{c_{air}}\right)^2 - \left(\frac{t'_{01}}{a}\right)^2}} . \tag{9.30}$$

Loss tangent uncertainty due to the uncertainties in the measured parameters is shown in Fig. 9.2. We can observe a highly frequency-dependent behavior in loss tangent uncertainty due to cavity dimensions uncertainties. Typically we measure samples with much lower loss than our hypothetical 10-mm thick sample with permittivity $10 - j0.001$, and loss tangent uncertainty is usually dominated by uncertainties in the measurement of resonance Q.

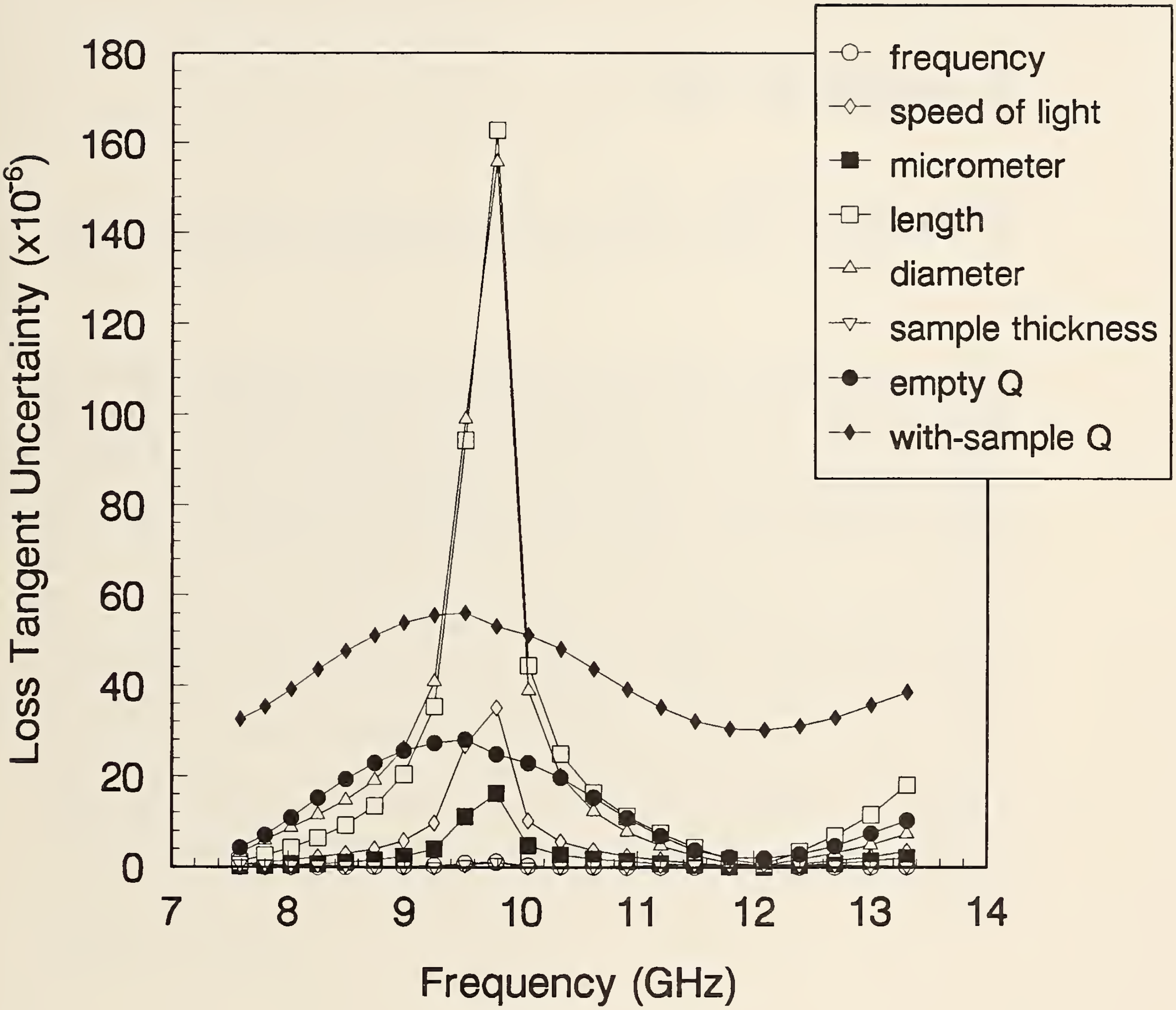

Figure 9.2: Loss tangent uncertainty $\Delta(\tan\delta)$ versus frequency due to uncertainties in various measured parameters for a 10-mm thick sample with $\epsilon'_R = 10 - j0.001$.

Chapter 10

Future Work

The change in length of the cavity is monitored by the sensing micrometer. As described in Sec. 4.1.2, the sensing micrometer's tip gradually wears a small pit into the extended finger of the tuner-endplate yoke. The yoke is made of aluminum and is therefore softer than the sensing micrometer's tip. To improve the repeatability of cavity length measurement, we will press fit a hardened steel plug into the yoke for the sensing micrometer to press against.

The uncertainty estimates given in Chapter 9 assume that systematic errors have been accounted for. To determine whether our accuracy estimates are truly correct, we must compare our results with the results of other instruments of similar accuracy. The glass results given in Sec. 8.1 compare favorably with results from an international intercomparison completed in 1974. NIST plans to engage in future international comparisons not only to qualify the NIST resonator for calibration service, but also to qualify our stock of cross-linked polystyrene, fused silica, and alumina as a Standard Reference Material.

Acknowledgements

The authors express their sincere appreciation to Hans J. Liebe of the U.S. Department of Commerce Institute for Telecommunication Services for his valuable contributions in the design and evaluation of the NIST cavity resonator. Thanks also go to Ulrich Stumper of the German Physikalisch Technische Bendesanstalt for his consultations on the initial design of the NIST resonator. We also express our appreciation to William A. Kissick. His leadership of the NIST Electromagnetic Properties of Materials Program and of the Broadband Microwave Metrology Group has contributed a great deal to the progress and successes of many individuals. Thanks also go to Howard E. Bussey, whose many contributions to the field of electromagnetic properties of materials measurements continue to be standards by which new techniques are judged.

The authors thank Lydell L. Frasch of Boeing Defense and Space Group, Charles Wilker and Ron Riegert of E.I. du Pont de Nemours & Company, and Mark D.A. Rosen and Mark Lizza of Grumman Corporate Research Center for their careful review of this Technical Note. Critical technical review of NIST publications is an important component of NIST manuscript review, and their contribution is greatly appreciated.

References Cited

[1] F. Horner, T. Taylor, R. Dunsmuir, J. Lamb, and W. Jackson, "Resonance methods of dielectric measurement at centimetre wavelengths," *J. IEE*, vol. 93, Pt. III, pp. 53–68, 1946.

[2] R.J. Cook, "Microwave cavity methods," in *High frequency dielectric measurement*, (J. Chamberlain and G.W. Chantry, ed.), pp. 12–27, Guildford, U.K., IPC Science and Technology Press, 1973, Mar. 1972.

[3] E. Ni and U. Stumper, "Permittivity measurements using a frequency-tuned microwave TE_{01} cavity resonator," *IEE Proceedings*, vol. 132, Pt. H, pp. 27–32, Feb. 1985.

[4] R. G. Geyer, "Dielectric Characterization and Reference Materials," NIST-TN 1338, Natl. Inst. Stand. Technol., 1990.

[5] R.F. Harrington, *Time-Harmonic Electromagnetic Fields*, New York: McGraw-Hill Publishing Co., 1961.

[6] S. Morgan and J. Young, "Helix waveguide," *Bell System Tech. J.*, vol. 35, pp. 1347–1384, 1956.

[7] H.G. Unger, "Helix waveguide theory and application," *Bell System Tech. J.*, vol. 37, pp. 1599–1647, 1958.

[8] D. Young, "Measured TE_{01} attenuation in helix waveguide with controlled straightness deviations," *Bell System Tech. J.*, vol. 44, pp. 273–282, 1965.

[9] R.J. Cook and R.G. Jones, "Precise Dielectric Measurement Techniques for the Frequency Range 10 GHz to 150 GHz," pp. 528–531, Proc. 8th European Microwave Conf., 1978.

[10] R. Waldron, *Theory of Guided Electromagnetic Waves*, New York: Van Nostrand Reinhold Co., 1970.

[11] R. Waldron, "A helical coordinate system and its applications in electromagnetic theory," *Quart. J. Mech. Appl. Math*, vol. 11, pp. 438–461, 1958.

[12] R. Waldron and D. Bowe, "Normal modes of helix waveguide.," *Marconi Rev.*, vol. 28, pp. 29–64, 1965.

[13] M. Shimba, "Measurement of spurious mode losses in helix waveguide," *Rev. Electron. Commun. Lab., Tokyo*, vol. 11, pp. 111–120, 1962.

[14] Y. Kazantsev, I. Kaznacheev, and V. Meriakri, "Investigation of helix waveguide," *Radio Eng. Electron. (translation of Radiotekh. Elektron.)*, vol. 4, pp. 88–94, 1959.

[15] K. Mikoshiba, "TE_{01} loss characteristics in multimode waveguide with random imperfections," *J. Inst. Electron. Comm. Eng. of Japan*, vol. 4, pp. 2–3, 1964.

[16] G. Piefke, "The influence of helix wire diameter on the modes in a helix waveguide," *IEE paper No. 2858*, vol. E, pp. 110–118, Sep. 1959.

[17] R. Waldron, D. Bowe, P. Wackrill, and B. Wescott, "Helix waveguide theory," in *Proc. IEE Conf. Transmission Aspects of Communication Networks*, pp. 210–215, 1964.

[18] K. Noda, K. Yamaguchi, and N. Suzuki, "Circular electric wave transmission through helix waveguide," *Rev. Electron. Commun. Lab., Tokyo*, vol. 10, pp. 49–70, 1962.

[19] D. Kajfez, "Correction for measured resonant frequency of unloaded cavity," *Electron. Lett.*, vol. 20, no. 2, pp. 81–82, 1984.

[20] E.L. Ginzton, *Microwave Measurements*, *International Series in Pure and Applied Physics*, New York: McGraw-Hill, 1957.

[21] M. Sucher and J. Fox, *Handbook of Microwave Measurements*, Vol. II, Brooklyn, N.Y.: Polytechnic Press, 1963.

[22] D. Kajfez and E. Hwan, "Q-factor measurement with a network analyzer," *IEEE Trans. Microwave Theory Tech.*, vol. 32, no. 7, pp. 666–670, 1984.

[23] A.J. Estin, "Resonator characterization and measurement," Cyberlink Corporation, Boulder, CO, Final Report to Sponsor, July 1990.

[24] A.J. Estin and M.D. Janezic, "Improvements in dielectric measurements with a resonant cavity," in *IEEE IMTC-S Conf. Proc.*, Atlanta, GA, May 1991.

[25] D. Kajfez, "Incremental frequency rule for computing the Q-factor of a shielded TE_{0mp} dielectric resonator," *IEEE Trans. Microwave Theory Tech.*, vol. 32, pp. 941–943, Aug. 1984.

[26] H.A. Wheeler, "Formulas for the skin effect," *Proc. IRE*, vol. 30, pp. 412–424, Sep. 1942.

[27] H.E. Bussey, J.E. Gray, E.C. Bamberger, E. Rushton, G. Russell, B.W. Petley, and D. Morris, "International comparison of dielectric measurements," *IEEE Trans. Instrum. Meas. Technol.*, vol. IM-13, pp. 305–311, Dec. 1964.

[28] J.C. Gallop and W.J. Radcliffe, "Dimensional measurement by microwave resonances," *J. Phys. E: Sci. Instrum.*, vol. 14, pp. 461–463, 1981.

[29] J.C. Slater, *Microwave Electronics*, New York: D. Van Nostrand Company, Inc., 1950, p. 75.

[30] H. Bussey, "Standards and measurements of microwave surface impedance, skin depth, conductivity and Q," *IRE Trans. Instrum.*, vol. I-9, pp. 171–175, Sep. 1960.

[31] H. Bethe and J. Schwinger, "Perturbation theory for cavities," Tech. Rep. D1 117, Natl. Defense Res. Committee, Washington, DC, Mar. 1943.

[32] P. Boggs, R. Byrd, J. Donaldson, and R. Schnabel, "Algorithm 676 — ODRPACK: software for weighted orthogonal distance regression," *ACM Trans. Mathematical Software*, vol. 15, no. 4, pp. 348–364, 1989.

[33] C. Montgomery, R. Dicke, and E. Purcell, *Principles of Microwave Circuits*. Vol. 8 of *MIT Radiation Laboratory Series*, McGraw-Hill Book Co., Inc., 1948.

[34] E. Vanzura and J. Rogers, "Resonant circuit model evaluation using reflected S-parameters," in *IEEE IMTC-S Conf. Proc.*, Atlanta, GA, May 1991.

[35] H. Nussenzvieg, *Causality and Dispersion Relations*, Vol. 95 of *Mathematics in Science and Engineering*, New York: Academic Press, Inc., 1972.

[36] Von G. Schulten, "Resonatoren fur millimeterwellen und ihre verwendung zur beobachtung von gasresonanzen," *Frequenz*, vol. 20, no. 1, pp. 10–22, 1966.

[37] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling, *Numerical Recipes: The Art of Scientific Computing*, Cambridge: Cambridge University Press, 1986.

[38] E.J. Vanzura, "Creating CSUBs written in FORTRAN that run in BASIC," in *INTEREX Proceedings of the 1988 Conference of HP Technical Computer Users*, INTEREX International Association of Hewlett-Packard Computer Users, Paper 20, 18 pages, Aug. 1988.

[39] E.J. Vanzura, "Creating CSUBs in BASIC," *HP Design and Automation*, Oct. and Nov. 1988, Austin: Wilson Publications, Inc.

[40] H. Liebe, "An updated model for millimeter wave propagation in moist air," *Radio Science*, vol. 20, pp. 1069–1089, Sep.–Oct. 1985.

[41] N.R. Draper and H. Smith, *Applied Regression Analysis*, J. Wiley and Sons, Inc., second ed., 1981.

[42] J.D. Jackson, *Classical Electrodynamics*, New York: J. Wiley and Sons, Inc., second ed., 1975.

[43] H. E. Bussey, D. Morris, and E. Zal'tsman, "International comparison of complex permittivity measurement at 9 GHz," *IEEE Trans. Instrum. Meas. Technol.*, vol. IM-23, pp. 235–239, Sep. 1974.

[44] P.C.R. Grivet *Acad. Sci. (Paris)*, vol. 218, p. 71, 1944.

[45] H. Bussey and J. Gray, "Measurement and standardization of dielectric samples," *IRE Trans. Instrum.*, vol. I-11, pp. 162–165, Dec. 1962.

Appendix A

Partial Filling Factor for Loss Determination

Previously we showed that the unloaded quality factor of the cavity with sample inserted is

$$\frac{1}{Q_{0,\text{ sample}}} = \frac{1}{Q_C} + \frac{1}{Q_S} \,. \tag{A.1}$$

where $\tan\delta = 1/Q_S$. The empty unloaded cavity quality factor is

$$\frac{1}{Q_{0,\text{ empty}}} = \frac{1}{Q_C} \,. \tag{A.2}$$

In practice, a loaded cavity quality factor is always measured, which includes coupling port losses Q_E

$$\frac{1}{Q_L} = \frac{1}{Q_0} + \frac{1}{Q_E} \,. \tag{A.3}$$

where Q_0 is either with or without sample insertion. The loaded quality factor is related to the unloaded quality factor by (2.80).

The analysis in Secs. 2.2.5 and 2.2.6 derived the theoretical Q_C and the measurement approach for evaluating total power loss in the cavity walls, as well as a technique for separating sidewall from end plate losses. However, the loss in the sidewall of the cavity is different when the sample is inserted as opposed to when the cavity is empty (air filled). Therefore, it is necessary to determine the energy stored and power dissipated both (1) in the sample of arbitrary thickness having relative complex permittivity $\epsilon_R' - j\epsilon_R''$ and (2) in the air-filled portion of the cavity when the sample is inserted. This allows us to account for the fact that the cavity is only partially filled with dielectric.

Reference [2] shows that the theoretical quality factor Q_0 of a cylindrical cavity of radius a that is partially filled with a sample of thickness b having a relative complex permittivity $\epsilon_R^* = \epsilon'_R - j\epsilon''_R$, is given by

$$Q_0 = \frac{\{F(2b-G) + [2(L-b) - U]/\epsilon'_R\} a\omega^2 \mu'_R \epsilon'_R}{\{\delta_s c^2 \left(k_c^2 [(2b-G)F + 2(L-b) - U] + 2a [F\beta_1^2 + \beta_0^2]\right) + a\omega^2 \mu'_R \epsilon'_R F(2b-G) \tan\delta\}} , \quad (A.4)$$

where c is the speed of light in air, ω is angular frequency for any TE_{01p} resonant mode, δ_s is the effective penetration depth given by (2.123) of the cavity's wall and end plates, and

$$F = \frac{\sin^2 [\beta_0(L-b)]}{\sin^2 [\beta_1 b]} , \quad (A.5)$$

$$G = \frac{\sin [2\beta_1 b]}{\beta_1} , \quad (A.6)$$

$$U = \frac{\sin [2\beta_0(L-b)]}{\beta_0} . \quad (A.7)$$

Equation (A.4) is valid for a dielectrically lossy sample and takes into account the effect of finite wall and end plate loss intrinsic to the cavity. The measured unloaded quality factor $Q_{0,m}$ with the dielectric sample is generally somewhat lower than the theoretical Q_0 in (A.4). From (2.123) and (A.4) Cook [2] has further shown that the loss tangent may be precisely calculated from

$$\tan\delta = \left\{ \frac{F(2b-G) + [2(L-b) - U]/\epsilon'_R}{F(2b-G)} \right\} \left[\frac{1}{Q_{0,m}} - \frac{1}{\bar{Q}} \right] , \quad (A.8)$$

where

$$\bar{Q} = \frac{\{F(2b-G) + [2(L-b) - U]/\epsilon'_R\} a\omega^2 \mu'_R \epsilon'_R}{\delta_{s,m} c^2 \left[k_c^2 [F(2b-G) + 2(L-b) - U] + 2a (F\beta_1^2 + \beta_0^2)\right]} . \quad (A.9)$$

Equation (A.8) is similar to previous analysis except it now contains a filling factor because the specimen only partly fills the cavity.

Appendix B

Radial Air Gap Correction

One possible source of error in dielectric property measurements at microwave frequencies is that due to the air gap between the sample under test and the wall of the waveguide or resonator fixture. The solid dielectric under test is usually machined into the shape of a parallel-sided disk for a right-circular cylindrical resonator, and clearance must be left between the sides of the disk and the wall of the resonator so that the disk may move freely in the resonator. Exactly how much clearance is permissible before the error in dielectric characterization of the sample becomes significant is a problem that must be examined.

Consider two situations for our cavity resonator in which the walls are modelled as perfect electrical conductors. The first case (Fig. B.1) illustrates a dielectric sample in a cylindrical resonator with no air gap. The second case (Fig. B.2) shows the actual situation in which an air gap exists. The presence of this gap results in an increased length of $L_1 + \Delta L$ at resonance, where $L_1 = L - b$ for the ideal case. This, in turn, leads to a measured permittivity of the sample under test that is too small.

One approach to correct for the presence of an air gap is to consider the change in resonant frequency due to the gap with perturbation theory, where the ideal situation shown in Fig. B.1 is perturbed. Since we know the measured length $L_1 + \Delta L$ at measured resonant frequency f_m and the radii of both the sample under test and dielectric resonator, we can estimate the resonant frequency for the resonator with no air gap but of length $L_1 + \Delta L$ for use in the transcendental equation for permittivity calculation. Here a method described by Grivet [44] and Bussey [45] is outlined.

We noted that at resonance the total energy interior to the cavity is constant, as a function of time and that the average values of the energy stored in the

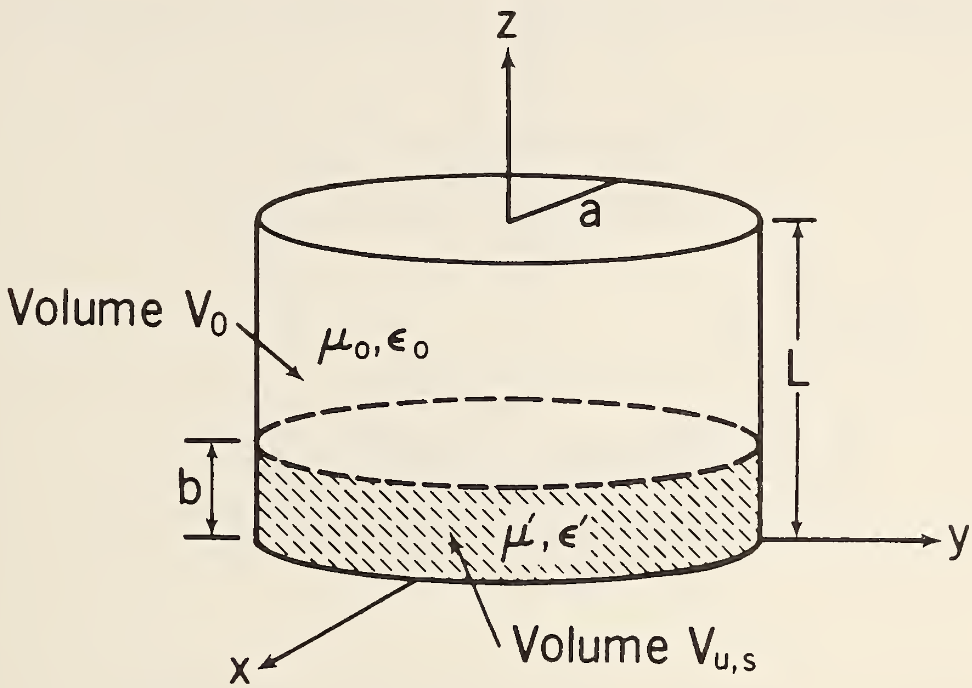

Figure B.1: Unperturbed cylindrical cavity resonator with no air gap present between sample under test and resonator side wall.

electric or magnetic field are equal. Hence, for TE_{01p} mode structure,

$$\iiint_{\text{cavity}} \epsilon' |E_\phi E_\phi^*| dV = \iiint_{\text{cavity}} \mu |\vec{H}\vec{H}^*| dV \ , \tag{B.1}$$

where the * denotes complex conjugate and the integrals are taken over the entire volume of the cavity. For a magnetically impermeable sample under test, μ may be taken outside of the above integral and since, by Faraday's law,

$$\nabla \times \vec{E} = -j2\pi f \mu \vec{H} \ , \tag{B.2}$$

(B.1) can be written

$$\iiint_{\text{cavity}} \epsilon' |\vec{E}|^2 dV = -\frac{1}{4\pi^2 f^2 \mu} \iiint_{\text{cavity}} \left[\nabla \times \vec{E}\right]^2 dV \ . \tag{B.3}$$

If we now make the assumption that the electric field in the perturbed cavity is approximately that of the unperturbed, we may write

$$\vec{E} \approx \vec{E}_u \ , \tag{B.4}$$

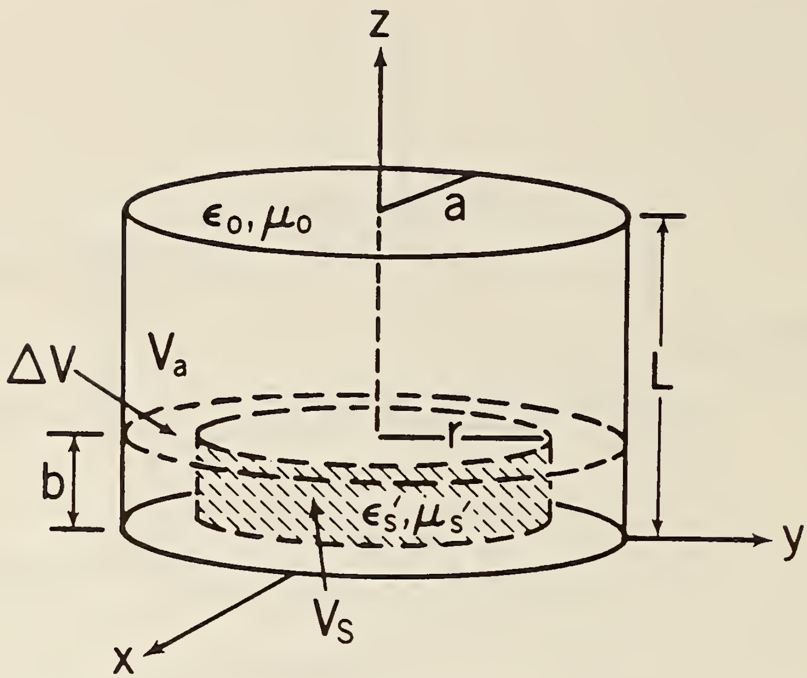

Figure B.2: Perturbed cylindrical cavity with uniform air gap between sample and resonator wall.

since we know that $\vec{E}_u$ satisfies the correct boundary conditions at the conducting walls of the cavity and is continuous at the boundary of the sample under test. This approximation is valid only if the air gap is not too large relative to λ_g and if the air gap is uniform, so azimuthal symmetry is maintained. From (B.4),

$$\nabla \times \vec{E}_u = -j2\pi f_u \mu \vec{H}_u \ , \tag{B.5}$$

where f_u denotes the unperturbed resonant frequency. Hence,

$$\begin{aligned} f_m^2 \Bigg\{ \iiint_{V_0} \epsilon_0 |\vec{E}_{u,\text{ air}}|^2 dV + \iiint_{V_{u,s}-\Delta V} \epsilon_s' |\vec{E}_{u,s}|^2 dV \\ + \iiint_{\Delta V} \epsilon_0 |\vec{E}_{u,s}|^2 dV \Bigg\} = \\ f_u^2 \Bigg\{ \iiint_{V_0} \mu_0 |\vec{H}_{u,\text{ air}}|^2 dV + \iiint_{V_{u,s}-\Delta V} \mu_0 |\vec{H}_{u,s}|^2 dV \\ + \iiint_{\Delta V} \mu_0 |\vec{H}_{u,s}|^2 dV \Bigg\} \ , \end{aligned} \tag{B.6}$$

where f_m represents the measured resonant frequency, the subscripts u and s denote unperturbed and sample, and where the integration over ΔV is an integration over the volume of the air gap which is

$$\Delta V = \pi a^2 b - \pi r^2 b = \pi b \left(a^2 - r^2 \right) \ , \tag{B.7}$$

where a and r are the respective radii of the resonator and the sample under test. Equation (B.6) may be simplified as follows:

$$f_m^2\left\{\iiint_{V_0}\epsilon_0|\vec{E}_{u,\text{ air}}|^2dV+\iiint_{V_{u,s}}\epsilon_s'|\vec{E}_{u,s}|^2dV\right.$$
$$\left.+\iiint_{\Delta V}(\epsilon_0-\epsilon_s')\,|\vec{E}_{u,s}|^2dV\right\}$$
$$= f_u^2\left\{\iiint_{V_0}\mu|\vec{H}_{u,\text{ air}}|^2dV+\iiint_{V_{u,s}}\mu|\vec{H}_{u,s}|^2dV\right\}. \tag{B.8}$$

We now note that

$$f_u^2\left\{\iiint_{V_0}\mu|\vec{H}_{u,\text{ air}}|^2dV+\iiint_{V_{u,s}}\mu|\vec{H}_{u,s}|^2dV\right\}$$
$$\equiv f_m^2\left\{\iiint_{V_0}\epsilon_0|\vec{E}_{u,\text{ air}}|^2dV+\iiint_{V_{u,s}}\epsilon_s'|\vec{E}_{u,s}|^2dV\right\}, \tag{B.9}$$

from (B.1), so

$$f_m^2\left\{\iiint_{V_0}\epsilon_0|\vec{E}_{u,\text{ air}}|^2dV+\iiint_{V_{u,s}}\epsilon_s'|\vec{E}_{u,s}|^2dV\right.$$
$$\left.-\iiint_{\Delta V}(\epsilon_s'-\epsilon_0)\,|\vec{E}_{u,s}|^2dV\right\}$$
$$= f_u^2\left\{\iiint_{V_0}\epsilon_0|\vec{E}_{u,\text{ air}}|^2dV+\iiint_{V_{u,s}}\epsilon_s'|\vec{E}_{u,s}|^2dV\right\}. \tag{B.10}$$

We now have a correction formula for the unperturbed resonant frequency f_u in terms of the measured resonant frequency f_m in the presence of an air gap of volume ΔV; that is,

$$f_u=f_m\left\{1-\frac{\iiint_{\Delta V}(\epsilon_s'-\epsilon_0)\,|\vec{E}_{u,s}|^2dV}{\iiint_{V_0}\epsilon_0|\vec{E}_{u,\text{ air}}|^2dV+\iiint_{V_{u,s}}\epsilon_s'|\vec{E}_{u,s}|^2dV}\right\}^{1/2}. \tag{B.11}$$

Of course, when there is no air gap $\Delta V = 0$ and $f_u = f_m$. For the mode-filtered TE_{01p} cavity of concern here, we have only an azimuthal component of the electric field that is of concern in the above integrations.

If an air gap is present between the sample under test and wall of the waveguide of the cavity resonator, the cavity will be at resonance at an increased length (compared to the situation where no gap exists). This leads to a measured dielectric constant that is too small. The procedure from the analysis above, which is valid if the air gap is not too large relative to the guide wavelength and if the air gap is uniform so that azimuthal symmetry is maintained, is as follows:

1. Measure the resonant frequency f_m and measured length of the cavity at resonance.

2. Compute the unperturbed resonant frequency f_u for a resonator with no air gap but with the length measured in step 1 above.

3. Use the unperturbed frequency derived in step 2 for ϵ'_R determination in (2.46).

An alternate form of (B.11) for computing the correction to the measured relative dielectric constant in the presence of a radial air gap for TE_{01p} mode is

$$\Delta\epsilon'_R = (\epsilon'_R - 1)\frac{\int_r^a \int_0^{2\pi} \left[J_1(k_c r_1)\right]^2 \sin^2(\beta_1 z)\, r_1 dr_1 d\phi}{\int_0^a \int_0^{2\pi} \left[J_1(k_c r_1)\right]^2 \sin^2(\beta_1 z)\, r_1 dr_1 d\phi}, \tag{B.12}$$

which reduces to

$$\Delta\epsilon'_R = \frac{(\epsilon'_R - 1)}{2}\left\{1 - \left(\frac{r}{a}\right)^2 \frac{\Psi}{J_0^2(k_c a)}\right\}, \tag{B.13}$$

with

$$\Psi = J_0^2(k_c r) - \frac{2}{k_c r} J_0(k_c r) J_1(k_c r) + J_1^2(k_c r)\ . \tag{B.14}$$

Appendix C

60-mm Cavity Drawings

To be consistent with shop tooling, dimensions for resonator parts were specified in inches. Apologies are offered for our deviation from Standard International units. In the following drawings, dimensions are in inches unless otherwise specified.

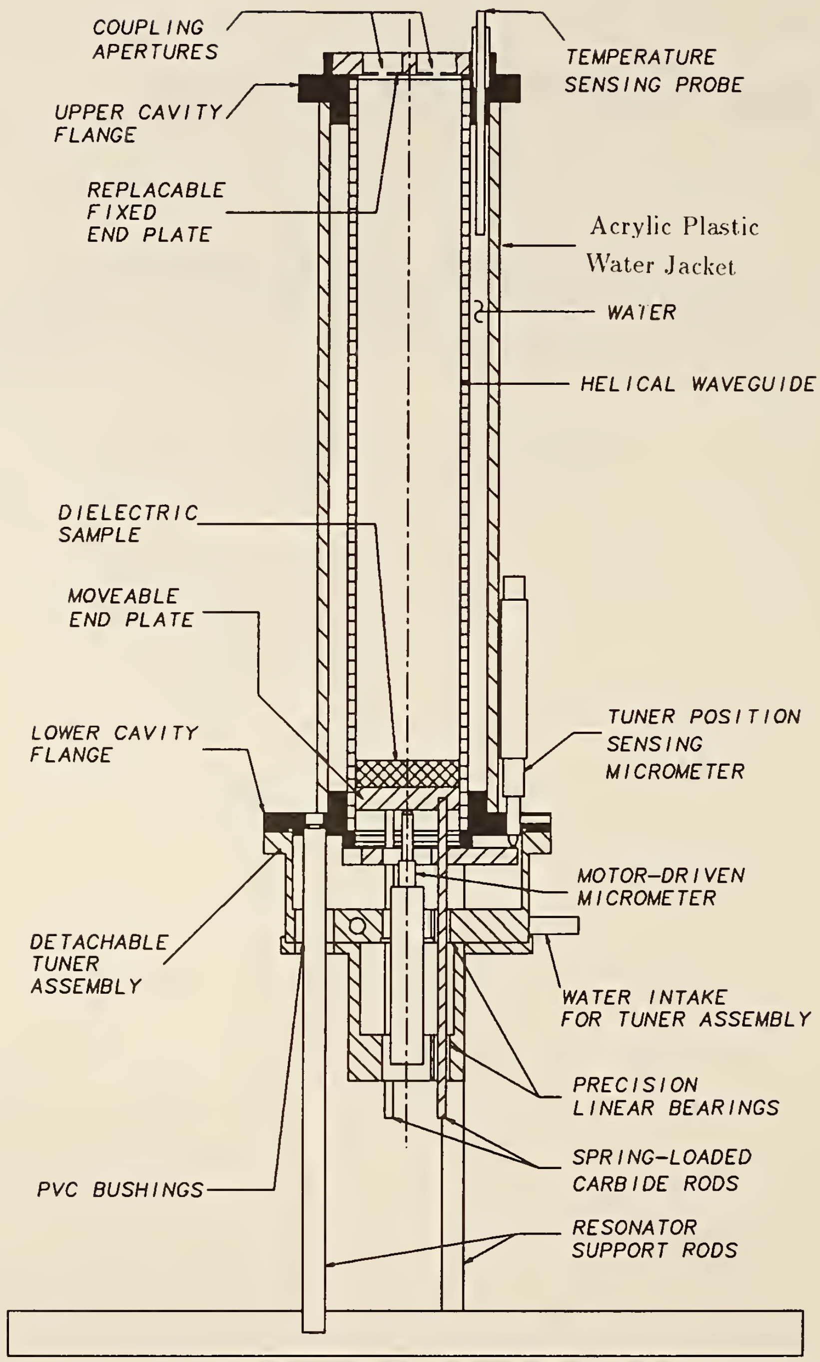

Figure C.1: Cutaway assembly drawing of NIST cavity resonator.

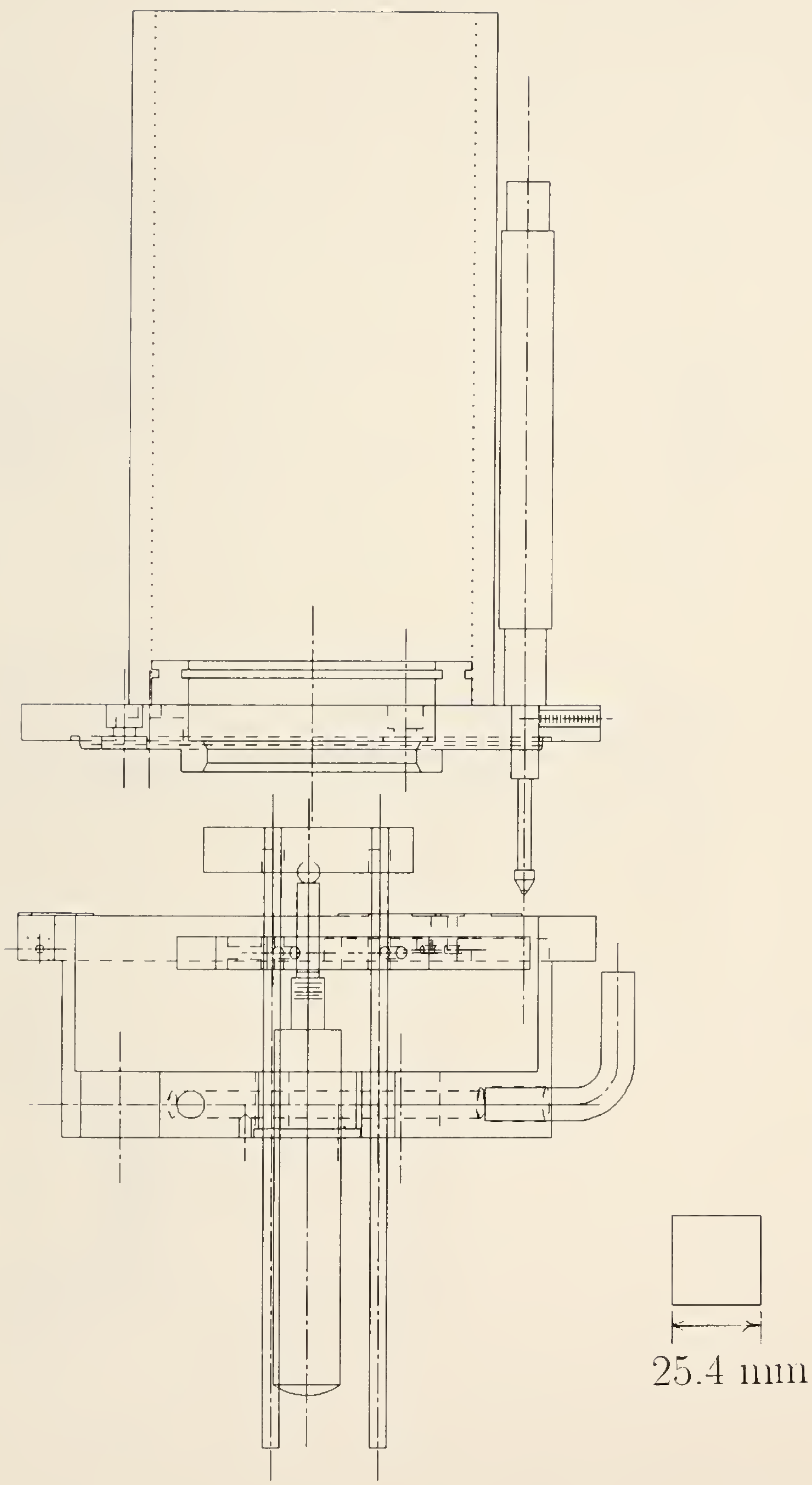

Figure C.2: Detail of tuner assembly and flange.

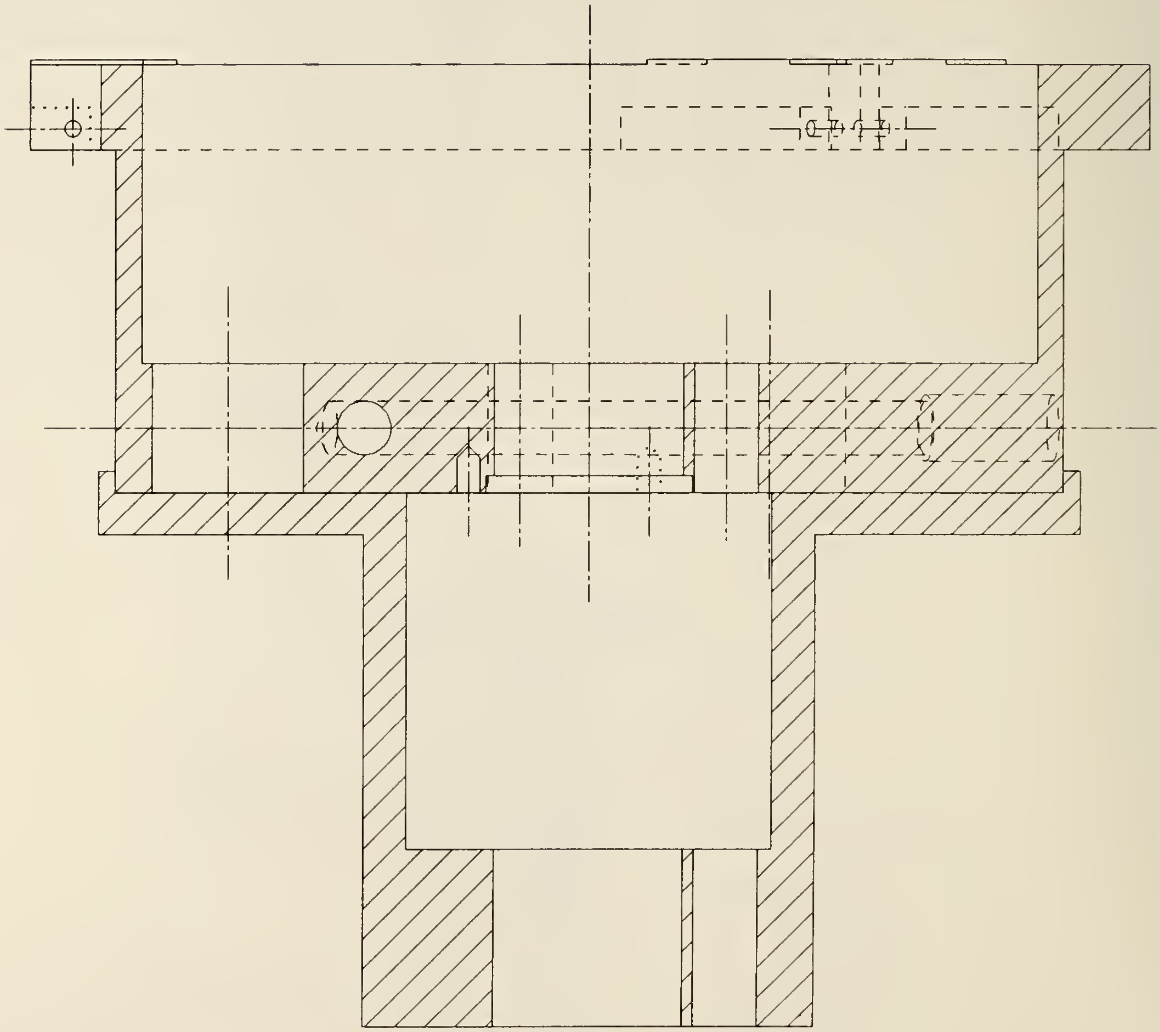

Figure C.3: Cutaway side view of tuner-base and extension.

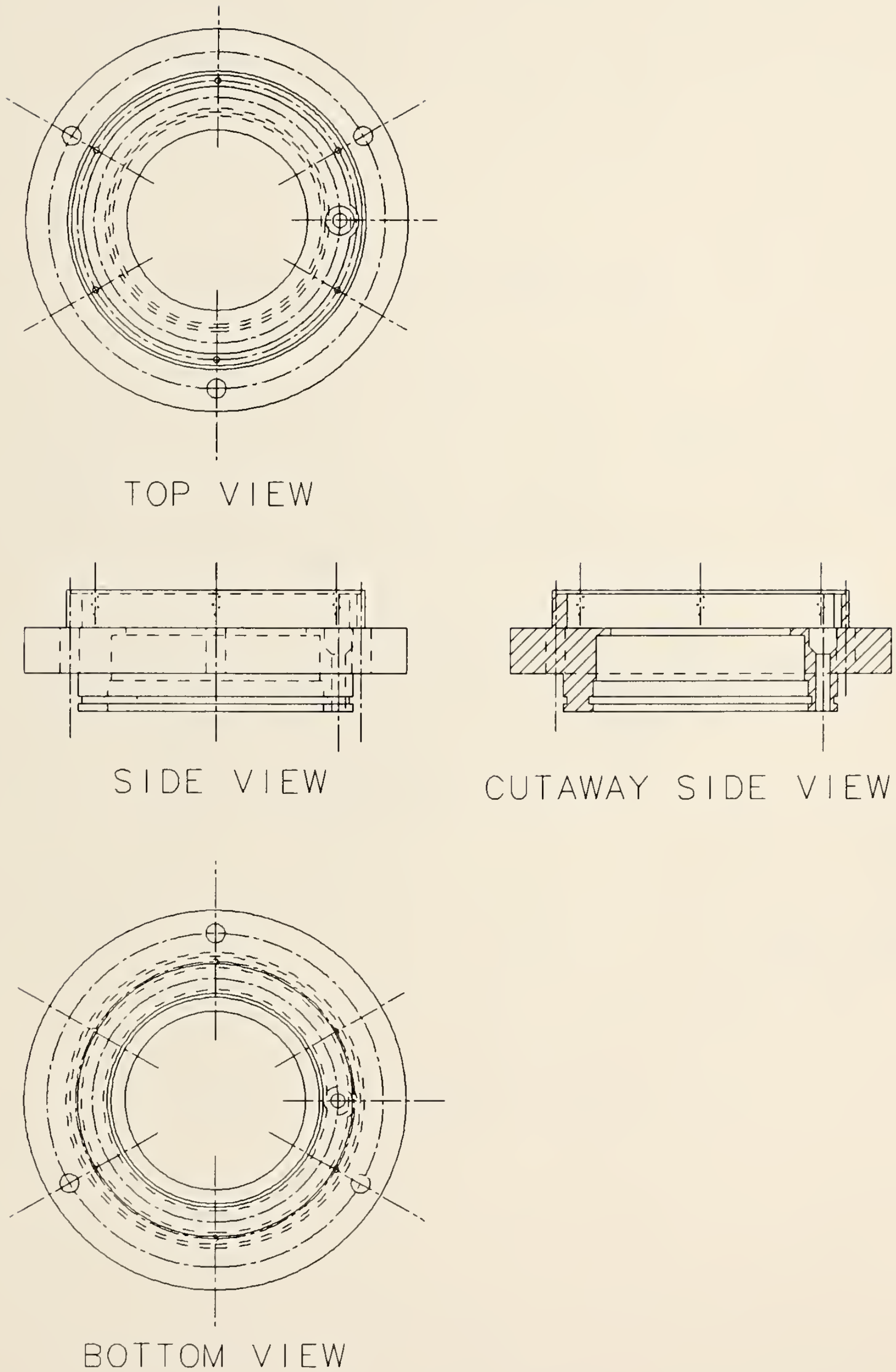

Figure C.4: Orthogonal views of coupling flange.

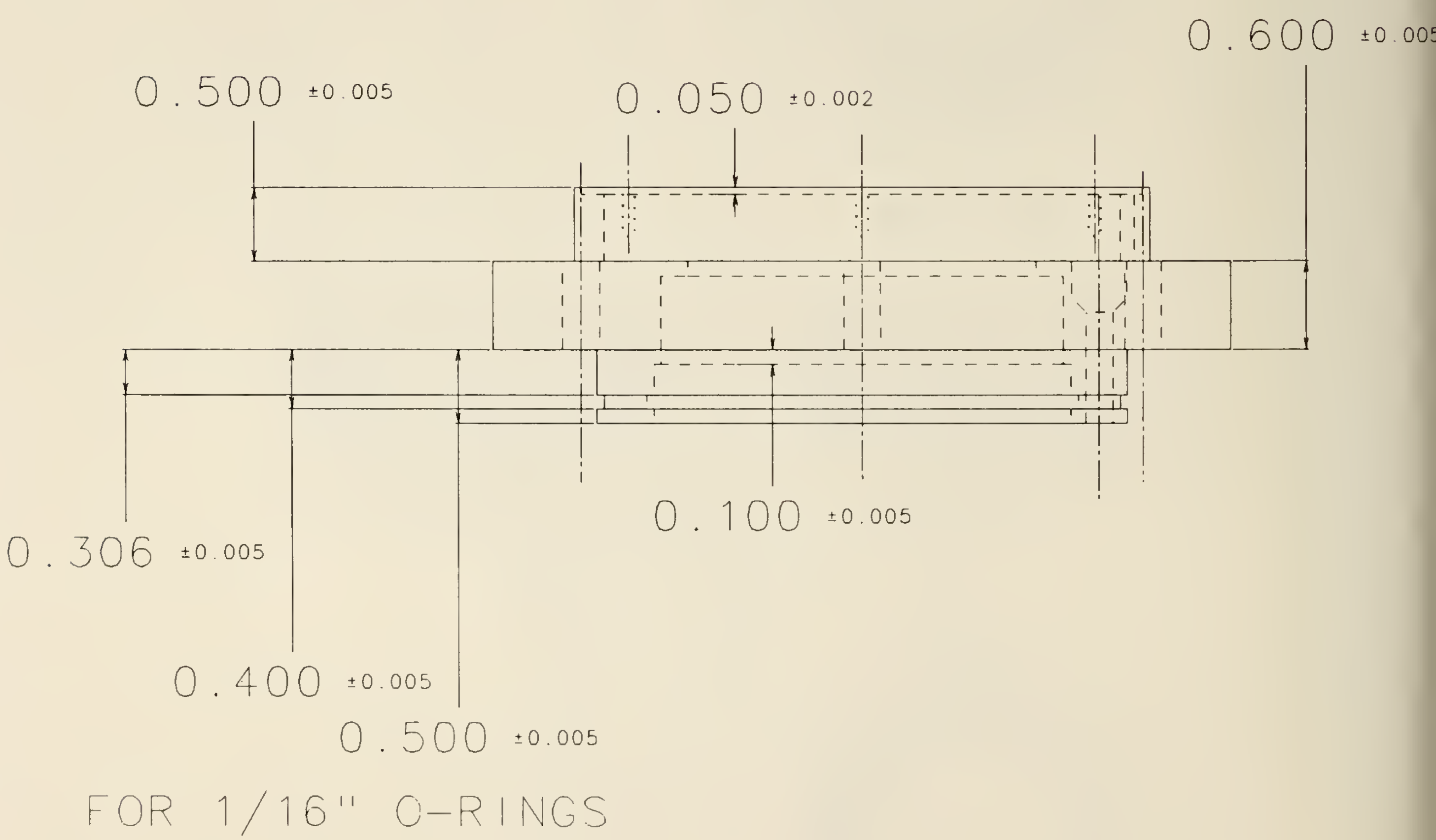

Figure C.5: Side view of coupling flange.

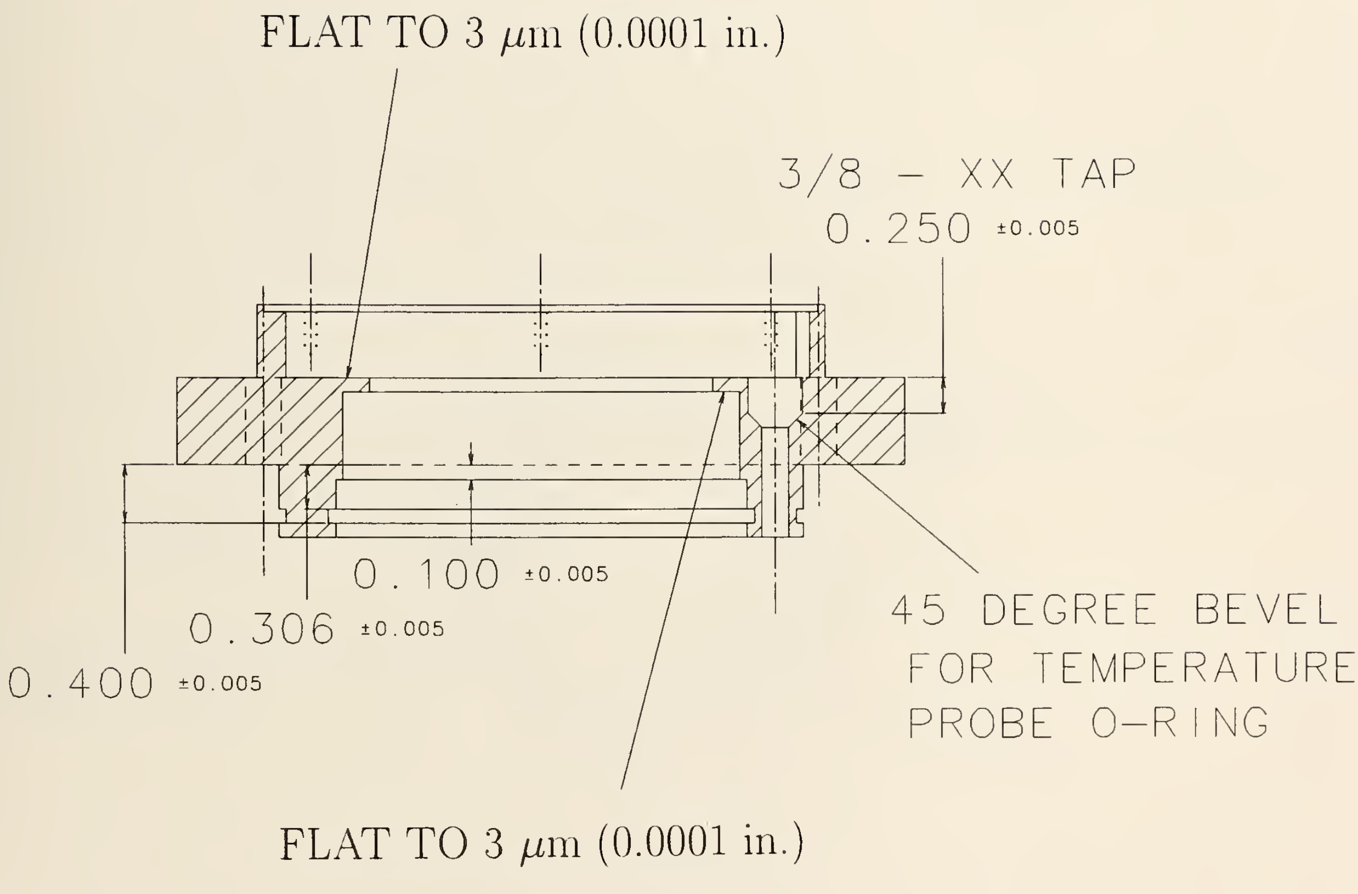

Figure C.6: Cutaway side view of coupling flange.

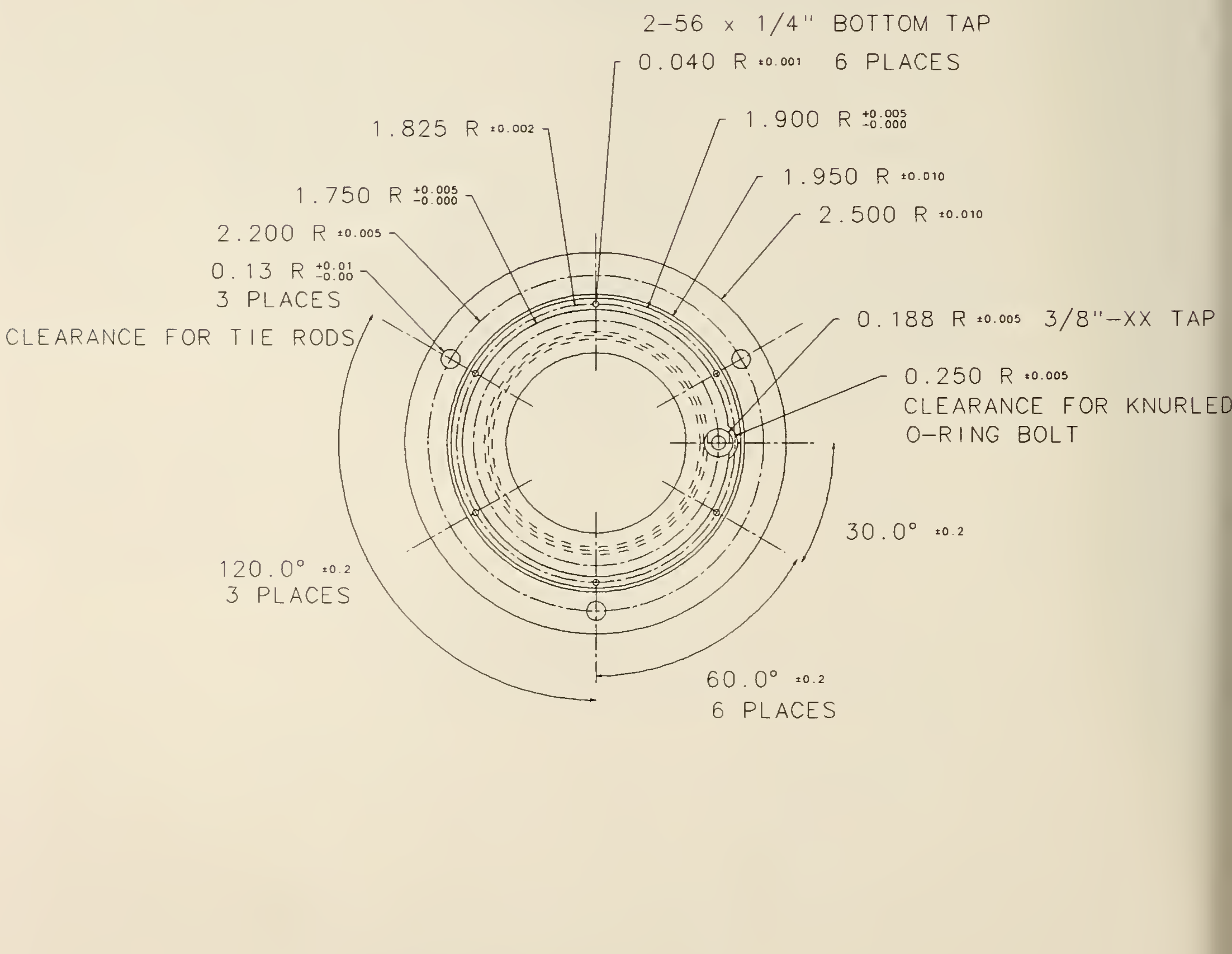

Figure C.7: Top view of coupling flange.

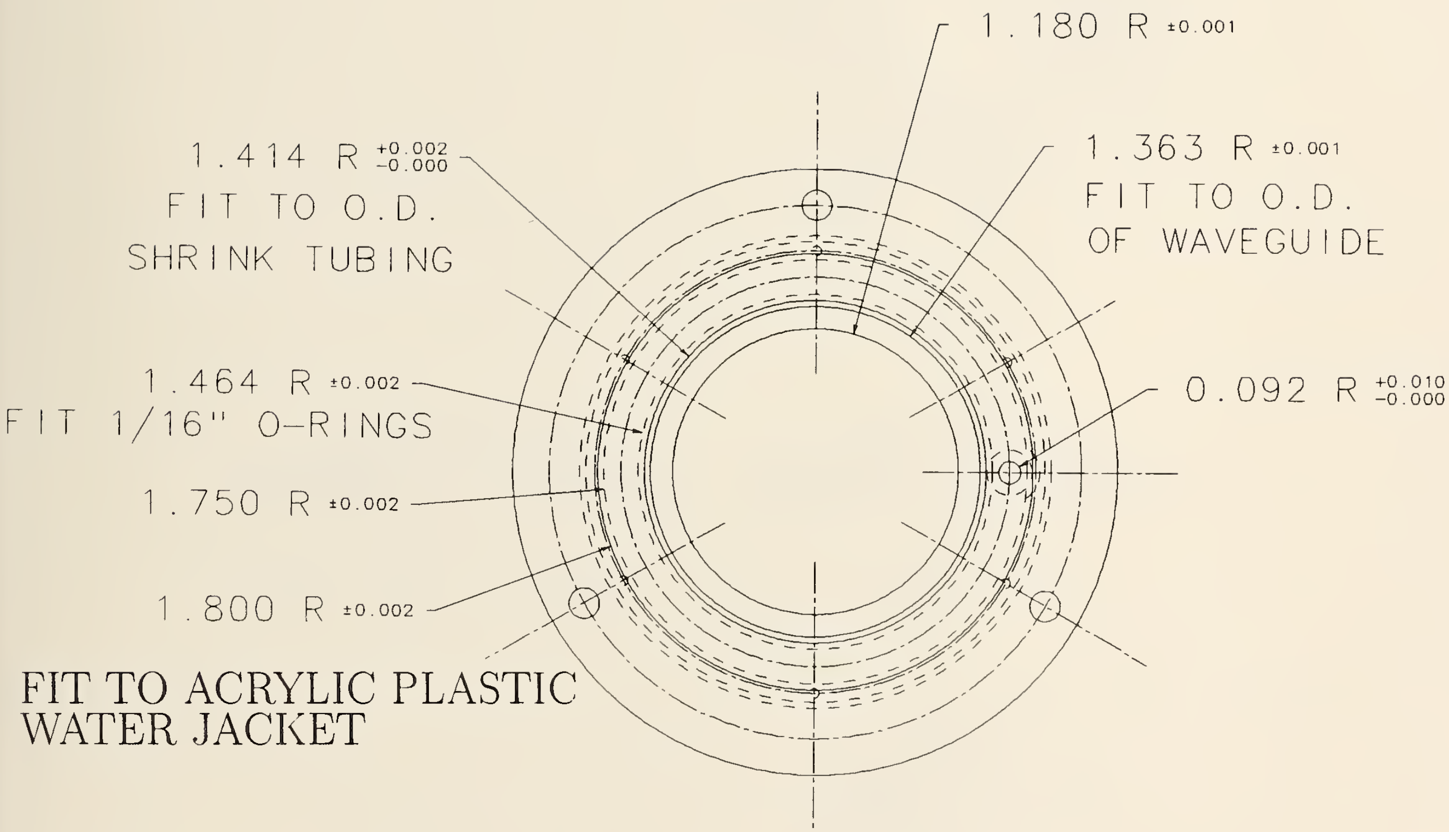

Figure C.8: Bottom view of coupling flange.

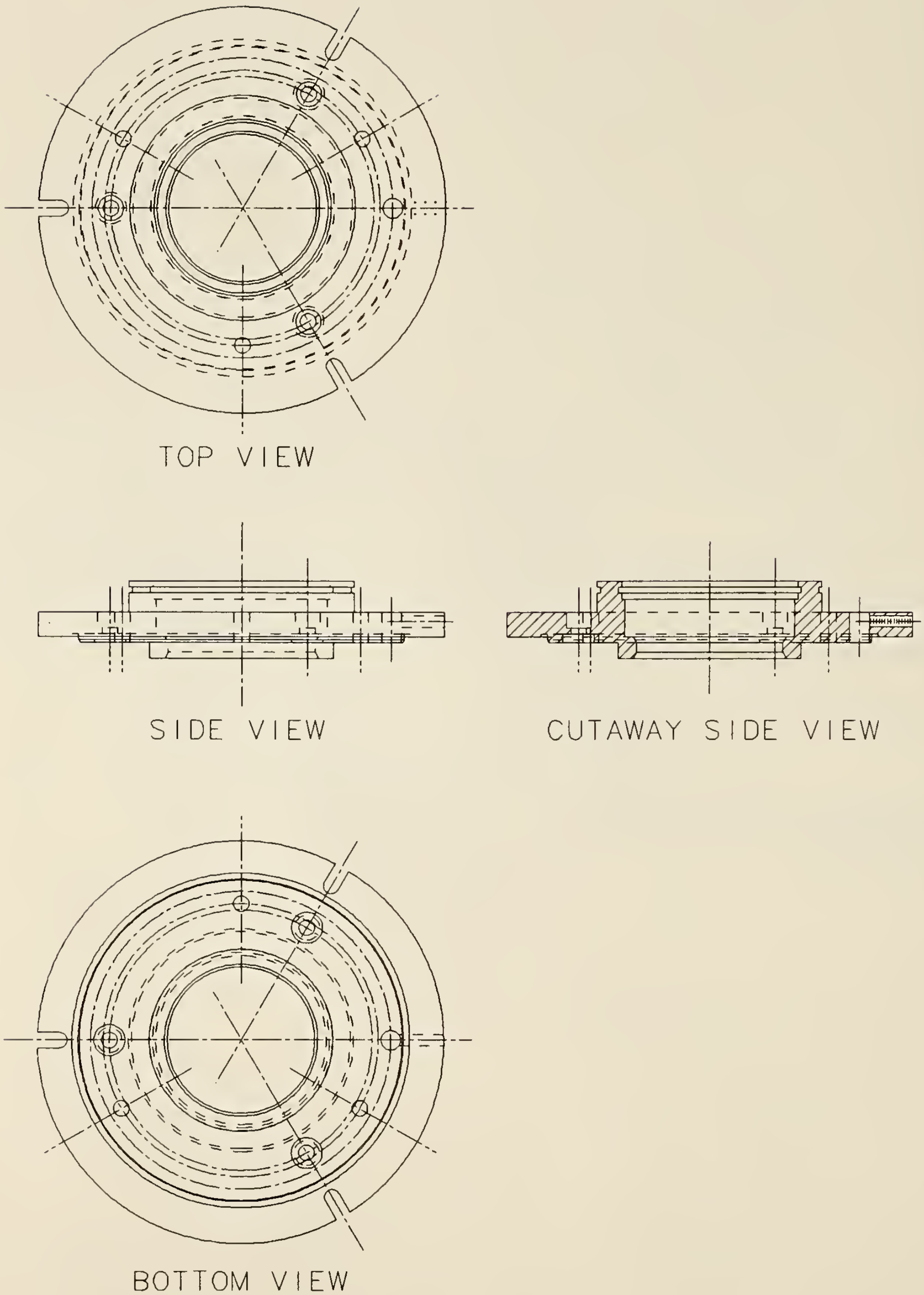

Figure C.9: Orthogonal views of tuner flange.

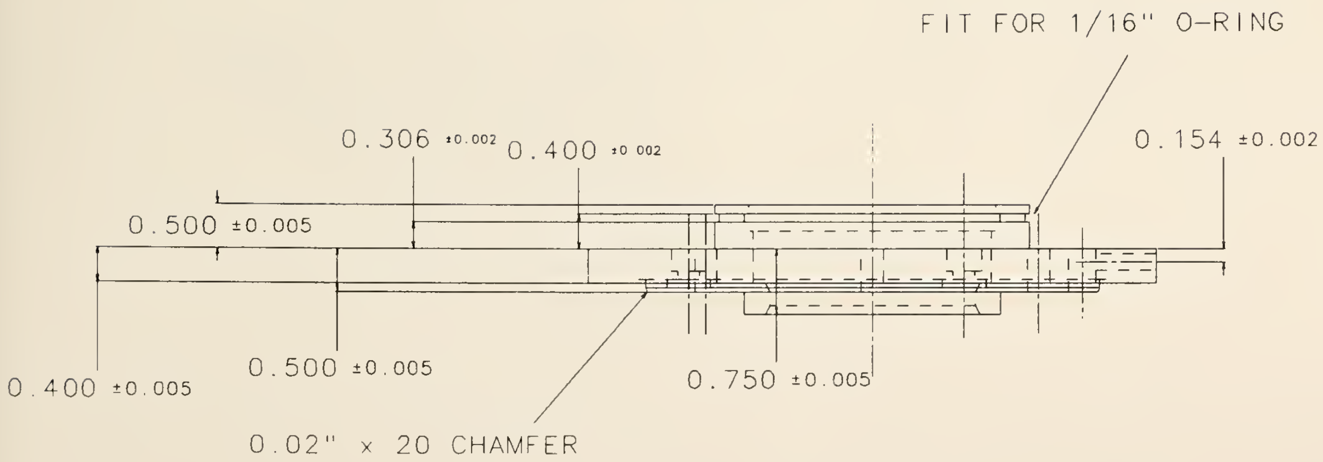

Figure C.10: Side view of coupling flange.

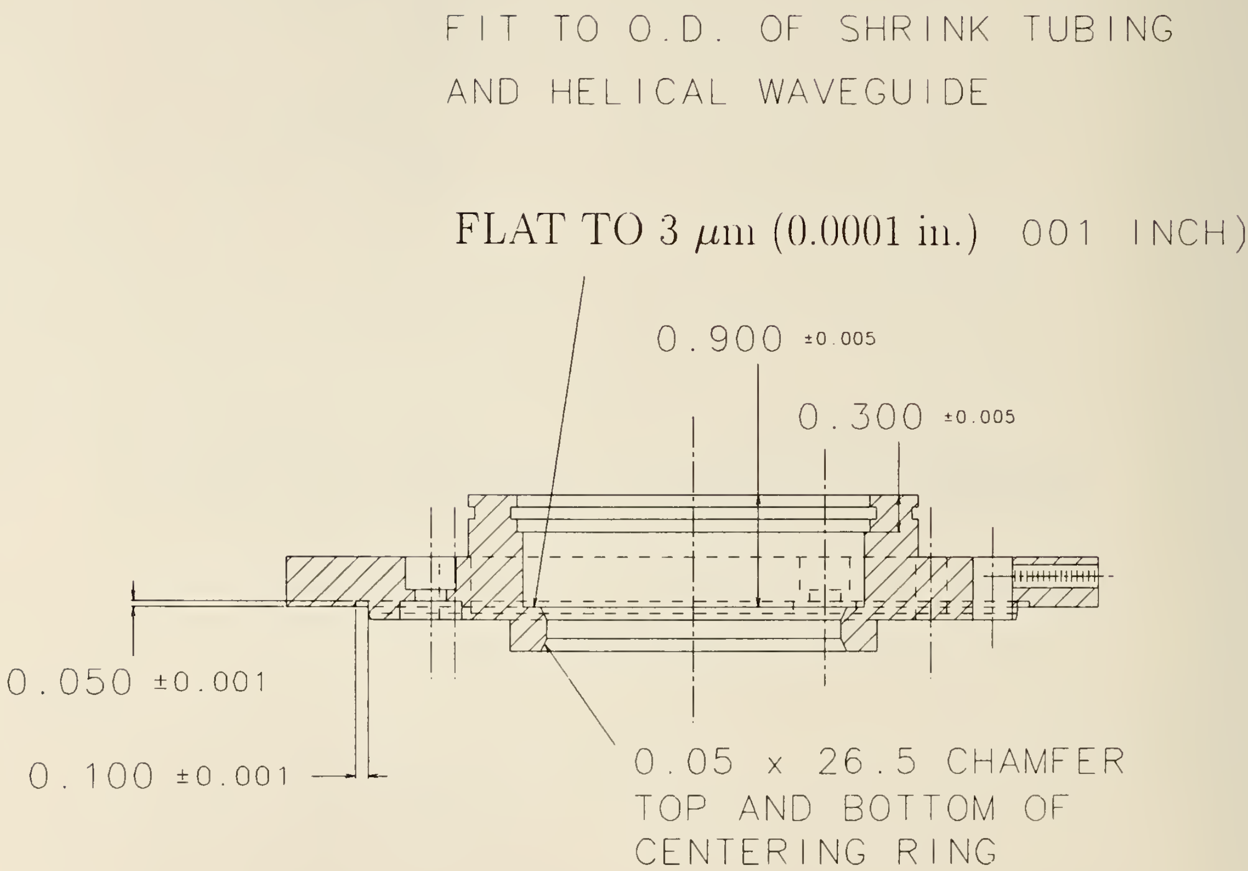

Figure C.11: Cutaway side view of coupling flange.

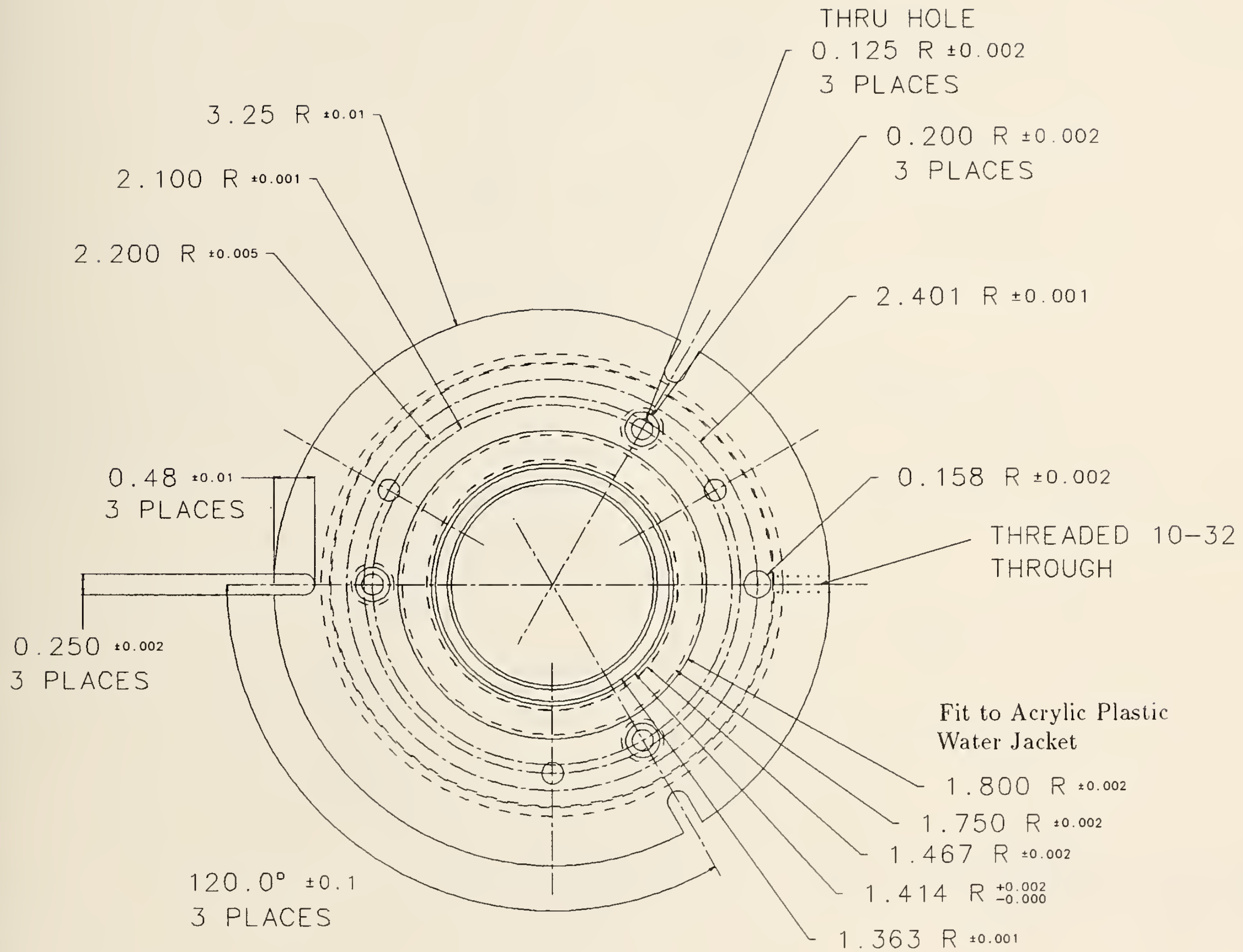

Figure C.12: Top view of tuner flange.

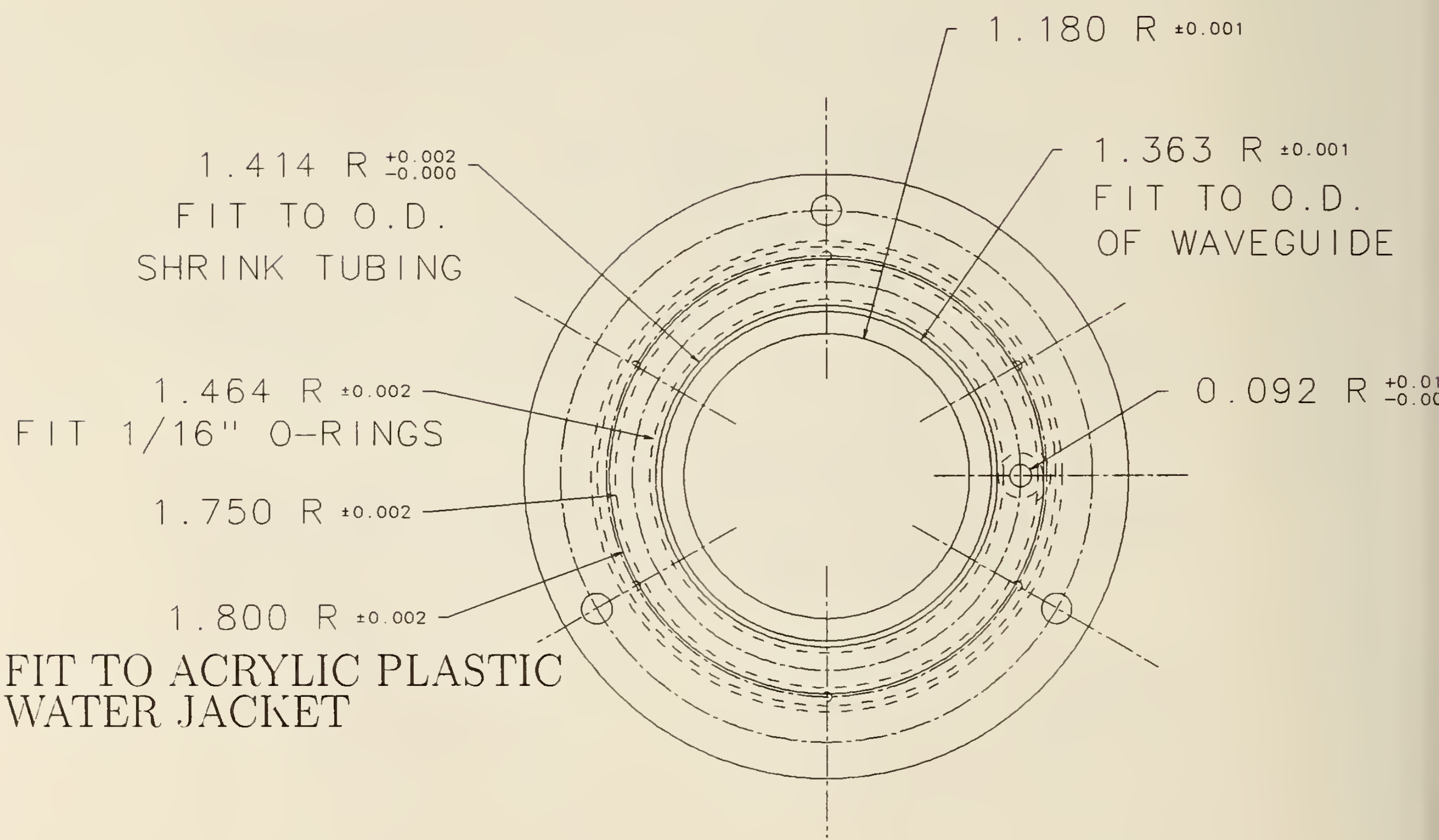

Figure C.13: Bottom view of tuner flange.

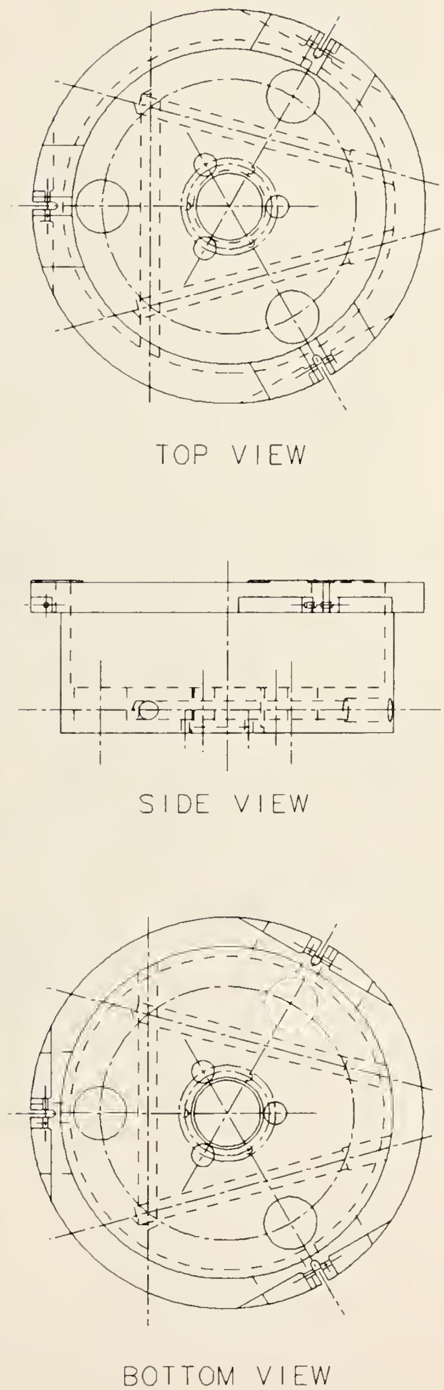

Figure C.14: Orthogonal views of tuner base.

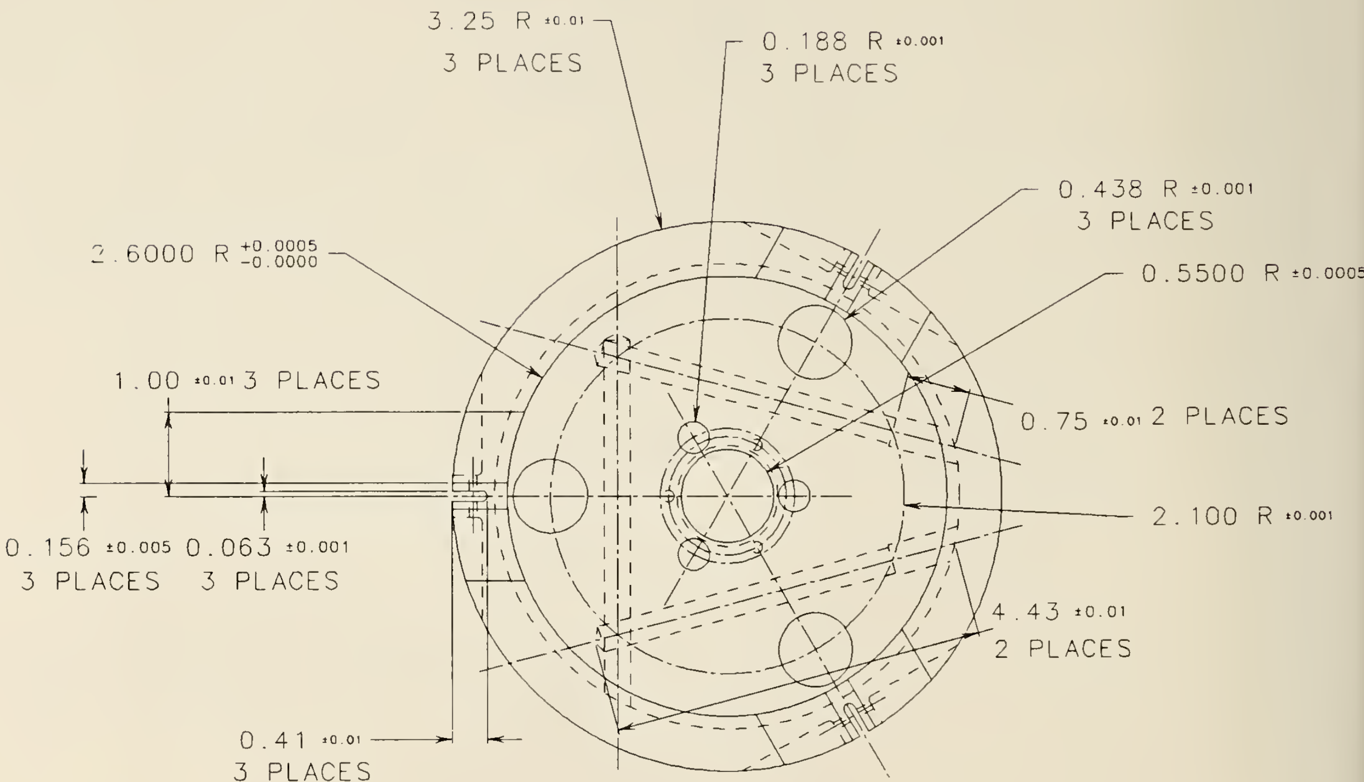

Figure C.15: Top view of tuner base.

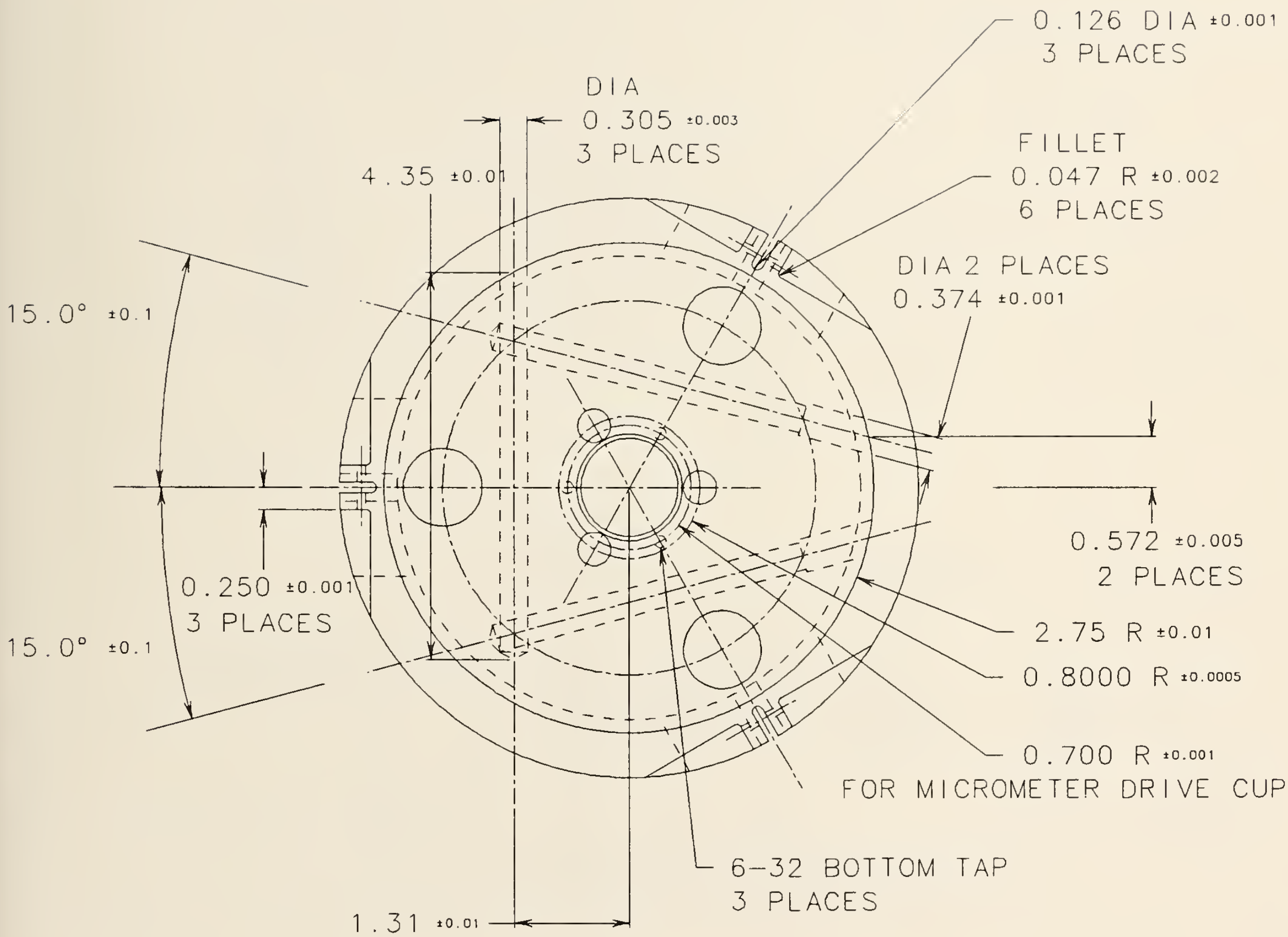

Figure C.16: Bottom view of tuner base.

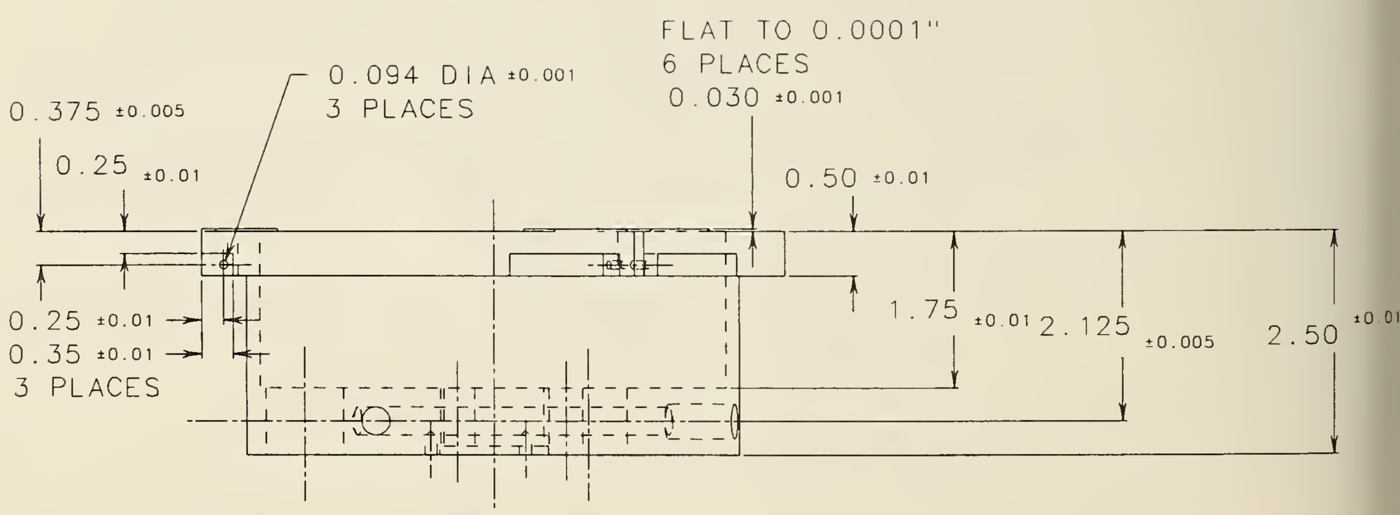

Figure C.17: Side view of tuner base.

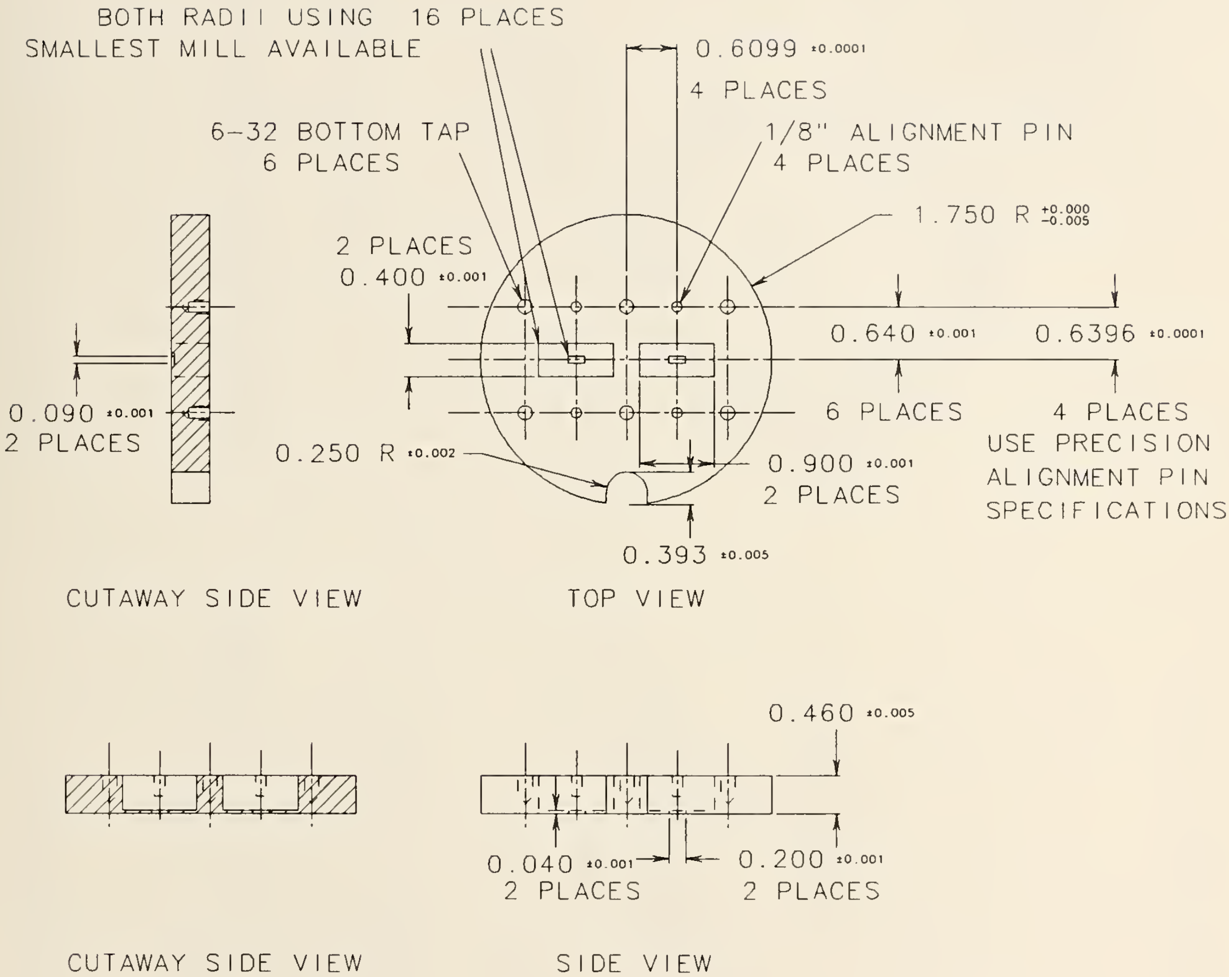

Figure C.18: Coupling endplate.

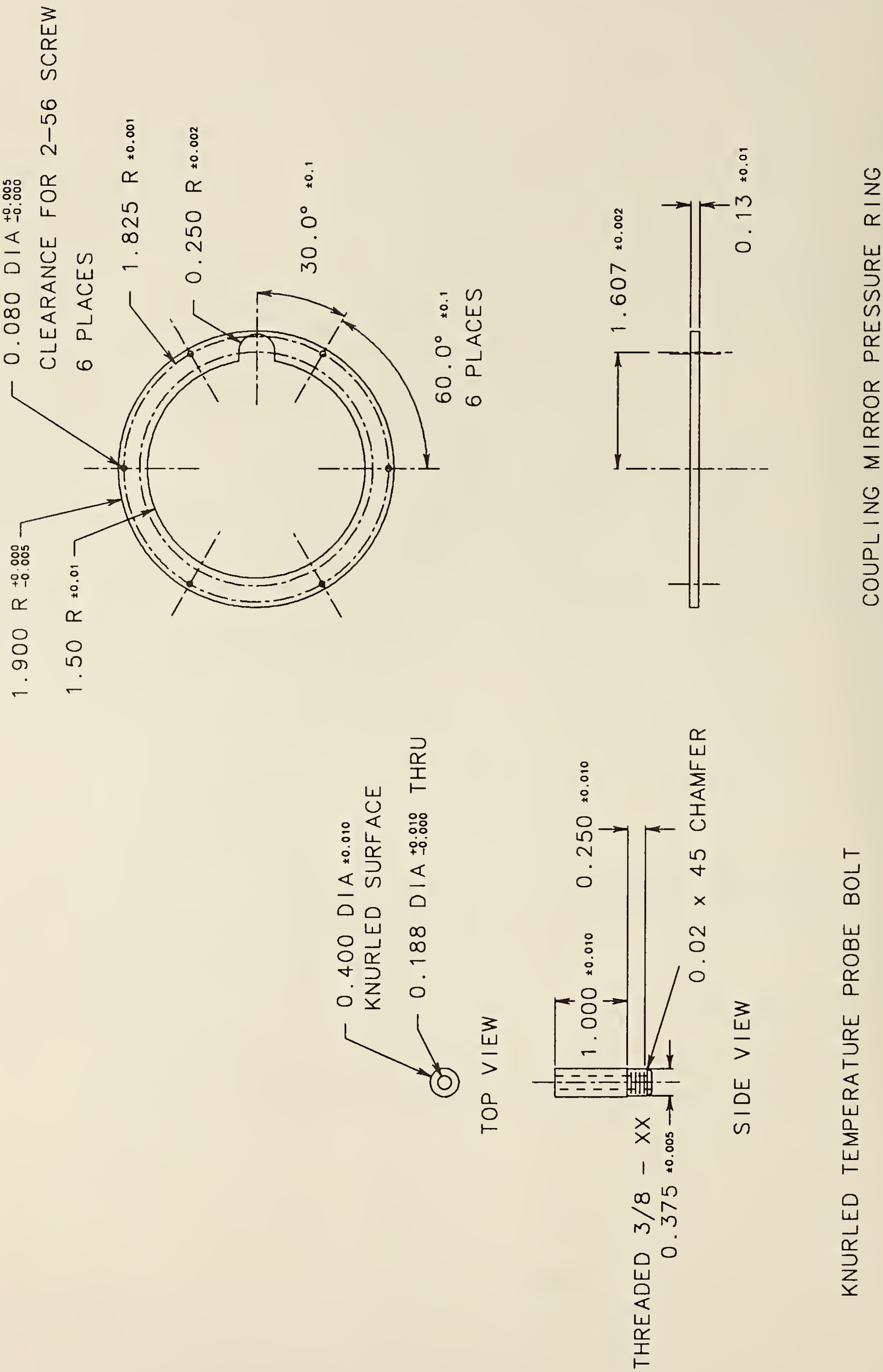

Figure C.19: Temperature-probe bolt and coupling-endplate pressure ring.

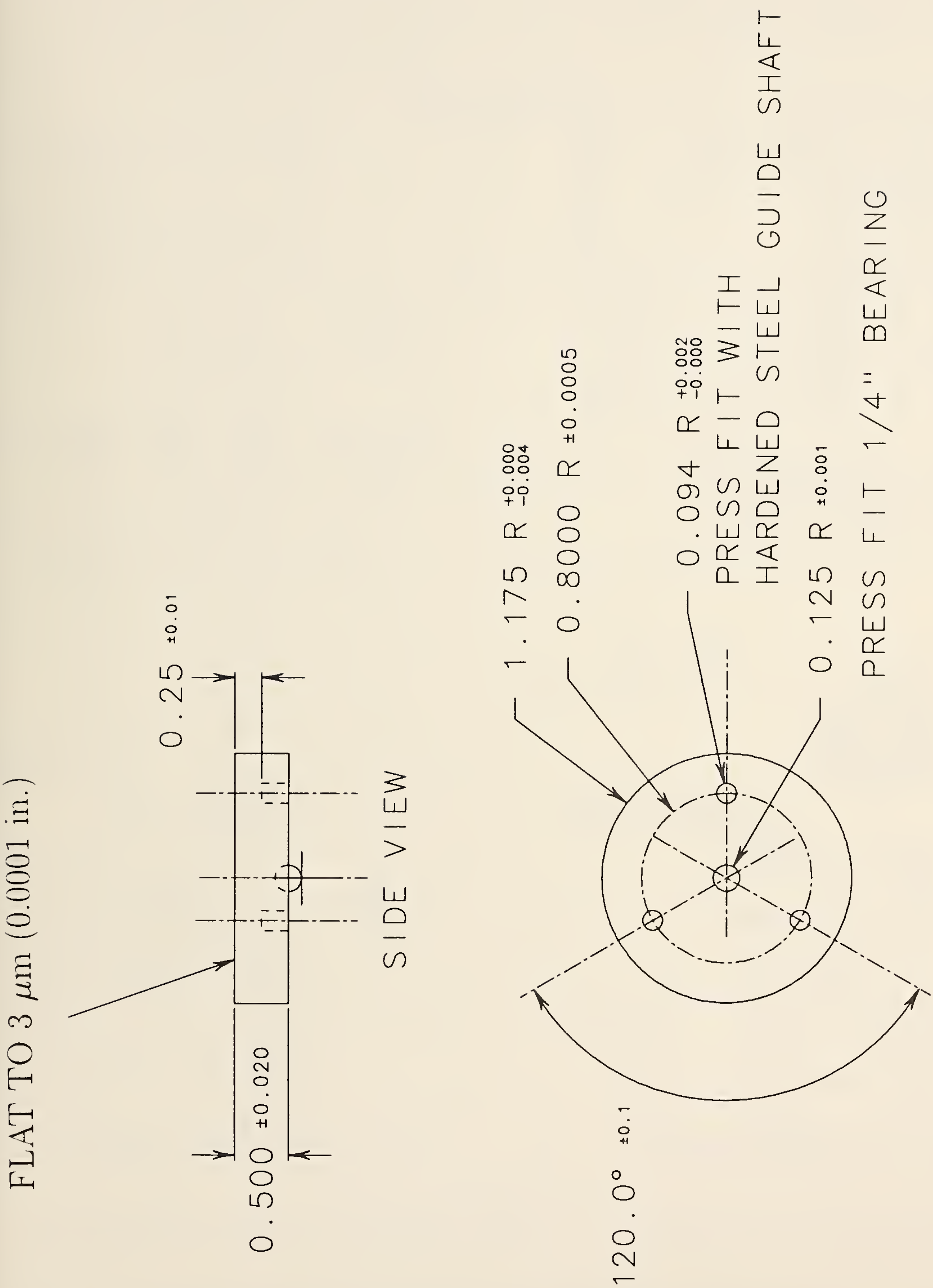

Figure C.20: Tuner-endplate.

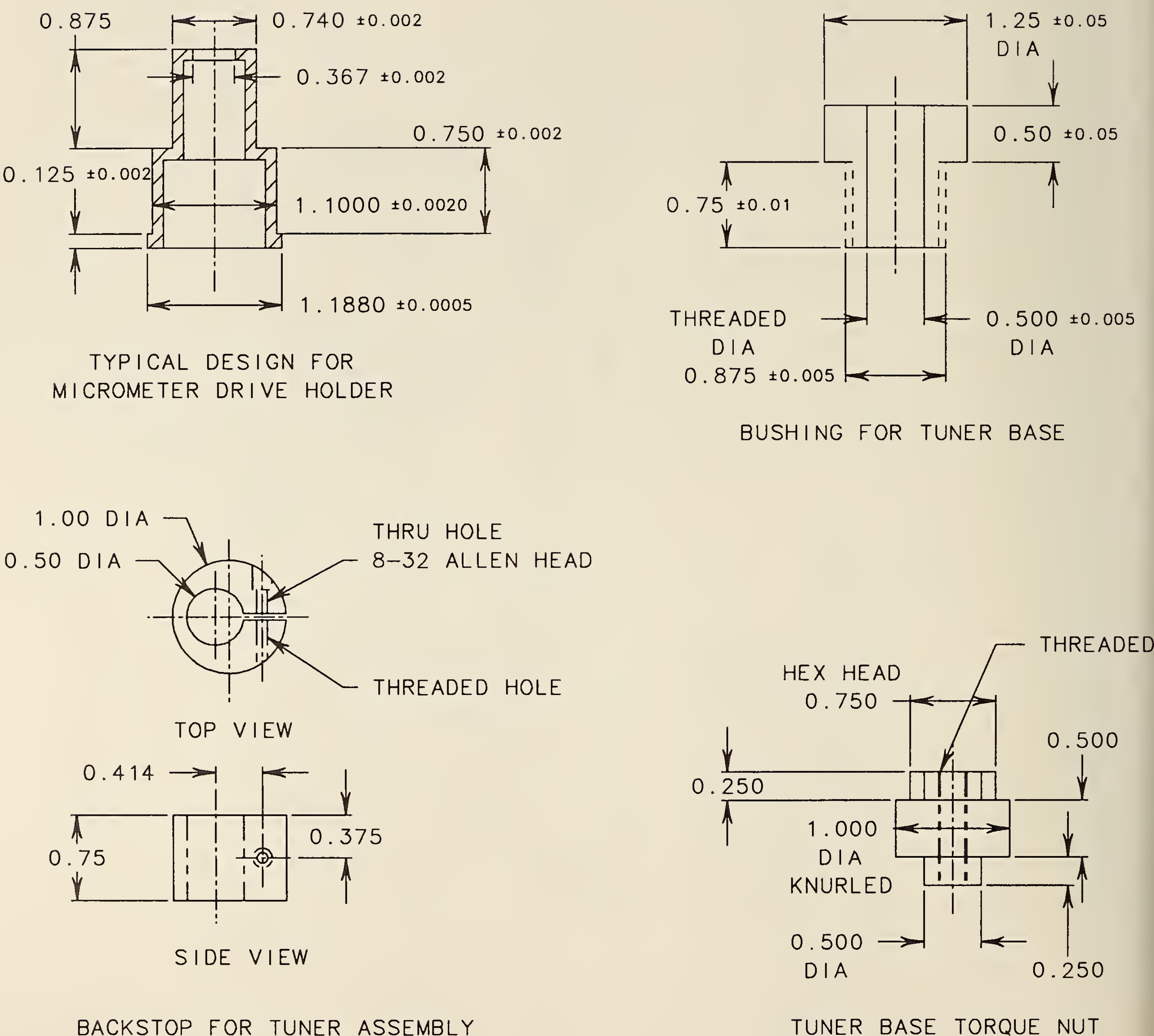

Figure C.21: Tuner assembly backstop, torque nut, micrometer-drive holder and sliding bushing.

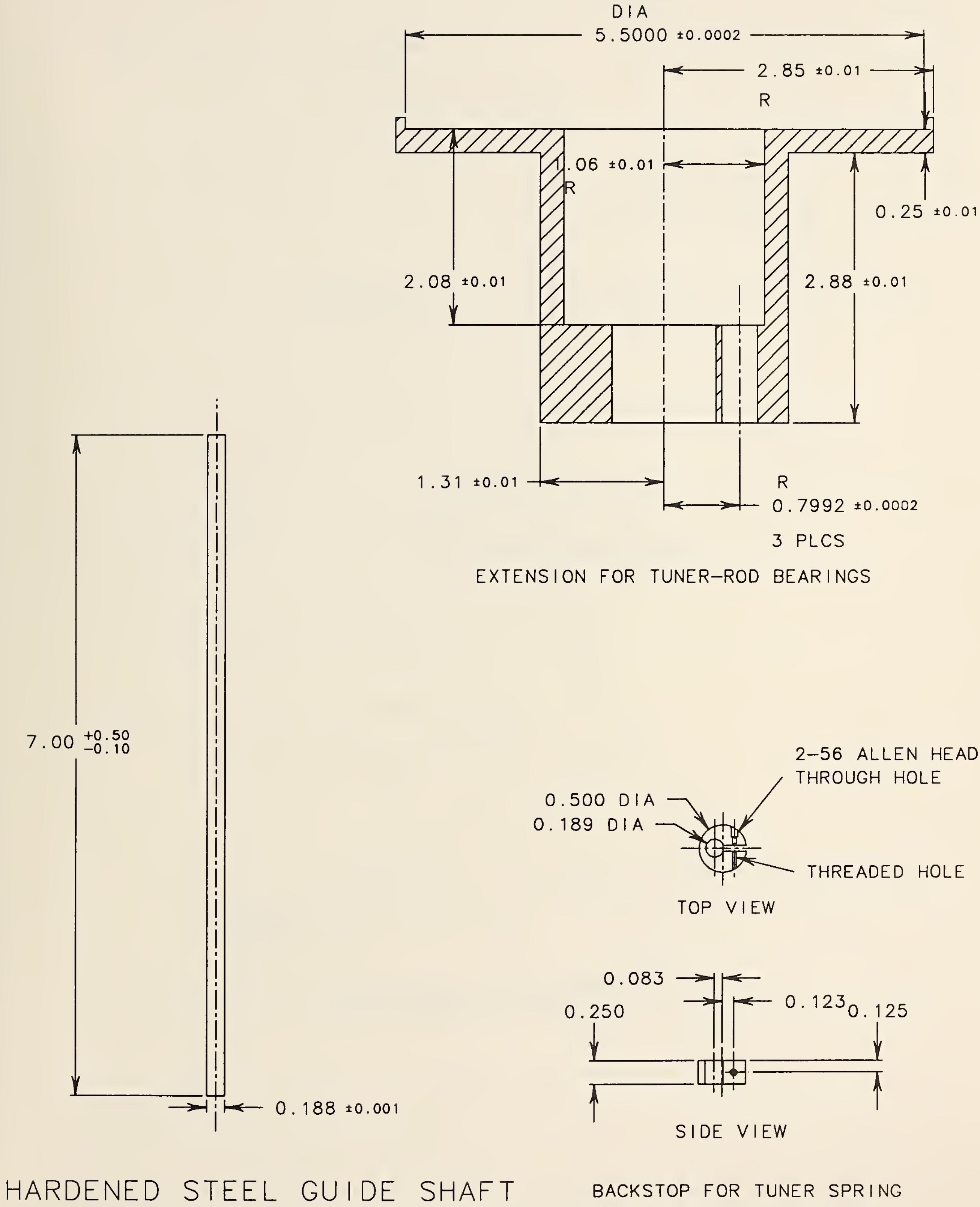

Figure C.22: Tuner guide shaft, spring backstop and extension.

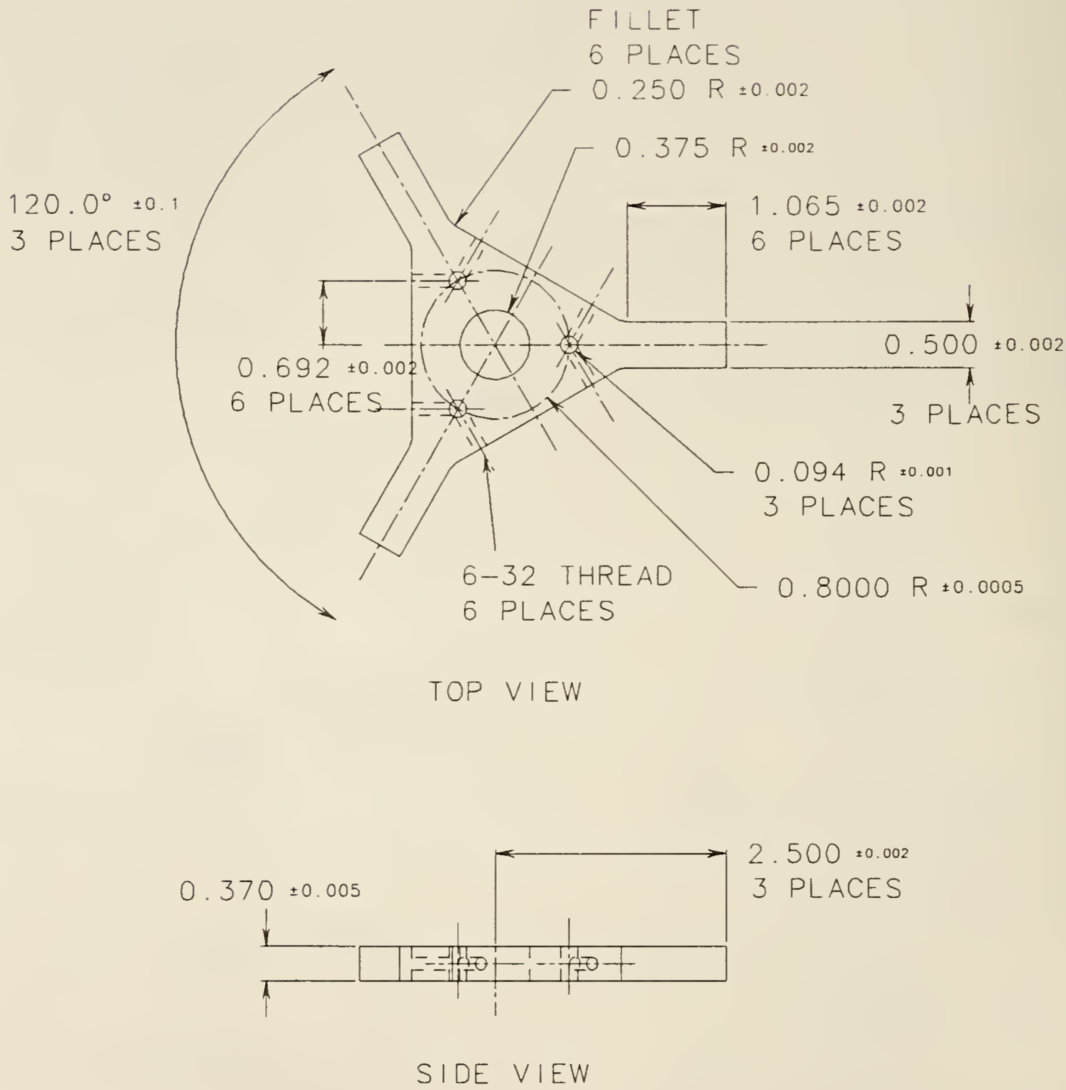

Figure C.23: Measurement-micrometer reference yoke.

Other resonator parts not shown are:

1. 1 ea. 60-mm diameter helical waveguide.

   faced off perpendicular to axis.

   outside diameter turned to be concentric with inner diameter.

   ends flat and perpendicular to axis to within 3 μm (0.0001 in).

2. 1 ea. 4 in outside-diameter, 3.5 in inside-diameter acrylic plastic tubing with tangential inlet and outlet ducts at opposite ends.

3. 1 ea. plug for tuner-base water channel.

4. 6 ea. 2-56 machine screws for coupling-endplate pressure ring.

5. 3 ea. 2-56 set-screws for tuner-endplate.

6. 3 ea. Allen-head $\frac{1}{4} - 20$ bolts to attach stand to tuner flange.

7. 1 ea. 10-32 knurled screw for micrometer attachment.

8. 2 ea. $\frac{1}{16}$-in inside flange o-rings.

9. 2 ea. $\frac{1}{16}$-in outside flange o-rings.

10. 1 ea. temperature probe o-ring.

11. 3 ea. $\frac{1}{4}$-in diameter tie rods. Both ends threaded $\frac{1}{4} - 20$. For attaching coupling and tuner flanges.

12. 3 ea. washers and $\frac{1}{4} - 20$ tie rods.

13. 1 ea. base for resonator stand. 18-in diameter, 1-in thick aluminum with 3-in diameter center-hole and 3-$\frac{1}{2}$-in diameter holes located 120° apart at a 2.1-in radius.

14. 3 ea. stands to connect from base to tuner flange. 12-in long, $\frac{1}{2} \pm 0.01$ in diameter stainless steel rods.

Appendix D

MEAS_RES Program Listing

```
105!            PURGE "MEAS_RES"
110!            RE-STORE "MEAS_RES"
115    GOSUB Init_com
120    GOSUB Whatkindofmeas
125 !      Whatkindofmeas$="LIST","SWEEP", or "TRIGGER","DISKDATA"
130    SELECT Whatkindofmeas$
135    CASE "LIST"
140        GOSUB Measurelist
145    CASE "SWEEP"
150        GOSUB Measuresweep
155    CASE "TRIGGER"
160        GOSUB Measuretrigger
165    CASE "LISTTRIGGER"
170        GOSUB Measurelistrig
175    CASE "DUMMY"
180        GOSUB Measuredumdata
185    CASE "DISKDATA"
190        GOSUB Measurediskdata
195    CASE "BAILOUT"
200    CASE ELSE
205        DISP "Hey! ";Whatkindofmeas$;" isn't a measurement option. PROGRAM ST
  => OPPED."
210        BEEP
215        STOP
220    END SELECT
225    GOTO End_program
230  !
235  ! ////////////////////////////////////////////////////////////
240  !
245 Whatkindofmeas: !
250    Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
255    CLEAR SCREEN
260    PRINT "This program allows you to read resonance data from a network anal
  => yzer"
265    PRINT "There are three ways for you to do this:"
270    PRINT
275    PRINT "1) Measure resonances at frequencies given in DATA statements."
280    PRINT "   Edit line label 'Resonance_data' and enter the frequencies befo
  => re running."
285    PRINT "2) Sweep a band of frequencies, searching for resonances."
290    PRINT "   Edit line label 'Init_sweep' to change the sweep parameters."
295    PRINT "3) Measure resonances that you set up manually on the network anal
  => yzer."
300    PRINT "   Simply find a resonance on the screen and press a softkey to me
  => asure."
301 Menu: Reset_keys=0
303    IF Printer_on THEN
304    PRINT TABXY(1,12);"Printer is ON "
305    ELSE
306    PRINT TABXY(1,12);"Printer is OFF"
307    END IF
308    DISP "Which kind of measurement do you want to perform?"
310    ON KEY 1 LABEL "End    Program",Prty GOSUB Bailout
315    ON KEY 2 LABEL "DATA   LIST",Prty GOSUB Datalist
320    ON KEY 3 LABEL "SWEEP  NWA",Prty GOSUB Sweep
325    ON KEY 4 LABEL "MEAS ONTRIGGER",Prty GOSUB Trigger
330    ON KEY 5 LABEL "DUMMY  DATA",Prty GOSUB Dummydata
335    ON KEY 5 LABEL "DISK   DATA",Prty GOSUB Diskdata
340    ON KEY 6 LABEL "LIST ->TRIGGER",Prty GOSUB Measurelisttrig
345    ON KEY 7 LABEL "DELSUB MODESUB",Prty GOSUB Delsub_modesub
346    IF NOT Printer_on THEN
348    ON KEY 8 LABEL "Turn ONPrinter",Prty GOSUB Toggle_printer
349    ELSE
350    ON KEY 8 LABEL "TurnOFFPrinter",Prty GOSUB Toggle_printer
351    END IF
352    Selected=0
355    LOOP
360    EXIT IF Selected
361    IF Reset_keys THEN GOTO Menu
365    END LOOP
370    OFF KEY
375    CLEAR SCREEN
380    RETURN
385  !
390  ! ////////////////////////////////////////////////////////////
395  !
396 Abort: Abort=1
397        RETURN
400 Bailout:Selected=1
405    Whatkindofmeas$="BAILOUT"
410    RETURN
415  !
420 Datalist:Selected=1
425    Whatkindofmeas$="LIST"
430    RETURN
435  !
440 Sweep:Selected=1
445    Whatkindofmeas$="SWEEP"
450    RETURN
455  !
460 Trigger:Selected=1
465    Whatkindofmeas$="TRIGGER"
470    RETURN
475  !
480 Dummydata:Selected=1
485    Whatkindofmeas$="DUMMY"
490    RETURN
495 Diskdata:Selected=1
500    Whatkindofmeas$="DISKDATA"
505    RETURN
510 Delsub_modesub: !
515    DISP "PLEASE WAIT WHILE THE ROUTINES ARE DELETED....."
520    DELSUB Resonance_data
525    DELSUB Empty_cavity
530    DELSUB Loaded_cavity
535    DELSUB Var_freq_empty
540    DELSUB Var_freq_loaded
545    DELSUB Var_len_empty
550    DELSUB Var_len_loaded
555    DELSUB Calc_c
560    DELSUB Calc_mode_freq
565    DISP
570    CLEAR SCREEN
575    RETURN
576  !
577  ! ////////////////////////////////////////////////////////////
578  !
579 Toggle_printer: OFF KEY 8
580    IF Printer_on THEN
581      PRINT TABXY(1,12);"Printer is OFF"
582      Printer_on=0
584      ELSE
585      PRINT TABXY(1,12);"Printer is ON "
586      Printer_on=1
588      END IF
589      Reset_keys=1
590      RETURN
591  !
592  ! ////////////////////////////////////////////////////////////
593  !
595 Measurelist:OFF KEY
600    Sss=1
```

```
605    Speed=0
610    MAT Te01= (0)
615    MAT Te02= (0)
620    MAT Te12= (0)
625    MAT Te13= (0)
630    MAT Freq= (0)
635    GOSUB Resonance_data
640    FOR I=1 TO Num_peaks
645        GOSUB Get_resonance
650        CALL Zoom_on_peak
655        GOSUB Measure_sparms
660        CALL Optimizedisplay
665        GOSUB Set_startstop
670        CALL Fit_sparms
675        CALL Phase_delay
680        CALL Fit_sparms
685    NEXT I
690    IF Printer_on THEN OUTPUT Print_addr;CHR$(12);
695    RETURN
700  !
705  ! ////////////////////////////////////////////////////////////
710  !
715 Measurelisttrig:OFF KEY
720    Sss=1
725    RESTORE Resonance_data
730    READ Num_peaks
735    FOR I=1 TO Num_peaks
740        READ Freq(I)
745    NEXT I
750    FOR I=1 TO Num_peaks
755        GOSUB Give_display
760        CALL Zoom_on_peak
765        GOSUB Measure_sparms
770        CALL Optimizedisplay
775        GOSUB Set_startstop
780        CALL Fit_sparms
785    NEXT I
790    IF Printer_on THEN OUTPUT Print_addr;CHR$(12);
795    RETURN
800  !
805  ! ////////////////////////////////////////////////////////////
810  !
815 Give_display: !
820    CALL Set_nwa("SWEEP=RAMP","S=21")
825    CALL Set_nwa("SPAN=1E5","SCALE=10","CENTER="&VAL$(Freq(I)))
830    CALL Set_nwa("REFV=-40","AVER=OFF","NUMG=1")
835    CALL Optimizedisplay
840    CALL Set_nwa("SWEEP=CONT")
845    Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
850    ON KEY 6 LABEL "READY  TO MEAS",Prty GOTO Get_back
855 Sitnspin:GOTO Sitnspin
860 Get_back:OFF KEY
865    CALL Set_nwa("SPAN=3E7")
870    RETURN
875  !
880  ! ////////////////////////////////////////////////////////////
885  !
890 Get_resonance: !
895    CALL Set_nwa("SWEEP=CONT","SWEEP=RAMP","S=21")
900    CALL Set_nwa("SPAN=3E7","SCALE=5","CENTER="&VAL$(Freq(I)))
905    CALL Set_nwa("REFV=-40","AVER=16","NUMG=17")
910    RETURN
915  ! ////////////////////////////////////////////////////////////
920  !
925 Find_resonance: !
930    CALL Set_nwa("SWEEP=CONT","SWEEP=RAMP","S=21")
935    CALL Set_nwa("SPAN=3E7","SCALE=5","CENTER="&VAL$(Freq(I)))
940    CALL Set_nwa("REFV=-40","AVER=16","NUMG=17")
945    Set_nwa("MARK=CONT","MARK=1","MARK=MAX")
950    Read_nwa("MARK=1")
955    Delta=-3
960    Set_nwa("DEL REF=1","MARKER TARGET="&VAL$(Delta))
965    Set_nwa("MARK=2","MARK=MAX","MARKER=TARGET","MARKER=SEARCH LEFT")
970    Set_nwa("MARK=3","MARK=MAX","MARKER=TARGET")
975    Read_nwa("MARK=2","MARK=3")
980    Set_nwa("DELTA OFF=")
985    IF ABS(Marker(2,1))>5000 AND ABS(Marker(3,1))>5000 THEN
990        IF ABS(Marker(2,1))<ABS(.9*Marker(3,1)) THEN
995            Span=2*ABS(Marker(3,1))
1000       ELSE
1005           Span=2*ABS(Marker(2,1))
1010       END IF
1015       CALL Zoom_on_peak
1020   ELSE
1025       Span=999999
1030       BEEP
1035       PRINT "Span cannot be determined from marker information."
1040       PRINT "No resonance was found at ";Freq(I)/1.E+9;" GHz."
1045       IF Printer_on THEN OUTPUT Print_addr;"No resonance was found at ";Fre
  => q(I)/1.E+9;" GHz."
1050       DISP "No resonance was found at ";Freq(I)/1.E+9;" GHz."
1055!          PAUSE
1060   END IF
1065   RETURN
1070 !
1075 ! ////////////////////////////////////////////////////////////
1080 !
1085 Measuresweep:OFF KEY
1090   CALL Sweep_nwa
1095   RETURN
1100 !
1105 ! ////////////////////////////////////////////////////////////
1110 !
1115 Measuretrigger:OFF KEY
1120   Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1125   DISP "When you have a resonance on the network analyzer CRT, press 'READ
  => PEAK'."
1130   ON KEY 0 LABEL "END    PROG",Prty GOTO Triggerdone
1135   ON KEY 1 LABEL "READ   S11",Prty GOSUB Read_s11res
1140   ON KEY 2 LABEL "READ   S21",Prty GOSUB Read_s21res
1145   ON KEY 3 LABEL "READ   S12",Prty GOSUB Read_s12res
1150   ON KEY 4 LABEL "READ   S22",Prty GOSUB Read_s22res
1151   ON KEY 5 LABEL "READ   S11/S22",Prty GOSUB Read_s11s22res
1152   ON KEY 15,Prty GOSUB Read_allsparms
1155 !
1160   ON KEY 6 LABEL "ZOOM   S11",Prty GOSUB Zoomread_s11res
1165   ON KEY 7 LABEL "ZOOM   S21",Prty GOSUB Zoomread_s21res
1170   ON KEY 8 LABEL "ZOOM   S12",Prty GOSUB Zoomread_s12res
1175   ON KEY 9 LABEL "ZOOM   S22",Prty GOSUB Zoomread_s22res
1180   Reset_keys=0
1185   LOOP
1190       IF Reset_keys THEN GOTO Measuretrigger
1195   END LOOP
1200 Triggerdone:OFF KEY
1205   RETURN
1210 !
1215 ! ////////////////////////////////////////////////////////////
1220 !
1225 Measurediskdata:OFF KEY
1230 Get_data: !
1235   DISP "Diskdata is in new or old format?"
1240   Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
```

```
1245   ON KEY 0 LABEL "ABORT THIS",Prty GOSUB Done_return
1250   ON KEY 2 LABEL "OLD   SPARMS",Prty GOSUB Get_old_sparms
1255   ON KEY 3 LABEL "NEW   SPARMS",Prty GOSUB Get_new_sparms
1260   Done=0
1265   Reset_keys=0
1270   REPEAT
1275       IF Reset_keys THEN GOTO Get_data
1280   UNTIL Done
1285   RETURN
1290 !
1295 Done_return:Done=1
1300   RETURN
1305 !
1310 ! ///////////////////////////////////////////////////////////////
1315 !
1320 Get_old_sparms:OFF KEY
1325   CALL Load_sparms
1330 ! CALL Phase_delay
1335   RESTORE Diskdatas
1340   Speed=0
1345   Sselect=1   !S21
1350   Start_index=1
1355   Stop_index=Dcount
1360   GOSUB Diskdata_keys
1365   Done=1
1370   RETURN
1375 !
1380 ! ///////////////////////////////////////////////////////////////
1385 !
1390 Get_new_sparms:OFF KEY
1395 ! ALLOCATE Desc$[160]
1400 ! REPEAT
1405 !     LINPUT "Enter a description of this data. When finished enter a null
 => string.",Desc$
1410 !     PRINT Desc$
1415 !     IF Printer_on THEN OUTPUT Print_addr;Desc$
1420 ! UNTIL Desc$=""
1425 ! DEALLOCATE Desc$
1430 ! N=6
1435 ! OUTPUT 2 USING "K,#";N
1440 ! INPUT "Please enter the number of files",N
1441   ON ERROR GOSUB Change_disk
1443   RESTORE Mode_numbers
1444   READ Num_settings
1445   FOR J=1 TO Num_settings
1446       READ Micr,Nummodes
1447       IF Printer_on THEN OUTPUT Print_addr;"Micr:";Micr;"  Nummodes:";Nummo
 => des;"  30 Nov 1990 Test of micrometer travel accuracy"
1448       PRINT "Micr:";Micr;"  Nummodes:";Nummodes;"  30 Nov 1990 Test of micr
 => ometer travel accuracy"
1449       FOR I=1 TO Nummodes
1453           Filename$=VAL$(Micr)&"30Nov90"&VAL$(I)
1455           Diskdrive$=":,1400,1"
1460           CALL New_load_sparms
1465           Sselect=1          !S21 is hardcoded
1470           Start_index=1
1475           Stop_index=Dcount
1480           CALL Fit_sparms
1485           Peak(I,1)=A(4)
1490           Peak(I,2)=SQRT(A(1)^2+A(2)^2)
1495           Peak(I,3)=A(4)/A(3)
1500       NEXT I
1505       IF Printer_on THEN OUTPUT Print_addr;CHR$(12);
1506       N=Nummodes
1510       ALLOCATE Basket(N,2),Id$[40]
1515       MAT Basket(*,1)= Peak(1:N,1)
1520       MAT Basket(*,2)= Peak(1:N,3)
1525       CLEAR SCREEN
1526       Filename$=VAL$(Micr)&"30Nov90"&VAL$(J)
1527       Id$=Filename$
1528       Diskdrive$=":,1404"
1530 ! CALL Select_disk
1535 ! IF Diskdrive$="NO DISK" THEN GOTO Skip_peak_save
1540 ! CALL Enterfilename("ABORT","SAVING resonance parameters in a file")
1545 ! IF Filename$="" THEN GOTO Skip_peak_save
1550 ! CALL Enter_id(Id$,"resonance parameter file")
1555 ! IF Id$="" THEN GOTO Skip_peak_save
1560       CALL Save_file(Basket(*),Nummodes,Id$)
1565 Skip_peak_save:DEALLOCATE Id$,Basket(*)
1566   NEXT J
1567   OFF ERROR
1570   Reset_keys=1
1575   RETURN
1580 !
1581 Change_disk:PRINT "please insert next disk"
1582   DISP ERRM$
1583   BEEP
1585   PAUSE
1586   RETURN
1587 ! ///////////////////////////////////////////////////////////////
1590 !
1595 Diskdata_keys:OFF KEY
1600   GOSUB Diskdata_menu
1605   DISP "Which S-parameter would you like to fit?"
1610   Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1615   ON KEY 1 LABEL "Bailout",Prty GOSUB Bailout
1620   ON KEY 6 LABEL "FIT    S11",Prty GOSUB Fit_s11
1625   ON KEY 7 LABEL "FIT    S12",Prty GOSUB Fit_s12
1630   ON KEY 8 LABEL "FIT    S21",Prty GOSUB Fit_s21
1635   ON KEY 9 LABEL "FIT    S22",Prty GOSUB Fit_s22
1636   ON KEY 3 LABEL "SAVE   SPARMS",Prty GOSUB Save_new_sparms
1640   ON KEY 4 LABEL "START  INDEX",Prty GOSUB Fix_start_index
1645   ON KEY 5 LABEL "STOP   INDEX",Prty GOSUB Fix_stop_index
1650   Done=0
1655   Selected=0
1660   LOOP
1665       IF Done THEN GOTO Diskdata_keys
1670   EXIT IF Selected
1675   END LOOP
1680   OFF KEY
1685   CLEAR SCREEN
1690   RETURN
1695 !
1700 Fit_s11: !
1705   Sss=2
1710   Sparm$="S11"
1715   MAT Sparm= S11
1720   MAT Frequency= Freq
1725   CALL Fit_sparms
1730   Done=1
1735   RETURN
1740 Fit_s12: !
1745   Sss=1
1750   Sparm$="S12"
1755   MAT Sparm= S12
1760   MAT Frequency= Freq
1765   CALL Fit_sparms
1770   Done=1
1775   RETURN
1780 Fit_s21: !
1785   Sss=1
1790   Sparm$="S21"
```

```
1795   MAT Sparm= S21
1800   MAT Frequency= Freq
1805   CALL Fit_sparms
1810   Done=1
1815   RETURN
1820 Fit_s22: !
1825   Sss=2
1830   Sparm$="S22"
1835   MAT Sparm= S22
1840   MAT Frequency= Freq
1845   CALL Fit_sparms
1850   Done=1
1855   RETURN
1860 Fix_start_index:Si=Start_index
1865   CALL Test_real(1,Stop_index-1.,"Start_index",Si)
1870   Start_index=Si
1875   Done=1
1880   RETURN
1885 Fix_stop_index:Si=Stop_index
1890   CALL Enter_real(Start_index+1.,Dcount,"Stop_index",Si)
1895   Done=1
1900   RETURN
1905 Diskdatas:DATA 1,33,34,66,67,99,100,132,133,165,166,198,199,231,232,264,265
  => ,297,298,330,331,363,364,396
1910   RETURN
1915 !
1916 Save_new_sparms:OFF KEY
1917   CLEAR SCREEN
1918   Ssave=Svalid
1920   Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1921 Setup_save:DISP "Which S-parameters do you want to save?"
1922   IF NOT BIT(Svalid,0) THEN
1923       PRINT TABXY(0,1);"S11 is not valid, cannot be saved"
1924   ELSE
1925       IF BIT(Ssave,0) THEN
1926           PRINT TABXY(0,1);CHR$(129);"S11 will be saved    ";CHR$(128)
1927           ON KEY 6 LABEL "noSave S11",Prty GOSUB Toggle_s11
1928       ELSE
1929           PRINT TABXY(0,1);"S11 will NOT be saved"
1930           ON KEY 6 LABEL "  Save S11",Prty GOSUB Toggle_s11
1931       END IF
1932   END IF
1933 !
1934   IF NOT BIT(Svalid,1) THEN
1935       PRINT TABXY(0,2);"S21 is not valid, cannot be saved"
1936   ELSE
1937       IF BIT(Ssave,1) THEN
1939           PRINT TABXY(0,2);CHR$(129);"S21 will be saved    ";CHR$(128)
1940           ON KEY 7 LABEL "noSave S21",Prty GOSUB Toggle_s21
1941       ELSE
1942           PRINT TABXY(0,2);"S21 will NOT be saved"
1943           ON KEY 7 LABEL "  Save S21",Prty GOSUB Toggle_s21
1944       END IF
1945   END IF
1947 !
1948   IF NOT BIT(Svalid,2) THEN
1949       PRINT TABXY(0,3);"S12 is not valid, cannot be saved"
1950   ELSE
1952       IF BIT(Ssave,2) THEN
1953           PRINT TABXY(0,3);CHR$(129);"S12 will be saved    ";CHR$(128)
1954           ON KEY 8 LABEL "noSave S12",Prty GOSUB Toggle_s12
1955       ELSE
1956           PRINT TABXY(0,3);"S12 will NOT be saved"
1957           ON KEY 8 LABEL "  Save S12",Prty GOSUB Toggle_s12
1958       END IF
1959   END IF
1961 !
1962   IF NOT BIT(Svalid,3) THEN
1963       PRINT TABXY(0,4);"S22 is not valid, cannot be saved"
1964   ELSE
1965       IF BIT(Ssave,3) THEN
1967           PRINT TABXY(0,4);CHR$(129);"S22 will be saved    ";CHR$(128)
1968           ON KEY 9 LABEL "noSave S22",Prty GOSUB Toggle_s22
1969       ELSE
1970           PRINT TABXY(0,4);"S22 will NOT be saved"
1971           ON KEY 9 LABEL "  Save S22",Prty GOSUB Toggle_s22
1972       END IF
1973   END IF
1974   ON KEY 1 LABEL "ABORT",Prty GOSUB Abort
1975   ON KEY 2 LABEL "ALL OK",Prty GOSUB Setup_done
1977 !
1978   Reset_keys=0
1979   Done=0
1980   LOOP
1981       IF Reset_keys THEN GOTO Setup_save
1982   EXIT IF Done
1983   END LOOP
1984   Done=0
1985   Diskdrive$=":,1400,1"
1986   CALL Enterfilename("VALID")
1988   CALL New_save_sparms(Ssave)
1989   Done=1
1990   RETURN
1991 !
1992 Setup_done:OFF KEY
1993           Done=1
1994           RETURN
1995 !
1996 ! ///////////////////////////////////////////////////////////////
1997 !
1998 Toggle_s11:!
1999       IF BIT(Ssave,0) THEN
2000       Ssave=Ssave-1
2001       ELSE
2002       Ssave=Ssave+1
2003       END IF
2004       Reset_keys=1
2005       RETURN
2006 !
2007 Toggle_s21:!
2008       IF BIT(Ssave,1) THEN
2009       Ssave=Ssave-2
2010       ELSE
2011       Ssave=Ssave+2
2012       END IF
2013       Reset_keys=1
2014       RETURN
2015 !
2016 Toggle_s12:!
2017       IF BIT(Ssave,2) THEN
2018       Ssave=Ssave-4
2019       ELSE
2020       Ssave=Ssave+4
2021       END IF
2022       Reset_keys=1
2023       RETURN
2024 !
2025 Toggle_s22:!
2026       IF BIT(Ssave,3) THEN
2027       Ssave=Ssave-8
2028       ELSE
2029       Ssave=Ssave+8
```

```
2030       END IF
2031       Reset_keys=1
2032       RETURN
2033 !
2034 ! ///////////////////////////////////////////////////////////
2035 !
2036 Diskdata_menu:PRINT TABXY(1,1)
2037          PRINT "S PARAMETER FITTING FROM DISK DATA"
2038          PRINT
2039          PRINT USING Image1;Filename$&Diskdrive$
2040 ! PRINT USING Image2;Title$
2041 ! PRINT USING Image6;DATE$(DATE(Measure_time$))
2042          PRINT
2043          PRINT USING Image3;Dcount
2044          PRINT
2045          PRINT USING Image4;Start_index
2046          PRINT USING Image5;Stop_index
2047 Image1:   IMAGE"FILENAME       : ",K
2048 Image2:   IMAGE"DESCRIPTION    : ",K
2049 Image3:   IMAGE"DATACOUNT      : ",3D
2050 Image4:   IMAGE"START INDEX    : ",3D
2051 Image5:   IMAGE"STOP  INDEX    : ",3D
2052 Image6:   IMAGE"DATE           : ",K
2053          RETURN
2054 ! ///////////////////////////////////////////////////////////
2055 !
2056 Measuredumdata:OFF KEY
2057          Span=2.E+5
2058          F0=(1.200E+10)
2059          Q0=90000
2060          Ql=79000
2061          G0=F0/Q0
2062          Gl=F0/Ql
2063          Degrees=38
2064          Mag=.7
2065          Num_points=50
2066          Start_index=1
2067          Stop_index=50
2068 !     ALLOCATE X(Num_points),COMPLEX S21(Num_points),
2069          FOR I=1 TO Num_points
2070              X(I)=Span*I/Num_points+F0-(Span/2)
2071              Frequency(I)=X(I)
2072              Ql2f=2*Ql*(X(I)-F0)/F0
2073              P=CMPLX(COS(Degrees),SIN(Degrees))
2074 !        S21(I)=Mag*(1-G0/Gl)*P/CMPLX(1,2*(X(I)-F0)/Gl)
2075 !        S11(I)=Mag*CMPLX(Ql/Q0+Ql2f*Ql2f,Ql2f*(1-Ql/Q0))*P/(1+Ql2f*Ql2f)
2076 !     S21(I)=(1-(Ql/Q0))/CMPLX(1,2*Ql*(F(I)-F0)/F0)
2077 !     S11(I)=Mag*CMPLX(Ql/Q0,2*Ql*(F(I)-F0)/F0)/CMPLX(1,2*Ql*(F(I)-F0)/F0)*
  => CMPLX(COS(Degrees),SIN(Degrees))
2078              Rndmag=.0001*(RND-.5)
2079              Rndphase=.001*(RND-.5)
2080              Sparm(I)=S21(I)+Rndmag*(CMPLX(COS(Rndphase),SIN(Rndphase)))
2081          NEXT I
2082          CALL Fit_sparms
2083 ! PRINT USING "2(K,MD.3DE,2X),K,MD.9DE";"Q0=",A(1),"Ql=",A(2),"F0=",A(3)
2084 ! PRINT USING "2(K,MD.3DE,2X),K,MD.9DE";"UQ0=",Ua,"UQl=",Ub,"UF0=",Uc
2085          RETURN
2086 !
2087 ! ///////////////////////////////////////////////////////////
2088 !
2089 Read_s11s22res:OFF KEY
2090               INPUT "ENTER THE NUMBER OF AVERAGES",Avs
2091               OUTPUT 716;"AVERON "&VAL$(Avs)
2092 !
2093               OUTPUT 716;"S11;"
2095               OUTPUT 716;"CORRON;CALS4;"
2096               GOSUB Read_s11res
2097 !
2098 !             OUTPUT 716;"CORRON;CALS4;"
2099 !             GOSUB Read_s21res
2100 !
2101               OUTPUT 716;"S22;"
2103               OUTPUT 716;"CORRON;CALS5;"
2104               GOSUB Read_s22res
2105 !
2106 !             OUTPUT 716;"CORRON;CALS4;"
2107 !             GOSUB Read_s12res
2108 !
2110               GOSUB Save_new_sparms
2111               Reset_keys=1
2112               RETURN
2113 !
2114 ! ///////////////////////////////////////////////////////////
2115 !
2116 Read_allsparms:OFF KEY
2117               GOSUB Read_s11res
2118               GOSUB Read_s21res
2119               GOSUB Read_s12res
2120               GOSUB Read_s22res
2121               GOSUB Save_new_sparms
2122               Reset_keys=1
2123               RETURN
2124 !
2125 ! ///////////////////////////////////////////////////////////
2126 !
2127 Zoomread_s11res:OFF KEY
2128          CALL Zoom_on_peak
2129 Read_s11res:OFF KEY
2130          Sparm$="S11"
2131          Sss=3
2132          GOSUB Measure_sparms
2133          CALL Optimizedisplay
2134          GOSUB Set_startstop
2135          CALL Fit_sparms
2136          GOSUB Give_results
2137          Reset_keys=1
2138          RETURN
2139 !
2140 ! ///////////////////////////////////////////////////////////
2141 !
2142 Zoomread_s21res:OFF KEY
2143          CALL Zoom_on_peak
2144 Read_s21res:OFF KEY
2145          Sparm$="S21"
2146          Sss=1
2147          GOSUB Measure_sparms
2148          CALL Optimizedisplay
2149 !        GOSUB Set_startstop
2150 !        CALL Set_nwa("SPAN=100MHZ","SWEEP=RAMP","AVER=OFF","SWEEP=SING")
2151 !        CALL Set_nwa("SWEEP=CONT","AUTOSCALE=","LOCAL=")
2152 !        CALL Fit_sparms
2153 !        GOSUB Give_results
2154          Reset_keys=1
2155          RETURN
2156 !
2157 ! ///////////////////////////////////////////////////////////
2158 !
2159 Zoomread_s12res:OFF KEY
2160          CALL Zoom_on_peak
2161 Read_s12res:OFF KEY
2162          Sparm$="S12"
2163          Sss=1
```

```
            GOSUB Measure_sparms
            CALL Optimizedisplay
!           CALL Fit_sparms
!           GOSUB Set_startstop
!           GOSUB Give_results
            Reset_keys=1
            RETURN
!
! ///////////////////////////////////////////////////////////////
!
Zoomread_s22res:OFF KEY
            CALL Zoom_on_peak
Read_s22res:OFF KEY
            Sparm$="S22"
            Sss=3
            GOSUB Measure_sparms
            CALL Optimizedisplay
!           CALL Fit_sparms
!           GOSUB Give_results
            Reset_keys=1
            RETURN
!
! ///////////////////////////////////////////////////////////////
!
Measure_sparms:!
            SELECT Sparm$
            CASE "S11"
                 Set_nwa("S=11","LOG MAG=")
            CASE "S21"
                 Set_nwa("S=21","LOG MAG=")
            CASE "S12"
                 Set_nwa("S=12","LOG MAG=")
            CASE "S22"
                 Set_nwa("S=22","LOG MAG=")
            END SELECT
!           CALL Set_nwa("SWEEP=STEP")
            CALL Set_nwa("NUMBER OF GROUPS=1")
            CALL Read_nwa("S=")
            RETURN
!
! ///////////////////////////////////////////////////////////////
!
Set_startstop:!
            Start_index=1
            Stop_index=Dcount
            RETURN
!
! ///////////////////////////////////////////////////////////////
!
!
Give_results:!
            SELECT Sss
            CASE 1      !S21 RESONANCE
                 IF Printer_on THEN
                      PRINTER IS Print_addr
                      PRINT USING "K,D.5DESZ,K,D.9DE";"Resonance 3dB Span: ",A(
 => 4)/A(3),"  Unloaded resonant frequency: ",A(4)+A(4)/(A(3)*2)
                      PRINT
                      PRINTER IS 1
                 END IF
            END SELECT
            RETURN
! ///////////////////////////////////////////////////////////////
!
Send_to_local:!
            Set_nwa("SWEEP=CONT","AVER=OFF","SWEEP=RAMP","LOCAL=")
            RETURN
!
! ///////////////////////////////////////////////////////////////
!
Init_com:  !
!
            OPTION BASE 1
            COM /S_array/ INTEGER Dcount,Svalid,REAL S(801,5,2)
            COM /Fittype/ INTEGER Sselect,Start_index,Stop_index
! COM /Sparm/ Frequency(801),COMPLEX Sparm(801),Sparm$[3],INTEGER Num_points
            COM /Cdata/ INTEGER Mfit(2),Lista(5,2),Ma(2),REAL A(5),Ua(5),INTE
 => GER Ndata(2)
            COM /Cstats/ Alamda,Chisq,INTEGER Nca(2),REAL Alpha(5,5),Covar(5,
 => 5)
            COM /Files/ Diskdrive$[20],Filename$[10]
            COM /Bugs/ INTEGER Bug1,Bug2,Bug3,Printer
            COM /Nwa_data/ Start_freq,Stop_freq,Marker(5,3)
            COM /Peak_data/ Peak(100,3),Upeak(100,3),INTEGER Peak_index,Num_p
 => eaks
            COM /Resonance/ Kx,Ky,Q,F0,Ukx,Uky,Uq,Uf0
!
            COM /History/ Status$[1],Time_orgn$[8],Date_orgn$[11]
            COM /History/ Time_chng$[8],Date_chng$[11],Description$[160]
            COM /Labels/ Labels$(30)[60],INTEGER Lbl_count,REAL Lbl_addr(30,6
 => )
            COM /Data_param/ INTEGER Data_count,Filesize,Curvecount,Roster(17
 => ,4)
            COM /Data_param/ REAL Sym_size,Symbol$(17)[2],Curve_id$(17)[40]
            COM /Data_param/ REAL Xmin_data,Xmax_data
            COM /Data_param/ REAL Ymin_data,Ymax_data
            COM /Ana_data/ INTEGER Prog_id,Sweep_type,Sweep_mode,Datacount
            COM /Ana_data/ INTEGER S11_valid,S21_valid,S12_valid,S22_valid,Po
 => rts
            COM /Ana_data/ REAL Z0,Start,Stop
            COM /Ana_data/ Title$[55],Company$[30],Operator_name$[30],Measure
 => _time$[30]
            COM /Sparms/ REAL Freq(801),COMPLEX S11(801),S21(801),S12(801),S2
 => 2(801)
            COM /Graphics/ INTEGER Speed
            COM /Background/ Graphtype$[12],Margins$(2)[10],Papersize$[1]
            COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
            COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
            COM /Background/ Xgrid_tick$[4],INTEGER Xmajor,Xminor
            COM /Background/ Ygrid_tick$[4],INTEGER Ymajor,Yminor
            COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
            COM /Axes_labels/ Print_xlabels$[3],Print_ylabels$[3]
            COM /Axes_labels/ Sig_digits$[2],REAL Xlcsize,Ylcsize
            COM /Windowspace/ REAL Xmin,Xmid,Xmax,Ymin,Ymid,Ymax! graph edges
 =>  UDU
            COM /Windowspace/ REAL Xleft,Xright,Ybottom,Ytop! paper edges UDU
 => s
            COM /Log_scale/ REAL Xcycles,Xbegin,Ycycles,Ybegin
            COM /Plot_device/ Plot_lang$[10],INTEGER Plot_addr
            COM /Hard_space/ REAL Xview_lft,Xview_rt,Yview_btm,Yview_top
            COM /Hard_space/ REAL Viewscale,Aspect_ratio
            COM /Frame_onoff/ INTEGER Frame_flag
            COM /Clear_space/ REAL Space_lft,Space_rt,Space_btm,Space_top
  ! MODEHELPER COM
            COM /Frequency_info/ REAL Start_frequency,Stop_frequency,Incremen
 => t_freq
            COM /Length_info/ REAL Start_length,Stop_length,Increment_len
            COM /Cavity_info/ REAL Diameter,Length
            COM /Lab_info/ REAL Rel_humidity,Temperature,Pressure,C
            COM /Sample_info/ Sample_desc$[40],REAL Thickness,Epsilon
            COM /Bessel_root/ REAL Bessel_root
            COM /Modes/ Mode$[10]
```

```
          COM /Modes/ REAL Te01(50,2),Te02(50,2)
          COM /Modes/ REAL Te12(50,2),Te13(50,2)
          COM /Output/ INTEGER Print_addr,Plotter,Printer_on,Plotter_on
          INTEGER Num_settings,Micr,Nummodes,Ssave
!     !
Init_var:!
          Speed=1
          Printer=701
          Print_addr=Printer
          Plotter=705
          Printer_on=1
          Plotter_on=0
          COMPLEX Cnum,P
          Svalid=0
          Ssave=0
          RETURN
!
Data:!
Resonance_data:!
          DISP "PLEASE WAIT WHILE ROUTINES ARE LOADED....."
          LOADSUB Resonance_data FROM "MODESUB:,1400,0,3"
          LOADSUB Empty_cavity FROM "MODESUB:,1400,0,3"
          LOADSUB Loaded_cavity FROM "MODESUB:,1400,0,3"
          LOADSUB Var_freq_empty FROM "MODESUB:,1400,0,3"
          LOADSUB Var_freq_loaded FROM "MODESUB:,1400,0,3"
          LOADSUB Var_len_empty FROM "MODESUB:,1400,0,3"
          LOADSUB Var_len_loaded FROM "MODESUB:,1400,0,3"
          LOADSUB Calc_c FROM "MODESUB:,1400,0,3"
          LOADSUB Calc_mode_freq FROM "MODESUB:,1400,0,3"
          DISP
          CALL Resonance_data
          I=1
          J=0
          WHILE Te01(I,1)=0
              I=I+1
          END WHILE
          WHILE Te01(I,1)<>0
              J=J+1
              Freq(J)=Te01(I,1)
              I=I+1
          END WHILE
          Num_peaks=J
          DISP "PLEASE WAIT WHILE ROUTINES ARE DELETED....."
          DELSUB Resonance_data
          DELSUB Empty_cavity
          DELSUB Loaded_cavity
          DELSUB Var_freq_empty
          DELSUB Var_freq_loaded
          DELSUB Var_len_empty
          DELSUB Var_len_loaded
          DELSUB Calc_c
          DELSUB Calc_mode_freq
          DISP
          RETURN
!
! ////////////////////////////////////////////////////////////
!
End_program:  !
          DISP "The program has ended."
!
Mode_numbers:DATA 29,0,16,2,16,4,16,6,16,8,17,10,16,12,15,14,16,16,17,18,14
 => ,20,14,22,14,24,15,23,15,21,15,19,14,17,16,15,15,13,16,11,16
          DATA 9,15,7,15,5,16,3,18,1,17,2,16,4,16,6,16,8,16
          END
!
! ******************************************************************
!
   SUB Set_nwa(Cmd1$,OPTIONAL Cmd2$,Cmd3$,Cmd4$,Cmd5$,Cmd6$)
Set_nwa:  !
      OPTION BASE 1
      ASSIGN @Nwa TO 716
      ALLOCATE C$(1:6)[35],Hp_ib$[35]
      ON NPAR GOTO C1,C2,C3,C4,C5,C6
C6:   C$(6)=Cmd6$
C5:   C$(5)=Cmd5$
C4:   C$(4)=Cmd4$
C3:   C$(3)=Cmd3$
C2:   C$(2)=Cmd2$
C1:   C$(1)=Cmd1$
!
      FOR I=1 TO NPAR
          Function$=C$(I)[1,POS(C$(I),"=")-1]
          Value$=C$(I)[POS(C$(I),"=")+1]
!
          SELECT Function$
          CASE "PRES","PRESET"
              Hp_ib$="PRES;"
          CASE "START","START FREQ"
              Hp_ib$="STAR"&Value$&";"
          CASE "STOP","STOP FREQ"
              Hp_ib$="STOP"&Value$&";"
          CASE "CENTER","CF"
              SELECT Value$
              CASE "MARK","MARKER"
                  Hp_ib$="CENT;EQUA;"
              CASE ELSE
                  Hp_ib$="CENT"&Value$&";"
              END SELECT
          CASE "SPAN","SP"
              SELECT Value$
              CASE "MARK","MARKER"
                  Hp_ib$="SPAN;EQUA;"
              CASE ELSE
                  Hp_ib$="SPAN"&Value$&";"
              END SELECT
Marker:!
          CASE "MARKER","MARK"
              SELECT Value$
              CASE "MAX","MAXIMUM","MAXI"
                  Hp_ib$="MARKMAXI;"
              CASE "MIN","MINIMUM","MINI"
                  Hp_ib$="MARKMINI;"
              CASE "TARG","TARGET"
                  Hp_ib$="MARKTARG;"
              CASE "CONT","CONTINUOUS"
                  Hp_ib$="MARKCONT;"
              CASE "DISC","DISCRETE"
                  Hp_ib$="MARKDISC;"
              CASE "SEARCH RIGHT"
                  Hp_ib$="SEAR;"
              CASE "SEARCH LEFT"
                  Hp_ib$="SEAL;"
              CASE ELSE
                  Hp_ib$="MARK"&Value$&";"
              END SELECT
          CASE "MARKER TARGET","TARGET VALUE","TARV"
              Hp_ib$="TARV"&Value$&";"
          CASE "DELTA REF","DELTA REFERENCE","DEL REF","DELR"
              Hp_ib$="DELR"&Value$&";"
          CASE "DEL OFF","DELTA OFF","DELO"
              Hp_ib$="DELO;"
          CASE "AVER","AVERAGING","AVERAGE","AVG"
```

```
2415          IF Value$="OFF" THEN
2416              Hp_ib$="AVEROFF;"
2417          ELSE
2418              Hp_ib$="AVERON"&Value$&";"
2419          END IF
2420      CASE "NUMG","NUM GROUPS","NUMBER OF GROUPS"
2421          Hp_ib$="NUMG"&Value$&";"
2422      CASE "REF POS","REF POSN","REFERENCE POSITION","REFP"
2423          Hp_ib$="REFP"&Value$&";"
2424      CASE "REF VAL","REFERENCE VALUE","REF VALUE","REFV"
2425          Hp_ib$="REFV"&Value$&";"
2426      CASE "WAIT"
2427          Hp_ib$="WAIT;"
2428      CASE "UP","STEP UP"
2429          Hp_ib$="UP;"
2430      CASE "DOWN","STEP DOWN"
2431          Hp_ib$="DOWN;"
2432      CASE "LOGM","LOG MAG"
2433          Hp_ib$="LOGM;"
2434          IF Value$<>"" THEN Hp_ib$=Hp_ib$&" SCAL"&Value$&";"
2435      CASE "IMAG","CARTESIAN","X-jY"
2436          Hp_ib$="IMAG;"
2437      CASE "PHASE"
2438          Hp_ib$="PHAS;"
2439      CASE "SMITH CHART","SMITH"
2440          Hp_ib$="SMIC;"
2441      CASE "SWEEP"
2442          SELECT Value$
2443          CASE "CONT","CONTINUOUS"
2444              Hp_ib$="CONT;"
2445          CASE "SING","SINGLE"
2446              Hp_ib$="SING;"
2447          CASE "STEP"
2448              Hp_ib$="STEP;"
2449          CASE "RAMP"
2450              Hp_ib$="RAMP;"
2451          CASE "HOLD"
2452              Hp_ib$="HOLD;"
2453          END SELECT
2454      CASE "S","s"
2455          Hp_ib$=Function$&Value$&";"
2456      CASE "SCALE","SCAL"
2457          Hp_ib$="SCAL"&Value$&";"
2458      CASE "AUTOSCALE","AUTO"
2459          Hp_ib$="AUTO;"
2460      CASE "LOCAL"
2461          Hp_ib$=""
2462          LOCAL @Nwa
2463      CASE ELSE
2464          PRINT Function$;" IS NOT DEFINED IN SET_NWA"
2465      END SELECT
2466      IF Hp_ib$<>"" THEN OUTPUT @Nwa;Hp_ib$
2467    NEXT I
2468    ASSIGN @Nwa TO *
2469  SUBEND
2470 !
2471 ! ***************************************************************
2472 !
2473  SUB Read_nwa(Cmd1$,OPTIONAL Cmd2$,Cmd3$,Cmd4$,Cmd5$,Cmd6$)
2474    OPTION BASE 1
2475    COM /Nwa_data/ Start_freq,Stop_freq,Marker(*)
2476    COM /S_array/ INTEGER Dcount,Svalid,REAL S(801,5,2)
2477    DIM Sparm$[5],Swem$[20]
2478 Read_nwa:  !
2479    ASSIGN @Nwa TO 716
2480    ASSIGN @Nwa_data1 TO 716;FORMAT ON
2481    ASSIGN @Nwa_data2 TO 716;FORMAT OFF
2482    REAL Freq
2483    INTEGER Preamble,Bytes
2484    ALLOCATE C$(1:6)[35],Hp_ib$[35]
2485    ON NPAR GOTO C1,C2,C3,C4,C5,C6
2486 C6:   C$(6)=Cmd6$
2487 C5:   C$(5)=Cmd5$
2488 C4:   C$(4)=Cmd4$
2489 C3:   C$(3)=Cmd3$
2490 C2:   C$(2)=Cmd2$
2491 C1:   C$(1)=Cmd1$
2492!
2493    FOR I=1 TO NPAR
2494      Function$=C$(I)[1,POS(C$(I),"=")-1]
2495      Value$=C$(I)[POS(C$(I),"=")+1]
2496!
2497      SELECT Function$
2498      CASE "MARKER","MARK"
2499          Hp_ib$="MARK"&Value$&";"
2500          OUTPUT @Nwa;Hp_ib$
2501          OUTPUT @Nwa;"OUTPACTI;"
2502          ENTER @Nwa_data1;Marker(VAL(Value$),1)
2503          OUTPUT @Nwa;"OUTPMARK;"
2504          ENTER @Nwa_data1;Marker(VAL(Value$),2),Marker(VAL(Value$),3)
2505      CASE "START"
2506          OUTPUT @Nwa;"STAR; OUTPACTI;"
2507          ENTER @Nwa_data1;Start_freq
2508      CASE "STOP"
2509          OUTPUT @Nwa;"STOP; OUTPACTI;"
2510          ENTER @Nwa_data1;Stop_freq
2511      CASE "S","s"
2512          GOSUB Get_sparm
2513      CASE ELSE
2514          PRINT Function$;" NOT DEFINED IN READ ANA"
2515      END SELECT
2516    NEXT I
2517    ASSIGN @Nwa TO *
2518    ASSIGN @Nwa_data1 TO *
2519    ASSIGN @Nwa_data2 TO *
2520    SUBEXIT
2521 !
2522 ! ///////////////////////////////////////////////////////////////
2523 !
2524 Get_sparm:!This subroutine gets the S-parameter data that is on the NWA CRT
2525    OUTPUT @Nwa;"PARA?"            !Find out what Sparm is on the CRT
2526    ENTER @Nwa;Sparm$
2527    Sparm$=Sparm$[2,4]
2528    OUTPUT @Nwa;"SWEM?"           !What is the sweep mode?
2529    ENTER @Nwa;Swem$              !RAMP, STEP or FREQUENCY LIST
2530    Swem$=Swem$[2,LEN(Swem$)-1]
2531    SELECT Sparm$
2532    CASE "S11"
2533      N=1                  !Set index for the S(*) array
2534    CASE "S21"
2535      N=2
2536    CASE "S12"
2537      N=3
2538    CASE "S22"
2539      N=4
2540    CASE ELSE
2541      BEEP
2542      DISP "Sparm$ is invalid in Read_nwa"
2543      PAUSE
2544    END SELECT
2545 !
2546    OUTPUT @Nwa;"FORM3; OUTPDATA;"
```

```
2547       ENTER @Nwa_data2;Preamble;Bytes
2548       Dcount=Bytes/16
2549       ALLOCATE S1(1:Dcount,2),F(1:Dcount)
2550       ENTER @Nwa_data2;S1(*)
2551       SELECT Swem$
2552       CASE "RAMP","STEP"
2553           CALL Read_nwa("START=","STOP=")
2554           FOR J=1 TO Dcount
2555               F(J)=Start_freq+(J-1)*(Stop_freq-Start_freq)/(Dcount-1)
2556           NEXT J
2557       CASE "FREQUENCY LIST"
2558           OUTPUT @Nwa;"FORM3;OUTPFREL;"
2559           ENTER @Nwa;Preamble,Bytes,F(*)
2560       CASE ELSE
2561           BEEP
2562           DISP "Swem$ has not been properly selected in Read_nwa"
2563           PAUSE
2564       END SELECT
2565       MAT S(1:Dcount,N,1)= S1(1:Dcount,1)
2566       MAT S(1:Dcount,N,2)= S1(1:Dcount,2)
2567       MAT S(1:Dcount,5,1)= F
2568       DEALLOCATE F(*),S1(*)
2569       IF NOT BIT(Svalid,N-1) THEN Svalid=Svalid+2^(N-1)
2570       RETURN
2571   SUBEND
2572 !
2573 ! ****************************************************************
2574 !
2575   SUB Zoom_on_peak
2576 !This subprogram assumes there is a resonance peak on the network analyzer
2577 !display.  The routine will zoom in on the center frequency by reducing the
2578 !span until the resonance "Delta" dB points are shown.  Averaging depends
2579 !on signal level.  Step frequency mode is used only for the last sweep.
2580 !The S(I,J,K) array is dimensioned (801,5,2) (1:Dcount,Ns,REAL/IMAG)
2581 !Ns= 1:S11 2:S21 3:S12 4:S22 5:Frequency(for K=1)
2582 !Written by Eric J. Vanzura NIST 723.02
2583       OPTION BASE 1
2584       ASSIGN @Nwa TO 716
2585       COM /Nwa_data/ Start_freq,Stop_freq,Marker(*)
2586 !     COM /Sparm/ Frequency(801),COMPLEX Sparm(801),Sparm$[3],INTEGER Num_p
  => oints
2587       COM /S_array/ INTEGER Dcount,Svalid,REAL S(801,5,2)
2588 Zoom_on_peak:     !
2589       Lastscale=.5
2590       Delta=-3
2591       Spanratio=1.5
2592 !
2593       Scale=20*Lastscale
2594       Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
2595       DISP "Zooming in on peak"
2596 !  Resonance data already exists on NWA CRT.
2597       Averages=4
2598       Nwa_span=5.E+6
2599       GOSUB Take_sweeps
2600       CALL Optimizedisplay
2601       GOSUB Find_deltaspan
2602       GOSUB Print_peak_info
2603       Scale=Lastscale*10
2604       GOSUB Set_new_nwaspan
2605       GOSUB Select_averages
2606       Set_nwa("REF VAL="&VAL$(Level+Scale),"SCALE="&VAL$(Scale))
2607       GOSUB Take_sweeps
2608       CALL Optimizedisplay
2609       GOSUB Find_deltaspan
2610       GOSUB Print_peak_info
2611       Scale=Lastscale
2612       GOSUB Set_new_nwaspan
2613       GOSUB Select_averages
2614       Set_nwa("REF VAL="&VAL$(Level+Scale),"SCALE="&VAL$(Scale))
2615       GOSUB Take_sweeps
2616       CALL Optimizedisplay
2617       GOSUB Find_deltaspan
2618       GOSUB Print_peak_info
2619       SUBEXIT
2620 !
2621 ! ///////////////////////////////////////////////////////////////
2622 !
2623 Set_new_nwaspan:!
2624       SELECT Scale
2625       CASE >Lastscale
2626           IF Span<5000 THEN
2627               Span=3.E+6
2628               Nwa_span=Span
2629           ELSE
2630               Nwa_span=Span*3
2631           END IF
2632       CASE =Lastscale
2633           Nwa_span=Spanratio*Span
2634       END SELECT
2635       RETURN
2636 !
2637 ! ///////////////////////////////////////////////////////////
2638 !
2639 Find_deltaspan:!
2640       Set_nwa("MARK=CONT","MARK=1","MARK=MAX")
2641       Read_nwa("MARK=1")
2642       Freq=Marker(1,1)
2643       Level=Marker(1,2)
2644       Set_nwa("DEL REF=1","MARKER TARGET="&VAL$(Delta))
2645       Set_nwa("MARK=2","MARK=MAX","MARKER=TARGET","MARKER=SEARCH LEFT")
2646       Set_nwa("MARK=3","MARK=MAX","MARKER=TARGET")
2647       Read_nwa("MARK=2","MARK=3")
2648       Set_nwa("DELTA OFF=")
2649       IF ABS(Marker(2,1))>1000 AND ABS(Marker(3,1))>1000 THEN
2650           IF ABS(Marker(2,1))<ABS(.9*Marker(3,1)) THEN
2651 !             Span=2*ABS(Marker(3,1))
2652               Span=ABS(Marker(2,1))+ABS(Marker(3,1))
2653           ELSE
2654 !             Span=2*ABS(Marker(2,1))
2655               Span=ABS(Marker(2,1))+ABS(Marker(3,1))
2656           END IF
2657       ELSE
2658           Span=999999
2659           BEEP
2660           PRINT "Span cannot be determined from marker information";
2661           PRINT " in Find_deltaspan."
2662           PAUSE
2663       END IF
2664       RETURN
2665 !
2666 ! ///////////////////////////////////////////////////////////
2667 !
2668 Print_peak_info:!
2669       IF Freq<1.E+10 THEN
2670           PRINT USING "K,X,D.9DE,2X,K,S2D.2D,2X,K,8D,K,X,D.9DE";"Freq=",Fre
  => q/1.E+9,"Level=",Level,"Span=",Span,"Freq+(Span/2)=",(Freq+Span/2)/1.E+9
2671       ELSE
2672           PRINT USING "K,X,DD.9DE,2X,K,S2D.2D,2X,K,8D,K,X,DD.9DE";"Freq=",F
  => req/1.E+9,"Level=",Level,"Span=",Span,"Freq+(Span/2)=",(Freq+Span/2)/1.E+9
2673       END IF
2674       RETURN
2675 !
```

```
2676 ! ////////////////////////////////////////////////////////////
2677 !
2678 Take_sweeps:!
2679       Set_nwa("MARK=1","MARK=MAX","CENTER=MARKER","SPAN="&VAL$(Nwa_span))
2680       Set_nwa("AVER="&VAL$(Averages))
2681       IF Scale=Lastscale THEN
2682           Set_nwa("SWEEP=STEP","NUMG=1")
2683       ELSE
2684           Set_nwa("SWEEP=RAMP","NUMG="&VAL$(Averages+1))
2685       END IF
2686       RETURN
2687 !
2688 ! ////////////////////////////////////////////////////////////
2689 !
2690 Select_averages:!          !
2691       SELECT Level         !Level of marker 1
2692       CASE >-30
2693           Averages=2
2694       CASE >-42
2695           Averages=4
2696       CASE >-50
2697           Averages=8
2698       CASE >-60
2699           Averages=16
2700       CASE <=-60
2701           Averages=32
2702       CASE ELSE
2703           PRINT "Select_averages IS BAD"
2704       END SELECT
2705       IF Scale=Lastscale THEN Averages=Averages*4
2706       PRINT "LEVEL=";Level,"AVERAGES=";Averages
2707       RETURN
2708   SUBEND
2709   !
2710   ! ************************************************************
2711   !
2712   SUB Sweep_nwa
2713 Sweep_nwa:  !
2714       GOSUB Init_sweep_nwa
2715       GOSUB Print_info
2716       CALL Set_nwa("PRES=","WAIT=","S=21","REF POS=10")
2717       Startwind=Startfreq
2718       WHILE Startwind<Stopfreq
2719           DISP "Searching for a peak ";Startwind/1.E+9;" to ";(Startwind+St
  => epfreq)/1.E+9;" GHz"
2720           Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
2721           ON KEY 1 LABEL "ZOOM ONPEAK",Prty GOTO Zoomonpeak
2722           ON KEY 2 LABEL "SKIP   WINDOW",Prty+1 RECOVER Next_step
2723           CALL Set_nwa("SWEEP=CONT","SWEEP=RAMP")
2724           CALL Set_nwa("START FREQ="&VAL$(Startwind),"STOP FREQ="&VAL$(Star
  => twind+Stepfreq))
2725           CALL Set_nwa("S=21","REFV=0","SCALE=10","AVER=16","NUMG=17")
2726           CALL Set_nwa("MARK=1","MARK=MAX","MARK=2","MARK=MIN")
2727           CALL Read_nwa("MARK=1","MARK=2")
2728           IF Marker(1,2)<Min_peak_val THEN GOTO Next_step
2729           IF Marker(1,2)-Marker(2,2)<Min_sig_to_noi THEN GOTO Next_step
2730 Zoomonpeak:OFF KEY 1
2731           CALL Zoom_on_peak  !Sets up S21 for data aquisition
2732           CALL Read_nwa("S=")
2733           Num_peaks=Num_peaks+1
2734           Sselect=1      ! 0:S11 1:S21 2:S12 3:S22
2735           IF Save_fo_later THEN
2736               GOSUB Save_fo_later
2737           ELSE
2738               Start_index=1
2739               Stop_Index=Dcount
2740               CALL Fit_sparms
2741               GOSUB Print_3dbspan
2742               PRINT
2743               IF Printer_on THEN
2744                   PRINTER IS Print_addr
2745                   GOSUB Print_3dbspan
2746                   PRINT
2747                   PRINTER IS 1
2748               END IF
2749               GOSUB Add_peak_2_list
2750           END IF
2751 Next_step:OFF KEY
2752           Startwind=Startwind+Stepfreq
2753       END WHILE
2754       GOTO End_sweep_nwa
2755 !
2756 ! ////////////////////////////////////////////////////////////
2757 !
2758 Print_3dbspan:!
2759       SELECT Sss
2760       CASE 1
2761           PRINT USING "K,D.5DESZ,K,D.9DE";"3dB span:",A(4)/A(3)," F0+(Span/
  => 2):",A(4)+A(4)/(A(3)*2)/1.E+9
2762       CASE ELSE
2763           DISP "You need to add a PRINT statement in Sweep_nwa"
2764       END SELECT
2765       RETURN
2766 !
2767 ! ////////////////////////////////////////////////////////////
2768 !
2769 Save_fo_later:!
2770       SELECT TRIM$(SYSTEM$("SYSTEM ID"))
2771       CASE "S300:20"    ! The Fast HPIB
2772           Diskdrive$=":,1400,1"
2773       CASE "9836C"      ! The Wiper card
2774           Diskdrive$=":,1500,2"
2775       CASE ELSE
2776           BEEP
2777           DISP "Save_fo_later ain't set up fo yo computer!"
2778           PAUSE
2779       END SELECT
2780       A$=DATE$(TIMEDATE)
2781       Filename$=TRIM$(A$[1,2])&A$[4,6]&A$[10,11]&VAL$(Num_peaks)
2782       Ssave=2 Sselect
2783 !     CALL New_save_sparms(Ssave)
2784       WAIT 5
2786       PRINT Filename$&Diskdrive$;" has been saved"
2787       DISP Filename$&Diskdrive$;" has been saved"
2788       RETURN
2789!
2790! ////////////////////////////////////////////////////////////
2791!
2792 Init_sweep_nwa:OPTION BASE 1
2793       INTEGER Done,Reset_keys,Bad_number,Ssave
2794       REAL Startfreq,Stopfreq,Stepfreq,Minfreq,Maxfreq
2795       REAL Min_peak_val,Min_sig_to_noi
2796       REAL Freq,Level,Span
2797       DIM Id$[40]
2798 Init_com:DEG
2799       COM /History/ Status$[1],Time_orgn$[8],Date_orgn$[11]
2800       COM /History/ Time_chng$[8],Date_chng$[11],Description$[160]
2801       COM /Labels/ Labels$(30)[60],INTEGER Lbl_count,REAL Lbl_addr(30,6)
2802       COM /Data_param/ INTEGER Data_count,Filesize,Curvecount,Roster(17,4)
2803       COM /Data_param/ REAL Sym_size,Symbol$(17)[2],Curve_id$(17)[40]
2804       COM /Data_param/ REAL Xmin_data,Xmax_data
2805       COM /Data_param/ REAL Ymin_data,Ymax_data
```

```
2806      COM /Background/ Graphtype$[12],Margins$(2)[10],Papersize$[1]
2807      COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
2808      COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
2809      COM /Background/ Xgrid_tick$[4],INTEGER Xmajor,Xminor
2810      COM /Background/ Ygrid_tick$[4],INTEGER Ymajor,Yminor
2811      COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
2812      COM /Axes_labels/ Print_xlabels$[3],Print_ylabels$[3]
2813      COM /Axes_labels/ Sig_digits$[2],REAL Xlcsize,Ylcsize
2814      COM /Windowspace/ REAL Xmin,Xmid,Xmax,Ymin,Ymid,Ymax! graph edges UDU
2815      COM /Windowspace/ REAL Xleft,Xright,Ybottom,Ytop! paper edges UDUs
2816      COM /Log_scale/ REAL Xcycles,Xbegin,Ycycles,Ybegin
2817      COM /Plot_device/ Plot_lang$[10],INTEGER Plot_addr
2818      COM /Hard_space/ REAL Xview_lft,Xview_rt,Yview_btm,Yview_top
2819      COM /Hard_space/ REAL Viewscale,Aspect_ratio
2820      COM /Frame_onoff/ INTEGER Frame_flag
2821      COM /Clear_space/ REAL Space_lft,Space_rt,Space_btm,Space_top
2822 !    !
2823      COM /S_array/ INTEGER Dcount,Svalid,REAL S(801,5,2)
2824      COM /Fittype/ INTEGER Sselect,Start_index,Stop_index
2825      COM /Cdata/ INTEGER Mfit(2),Lista(5,2),Ma(2),REAL A(5),Ua(5),INTEGER
  => Ndata(2)
2826      COM /Cstats/ Alamda,Chisq,INTEGER Nca(2),REAL Alpha(5,5),Covar(5,5)
2827      COM /Files/ Diskdrive$[20],Filename$[10]
2828      COM /Bugs/ INTEGER Bug1,Bug2,Bug3,Printer
2829      COM /Nwa_data/ Start_freq,Stop_freq,Marker(*)
2830      COM /Peak_data/ Peak(*),Upeak(*),INTEGER Peak_index,Num_peaks
2831 !    COM /Resonance/ Kx,Ky,Q,F0,Ukx,Uky,Uq,Uf0
2832      COM /Output/ INTEGER Print_addr,Plotter,Printer_on,Plotter_on
2833      ASSIGN @Nwa TO 716
2834      ASSIGN @Nwa_data TO 716;FORMAT ON
2835 Init_vals:!
2836      Minfreq=6.55E+9
2837      Maxfreq=1.35E+10
2838      Startfreq=8.2E+9
2839      Stopfreq=1.24E+10
2840      Stepfreq=8.0E+7
2841      Min_peak_val=-55
2842      Min_sig_to_noi=10
2843      Num_peaks=0
2844      Peak_index=0
2845      Save_fo_later=1
2846 Init_sweep:OFF KEY
2847      GOSUB Sweep_menu
2848      Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
2849      ON KEY 0 LABEL "OK/DONE",Prty GOSUB Done_return
2850      ON KEY 1 LABEL "Startfreq",Prty GOSUB Startfreq
2851      ON KEY 2 LABEL "Stopfreq",Prty GOSUB Stopfreq
2852      ON KEY 3 LABEL "Stepfreq",Prty GOSUB Stepfreq
2853      ON KEY 6 LABEL "Min_peak_val",Prty GOSUB Min_peak_val
2854      ON KEY 7 LABEL "Min_sig_to_noi",Prty GOSUB Min_sig_to_noi
2855      IF Printer_on THEN
2856         ON KEY 8 LABEL "No printer",Prty GOSUB Tog_printer
2857      ELSE
2858         ON KEY 8 LABEL "Printer ON",Prty GOSUB Tog_printer
2859      END IF
2860      IF Save_fo_later THEN
2861         ON KEY 9 LABEL "Compute Sparms",Prty GOSUB Tog_savefolater
2862      ELSE
2863         ON KEY 9 LABEL "Save Sparms",Prty GOSUB Tog_savefolater
2864      END IF
2865      Done=0
2866      Reset_keys=0
2867      REPEAT
2868         IF Reset_keys THEN GOTO Init_sweep
2869      UNTIL Done
2870      OFF KEY
2871      RETURN
2872 !
2873 ! //////////////////////////////////////////////////////////////
2874 !
2875 Tog_printer:OFF KEY 8
2876      IF Printer_on THEN
2877         Printer_on=0
2878      ELSE
2879         Printer_on=1
2880      END IF
2881      Reset_keys=1
2882      RETURN
2883 !
2884 Done_return:Done=1
2885      RETURN
2886 !
2887 Startfreq:CALL Enter_real(Minfreq,Stopfreq,"Sweep start frequency",Startfre
  => q)
2888      Reset_keys=1
2889      RETURN
2890 !
2891 Stopfreq:CALL Enter_real(Startfreq,Maxfreq,"Sweep stop frequency",Stopfreq)
2892      Reset_keys=1
2893      RETURN
2894 !
2895 Stepfreq:CALL Enter_real(1.E+4,3.E+8,"Sweep step span (Hz)",Stepfreq)
2896      Reset_keys=1
2897      RETURN
2898 !
2899 Min_peak_val:CALL Enter_real(-120,20,"Minimum resonance peak value (dB)",Mi
  => n_peak_val)
2900      Reset_keys=1
2901      RETURN
2902 !
2903 Min_sig_to_noi:CALL Enter_real(3,50,"Minimum signal-to-noise ratio",Min_sig
  => _to_noi)
2904      Reset_keys=1
2905      RETURN
2906 !
2907 Tog_savefolater:Reset_keys=1
2908      IF Save_fo_later THEN
2909         Save_fo_later=0
2910      ELSE
2911         Save_fo_later=1
2912      END IF
2913      RETURN
2914 !
2915 Sweep_menu:PRINT TABXY(1,1);
2916      IF Startfreq>1.E+10 THEN
2917         PRINT USING "K,2D.6DE,K";"Startfreq=",Startfreq," Hz"
2918      ELSE
2919         PRINT USING "K,D.6DE,K";"Startfreq=",Startfreq," Hz"
2920      END IF
2921      IF Stopfreq>1.E+10 THEN
2922         PRINT USING "K,2D.6DE,K";"Stopfreq=",Stopfreq," Hz"
2923      ELSE
2924         PRINT USING "K,D.6DE,K";"Stopfreq=",Stopfreq," Hz"
2925      END IF
2926      PRINT USING "K,D.2DE,K";"Stepfreq=",Stepfreq," Hz"
2927      PRINT
2928      PRINT USING "K,M3D,K";"Min_peak_val=",Min_peak_val," dB"
2929      PRINT USING "K,M3D,K";"Min_sig_to_noi=",Min_sig_to_noi," dB"
2930      PRINT
2931      IF Printer_on THEN
2932         PRINT "Printer is ON              Print_addr=";Print_addr;"   "
2933      ELSE
```

```
2934          PRINT "Printer is OFF              Print_addr=";Print_addr;"   "
2935      END IF
2936      PRINT
2937      IF Save_fo_later THEN
2938          PRINT "The S-parameters will be saved on disk. Resonance paramete
 => rs won't be computed."
2939      ELSE
2940          PRINT "The resonance parameters will be computed.
 =>              "
2941      END IF
2942      RETURN
2943 !
2944 ! ///////////////////////////////////////////////////////////////
2945 !
2946 Print_info:!
2947      ALLOCATE Desc$[160]
2948      REPEAT
2949          LINPUT "Enter a description of this sweep. When finished enter a
 => null string.",Desc$
2950          PRINT Desc$
2951          IF Printer_on THEN OUTPUT Print_addr;Desc$
2952      UNTIL Desc$=""
2953      DEALLOCATE Desc$
2954      RETURN
2955 !
2956 ! ///////////////////////////////////////////////////////////////
2957 !
2958 Add_peak_2_list:!
2959      SELECT Sselect
2960      CASE 1,2
2961          Peak(Num_peaks,1)=A(3)
2962          Peak(Num_peaks,2)=SQRT(A(1)^2+A(2)^2)
2963          Peak(Num_peaks,3)=A(3)/A(4)
2964      CASE ELSE
2965          BEEP
2966          DISP "Sweep_nwa needs to have S11 measurements added"
2967          PAUSE
2968      END SELECT
2969      RETURN
2970 !
2971 ! ///////////////////////////////////////////////////////////////
2972 !
2973 Save_sweepfreqs:OFF KEY
2974      IF Save_fo_later THEN RETURN
2975      Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
2976      ON KEY 0 LABEL "NO",Prty GOSUB Done_return
2977      ON KEY 2 LABEL "YES",Prty GOSUB Freqs_to_disk
2978      DISP "Do you want to save the resonant frequency list to disk?"
2979      Done=0
2980      LOOP
2981      EXIT IF Done
2982      END LOOP
2983      RETURN
2984 !
2985 ! ///////////////////////////////////////////////////////////////
2986 !
2987 Freqs_to_disk:OFF KEY
2988      ALLOCATE Basket(Num_peaks,2)
2989      MAT Basket(1:Num_peaks,1)= Peak(1:Num_peaks,1)
2990      MAT Basket(1:Num_peaks,2)= Peak(1:Num_peaks,3)
2991      CALL Enter_id(Id$,"the resonant frequency list")
2992      IF Id$="" THEN GOTO Abort_savefreqs
2993      CALL Select_disk
2994      IF Diskdrive$="NO DISK" THEN GOTO Abort_savefreqs
2995      CALL Enterfilename("ABORT"," resonant frequency list")
2996      IF Filename$="" THEN GOTO Abort_savefreqs
2997      CALL Save_file(Basket(*),Num_peaks,Id$)
2998 Abort_savefreqs:!
2999      Done=1
3000      RETURN
3001 !
3002 ! ///////////////////////////////////////////////////////////////
3003 !
3004 End_sweep_nwa:!
3005 !    CALL Set_nwa("PRES=")
3006      IF Save_fo_later THEN     !This array hasn't been used
3007          Peak(Num_peaks,1)=1.063E+10
3008          Peak(Num_peaks,3)=2.E+7
3009          Peak(Num_peaks,2)=-25
3010      END IF
3011      Peak_index=1
3012      CALL Set_nwa("CENTER="&VAL$(Peak(Peak_index,1)))
3013      CALL Set_nwa("SPAN="&VAL$(Peak(Peak_index,3)*5))
3014      CALL Set_nwa("REF VAL="&VAL$(Peak(Peak_index,2)+5),"SCALE=5")
3015      CALL Set_nwa("S=21","REF POS=10")
3016      LOCAL @Nwa
3017      IF Printer_on THEN OUTPUT Print_addr;CHR$(12);
3018      BEEP
3019!     GOSUB Save_sweepfreqs
3020      DISP "I am finished with the experiment oh great master."
3021   SUBEND
3022 !
3023 ! ***************************************************************
3024 !
3025   SUB Pack_data(File(*),OPTIONAL REAL F1(*),INTEGER Data_cnt,Pen,Id$)
3026 Pack_data:  ! Original: 01 Jun 1987    G. Koepke
3027             ! Revision: 07 Aug 1987
3028      ! This routine will take up to 17 independent data files and pack
3029      ! the information into File(*) using GRAPH_DATA master file format.
3030      ! The Roster(*) information will be generated for use by Plot_all.
3031      !
3032      ! Roster(i,1) = Curve number 1,2,3,...,17
3033      ! Roster(i,2) = Start address in File(x,*); = x
3034      ! Roster(i,3) = Datacount for curve i
3035      ! Roster(i,4) = PEN number for this curve
3036      !
3037      ! Symbol$(i)="" or "Y" => no symbol, connect pts
3038      ! Symbol$(i)="*Y" =>  * symbol, connect pts
3039      ! Symbol$(i)="*N" =>  * symbol, do not connect pts
3040      !
3041      OPTION BASE 1
3042      COM /Data_param/ INTEGER Datacount,Filesize,Curvecount,Roster(*)
3043      COM /Data_param/ REAL Sym_size,Symbol$(*),Curve_id$(*)
3044      COM /Data_param/ REAL Xmin_data,Xmax_data
3045      COM /Data_param/ REAL Ymin_data,Ymax_data
3046      COM /Bugs/ INTEGER Bug1,Bug2,Bug3,Printer
3047      INTEGER I,J,C
3048      Filesize=SIZE(File,1)
3049      IF NPAR=1 THEN   ! Clear data parameters and exit
3050          Datacount=0
3051          Curvecount=0
3052          MAT Roster= (0)
3053          MAT Symbol$= ("")
3054          MAT Curve_id$= ("")
3055          SUBEXIT
3056      END IF
3057      IF Data_cnt<1 THEN
3058          DISP " NO data in this file  (packing aborted...): ";Id$
3059          BEEP
3060          PAUSE
3061          SUBEXIT
3062      END IF
```

```
3063      !
3064      ! Add new file to the data being plotted.
3065      !
3066      Timer=TIMEDATE
3067      DISP " Packing ... ";Id$
3068      Pen=MIN(MAX(Pen,1),8)
3069      SELECT Curvecount
3070      CASE 0
3071          Curvecount=1
3072          I=Curvecount
3073          Roster(I,1)=1
3074          Roster(I,2)=1
3075      CASE <17
3076          Curvecount=Curvecount+1
3077          I=Curvecount
3078          Roster(I,1)=I
3079          Roster(I,2)=Roster(I-1,2)+Roster(I-1,3)
3080      CASE ELSE
3081          DISP " CURVE limit has been reached, new data discarded. ";
3082          DISP " (continue) "
3083          BEEP
3084          PAUSE
3085          DISP CHR$(12);
3086          SUBEXIT
3087      END SELECT
3088      Roster(I,3)=Data_cnt
3089      Roster(I,4)=Pen
3090      Symbol$(I)=""
3091      Curve_id$(I)=Id$
3092      !
3093      IF Roster(I,2)+Roster(I,3)-1>Filesize THEN
3094          DISP " DATA FILE overflow, new data discarded. ";
3095          DISP " (continue) "
3096          BEEP
3097          PAUSE
3098          Curvecount=Curvecount-1
3099          DISP CHR$(12)
3100          SUBEXIT
3101      END IF
3102      !
3103      ! Copy data into File(*)
3104      !
3105      C=1
3106      FOR J=Roster(I,2) TO Roster(I,2)+Roster(I,3)-1
3107          File(J,1)=F1(C,1)
3108          File(J,2)=F1(C,2)
3109          C=C+1
3110      NEXT J
3111      LOOP
3112      EXIT IF TIMEDATE-Timer>1
3113      END LOOP
3114      DISP CHR$(12)
3115      SUBEXIT
3116      !
3117  SUBEND
3118  !
3119  ! ************************************************************
3120  !
3121  SUB Init_graphics(Label1$,Label2$,Label3$)
3122  !
3123 Init_graphics:  ! Original: 01 Jun 1987   G. Koepke
3124                 ! Revision: 07 Aug 1987
3125      ! Roster(i,1) = Curve number 1,2,3,...,17
3126      ! Roster(i,2) = Start address in File(x,*); = x
3127      ! Roster(i,3) = Datacount for curve i
3128      ! Roster(i,4) = PEN number for this curve
3129      !
3130      ! Symbol$(i)="" or "Y" => no symbol, connect pts
3131      ! Symbol$(i)="*Y" =>  * symbol, connect pts
3132      ! Symbol$(i)="*N" =>  * symbol, do not connect pts
3133      ! Lbl_addr: x, y, pen, size, LDIR, LORG
3134      !
3135      COM /Labels/ Labels$(*),INTEGER Lbl_count,REAL Lbl_addr(*)
3136      COM /Data_param/ INTEGER Datacount,Filesize,Curvecount,Roster(*)
3137      COM /Data_param/ REAL Sym_size,Symbol$(*),Curve_id$(*)
3138      COM /Data_param/ REAL Xmin_data,Xmax_data
3139      COM /Data_param/ REAL Ymin_data,Ymax_data
3140      COM /Background/ Graphtype$,Margins$(*),Papersize$
3141      COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
3142      COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
3143      COM /Background/ Xgrid_tick$,INTEGER Xmajor,Xminor
3144      COM /Background/ Ygrid_tick$,INTEGER Ymajor,Yminor
3145      COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
3146      !
3147      COM /Axes_labels/ Print_xlabels$,Print_ylabels$
3148      COM /Axes_labels/ Sig_digits$,REAL Xlcsize,Ylcsize
3149      !
3150      COM /Windowspace/ REAL Xmin,Xmid,Xmax,Ymin,Ymid,Ymax!graph edges UDU
3151      COM /Windowspace/ REAL Xleft,Xright,Ybottom,Ytop! paper edges UDUs
3152      COM /Plot_device/ Plot_lang$,INTEGER Plot_addr
3153      COM /Hard_space/ REAL Xview_lft,Xview_rt,Yview_btm,Yview_top
3154      COM /Hard_space/ REAL Viewscale,Aspect_ratio
3155      COM /Frame_onoff/ INTEGER Frame_flag
3156      COM /Clear_space/ REAL Space_lft,Space_rt,Space_btm,Space_top
3157      COM /Bugs/ INTEGER Bug1,Bug2,Bug3,Printer
3158      !
3159      ! //////////////// INITIAL VALUES ////////////////////////////
3160      !
3161 Ginit_clear:GINIT  ! Clear all graphics
3162      GCLEAR
3163      OUTPUT 2 USING "#,K";" K"   ! Clear the screen
3164 Initial_values:      !DEFINE ALL INITIAL VALUES
3165      !
3166      Frame_flag=1          ! Completely frame the data area.
3167      Print_xlabels$="YES"! AND print labels.
3168      Print_ylabels$="YES"
3169      Sig_digits$="FF"      ! Select free format for X & Y labels.
3170      Xlcsize=.035          ! Label size factor for axes labels.
3171      Ylcsize=.035          ! graph_data default size = .032
3172 !    Plot_lang$="INTERNAL"! OR "HPGL"
3173 !    Plot_addr=3           ! OR 705 (HPIB ADDR OF PLOTTER)
3174      Margins$(1)="HORIZONTAL"! OR "VERTICAL"  for plotter only
3175      Margins$(2)="FULL"   ! OR "BOUND LEFT"
3176                           ! OR "BOUND TOP"
3177                           ! OR "SQUARE"
3178                           ! OR "USER"
3179      Papersize$="4"        ! 8.5x11
3180                           ! OR "3", 11x17
3181      Backgnd_pen=1         ! 1 to 8
3182      Auto_time=1           ! Automatically label time/date on graphs.
3183      Auto_file=0           ! Automatically label last file name.
3184      Pen_speed=20.0        ! .38 cm/s TO 38.1 cm/s
3185      !
3186      ! SET VIEWPORT PARAMETERS
3187      !
3188      Xview_lft=34.4
3189      Xview_rt=100*RATIO
3190      Yview_btm=24
3191      Yview_top=100
3192      ! Clear space around labels in terms of frac of scale
3193      Space_lft=.21
3194      Space_rt=.02
```

```
3195        Space_btm=.11
3196        Space_top=.10
3197        !
3198        ! Background parameters...........................
3199        !
3200        Graphtype$="LINEAR"
3201        Xgrid_tick$="TICK"
3202        Ygrid_tick$="TICK"
3203        Xmajor=6
3204        Ymajor=1
3205        Xminor=3
3206        Yminor=1
3207        Xmin_graph=0
3208        Xmax_graph=1.8E+10
3209        Ymin_graph=0
3210        Ymax_graph=1
3211        X_cross_y=Ymin_graph
3212        Y_cross_x=Xmin_graph
3213        !
3214        Xmin=Xmin_graph
3215        Xmax=Xmax_graph
3216        Xmid=Xmin+((Xmax-Xmin)/2)
3217        Ymin=Ymin_graph
3218        Ymax=Ymax_graph
3219        Ymid=Ymin+((Ymax-Ymin)/2)
3220        !
3221        ! Label parameters................................
3222        !
3223        Lbl_count=0
3224        MAT Labels$= ("")
3225        MAT Lbl_addr= (0)
3226        !
3227        ! Insert desired labels here using UDUs defined above.
3228        !
3229        Lbl_count=3
3230 !      Labels$(1)="Rotation angle (degrees)"
3231        Labels$(1)=Label1$
3232        Lbl_addr(1,1)=9.E+9 ! X location UDUs
3233        Lbl_addr(1,2)=-.05  ! Y location UDUs
3234        Lbl_addr(1,3)=5     ! PEN
3235        Lbl_addr(1,4)=.035  ! SIZE factor
3236        Lbl_addr(1,5)=0     ! LDIR
3237        Lbl_addr(1,6)=6     ! LORG
3238        !
3239 !      Labels$(2)="Relative amplitude (dB)"
3240        Labels$(2)=Label2$
3241        Lbl_addr(2,1)=-2.E+9! X location UDUs
3242        Lbl_addr(2,2)=.5    ! Y location UDUs
3243        Lbl_addr(2,3)=5     ! PEN
3244        Lbl_addr(2,4)=.035  ! SIZE factor
3245        Lbl_addr(2,5)=90    ! LDIR
3246        Lbl_addr(2,6)=4     ! LORG
3247        !
3248 !      Labels$(3)="Probe Calibration Data"
3249        Labels$(3)=Label3$
3250        Lbl_addr(3,1)=9.E+9 ! X location UDUs
3251        Lbl_addr(3,2)=1.02  ! Y location UDUs
3252        Lbl_addr(3,3)=5     ! PEN
3253        Lbl_addr(3,4)=.040  ! SIZE factor
3254        Lbl_addr(3,5)=0     ! LDIR
3255        Lbl_addr(3,6)=4     ! LORG
3256        !
3257        SUBEXIT
3258    SUBEND
3259    !
3260    ! ****************************************************************
3261    !
3262    SUB Plot_all(File(*))
3263 Plot_all:  ! Original: 13 Nov 1984
3264            ! Revision: 06 Aug 1987
3265        OPTION BASE 1
3266        DEG
3267        !
3268        COM /Sys/ Sys_id$[10]
3269        COM /Labels/ Labels$(*),INTEGER Lbl_count,REAL Lbl_addr(*)
3270        COM /Data_param/ INTEGER Datacount,Filesize,Curvecount,Roster(*)
3271        COM /Data_param/ REAL Sym_size,Symbol$(*),Curve_id$(*)
3272        COM /Data_param/ REAL Xmin_data,Xmax_data
3273        COM /Data_param/ REAL Ymin_data,Ymax_data
3274        COM /Background/ Graphtype$,Margins$(*),Papersize$
3275        COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
3276        COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
3277        COM /Background/ Xgrid_tick$,INTEGER Xmajor,Xminor
3278        COM /Background/ Ygrid_tick$,INTEGER Ymajor,Yminor
3279        COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
3280        !
3281        COM /Axes_labels/ Print_xlabels$,Print_ylabels$
3282        COM /Axes_labels/ Sig_digits$,REAL Xlcsize,Ylcsize
3283        !
3284        COM /Windowspace/ REAL Xmin,Xmid,Xmax,Ymin,Ymid,Ymax!graph edges UDU
3285        COM /Windowspace/ REAL Xleft,Xright,Ybottom,Ytop! paper edges UDUs
3286        COM /Plot_device/ Plot_lang$,INTEGER Plot_addr
3287        COM /Hard_space/ REAL Xview_lft,Xview_rt,Yview_btm,Yview_top
3288        COM /Hard_space/ REAL Viewscale,Aspect_ratio
3289        COM /Frame_onoff/ INTEGER Frame_flag
3290        COM /Clear_space/ REAL Space_lft,Space_rt,Space_btm,Space_top
3291        COM /Bugs/ INTEGER Bug1,Bug2,Bug3,Printer
3292        COM /Files/ Diskdrive$,Filename$
3293        INTEGER I,J,Local_prty,Suspended,Outofbounds
3294        !
3295        Aspect_ratio=(Xview_rt-Xview_lft)/((Yview_top-Yview_btm)*2)
3296        Viewscale=MIN((Xview_rt-Xview_lft),(Yview_top-Yview_btm))
3297        Aspect_ratio=MIN(MAX(Aspect_ratio,.35),.65)
3298        !
3299        Local_prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
3300        IF Lbl_count>0 THEN
3301            ALLOCATE Lbl_ratio(Lbl_count,2)
3302            GOSUB Update_lbl_addr    ! includes Find_mid_pt
3303            DEALLOCATE Lbl_ratio(*)
3304        ELSE
3305            GOSUB Find_mid_pt
3306        END IF
3307        CALL Background(Suspended)
3308        IF Suspended THEN GOTO Nomoregraph
3309        IF Plot_addr<>3 THEN
3310            ON KEY 0 LABEL "ABORT  GRAPH",Local_prty+1 GOTO Nomoregraph
3311        END IF
3312        CLIP OFF
3313        GOSUB Label_labels
3314        GOSUB Plot_data
3315 Nomoregraph:IF Plot_addr<>3 THEN OFF KEY
3316        PENUP
3317        PEN 0
3318        CLIP OFF
3319        SUBEXIT
3320        !
3321        ! ///////////////////////////////////////////////////////////
3322        !
3323 Update_lbl_addr:! Make sure that the label addresses are current.
3324        FOR I=1 TO Lbl_count
3325            Lbl_ratio(I,1)=(Lbl_addr(I,1)-Xmin)/(Xmax-Xmin)
3326            Lbl_ratio(I,2)=(Lbl_addr(I,2)-Ymin)/(Ymax-Ymin)
```

```
      NEXT I
      GOSUB Find_mid_pt            ! define new max and min
      FOR I=1 TO Lbl_count
          Lbl_addr(I,1)=Lbl_ratio(I,1)*(Xmax-Xmin)+Xmin
          Lbl_addr(I,2)=Lbl_ratio(I,2)*(Ymax-Ymin)+Ymin
      NEXT I
      RETURN
      !
      ! ///////////////////////////////////////////////////////////
      !
Find_mid_pt:     !Interpret the graph type and the scaling done.
      SELECT Graphtype$
      CASE "LINEAR"
          Xmin=Xmin_graph
          Xmax=Xmax_graph
          Xmid=((Xmax-Xmin)/2)+Xmin
          Ymin=Ymin_graph
          Ymax=Ymax_graph
          Ymid=((Ymax-Ymin)/2)+Ymin
      CASE "SEMILOG X"
          Xmin=0
          Xmax=100
          Xmid=50
          Ymin=Ymin_graph
          Ymax=Ymax_graph
          Ymid=((Ymax-Ymin)/2)+Ymin
      CASE "SEMILOG Y"
          Xmin=Xmin_graph
          Xmax=Xmax_graph
          Xmid=((Xmax-Xmin)/2)+Xmin
          Ymin=0
          Ymax=100
          Ymid=50
      CASE "LOG LOG"
          Xmin=0
          Xmax=100
          Xmid=50
          Ymin=0
          Ymax=100
          Ymid=50
      CASE "POLAR"
          !
          BEEP
          DISP "POLAR PARAMETERS ARE NOT YET IMPLEMENTED!!!!"
          PAUSE
      END SELECT
      RETURN
      !
      ! ///////////////////////////////////////////////////////////
      !
Label_labels:      !ALL LABELS ARE APPLIED
      PEN Backgnd_pen
      CSIZE DROUND(.025*Viewscale,3),Aspect_ratio
      IF Auto_time THEN
          LORG 1
          MOVE Xleft,Ybottom
          LABEL " "&DATE$(TIMEDATE)&", "&TIME$(TIMEDATE)
          PENUP
      END IF
      IF Auto_file THEN
          LORG 7
          MOVE Xright,Ybottom
          IF LEN(Filename$)>0 THEN
              LABEL "File: "&Filename$&"  "
          END IF
          PENUP
      END IF
      FOR I=1 TO Lbl_count
          LDIR Lbl_addr(I,5)
          IF Lbl_addr(I,6)>0 THEN
              LORG Lbl_addr(I,6)
          ELSE
              LORG 1
          END IF
          CSIZE DROUND(Lbl_addr(I,4)*Viewscale,3),Aspect_ratio
          PEN DROUND(Lbl_addr(I,3),1)
          MOVE Lbl_addr(I,1),Lbl_addr(I,2)
          LABEL USING "#,K";Labels$(I)
          PENUP
      NEXT I
      RETURN
      !
      ! ///////////////////////////////////////////////////////////
      !
Plot_data:          !PLOT ALL CURVES
      Disptime=0.
      CLIP Xmin,Xmax,Ymin,Ymax
      LORG 5
      LDIR 0
      CSIZE DROUND(.025*Viewscale,3),Aspect_ratio
      FOR I=1 TO Curvecount
          Outofbounds=0
          PENUP
          PEN Roster(I,4)
          Xpos=File(Roster(I,2),1)
          Ypos=File(Roster(I,2),2)
          GOSUB Adjust_xy_pos
          IF Xpos>Xmax THEN
              GOSUB Skipcurvei
              GOTO Skipcurvej
          END IF
          Xpos=MAX(MIN(Xpos,Xmax),Xmin)
          Ypos=MAX(MIN(Ypos,Ymax),Ymin)
          MOVE Xpos,Ypos
          SELECT LEN(Symbol$(Roster(I,1)))
          CASE 2
              Mark$=Symbol$(Roster(I,1))[1,1]
              Connect$=Symbol$(Roster(I,1))[2,2]
          CASE 1
              Mark$=Symbol$(Roster(I,1))[1,1]
              Connect$="Y"
          CASE ELSE
              Mark$=""
              Connect$="Y"
          END SELECT
          !
          IF Mark$="" AND Connect$="Y" THEN
              !No symbol
              FOR J=Roster(I,2) TO Roster(I,2)+Roster(I,3)-1
                  Xpos=File(J,1)
                  Ypos=File(J,2)
                  GOSUB Adjust_xy_pos
                  IF Xpos>Xmax THEN
                      GOSUB Skipcurvei
                      GOTO Skipcurvej
                  END IF
                  IF NOT Outofbounds THEN
                      IF Xpos<Xmin OR Ypos<Ymin OR Ypos>Ymax THEN
                          Outofbounds=1
                          GOSUB Skipcurvei
                      END IF
                  END IF
```

```
3459                    Xpos=MAX(MIN(Xpos,Xmax),Xmin)
3460                    Ypos=MAX(MIN(Ypos,Ymax),Ymin)
3461                    DRAW Xpos,Ypos
3462                NEXT J
3463            ELSE
3464                !Put in symbol
3465                SELECT Connect$
3466                CASE "Y"
3467                    FOR J=Roster(I,2) TO Roster(I,2)+Roster(I,3)-1
3468                        Xpos=File(J,1)
3469                        Ypos=File(J,2)
3470                        GOSUB Adjust_xy_pos
3471                        IF Xpos>Xmax THEN
3472                            GOSUB Skipcurvei
3473                            GOTO Skipcurvej
3474                        END IF
3475                        IF NOT Outofbounds THEN
3476                            IF Xpos<Xmin OR Ypos<Ymin OR Ypos>Ymax THEN
3477                                Outofbounds=1
3478                                GOSUB Skipcurvei
3479                            END IF
3480                        END IF
3481                        Xpos=MAX(MIN(Xpos,Xmax),Xmin)
3482                        Ypos=MAX(MIN(Ypos,Ymax),Ymin)
3483                        DRAW Xpos,Ypos
3484                        LABEL USING "#,K";Mark$
3485                        PENUP
3486                        MOVE Xpos,Ypos
3487                    NEXT J
3488                CASE "N"
3489                    FOR J=Roster(I,2) TO Roster(I,2)+Roster(I,3)-1
3490                        Xpos=File(J,1)
3491                        Ypos=File(J,2)
3492                        GOSUB Adjust_xy_pos
3493                        IF Xpos>Xmax THEN
3494                            GOSUB Skipcurvei
3495                            GOTO Skipcurvej
3496                        END IF
3497                        IF NOT Outofbounds THEN
3498                            IF Xpos<Xmin OR Ypos<Ymin OR Ypos>Ymax THEN
3499                                Outofbounds=1
3500                                GOSUB Skipcurvei
3501                            END IF
3502                        END IF
3503                        Xpos=MAX(MIN(Xpos,Xmax),Xmin)
3504                        Ypos=MAX(MIN(Ypos,Ymax),Ymin)
3505                        MOVE Xpos,Ypos
3506                        LABEL USING "#,K";Mark$
3507                        PENUP
3508                    NEXT J
3509                END SELECT
3510            END IF
3511 Skipcurvej: !
3512            PENUP
3513        NEXT I
3514        IF Disptime>0 THEN
3515            LOOP
3516            EXIT IF TIMEDATE-Disptime>1.2
3517            END LOOP
3518            DISP CHR$(12)
3519        END IF
3520        RETURN
3521        !
3522        ! ////////////////////////////////////////////////////////////////
3523        !
3524 Skipcurvei: !
3525        DISP " DATA OUT OF RANGE ... NOT PLOTTED "
3526        IF Disptime<1.0E-10 THEN
3527            Disptime=TIMEDATE
3528        END IF
3529        RETURN
3530        !
3531        ! ////////////////////////////////////////////////////////////////
3532        !
3533 Adjust_xy_pos:      !CORRECT Xpos, Ypos FOR VARIOUS AXES
3534        SELECT Graphtype$
3535        CASE "LINEAR"
3536            !NO CHANGE
3537        CASE "SEMILOG X"
3538            Xpos=FNLin_map_logx(Xpos)
3539            IF Xpos<0 THEN Xpos=0
3540        CASE "SEMILOG Y"
3541            Ypos=FNLin_map_logy(Ypos)
3542            IF Ypos<0 THEN Ypos=0
3543        CASE "LOG LOG"
3544            Xpos=FNLin_map_logx(Xpos)
3545            Ypos=FNLin_map_logy(Ypos)
3546            IF Xpos<0 THEN Xpos=0
3547            IF Ypos<0 THEN Ypos=0
3548        CASE "POLAR"
3549            !NO CHANGE yet.
3550        END SELECT
3551        RETURN
3552        !
3553        ! ////////////////////////////////////////////////////////////////
3554        !
3555        SUBEXIT
3556    SUBEND
3557    !
3558    ! ****************************************************************
3559    !
3560    SUB Background(INTEGER Suspended)
3561 Background:  ! Original: 13 Nov 1984
3562               ! Revision: 06 Aug 1987
3563      !This SUB program is written to draw the background for the PLOT
3564      !program.  It draws LINEAR, SEMILOG, LOG LOG, and POLAR backgrounds.
3565      !Parameters are determined in the main program.
3566      !
3567      !Due to the complexity of LOG and POLAR coordinates, the TICKing is
3568      !done with MOVEs and DRAWs.  Also avoided are device dependent codes.
3569      !
3570        OPTION BASE 1
3571        DEG
3572        COM /Sys/ Sys_id$
3573        COM /Background/ Graphtype$,Margins$(*),Papersize$
3574        COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
3575        COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
3576        COM /Background/ Xgrid_tick$,INTEGER Xmajor,Xminor
3577        COM /Background/ Ygrid_tick$,INTEGER Ymajor,Yminor
3578        COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
3579        COM /Windowspace/ REAL Xmin,Xmid,Xmax,Ymin,Ymid,Ymax! graph edges
3580        COM /Windowspace/ REAL Edge_lft,Edge_rt,Edge_btm,Edge_top
3581        COM /Plot_device/ Plot_lang$[10],INTEGER Plot_addr
3582        COM /Hard_space/ REAL Xview_lft,Xview_rt,Yview_btm,Yview_top
3583        COM /Hard_space/ REAL Viewscale,Aspect_ratio
3584        COM /Log_scale/ REAL Xcycles,Xbegin,Ycycles,Ybegin
3585        COM /Axes_labels/ Print_xlabels$,Print_ylabels$
3586        COM /Axes_labels/ Sig_digits$,REAL Xlcsize,Ylcsize
3587        COM /Frame_onoff/ INTEGER Frame_flag
3588        COM /Clear_space/ REAL Space_lft,Space_rt,Space_btm,Space_top
3589        !
3590        COM /Bugs/ INTEGER Bug1,Bug2,Bug3,Printer
```

```
3591        DIM Digies$[1]
3592        INTEGER Tic_size,Toggle,Log_label_limit,Log_label_step
3593        INTEGER Txminor,Tyminor
3594        REAL Xcrossy,Ycrossx,Xming,Xmaxg,Yming,Ymaxg
3595        !
3596        GRAPHICS ON
3597        IF Print_xlabels$="" THEN Print_xlabels$="YES"
3598        IF Print_ylabels$="" THEN Print_ylabels$="YES"
3599        IF Plot_addr<>3 THEN
3600            Suspended=0
3601            CALL Hp7475a_setup(Suspended)
3602            IF Suspended THEN GOTO End_plotting
3603            ON KEY 0 LABEL "ABORT  GRAPH",Loacal_prty+1 GOTO End_plotting
3604        ELSE
3605            VIEWPORT Xview_lft,Xview_rt,Yview_btm,Yview_top
3606            IF Frame_flag THEN FRAME
3607        END IF
3608        Tic_size=DROUND(.04*Viewscale,3)
3609        LDIR 0
3610        Log_tic_size=.04   !Per cent of total range.
3611        Tic_ratio=(Yview_top-Yview_btm)/(Xview_rt-Xview_lft)
3612        Xming=Xmin_graph  !copy for use here
3613        Xmaxg=Xmax_graph
3614        Yming=Ymin_graph
3615        Ymaxg=Ymax_graph
3616        Xcrossy=X_cross_y
3617        Ycrossx=Y_cross_x
3618        IF Ymajor<1 THEN Ymajor=1
3619        IF Yminor<1 THEN Yminor=1
3620        IF Xmajor<1 THEN Xmajor=1
3621        IF Xminor<1 THEN Xminor=1
3622        IF Xgrid_tick$="GRID" AND Xminor<>1 THEN
3623            Txminor=1
3624        ELSE
3625            Txminor=Xminor
3626        END IF
3627        IF Ygrid_tick$="GRID" AND Yminor<>1 THEN
3628            Tyminor=1
3629        ELSE
3630            Tyminor=Yminor
3631        END IF
3632        !
3633        Log_label_limit=4
3634        !
3635        PEN Backgnd_pen
3636        SELECT Graphtype$
3637        CASE "LINEAR"
3638            ! Set up user units.  Outside edges
3639            Xtics=(Xmaxg-Xming)/(Xmajor*Txminor)
3640            Ytics=(Ymaxg-Yming)/(Ymajor*Tyminor)
3641            GOSUB Window_space
3642            GOSUB Draw_x_linear
3643            IF Print_xlabels$="YES" THEN GOSUB Label_x_linear
3644            GOSUB Draw_y_linear
3645            IF Print_ylabels$="YES" THEN GOSUB Label_y_linear
3646            !
3647        CASE "SEMILOG X"
3648            ! Set up user units.  Outside edges
3649            Xming=0.        !All log operations are mapped into
3650            Xmaxg=100.0     !this range.  Y axis range is unchanged.
3651            Xtic_size=Log_tic_size*(Ymaxg-Yming)
3652            Find_log_cycles(Xmin_graph,Xmax_graph,Xcycles,Xbegin,Xend)
3653            IF Y_cross_x<2*Xbegin THEN
3654                Ycrossx=0.
3655            ELSE
3656                Ycrossx=FNLin_map_logx(Y_cross_x)
3657            END IF
3658            Ytics=(Ymaxg-Yming)/(Ymajor*Tyminor)
3659            GOSUB Window_space
3660            GOSUB Draw_x_log
3661            IF Print_xlabels$="YES" THEN GOSUB Label_x_log
3662            GOSUB Draw_y_linear
3663            IF Print_ylabels$="YES" THEN GOSUB Label_y_linear
3664            !
3665        CASE "SEMILOG Y"
3666            ! Set up user units.  Outside edges
3667            Yming=0.       !All log operations are mapped into this
3668            Ymaxg=100.     !range.  X axis is unchanged.
3669            Ytic_size=Log_tic_size*(Xmaxg-Xming)*Tic_ratio
3670            Find_log_cycles(Ymin_graph,Ymax_graph,Ycycles,Ybegin,Yend)
3671            IF X_cross_y<2*Ybegin THEN
3672                Xcrossy=0.
3673            ELSE
3674                Xcrossy=FNLin_map_logy(X_cross_y)
3675            END IF
3676            Xtics=(Xmaxg-Xming)/(Xmajor*Txminor)
3677            GOSUB Window_space
3678            GOSUB Draw_x_linear
3679            IF Print_xlabels$="YES" THEN GOSUB Label_x_linear
3680            GOSUB Draw_y_log
3681            IF Print_ylabels$="YES" THEN GOSUB Label_y_log
3682            !
3683        CASE "LOG LOG"
3684            ! Set up user units.  Outside edges
3685            Xming=0.     !All log operations are mapped into this
3686            Xmaxg=100.   !range.   Both axes.
3687            Yming=0.
3688            Ymaxg=100.
3689            Xtic_size=Log_tic_size*(Ymaxg-Yming)
3690            Ytic_size=Log_tic_size*(Xmaxg-Xming)*Tic_ratio
3691            Find_log_cycles(Xmin_graph,Xmax_graph,Xcycles,Xbegin,Xend)
3692            Find_log_cycles(Ymin_graph,Ymax_graph,Ycycles,Ybegin,Yend)
3693            IF X_cross_y<2*Ybegin THEN
3694                Xcrossy=0.
3695            ELSE
3696                Xcrossy=FNLin_map_logy(X_cross_y)
3697            END IF
3698            IF Y_cross_x<2*Xbegin THEN
3699                Ycrossx=0.
3700            ELSE
3701                Ycrossx=FNLin_map_logx(Y_cross_x)
3702            END IF
3703            GOSUB Window_space
3704            GOSUB Draw_x_log
3705            IF Print_xlabels$="YES" THEN GOSUB Label_x_log
3706            GOSUB Draw_y_log
3707            IF Print_ylabels$="YES" THEN GOSUB Label_y_log
3708            !
3709        CASE "POLAR"
3710            !??????????
3711        END SELECT
3712        GOTO Back_done
3713 End_plotting:Suspended=1
3714 Back_done:   !
3715        IF Plot_addr<>3 THEN OFF KEY
3716        PEN 0
3717        SUBEXIT
3718     !
3719     ! ///////////////////////////////////////////////////////////////
3720     !
3721 Window_space:     !Allow room outside of curve box for labels.
3722                   !Also scale to curve box and turn on CLIP to these
```

```
3723                   !limits.
3724       IF Margins$(2)="SQUARE" THEN
3725           Edge_lft=Xming-.21*(Xmaxg-Xming)
3726           Edge_rt=Xmaxg+.09*(Xmaxg-Xming)
3727           Edge_btm=Yming-.15*(Ymaxg-Yming)
3728           Edge_top=Ymaxg+.15*(Ymaxg-Yming)
3729       ELSE
3730           Edge_lft=Xming-Space_lft*(Xmaxg-Xming)
3731           Edge_rt=Xmaxg+Space_rt*(Xmaxg-Xming)
3732           Edge_btm=Yming-Space_btm*(Ymaxg-Yming)
3733           Edge_top=Ymaxg+Space_top*(Ymaxg-Yming)
3734       END IF
3735       WINDOW Edge_lft,Edge_rt,Edge_btm,Edge_top
3736       CLIP Xming,Xmaxg,Yming,Ymaxg
3737       IF Frame_flag THEN FRAME
3738       RETURN
3739      !
3740      ! ////////////////////////////////////////////////////////////
3741      !
3742 Draw_x_linear:     !
3743           !Draw X (horizontal) axes.
3744       SELECT Xgrid_tick$
3745       CASE "TICK"
3746           AXES Xtics,0,Xming-Xtics*Txminor,Xcrossy,Txminor,Tyminor,Tic_size
3747           !Select opposite axis if necessary
3748           SELECT Xcrossy
3749           CASE <=Yming+1.0E-10*(Ymaxg-Yming)
3750               AXES Xtics,0,Xming-Xtics*Txminor,Ymaxg,Txminor,Tyminor,Tic_si
  => ze
3751           CASE >=Ymaxg-1.0E-10*(Ymaxg-Yming)
3752               AXES Xtics,0,Xming-Xtics*Txminor,Yming,Txminor,Tyminor,Tic_si
  => ze
3753           END SELECT
3754       CASE "GRID"
3755           GRID Xtics,0,Ycrossx,Xcrossy,Txminor,Tyminor,Tic_size
3756       END SELECT
3757       RETURN
3758       !
3759       ! ////////////////////////////////////////////////////////////
3760       !
3761 Draw_y_linear:      !
3762           !Draw Y (vertical) axes.
3763       SELECT Ygrid_tick$
3764       CASE "TICK"
3765           AXES 0,Ytics,Ycrossx,Yming-Ytics*Tyminor,Txminor,Tyminor,Tic_size
3766           !Select the opposite axis if necessary.
3767           SELECT Ycrossx
3768           CASE <=Xming+1.0E-10*(Xmaxg-Xming)
3769               AXES 0,Ytics,Xmaxg,Yming-Ytics*Tyminor,Txminor,Tyminor,Tic_si
  => ze
3770           CASE >=Xmaxg-1.0E-10*(Xmaxg-Xming)
3771               AXES 0,Ytics,Xming,Yming-Ytics*Tyminor,Txminor,Tyminor,Tic_si
  => ze
3772           END SELECT
3773       CASE "GRID"
3774           GRID 0,Ytics,Ycrossx,Xcrossy,Txminor,Tyminor,Tic_size
3775       END SELECT
3776       RETURN
3777       !
3778       ! ////////////////////////////////////////////////////////////
3779       !
3780 Draw_x_log:      !Draw log axis according to x parameters.
3781       Full_cycles=INT(Xcycles)
3782       Div_beyond=10*FRACT(Xcycles)
3783       SELECT Xgrid_tick$
3784       CASE "TICK"
3785           MOVE Xming,Xcrossy-(Xtic_size/2)
3786           RPLOT 0,Xtic_size,-1
3787           FOR C=1 TO Full_cycles
3788               FOR D=2 TO T0
3789                   Xloc=FNLin_map_logx(D*Xbegin*(10^(C-1)))
3790                   MOVE Xloc,Xcrossy-(Xtic_size/2)
3791                   RPLOT 0,Xtic_size,-1
3792               NEXT D
3793           NEXT C
3794           FOR D=1 TO Div_beyond
3795               Xloc=FNLin_map_logx((D+1)*Xbegin*(10^(Full_cycles)))
3796               MOVE Xloc,Xcrossy-(Xtic_size/2)
3797               RPLOT 0,Xtic_size,-1
3798           NEXT D
3799           MOVE Xmaxg,Xcrossy-(Xtic_size/2)
3800           RPLOT 0,Xtic_size,-1
3801           MOVE Xmaxg,Xcrossy
3802           DRAW Xming,Xcrossy
3803           PENUP
3804           !
3805           !CHECK FOR OPPOSITE AXIS.
3806           !
3807           IF Xcrossy=Yming OR Xcrossy=Ymaxg THEN
3808               !Repeat for the opposite axis
3809               IF Xcrossy=Yming THEN
3810                   Second_axis=Ymaxg
3811               ELSE
3812                   Second_axis=Yming
3813               END IF
3814               MOVE Xming,Second_axis-(Xtic_size/2)
3815               RPLOT 0,Xtic_size,-1
3816               FOR C=1 TO Full_cycles
3817                   FOR D=2 TO T0
3818                       Xloc=FNLin_map_logx(D*Xbegin*(10^(C-1)))
3819                       MOVE Xloc,Second_axis-(Xtic_size/2)
3820                       RPLOT 0,Xtic_size,-1
3821                   NEXT D
3822               NEXT C
3823               FOR D=1 TO Div_beyond
3824                   Xloc=FNLin_map_logx((D+1)*Xbegin*(10^(Full_cycles)))
3825                   MOVE Xloc,Second_axis-(Xtic_size/2)
3826                   RPLOT 0,Xtic_size,-1
3827               NEXT D
3828               MOVE Xmaxg,Second_axis-(Xtic_size/2)
3829               RPLOT 0,Xtic_size,-1
3830               MOVE Xmaxg,Second_axis
3831               DRAW Xming,Second_axis
3832               PENUP
3833           END IF
3834       CASE "GRID"
3835           MOVE Xming,Yming
3836           DRAW Xming,Ymaxg
3837           Toggle=+1      !Toggles between Ymaxg and Yming.
3838                          !+1=Ymaxg,  -1=Yming
3839           FOR C=1 TO Full_cycles
3840               FOR D=2 TO T0
3841                   Xloc=FNLin_map_logx(D*Xbegin*(10^(C-1)))
3842                   SELECT Toggle
3843                   CASE +1
3844                       MOVE Xloc,Ymaxg
3845                       DRAW Xloc,Yming
3846                       Toggle=-Toggle
3847                   CASE -1
3848                       MOVE Xloc,Yming
3849                       DRAW Xloc,Ymaxg
3850                       Toggle=-Toggle
```

```
3851                END SELECT
3852            NEXT D
3853        NEXT C
3854        FOR D=1 TO Div_beyond+1
3855            Xloc=FNLin_map_logx((D+1)*Xbegin*(10^(Full_cycles)))
3856            SELECT Toggle
3857            CASE +1
3858                MOVE Xloc,Ymaxg
3859                DRAW Xloc,Yming
3860                Toggle=-Toggle
3861            CASE -1
3862                MOVE Xloc,Yming
3863                DRAW Xloc,Ymaxg
3864                Toggle=-Toggle
3865            END SELECT
3866        NEXT D
3867    END SELECT
3868    RETURN
3869    !
3870    ! ////////////////////////////////////////////////////////////
3871    !
3872 Draw_y_log:      !Draw log axis according to Y parameters.
3873    Full_cycles=INT(Ycycles)
3874    Div_beyond=10*FRACT(Ycycles)
3875    SELECT Ygrid_tick$
3876    CASE "TICK"
3877        MOVE Ycrossx-(Ytic_size/2),Yming
3878        RPLOT Ytic_size,0,-1
3879        FOR C=1 TO Full_cycles
3880            FOR D=2 TO 10
3881                Yloc=FNLin_map_logy(D*Ybegin*(10^(C-1)))
3882                MOVE Ycrossx-(Ytic_size/2),Yloc
3883                RPLOT Ytic_size,0,-1
3884            NEXT D
3885        NEXT C
3886        FOR D=1 TO Div_beyond
3887            Yloc=FNLin_map_logy((D+1)*Ybegin*(10^(Full_cycles)))
3888            MOVE Ycrossx-(Ytic_size/2),Yloc
3889            RPLOT Ytic_size,0,-1
3890        NEXT D
3891        MOVE Ycrossx-(Ytic_size/2),Ymaxg
3892        RPLOT Xtic_size,0,-1
3893        MOVE Ycrossx,Ymaxg
3894        DRAW Ycrossx,Yming
3895        PENUP
3896        !
3897        !CHECK FOR OPPOSITE AXIS.
3898        !
3899        IF Ycrossx=Xming OR Ycrossx=Xmaxg THEN
3900            !Repeat for the opposite axis
3901            IF Ycrossx=Xming THEN
3902                Second_axis=Xmaxg
3903            ELSE
3904                Second_axis=Xming
3905            END IF
3906            MOVE Second_axis-(Ytic_size/2),Yming
3907            RPLOT Ytic_size,0,-1
3908            FOR C=1 TO Full_cycles
3909                FOR D=2 TO 10
3910                    Yloc=FNLin_map_logy(D*Ybegin*(10^(C-1)))
3911                    MOVE Second_axis-(Ytic_size/2),Yloc
3912                    RPLOT Ytic_size,0,-1
3913                NEXT D
3914            NEXT C
3915            FOR D=1 TO Div_beyond
3916                Yloc=FNLin_map_logy((D+1)*Ybegin*(10^(Full_cycles)))
3917                MOVE Second_axis-(Ytic_size/2),Yloc
3918                RPLOT Ytic_size,0,-1
3919            NEXT D
3920            MOVE Second_axis-(Ytic_size/2),Ymaxg
3921            RPLOT Ytic_size,0,-1
3922            MOVE Second_axis,Ymaxg
3923            DRAW Second_axis,Yming
3924            PENUP
3925        END IF
3926    CASE "GRID"
3927        MOVE Xming,Yming
3928        DRAW Xmaxg,Yming
3929        Toggle=+1       !Toggles between Ymaxg and Yming.
3930                        !+1=Ymaxg,  -1=Yming
3931        FOR C=1 TO Full_cycles
3932            FOR D=2 TO 10
3933                Yloc=FNLin_map_logy(D*Ybegin*(10^(C-1)))
3934                SELECT Toggle
3935                CASE +1
3936                    MOVE Xmaxg,Yloc
3937                    DRAW Xming,Yloc
3938                    Toggle=-Toggle
3939                CASE -1
3940                    MOVE Xming,Yloc
3941                    DRAW Xmaxg,Yloc
3942                    Toggle=-Toggle
3943                END SELECT
3944            NEXT D
3945        NEXT C
3946        FOR D=1 TO Div_beyond+1
3947            Yloc=FNLin_map_logy((D+1)*Ybegin*(10^(Full_cycles)))
3948            SELECT Toggle
3949            CASE +1
3950                MOVE Xmaxg,Yloc
3951                DRAW Xming,Yloc
3952                Toggle=-Toggle
3953            CASE -1
3954                MOVE Xming,Yloc
3955                DRAW Xmaxg,Yloc
3956                Toggle=-Toggle
3957            END SELECT
3958        NEXT D
3959    END SELECT
3960    RETURN
3961    !
3962    ! ////////////////////////////////////////////////////////////
3963    !
3964 Fixed_sig_digit: ! Assure equal significant digits to right of .
3965                  ! Sig_digits$ = 1, 2, 3, or 4
3966                  !
3967    SELECT ABS(Numeric_label)
3968    CASE 1.0 TO 9.99
3969        SELECT Numeric_label
3970        CASE PROUND(Numeric_label,-INT(VAL(Digies$)))
3971            SELECT Digies$
3972            CASE "1"
3973                LABEL USING "MD.D";Numeric_label
3974            CASE "2"
3975                LABEL USING "MD.DD";Numeric_label
3976            CASE "3"
3977                LABEL USING "MD.3D";Numeric_label
3978            CASE "4"
3979                LABEL USING "MD.4D";Numeric_label
3980            END SELECT
3981        CASE ELSE
3982            LABEL USING "MD.DDESZ";Numeric_label
```

```
          END SELECT
          !
      CASE 9.99 TO 99.99
          SELECT Numeric_label
          CASE PROUND(Numeric_label,-INT(VAL(Digies$)))
              SELECT Digies$
              CASE "1"
                  LABEL USING "M2D.D";Numeric_label
              CASE "2"
                  LABEL USING "M2D.DD";Numeric_label
              CASE "3"
                  LABEL USING "M2D.3D";Numeric_label
              CASE "4"
                  LABEL USING "M2D.4D";Numeric_label
              END SELECT
          CASE ELSE
              LABEL USING "MD.DDESZ";Numeric_label
          END SELECT
          !
      CASE 99.99 TO 999.99
          SELECT Numeric_label
          CASE PROUND(Numeric_label,-INT(VAL(Digies$)))
              SELECT Digies$
              CASE "1"
                  LABEL USING "M3D.D";Numeric_label
              CASE "2"
                  LABEL USING "M3D.DD";Numeric_label
              CASE "3"
                  LABEL USING "M3D.3D";Numeric_label
              CASE "4"
                  LABEL USING "M3D.4D";Numeric_label
              END SELECT
          CASE ELSE
              LABEL USING "MD.DDESZ";Numeric_label
          END SELECT
          !
      CASE 999.99 TO 9999.99
          SELECT Numeric_label
          CASE PROUND(Numeric_label,-INT(VAL(Digies$)))
              SELECT Digies$
              CASE "1"
                  LABEL USING "M4D.D";Numeric_label
              CASE "2"
                  LABEL USING "M4D.DD";Numeric_label
              CASE "3"
                  LABEL USING "M4D.3D";Numeric_label
              CASE "4"
                  LABEL USING "M4D.4D";Numeric_label
              END SELECT
          CASE ELSE
              LABEL USING "MD.DDESZ";Numeric_label
          END SELECT
          !
      CASE >9999.99
          SELECT ABS(Numeric_label)
          CASE <1.0E+10
              LABEL USING "MD.DDESZ";Numeric_label
          CASE <1.0E+100
              LABEL USING "MD.DDESZZ";Numeric_label
          CASE ELSE
              LABEL USING "MD.DDESZZZ";Numeric_label
          END SELECT
          !
          !++++++All values less that 1.0 ++++++++++
          !
      CASE .0001 TO 1.0
          SELECT Numeric_label
          CASE PROUND(Numeric_label,-INT(VAL(Digies$)))
              SELECT Digies$
              CASE "1"
                  LABEL USING "MZ.D";Numeric_label
              CASE "2"
                  LABEL USING "MZ.DD";Numeric_label
              CASE "3"
                  LABEL USING "MZ.3D";Numeric_label
              CASE "4"
                  LABEL USING "MZ.4D";Numeric_label
              END SELECT
          CASE ELSE
              LABEL USING "MD.DDESZ";Numeric_label
          END SELECT
          !
      CASE ELSE
          IF ABS(Numeric_label)>1.0E-99 THEN
              IF PROUND(Numeric_label,-9)=Numeric_label THEN
                  LABEL USING "MD.DDESZ";Numeric_label
              ELSE
                  LABEL USING "MD.DDESZZ";Numeric_label
              END IF
          ELSE
              SELECT Digies$
              CASE "1"
                  LABEL USING "Z.D";ABS(Numeric_label)
              CASE "2"
                  LABEL USING "Z.DD";ABS(Numeric_label)
              CASE "3"
                  LABEL USING "Z.3D";ABS(Numeric_label)
              CASE "4"
                  LABEL USING "Z.4D";ABS(Numeric_label)
              END SELECT
          END IF
      END SELECT
      RETURN
      !
      ! ////////////////////////////////////////////////////////////
      !
Label_format:      !Select the MINIMUM number of digits for label.
                   !Then label the graph at the current location.
                   !
      Numeric_label=DROUND(Numeric_label,8)
      IF Digies$<>"F" THEN   ! Not minimum digits
          GOSUB Fixed_sig_digit  ! Fill zeros to right of .
          RETURN
      END IF
      SELECT ABS(Numeric_label)
      CASE 1.0 TO 9.99
          IF INT(Numeric_label)=Numeric_label THEN
              LABEL USING "MD";Numeric_label
          ELSE
              SELECT Numeric_label
              CASE PROUND(Numeric_label,-1)
                  LABEL USING "MD.D";Numeric_label
              CASE PROUND(Numeric_label,-2)
                  LABEL USING "MD.DD";Numeric_label
              CASE PROUND(Numeric_label,-3)
                  LABEL USING "MD.DDD";Numeric_label
              CASE ELSE
                  LABEL USING "MD.DDESZ";Numeric_label
              END SELECT
          END IF
          !
      CASE 9.99 TO 99.99
```

```
            IF INT(Numeric_label)=DROUND(Numeric_label,5) THEN
                LABEL USING "MDD";Numeric_label
            ELSE
                SELECT Numeric_label
                CASE PROUND(Numeric_label,-1)
                    LABEL USING "M2D.D";Numeric_label
                CASE PROUND(Numeric_label,-2)
                    LABEL USING "M2D.DD";Numeric_label
                CASE PROUND(Numeric_label,-3)
                    LABEL USING "M2D.DDD";Numeric_label
                CASE ELSE
                    LABEL USING "MD.DDESZ";Numeric_label
                END SELECT
            END IF
            !
        CASE 99.99 TO 999.99
            IF INT(Numeric_label)=DROUND(Numeric_label,6) THEN
                LABEL USING "MDDD";Numeric_label
            ELSE
                SELECT Numeric_label
                CASE PROUND(Numeric_label,-1)
                    LABEL USING "M3D.D";Numeric_label
                CASE PROUND(Numeric_label,-2)
                    LABEL USING "M3D.DD";Numeric_label
                CASE ELSE
                    LABEL USING "MD.DDESZ";Numeric_label
                END SELECT
            END IF
            !
        CASE 999.99 TO 9999.99
            IF INT(Numeric_label)=DROUND(Numeric_label,7) THEN
                LABEL USING "M4D";Numeric_label
            ELSE
                SELECT Numeric_label
                CASE PROUND(Numeric_label,-1)
                    LABEL USING "M4D.D";Numeric_label
                CASE PROUND(Numeric_label,-2)
                    LABEL USING "M4D.DD";Numeric_label
                CASE ELSE
                    LABEL USING "MD.DDESZ";Numeric_label
                END SELECT
            END IF
            !
        CASE >9999.99
            SELECT ABS(Numeric_label)
            CASE <1.0E+10
                LABEL USING "MD.DDESZ";Numeric_label
            CASE <1.0E+100
                LABEL USING "MD.DDESZZ";Numeric_label
            CASE ELSE
                LABEL USING "MD.DDESZZZ";Numeric_label
            END SELECT
            !
            !++++++All values less that 1.0 ++++++++++
            !
        CASE .0001 TO 1.0
            SELECT Numeric_label
            CASE PROUND(Numeric_label,-1)
                LABEL USING "MZ.D";Numeric_label
            CASE PROUND(Numeric_label,-2)
                LABEL USING "MZ.2D";Numeric_label
            CASE PROUND(Numeric_label,-3)
                LABEL USING "MZ.3D";Numeric_label
            CASE PROUND(Numeric_label,-4)
                LABEL USING "MZ.4D";Numeric_label
            CASE ELSE
                LABEL USING "MD.DDESZ";Numeric_label
            END SELECT
            !
        CASE ELSE
            IF ABS(Numeric_label)>1.0E-99 THEN
                IF PROUND(Numeric_label,-9)=Numeric_label THEN
                    LABEL USING "MD.DDESZ";Numeric_label
                ELSE
                    LABEL USING "MD.DDESZZ";Numeric_label
                END IF
            ELSE
                LABEL USING "D";Numeric_label
            END IF
        END SELECT
        RETURN
        !
        ! ///////////////////////////////////////////////////////////////
        !
Label_x_linear:       !Put numeric labels at every MAJOR tick mark.
        Digies$=Sig_digits$[1;1]  ! Select X significant digits
        CSIZE DROUND(Viewscale*Xlcsize,3),Aspect_ratio
        CLIP OFF
        LDIR 0
        Tick=Xtics*Txminor  !Divisions for labeling
        LORG 6
        FOR Numeric_label=Xming TO Xmaxg-.5*Tick STEP Tick
            MOVE Numeric_label,Yming-.005*(Ymaxg-Yming)
            GOSUB Label_format
        NEXT Numeric_label
        MOVE Xmaxg,Yming-.005*(Ymaxg-Yming)
        LORG 9
        Numeric_label=Xmaxg
        GOSUB Label_format
        PENUP
        CLIP ON
        RETURN
        !
        ! ///////////////////////////////////////////////////////////////
        !
Label_y_linear:       !Put numeric labels at every MAJOR tick mark.
        Digies$=Sig_digits$[2;1]  ! Select Y significant digits
        CSIZE DROUND(Viewscale*Ylcsize,3),Aspect_ratio
        CLIP OFF
        LDIR 0
        MOVE Xming-.005*(Xmaxg-Xming),Yming
        LORG 7
        Numeric_label=Yming
        GOSUB Label_format
        Tick=Ytics*Tyminor  !Divisions for labeling
        LORG 8
        FOR Numeric_label=Yming+Tick TO Ymaxg-.5*Tick STEP Tick
            MOVE Xming-.005*(Xmaxg-Xming),Numeric_label
            GOSUB Label_format
        NEXT Numeric_label
        MOVE Xming-.005*(Xmaxg-Xming),Ymaxg
        LORG 9
        Numeric_label=Ymaxg
        GOSUB Label_format
        PENUP
        CLIP ON
        RETURN
        !
        ! ///////////////////////////////////////////////////////////////
        !
Label_x_log:          !Put numeric labels at every log cycle and end.
                      !If more that 4 log cycles then thin the labels.
```

```
                  !where Log_label_limit=4.
      Digies$=Sig_digits$[1;1]  ! Select X significant digits
      CSIZE DROUND(Viewscale*Xlcsize,3),Aspect_ratio
      CLIP OFF
      LORG 6
      Yloc=Yming-.005*(Ymaxg-Yming)
      !Label left corner
      Numeric_label=Xbegin
      Xloc=FNLin_map_logx(Numeric_label)
      MOVE Xloc,Yloc
      GOSUB Label_format
      !
      IF INT(Xcycles)<Log_label_limit THEN
          FOR C=1 TO INT(Xcycles)
              Numeric_label=Xbegin*(10^C)
              Xloc=FNLin_map_logx(Numeric_label)
              MOVE Xloc,Yloc
              IF C=INT(Xcycles) AND INT(Xcycles)=Xcycles THEN LORG 9
              GOSUB Label_format
          NEXT C
      ELSE
          Log_label_step=INT(DROUND((Xcycles/(Log_label_limit-1)+.5),1))
          IF Log_label_step<1 THEN Log_label_step=1
          FOR C=0 TO INT(Xcycles) STEP Log_label_step
              IF C>0 THEN
                  Numeric_label=Xbegin*(10^C)
                  Xloc=FNLin_map_logx(Numeric_label)
                  MOVE Xloc,Yloc
                  IF C=INT(Xcycles) AND INT(Xcycles)=Xcycles THEN LORG 9
                  GOSUB Label_format
              END IF
          NEXT C
      END IF
      IF INT(Xcycles)<>Xcycles AND INT(Xcycles)<Log_label_limit-1 THEN
          IF FRACT(Xcycles)>.31 THEN !Only label end point if >3/10 cycle
              LORG 9
              MOVE Xmaxg,Yloc
              Numeric_label=Xend
              GOSUB Label_format
          END IF
      END IF
      CLIP ON
      RETURN
      !
      ! ///////////////////////////////////////////////////////////////
      !
Label_y_log:        !Put numeric labels at every log cycle and end.
      Digies$=Sig_digits$[2;1]  ! Select Y significant digits
      CSIZE DROUND(Viewscale*Ylcsize,3),Aspect_ratio
      CLIP OFF
      LORG 7
      Xloc=Xming-.005*(Xmaxg-Xming)
      !Label lower corner
      MOVE Xloc,Yming
      Numeric_label=Ybegin
      GOSUB Label_format
      LORG 8
      IF INT(Ycycles)<Log_label_limit THEN
          FOR C=1 TO INT(Ycycles)
              Numeric_label=Ybegin*(10^C)
              Yloc=FNLin_map_logy(Numeric_label)
              MOVE Xloc,Yloc
              IF C=INT(Ycycles) AND INT(Ycycles)=Ycycles THEN LORG 9
              GOSUB Label_format
          NEXT C
      ELSE
          Log_label_step=INT(DROUND((Ycycles/(Log_label_limit-1)+.5),1))
          IF Log_label_step<1 THEN Log_label_step=1
          FOR C=0 TO INT(Ycycles) STEP Log_label_step
              IF C>0 THEN
                  Numeric_label=Ybegin*(10^C)
                  Yloc=FNLin_map_logy(Numeric_label)
                  MOVE Xloc,Yloc
                  IF C=INT(Ycycles) AND INT(Ycycles)=Ycycles THEN LORG 9
                  GOSUB Label_format
              END IF
          NEXT C
      END IF
      IF INT(Ycycles)<>Ycycles AND INT(Ycycles)<Log_label_limit-1 THEN
          LORG 9
          MOVE Xloc,Ymaxg
          Numeric_label=Yend
          GOSUB Label_format
      END IF
      CLIP ON
      RETURN
      !
      ! ///////////////////////////////////////////////////////////////
      !
  SUBEND
  !
  ! ***************************************************************
  !
  SUB Find_log_cycles(Low,High,Cycles,New_low,New_high)
Find_log_cycles:! Original: 13 Nov 1984   G. Koepke
                ! Revision: 06 Aug 1987
                ! Determine the number of LOG cycles that will cover
                ! the range of MIN (>0) to MAX for either axis.  There will
                ! be at least ONE cycle and may be stopped at the first
                ! 1/10 cycle above the scale MAX.
                ! The variable 'Cycles' has as the integer part the number
                ! of FULL cycles and as the fractional part the number of
                ! 1/10 cycle divisions beyond the last FULL cycle.
                ! 'New_low' give a lower value with one digit.
                ! 'New_high' gives the new upper limit value.
                !
      COM /Bugs/ INTEGER Bug1,Bug2,Bug3,Printer
      INTEGER Exponent,M,N
      Exponent=0
      IF Low<1.0E-50 THEN
          DISP " Range error for LOG plot, begin point too small.";
          DISP ".. (continue). "
          BEEP
          PAUSE
          DISP CHR$(12)
          Low=1.0E-50
      END IF
      N=0
      SELECT Low
      CASE <1.0
          REPEAT
              N=N+1
              Test=Low*10.0^(N)
              IF Test>=1.0 THEN
                  Exponent=-N
                  Test=DROUND(Test,12)
                  New_low=INT(Test)*10^(Exponent)
              END IF
          UNTIL Exponent<>0
      CASE >=10.
          REPEAT
              N=N+1
```

```
4379            Test=Low*10.0^(-N)
4380            IF Test<10. THEN
4381                Exponent=N
4382                Test=DROUND(Test,12)
4383                New_low=INT(Test)*10^(Exponent)
4384                IF Bug3 THEN
4385                    PRINT USING "3(MD.16DE)";New_low,Test,INT(Test)
4386                END IF
4387            END IF
4388        UNTIL Exponent<>0
4389    END SELECT
4390    !
4391    IF Low>=1.0 AND Low<10.0 THEN
4392        New_low=INT(Low)
4393    END IF
4394    New_low=DROUND(New_low,1)
4395    !
4396    !
4397    IF High<=10*New_low THEN
4398        Cycles=1.0
4399    ELSE
4400        N=1
4401        LOOP
4402            Test=DROUND(New_low*10.0^(N),3)
4403            IF Bug3 THEN
4404                PRINT "New_low=";New_low;" Full Cycles=";N;
4405                PRINT " Test=";Test;" Max=";High
4406            END IF
4407        EXIT IF High<=Test
4408            N=N+1
4409        END LOOP
4410        IF Test=High THEN Find_range
4411        IF Bug3 THEN PRINT
4412        !
4413        M=0    ! 1/10 Cycles
4414        LOOP   !Find the number of divisions above full cycles - 1
4415               !necessary to cover the range.
4416               !
4417            M=M+1
4418            Test=DROUND((M+1)*New_low*10.0^(N-1),3)
4419            IF Bug3 THEN
4420                PRINT "Cycles=";N-1;"1/10s=";M;" Test=";Test;" Max=";High
4421            END IF
4422        EXIT IF High<=Test
4423        EXIT IF M>=9
4424        END LOOP
4425 Find_range:!
4426        SELECT M
4427        CASE 0,>=9
4428            Cycles=N
4429        CASE <9
4430            Cycles=(N-1)+(M/10)
4431        END SELECT
4432    END IF
4433    New_high=New_low*(10.0*FRACT(Cycles)+1)*10.0^(INT(Cycles))
4434    SUBEXIT
4435  SUBEND
4436  !
4437  ! ****************************************************************
4438  !
4439  DEF FNLin_map_logx(V)
4440    COM /Log_scale/ REAL Xcycles,Xbegin,Ycycles,Ybegin
4441    !
4442    !P = position, in linear units on that axis, of the LGT(V)
4443    !V = value to be mapped on the LOG scale.
4444    !B = begin LOG value at linear 0.0, ie. 0.1, 0.003, 10, etc.
4445    !R = range of linear scale = 100.0
4446    !C = number of cycles nn.m, nn = whole cycles, m = divisions
4447    !    beyond last whole cycle.
4448    !
4449    R=100.0
4450    B=Xbegin
4451    C=Xcycles
4452    IF V<B THEN V=B/10
4453    P=(R/(INT(C)+LGT(10*FRACT(C)+1)))*LGT(V/B)
4454    RETURN P
4455  FNEND
4456  !
4457  ! ****************************************************************
4458  !
4459  DEF FNLin_map_logy(V)
4460    COM /Log_scale/ REAL Xcycles,Xbegin,Ycycles,Ybegin
4461    !
4462    !P = position, in linear units on that axis, of the LGT(V)
4463    !V = value to be mapped on the LOG scale.
4464    !B = begin LOG value at linear 0.0, ie. 0.1, 0.003, 10, etc.
4465    !R = range of linear scale = 100.0
4466    !C = number of cycles nn.m, nn = whole cycles, m = divisions
4467    !    beyond last whole cycle.
4468    !
4469    R=100.0
4470    B=Ybegin
4471    C=Ycycles
4472    IF V<B THEN V=B/10
4473    P=(R/(INT(C)+LGT(10*FRACT(C)+1)))*LGT(V/B)
4474    RETURN P
4475    !
4476  FNEND
4477  !
4478  ! ****************************************************************
4479  !
4480  SUB Hp7475a_setup(INTEGER Suspended)
4481 Hp7475a_setup:! Original: 13 Nov 1984
4482              ! Revision: 06 Aug 1987
4483    !Optimize use of the HP 7475a plotter to draw
4484    !various axes types.  This is the first step to
4485    !draw the background for the graph.
4486    !
4487    OPTION BASE 1
4488    DEG
4489    COM /Background/ Graphtype$,Margins$(*),Papersize$
4490    COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
4491    COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
4492    COM /Background/ Xgrid_tick$,INTEGER Xmajor,Xminor
4493    COM /Background/ Ygrid_tick$,INTEGER Ymajor,Yminor
4494    COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
4495    COM /Hard_space/ REAL Xview_lft,Xview_rt,Yview_btm,Yview_top
4496    COM /Hard_space/ REAL Viewscale,Aspect_ratio
4497    COM /Plot_device/ Plot_lang$,INTEGER Plot_addr
4498    !
4499    INTEGER Hllx,Hlly,Hurx,Hury
4500    DIM Output$[80]
4501    ASSIGN @Plotter TO Plot_addr
4502    !
4503    GOSUB Initial_7475a
4504    IF NOT Suspended THEN GOSUB Select_margin
4505    SUBEXIT
4506    !
4507    ! ////////////////////////////////////////////////////////////
4508    !
4509 Select_margin:      !Set margins and VIEWPORT.
4510    IF Graphtype$="POLAR" THEN   !Must define polar parameters
```

```
      !    Margins$(2)="SQUARE"     !through out program.
      !END IF
      IF Margins$(2)="USER DEFN" THEN
          GOSUB User
          GOTO Skipmargins
      END IF
      SELECT Margins$(1)
      CASE "HORIZONTAL"
          GOSUB Horizon_setup
          SELECT Margins$(2)
          CASE "BOUND TOP"
              GOSUB H_boundtop
          CASE "BOUND LEFT"
              GOSUB H_boundleft
          CASE "FULL"
              GOSUB H_full
          CASE "SQUARE"
              GOSUB H_square
          CASE ELSE
              GOSUB User
          END SELECT
      CASE "VERTICAL"
          GOSUB Vertical_setup
          SELECT Margins$(2)
          CASE "BOUND TOP"
              GOSUB V_boundtop
          CASE "BOUND LEFT"
              GOSUB V_boundleft
          CASE "FULL"
              GOSUB V_full
          CASE "SQUARE"
              GOSUB V_square
          CASE ELSE
              GOSUB User
          END SELECT
      END SELECT
Skipmargins:!
      !
      !Scale the Viewscale and Aspect_ratio for the lettering.
      !
      Xview_lft=Left_mar
      Xview_rt=Right_mar
      Yview_btm=Bottom_mar
      Yview_top=Top_mar
      SELECT Margins$(1)
      CASE "HORIZONTAL"
          Aspect_ratio=(Xview_rt-Xview_lft)/((Yview_top-Yview_btm)*2)
      CASE "VERTICAL"
          Aspect_ratio=(Yview_top-Yview_btm)/((Xview_rt-Xview_lft)*2)
      END SELECT
      Aspect_ratio=MIN(MAX(Aspect_ratio,.3),1.5)
      Viewscale=MIN((Xview_rt-Xview_lft),(Yview_top-Yview_btm))
      VIEWPORT Left_mar,Right_mar,Bottom_mar,Top_mar
      IF Bug2 THEN
          PRINTER IS Printer
          PRINT "P1, P2 coordinates are (x,y): ";Hllx,Hlly,Hurx,Hury
          PRINT "Viewport (Xl,Xr,Yb,Yt);";
          PRINT Left_mar,Right_mar,Bottom_mar,Top_mar
          PRINT USING "5/"
          PRINTER IS CRT
          BEEP
          PAUSE
      END IF
      RETURN
      !
      ! ///////////////////////////////////////////////////////////////
      !
Initial_7475a:!
      !Initialize 7475A plotter, and expand P1 & P2 to include maximum
      !plotting area.
      !
      ON TIMEOUT 7,12 GOTO Plotter_dead   !Plotter hangs bus!
      OUTPUT @Plotter;"IN"                 !Initialize 7475A plotter.
      OUTPUT @Plotter;"PS"&Papersize$      !Select paper size
      OUTPUT @Plotter;"VS"&VAL$(Pen_speed) !Set the pen to Pen_speed
                                           !and slow acceleration to .2
      IF Margins$(1)="VERTICAL" THEN
          OUTPUT @Plotter;"RO90;IP;IW"
      END IF
      IF Margins$(2)="USER DEFN" THEN
          OUTPUT @Plotter;"OP"          !Send the P1,P2 coordinates.
          ENTER @Plotter;Hllx,Hlly,Hurx,Hury
          OUTPUT @Plotter;"PU"
          OUTPUT @Plotter;"SP1"
Test_p1_p2:!
          Output$="PA"&VAL$(Hllx)&","&VAL$(Hlly)
          OUTPUT @Plotter;Output$
          BEEP
          DISP " MOVE PEN TO DESIRED LOCATION for LOWER LEFT CORNER.";
          DISP "  ...press CONTINUE. "
          PAUSE
          OUTPUT @Plotter;"OA"
          ENTER @Plotter;Hllx,Hlly,Pen_status
          !
          Output$="PA"&VAL$(Hurx)&","&VAL$(Hury)
          OUTPUT @Plotter;Output$
          BEEP
          DISP " MOVE PEN TO DESIRED LOCATION for UPPER RIGHT CORNER.";
          DISP "  ...press CONTINUE. "
          PAUSE
          OUTPUT @Plotter;"OA"
          ENTER @Plotter;Hurx,Hury,Pen_status
          IF Hurx<=Hllx OR Hury<=Hlly THEN
              DISP " BAD LIMITS....TRY AGAIN "
              BEEP 400,.3
              GOTO Test_p1_p2
              WAIT 1.8
          END IF
          DISP "Generating GRAPH on HP7475A plotter."
      ELSE
          OUTPUT @Plotter;"OH"          !Send the HARD CLIP limits.
          ENTER @Plotter;Hllx,Hlly,Hurx,Hury
      END IF
      !
      SELECT Margins$(2)  ! SELECT QUADRANTS
      CASE "LOW LEFT"
          Hurx=Hurx-(Hurx-Hllx)/2
          Hury=Hury-(Hury-Hlly)/2
      CASE "UP RIGHT"
          Hllx=Hllx+(Hurx-Hllx)/2
          Hlly=Hlly+(Hury-Hlly)/2
      CASE "UP LEFT"
          Hlly=Hlly+(Hury-Hlly)/2
          Hurx=Hurx-(Hurx-Hllx)/2
      CASE "LOW RIGHT"
          Hllx=Hllx+(Hurx-Hllx)/2
          Hury=Hury-(Hury-Hlly)/2
      CASE ELSE
          ! NO CHANGE
      END SELECT
      Output$="IP"&VAL$(Hllx)&","&VAL$(Hlly)&","&VAL$(Hurx)&","&VAL$(Hury)
      OUTPUT @Plotter;Output$           !Set P1 & P2 to defined limits.
```

```
4643      !
4644      !RESET all graphics operations, read in HARD CLIP limits and
4645      !scale the result to the same dimensions as the screen, which
4646      !is (X longer) 0,100*RATIO,0,100  or  (Y longer) 0,100,0,100/RATIO.
4647      !Both VIEWPORT and WINDOW operations are performed.
4648      !
4649       GINIT
4650       PLOTTER IS Plot_addr,Plot_lang$
4651       PENUP
4652       OFF TIMEOUT 7
4653       RETURN
4654       !++++++++++++++++++++++++++++++++++
4655 Plotter_dead:      !The bus is hung.
4656       DISP "The plotter is NOT responding!"
4657       BEEP
4658       WAIT 1.8
4659       DISP CHR$(12)
4660       Suspended=1
4661       OFF TIMEOUT 7
4662       RETURN
4663      !
4664      !/////////////////////////////////////////////////////////////////////
4665      !
4666 User:!This special margins routine will handle the USER DEFN hard
4667      !clip limits.
4668      !The shorter of Hurx-Hllx and Hury-Hlly is scaled to 100 units
4669      !by the PLOTTER IS command.  The longer is scaled to 100*RATIO units.
4670      !
4671       Left_mar=0
4672       Bottom_mar=0
4673       IF Hurx-Hllx>Hury-Hlly THEN
4674           Right_mar=100*RATIO
4675           Top_mar=100
4676       ELSE
4677           Right_mar=100
4678           Top_mar=100/RATIO
4679       END IF
4680       RETURN
4681      !
4682      ! //////////////////////////////////////////////////////////////
4683      !
4684 Horizon_setup:          !Determine the GDUs/cm for the appropriate paper.
4685       IF Papersize$="3" THEN
4686               ! 11x17 inch paper.
4687           Xlength=100*RATIO
4688           Xgdu_cm=100*RATIO/41.45
4689           Ylength=100
4690           Ygdu_cm=100/25.85
4691       ELSE          ! 8.5x11 inch.
4692           Xlength=100*RATIO
4693           Xgdu_cm=100*RATIO/25.82
4694           Ylength=100
4695           Ygdu_cm=100/19.85
4696       END IF
4697       RETURN
4698    !
4699    !//////////////////////////////////////////////////////////////////////
4700    !
4701 H_boundtop:     !Set margins for top binding.
4702       IF Papersize$="3" THEN
4703                   ! 11x17 inch paper.
4704           Left_mar=Xgdu_cm*1.17
4705           Right_mar=Xlength-Xgdu_cm*2.25
4706           Bottom_mar=Ygdu_cm*1.65
4707           Top_mar=Ylength-Ygdu_cm*2.75
4708                   !
4709       ELSE
4710               ! 8.5x11 inch paper.
4711           Left_mar=Xgdu_cm*1.6
4712           Right_mar=Xlength-Xgdu_cm*1.4
4713           Bottom_mar=Ygdu_cm*2.25
4714           Top_mar=Ylength-Ygdu_cm*2.39
4715               !
4716       END IF
4717       RETURN
4718    !
4719    !//////////////////////////////////////////////////////////////////////
4720    !
4721 H_boundleft:   !Set margins for left side binding.
4722       IF Papersize$="3" THEN
4723               ! 11x17 inch paper.
4724           Left_mar=Xgdu_cm*2.4
4725           Right_mar=Xlength-Xgdu_cm*2.25
4726           Bottom_mar=Ygdu_cm*1.65
4727           Top_mar=Ylength-Ygdu_cm*1.45
4728               !
4729       ELSE
4730               ! 8.5x11 inch paper.
4731           Left_mar=Xgdu_cm*2.89
4732           Right_mar=Xlength-Xgdu_cm*1.4
4733           Bottom_mar=Ygdu_cm*2.25
4734           Top_mar=Ylength-Ygdu_cm*1.15
4735               !
4736       END IF
4737       RETURN
4738    !
4739    !//////////////////////////////////////////////////////////////////
4740    !
4741 H_full:    !Set margins to fullest dimensions.
4742       IF Papersize$="3" THEN
4743               ! 11x17 inch paper.
4744           Left_mar=Xgdu_cm*0.
4745           Right_mar=Xlength-Xgdu_cm*1.22
4746           Bottom_mar=Ygdu_cm*.15
4747           Top_mar=Ylength-Ygdu_cm*0.
4748               !
4749       ELSE
4750               ! 8.5x11 inch paper.
4751           Left_mar=Xgdu_cm*.25
4752           Right_mar=Xlength-Xgdu_cm*0.
4753           Bottom_mar=Ygdu_cm*1.12
4754           Top_mar=Ylength-Ygdu_cm*0.
4755               !
4756       END IF
4757       RETURN
4758   !
4759   !///////////////////////////////////////////////////////////////////
4760   !
4761 H_square:      !Set margins for square centered on paper.
4762       IF Papersize$="3" THEN
4763               ! 11x17 inch paper.
4764           Left_mar=Xgdu_cm*7.35
4765           Right_mar=Xlength-Xgdu_cm*8.45
4766           Bottom_mar=Ygdu_cm*.15
4767           Top_mar=Ylength-Ygdu_cm*0.
4768               !
4769       ELSE
4770               ! 8.5x11 inch paper.
4771           Left_mar=Xgdu_cm*3.65
4772           Right_mar=Xlength-Xgdu_cm*3.4
4773           Bottom_mar=Ygdu_cm*1.12
4774           Top_mar=Ylength-Ygdu_cm*0.
```

```
4775                    !
4776      END IF
4777      RETURN
4778     !
4779     !///////////////////////////////////////////////////////////////////
4780     !
4781 Vertical_setup:            !Setup the vertical dimensions uniformly.
4782      IF Papersize$="3" THEN
4783              ! 11x17 inch paper.
4784          Xlength=100
4785          Xgdu_cm=100/25.85
4786          Ylength=100/RATIO
4787          Ygdu_cm=100/(RATIO*41.45)
4788      ELSE         ! 8.5x17 inch.
4789          Xlength=100
4790          Xgdu_cm=100/19.85
4791          Ylength=100/RATIO
4792          Ygdu_cm=100/(RATIO*25.82)
4793      END IF
4794      RETURN
4795     !
4796     !///////////////////////////////////////////////////////////////////
4797     !
4798           !
4799 V_boundtop:        !Set margins for top binding.
4800      IF Papersize$="3" THEN
4801                  ! 11x17 inch paper.
4802          Left_mar=Xgdu_cm*1.60
4803          Right_mar=Xlength-Xgdu_cm*1.45
4804          Bottom_mar=Ygdu_cm*2.30
4805          Top_mar=Ylength-Ygdu_cm*2.40
4806                  !
4807      ELSE
4808                  ! 8.5x11 inch paper.
4809          Left_mar=Xgdu_cm*1.11
4810          Right_mar=Xlength-Xgdu_cm*2.24
4811          Bottom_mar=Ygdu_cm*1.60
4812          Top_mar=Ylength-Ygdu_cm*2.67
4813                  !
4814      END IF
4815      RETURN
4816     !
4817     !//////////////////////////////////////////////////////////////////////
4818     !
4819 V_boundleft:             !Setup margins for left binding.
4820      IF Papersize$="3" THEN
4821                  ! 11x17 inch paper.
4822          Left_mar=Xgdu_cm*2.38
4823          Right_mar=Xlength-Xgdu_cm*1.45
4824          Bottom_mar=Ygdu_cm*2.30
4825          Top_mar=Ylength-Ygdu_cm*1.12
4826                  !
4827      ELSE
4828                  ! 8.5x11 inch paper.
4829          Left_mar=Xgdu_cm*2.37
4830          Right_mar=Xlength-Xgdu_cm*2.24
4831          Bottom_mar=Ygdu_cm*1.60
4832          Top_mar=Ylength-Ygdu_cm*1.40
4833                  !
4834      END IF
4835      RETURN
4836     I
4837     !//////////////////////////////////////////////////////////////////////
4838     I
4839 V_full:            !Set margins for fullest dimensions.
4840      IF Papersize$="3" THEN
4841                  ! 11x17 inch paper.
4842          Left_mar=Xgdu_cm*.15
4843          Right_mar=Xlength-Xgdu_cm*0.
4844          Bottom_mar=Ygdu_cm*1.15
4845          Top_mar=Ylength-Ygdu_cm*0.
4846                  !
4847      ELSE
4848                  ! 8.5x11 inch paper.
4849          Left_mar=Xgdu_cm*0.
4850          Right_mar=Xlength-Xgdu_cm*1.07
4851          Bottom_mar=Ygdu_cm*.20
4852          Top_mar=Ylength-Ygdu_cm*0.
4853                  !
4854      END IF
4855      RETURN
4856    !
4857    !///////////////////////////////////////////////////////////////////
4858    !
4859 V_square:          !Set margins for square centered on paper.
4860      IF Papersize$="3" THEN
4861                  ! 11x17 inch paper.
4862          Left_mar=Xgdu_cm*.15
4863          Right_mar=Xlength-Xgdu_cm*0.
4864          Bottom_mar=Ygdu_cm*8.6
4865          Top_mar=Ylength-Ygdu_cm*7.25
4866                  !
4867      ELSE
4868                  ! 8.5x11 inch paper.
4869          Left_mar=Xgdu_cm*0.
4870          Right_mar=Xlength-Xgdu_cm*1.07
4871          Bottom_mar=Ygdu_cm*3.67
4872          Top_mar=Ylength-Ygdu_cm*3.39
4873                  !
4874      END IF
4875      RETURN
4876      SUBEXIT
4877   SUBEND
4878   !
4879   ! *************************************************************
4880   !
4881   SUB Wipe_clean
4882 Wipe_clean:  ! Original: 13 Nov 1984
4883              ! Revision: 06 Aug 1987
4884              ! Clear the CRT and home the cursor.
4885      PRINTER IS CRT                !Interact with CRT
4886      CONTROL KBD,1;0                 !PRT ALL off.
4887      CONTROL CRT,4;0                 !DISPLAY FCTNS off.
4888      OUTPUT 2 USING "#,4A";" K T"    !Clear the display, home the cursor
4889      GINIT
4890      GCLEAR
4891      ALPHA ON
4892      GRAPHICS OFF
4893      SUBEXIT
4894   SUBEND
4895  !
4896  !*********************************************************************
4897  !
4898   SUB Autoscale(File(*),Ax$)
4899 Autoscale: !
4900   !
4901   ! Original: 01 Oct 1985, Eric Vanzura
4902   ! Revision: 13 Aug 1987, 11:00    by G. Koepke
4903   !
4904   ! This subprogram finds the max and min of X and/or Y axis data
4905   ! then selects the graphtype with it's starting and stopping points
4906   ! File(*) contains packed data of one or more curves.
```

```
! Ax$ = "X" if only autoscale X axis.
! Ax$ = "Y" if only autoscale Y axis.
! Ax$ = "XY" for both.
! IF Ax$ is not "XY" then Graphtype$ must be set to a valid choice.
! IF Ax$ is "NO" then autoscaling is disabled
!
! No SUB Programs are CALLed
    !
    ! Roster(i,1) = Curve number 1,2,3,...,17
    ! Roster(i,2) = Start address in File(x,*); = x
    ! Roster(i,3) = Datacount for curve i
    ! Roster(i,4) = PEN number for this curve
    !
    ! Symbol$(i)="" or "Y" => no symbol, connect pts
    ! Symbol$(i)="*Y" =>  * symbol, connect pts
    ! Symbol$(i)="*N" =>  * symbol, do not connect pts
    ! Lbl_addr: x, y, pen, size, LDIR, LORG
    !
    COM /Labels/ Labels$(*),INTEGER Lbl_count,REAL Lbl_addr(*)
    COM /Data_param/ INTEGER Datacount,Filesize,Curvecount,Roster(*)
    COM /Data_param/ REAL Sym_size,Symbol$(*),Curve_id$(*)
    COM /Data_param/ REAL Xmin_data,Xmax_data
    COM /Data_param/ REAL Ymin_data,Ymax_data
    COM /Background/ Graphtype$,Margins$(*),Papersize$
    COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
    COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
    COM /Background/ Xgrid_tick$,INTEGER Xmajor,Xminor
    COM /Background/ Ygrid_tick$,INTEGER Ymajor,Yminor
    COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
    !
    COM /Axes_labels/ Print_xlabels$,Print_ylabels$
    COM /Axes_labels/ Sig_digits$,REAL Xlcsize,Ylcsize
    !
    COM /Windowspace/ REAL Xmin,Xmid,Xmax,Ymin,Ymid,Ymax!graph edges UDU
    COM /Windowspace/ REAL Xleft,Xright,Ybottom,Ytop! paper edges UDUs
    COM /Plot_device/ Plot_lang$,INTEGER Plot_addr
    COM /Hard_space/ REAL Xview_lft,Xview_rt,Yview_btm,Yview_top
    COM /Hard_space/ REAL Viewscale,Aspect_ratio
    COM /Frame_onoff/ INTEGER Frame_flag
    COM /Clear_space/ REAL Space_lft,Space_rt,Space_btm,Space_top
    COM /Bugs/ INTEGER Bug1,Bug2,Bug3,Printer
    !
    REAL Max,Min          !dummy variables used for both axes
    REAL Range,Factor,Base
    INTEGER Major,Minor
    INTEGER I,J,Axis      ! Axis =1 for X and 2 for Y.
    INTEGER Total_data_pts,Number_below
    DIM Xscale$[6],Scale$[6]
    !
    IF Curvecount<1 OR Ax$="NO" THEN
        GRAPHICS OFF
        BEEP
        IF Ax$="NO" THEN
            DISP " Auto scale is disabled ... see BACKGROUND editor."
        ELSE
            DISP " NO DATA AVAILABLE FOR AUTO SCALING "
        END IF
        WAIT 2.0
        DISP CHR$(12)
        GRAPHICS ON
        SUBEXIT
    END IF
    IF Ax$<>"XY" THEN
        SELECT Graphtype$
        CASE "LINEAR"
            Xscale$="LINEAR"
            Scale$="LINEAR"
        CASE "SEMILOG X"
            Xscale$="LOG"
            Scale$="LINEAR"
        CASE "SEMILOG Y"
            Xscale$="LINEAR"
            Scale$="LOG"
        CASE "LOG LOG"
            Xscale$="LOG"
            Scale$="LOG"
        CASE ELSE
            DISP "Graph type improperly chosen! ";
            DISP " will be set to LINEAR ...continue"
            BEEP
            PAUSE
            DISP CHR$(12)
            Xscale$="LINEAR"
            Scale$="LINEAR"
        END SELECT
    END IF
    !
Selective_scale:!
    IF Ax$="X" OR Ax$="Y" OR Ax$="XY" THEN
        IF Ax$="X" OR Ax$="XY" THEN
            Axis=1          ! X-axis
            GOSUB Find_max_min
            Xmin_data=Min
            Xmax_data=Max
            GOSUB Choose_scale! Either log or linear parameters
            Xscale$=Scale$
            GOSUB Set_x_initial
        END IF
        IF Ax$="Y" OR Ax$="XY" THEN
            Axis=2          ! Y-axis
            GOSUB Find_max_min
            Ymin_data=Min
            Ymax_data=Max
            GOSUB Choose_scale! Either log or linear parameters -> Scale$
            GOSUB Set_y_initial
        END IF
    ELSE
        DISP " ERROR is setting axis selector for autoscale. ";
        DISP " will be set to XY ... continue "
        BEEP
        PAUSE
        DISP CHR$(12)
        Ax$="XY"
        GOTO Selective_scale
    END IF
    !
    ! Choose Graphtype$
    !
    SELECT Xscale$
    CASE "LOG"
        SELECT Scale$
        CASE "LOG"
            Graphtype$="LOG LOG"
        CASE "LINEAR"
            Graphtype$="SEMILOG X"
        CASE ELSE
            DISP "Auto select graph type error."
            BEEP
            PAUSE
        END SELECT
    CASE "LINEAR"
        SELECT Scale$
```

```
5039          CASE "LOG"
5040              Graphtype$="SEMILOG Y"
5041          CASE "LINEAR"
5042              Graphtype$="LINEAR"
5043          CASE ELSE
5044              DISP "Auto select graph type error."
5045              BEEP
5046              PAUSE
5047          END SELECT
5048      CASE ELSE
5049          DISP "Auto select graph type error."
5050          BEEP
5051          PAUSE
5052      END SELECT
5053      SUBEXIT
5054      !
5055      ! /////////////////////////////////////////////////////////////
5056      !
5057 Find_max_min: !
5058      Min=File(Roster(1,2),Axis)      ! 1st curve,1st data point
5059      Max=File(Roster(1,2),Axis)
5060      FOR I=1 TO Curvecount
5061          FOR J=Roster(I,2) TO Roster(I,2)+Roster(I,3)-1
5062              Max=MAX(Max,File(J,Axis))
5063              Min=MIN(Min,File(J,Axis))
5064          NEXT J
5065      NEXT I
5066      IF Max<=Min THEN Max=Min
5067      IF ABS(Max-Min)<1.0E-50 THEN
5068          IF ABS(Max)>1.0E-50 THEN
5069              Max=Max+.1*ABS(Max)
5070          ELSE
5071              Max=.1
5072          END IF
5073          IF ABS(Min)>1.0E-50 THEN
5074              Min=Min-.1*ABS(Min)
5075          ELSE
5076              Min=-.1
5077          END IF
5078      END IF
5079      RETURN
5080      !
5081      ! /////////////////////////////////////////////////////////////
5082      !
5083 Choose_scale: !
5084      Log_cutoff=(Max-Min)*.4+Min
5085      Number_below=0
5086      Total_data_pts=0
5087      FOR I=1 TO Curvecount
5088          Total_data_pts=Total_data_pts+Roster(I,3) !'Datacount'
5089          FOR J=Roster(I,2) TO Roster(I,2)+Roster(I,3)-1
5090              IF File(J,Axis)<Log_cutoff THEN
5091                  Number_below=Number_below+1
5092              ELSE
5093                  GOTO Next_crv
5094              END IF
5095          NEXT J
5096 Next_crv: !
5097      NEXT I
5098      IF Number_below/Total_data_pts<.5 OR Min<1.E-25 THEN
5099          Scale$="LINEAR"
5100      ELSE
5101          Scale$="LOG"
5102          RETURN
5103      END IF
5104      !   Find parameters for a linear scale (ideas by Wilber Anson)
5105      Range=Max-Min
5106      IF Range>1.0E-50 THEN
5107          Base=10^INT(LGT(Range))
5108          SELECT Range
5109          CASE <=2*Base
5110              Factor=Base/5
5111          CASE <=5*Base
5112              Factor=Base/2
5113          CASE <=10*Base
5114              Factor=Base
5115          END SELECT
5116      ELSE
5117          Factor=1
5118      END IF
5119      Min=Factor*INT(Min/Factor)
5120      Max=Factor*(INT(Max/Factor)+(Max/Factor<>INT(Max/Factor)))
5121      Major=INT((Max-Min)/Factor)
5122      Major=Major+(INT((Max-Min)/Factor)<>DROUND((Max-Min)/Factor,4))
5123      IF (INT(Major/2)=Major/2) THEN
5124          Major=Major/2
5125          Minor=2
5126      ELSE
5127          Minor=1
5128      END IF
5129      RETURN
5130      !
5131      ! /////////////////////////////////////////////////////////////
5132      !
5133 Set_x_initial: !
5134      Xmajor=Major
5135      Xminor=Minor
5136      Xmin_graph=Min
5137      Xmax_graph=Max
5138      IF Bug2 THEN
5139          PRINTER IS Printer
5140          PRINT "XMIN_GRAPH=";Xmin_graph,"XMAX_GRAPH=";Xmax_graph
5141          PRINT
5142          PRINTER IS CRT
5143      END IF
5144      Y_cross_x=Xmin_graph
5145      RETURN
5146      !
5147      ! /////////////////////////////////////////////////////////////
5148      !
5149 Set_y_initial: !
5150      Ymajor=Major
5151      Yminor=Minor
5152      Ymin_graph=Min
5153      Ymax_graph=Max
5154      IF Bug2 THEN
5155          PRINTER IS Printer
5156          PRINT "YMIN_GRAPH=";Ymin_graph,"YMAX_GRAPH=";Ymax_graph
5157          PRINT
5158          PRINTER IS CRT
5159      END IF
5160      X_cross_y=Ymin_graph
5161      RETURN
5162   SUBEND
5163   !
5164   ! ******************************************************
5165   !
5166   SUB Refresh_labels
5167      ! Original: 18 Aug 1987, G. Koepke (drawn from Graph Data)
5168      ! Revision: 18 Aug 1987
5169      ! Use this routine after Autoscale and before adding labels
5170      ! to maintain proper location on graph.
```

```
      !
      OPTION BASE 1
      DEG
      COM /Sys/ Sys_id$[10]
      COM /Labels/ Labels$(*),INTEGER Lbl_count,REAL Lbl_addr(*)
      COM /Data_param/ INTEGER Datacount,Filesize,Curvecount,Roster(*)
      COM /Data_param/ Real_sym_size,Symbol$(*),Curve_id$(*)
      COM /Data_param/ REAL Xmin_data,Xmax_data
      COM /Data_param/ REAL Ymin_data,Ymax_data
      COM /Background/ Graphtype$,Margins$(*),Papersize$
      COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
      COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
      COM /Background/ Xgrid_tick$,INTEGER Xmajor,Xminor
      COM /Background/ Ygrid_tick$,INTEGER Ymajor,Yminor
      COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
      !
      COM /Windowspace/ REAL Xmin,Xmid,Xmax,Ymin,Ymid,Ymax!graph edges UDU
      COM /Windowspace/ REAL Xleft,Xright,Ybottom,Ytop! paper edges UDUs
      IF Lbl_count>0 THEN
          ALLOCATE Lbl_ratio(Lbl_count,2)
          GOSUB Update_lbl_addr     ! includes Find_mid_pt
          DEALLOCATE Lbl_ratio(*)
      ELSE
          GOSUB Find_mid_pt
      END IF
      SUBEXIT
      !
      ! ///////////////////////////////////////////////////////////
      !
Update_lbl_addr:! Make sure that the label addresses are current.
      FOR I=1 TO Lbl_count
          Lbl_ratio(I,1)=(Lbl_addr(I,1)-Xmin)/(Xmax-Xmin)
          Lbl_ratio(I,2)=(Lbl_addr(I,2)-Ymin)/(Ymax-Ymin)
      NEXT I
      GOSUB Find_mid_pt              ! define new max and min
      FOR I=1 TO Lbl_count
          Lbl_addr(I,1)=Lbl_ratio(I,1)*(Xmax-Xmin)+Xmin
          Lbl_addr(I,2)=Lbl_ratio(I,2)*(Ymax-Ymin)+Ymin
      NEXT I
      RETURN
      !
      ! ///////////////////////////////////////////////////////////
      !
Find_mid_pt:      !Interpret the graph type and the scaling done.
      SELECT Graphtype$
      CASE "LINEAR"
          Xmin=Xmin_graph
          Xmax=Xmax_graph
          Xmid=((Xmax-Xmin)/2)+Xmin
          Ymin=Ymin_graph
          Ymax=Ymax_graph
          Ymid=((Ymax-Ymin)/2)+Ymin
      CASE "SEMILOG X"
          Xmin=0
          Xmax=100
          Xmid=50
          Ymin=Ymin_graph
          Ymax=Ymax_graph
          Ymid=((Ymax-Ymin)/2)+Ymin
      CASE "SEMILOG Y"
          Xmin=Xmin_graph
          Xmax=Xmax_graph
          Xmid=((Xmax-Xmin)/2)+Xmin
          Ymin=0
          Ymax=100
          Ymid=50
      CASE "LOG LOG"
          Xmin=0
          Xmax=100
          Xmid=50
          Ymin=0
          Ymax=100
          Ymid=50
      CASE "POLAR"
          !
          BEEP
          DISP "POLAR PARAMETERS ARE NOT YET IMPLEMENTED!!!!"
          PAUSE
      END SELECT
      RETURN
      !
      ! ///////////////////////////////////////////////////////////
      !
  SUBEND
!
! ************************************************************
!
  SUB Optimizedisplay
 !This subroutine makes the network analyzer display look nice.
      OPTION BASE 1
      COM /Nwa_data/ Start_freq,Stop_freq,Marker(5,3)
Optimizedisplay: !
      Set_nwa("MARK=4","MARK=MIN")
      Read_nwa("MARK=4")
      Min=Marker(4,2)
      Set_nwa("MARK=5","MARK=MAX")
      Read_nwa("MARK=5")
      Max=Marker(5,2)
      GOSUB Select_scale
      Set_nwa("SCALE="&VAL$(Scale))
      GOSUB Select_ref_val
      Set_nwa("REF POS=10","REF VAL="&VAL$(Ref_val))
      SUBEXIT
!
 ! ////////////////////////////////////////////////////////////
!
Select_scale: !
      SELECT Max-Min
      CASE <.5
          Scale=.05
      CASE <1
          Scale=.1
      CASE <2
          Scale=.2
      CASE <5
          Scale=.5
      CASE <10
          Scale=1
      CASE <20
          Scale=2
      CASE <50
          Scale=5
      CASE <100
          Scale=10
      CASE ELSE
          Scale=10
          BEEP
          PRINT "Proper scale was not found."
      END SELECT
      RETURN
 ! ////////////////////////////////////////////////////////////
Select_ref_val: !
```

```
5303      Ref_val=PROUND(((Max+Min)/2)+(Scale*5),-2)
5304      RETURN
5305   SUBEND
5306 !
5307  ! ****************************************************************
5308 !
5309   CSUB Mrqmin(X(*),Y(*),Sig(*),INTEGER Ndata(*),REAL A(*),INTEGER Ma(*),Lis
  => ta(*),Mfit(*),REAL Covar(*),Alpha(*),INTEGER Nca(*),REAL Chisq,Alamda,Sss,D
  => ebug)
5310   SUB Load_sparms
5311      OPTION BASE 1
5312      COM /History/ Status$[1],Time_orgn$[8],Date_orgn$[11]
5313      COM /History/ Time_chng$[8],Date_chng$[11],Description$[160]
5314      COM /Files/ Diskdrive$[20],Filename$[10]
5315      COM /Ana_data/ INTEGER Prog_id,Sweep_type,Sweep_mode,Datacount
5316      COM /Ana_data/ INTEGER S11_valid,S21_valid,S12_valid,S22_valid,Ports
5317      COM /Ana_data/ REAL Z0,Start,Stop
5318      COM /Ana_data/ Title$[55],Company$[30],Operator_name$[30],Measure_tim
  => e$[30]
5319      COM /S_array/ INTEGER Dcount,Svalid,REAL S(801,5,2)
5320 ! Sweep_type: 1=start/stop , 2=center/span
5321 ! Sweep_mode: 1=ramp , 2=step , 3=frequency list
5322      DIM Prompt$[30]
5323      INTEGER I,Num_sparms
5324 Load_sparms:!
5325      Prompt$="8510 S-PARAMETER DATA"
5326 !    CALL Select_disk(Prompt$)
5327      CALL Enterfilename("CAT",Prompt$)
5328      IF Filename$="" THEN
5329         CAT Diskdrive$
5330         CALL Enterfilename("ABORT",Prompt$)
5331         IF Filename$="" THEN
5332            BEEP
5333            DISP "PROGRAM ABORTED"
5334            STOP
5335         END IF
5336      END IF
5337      DISP CHR$(129);" Reading S-parameter data: ";CHR$(128);Filename$;Disk
  => drive$
5338      ON ERROR CALL Errortrap
5339      ASSIGN @Infile TO Filename$&Diskdrive$;FORMAT OFF
5340      OFF ERROR
5341      ON ERROR GOSUB Read_sparm_err
5342      ENTER @Infile;Prog_id
5343      ENTER @Infile;Title$
5344      ENTER @Infile;Company$
5345      ENTER @Infile;User_name$
5346      ENTER @Infile;Measure_time$
5347      ENTER @Infile;Z0
5348      ENTER @Infile;Sweep_type
5349      ENTER @Infile;Sweep_mode
5350      ENTER @Infile;Datacount,Start,Stop,S11_valid,S21_valid,S12_valid,S22_
  => valid,Ports
5351      Dcount=Datacount
5352      Num_sparms=S11_valid+S21_valid+S12_valid+S22_valid
5353      Svalid=0
5354      IF S11_valid THEN Svalid=Svalid+1
5355      IF S21_valid THEN Svalid=Svalid+2
5356      IF S12_valid THEN Svalid=Svalid+4
5357      IF S22_valid THEN Svalid=Svalid+8
5358      ALLOCATE REAL Local_array(Datacount,Num_sparms+1,2)
5359      ENTER @Infile;Local_array(*)
5360      OFF ERROR
5361      ASSIGN @Infile TO *
5362      Description$=Title$
5363      Time_orgn$=Measure_time$[15,22]
5364      Date_orgn$=Measure_time$[1,11]
5365      MAT S(1:Dcount,5,1)= Local_array(1:Dcount,5,1)
5366      FOR I=1 TO 4
5367         IF BIT(Svalid,I-1) THEN MAT S(1:Dcount,I,*)= Local_array(*,I,*)
5368      NEXT I
5369      DEALLOCATE Local_array(*)
5370      SUBEXIT !Load_sparms
5371!
5372! ////////////////////////////////////////////////////////////
5373!
5374 Read_sparm_err:!
5375      BEEP
5376      PRINT "Error number:";ERRN;ERRM$
5377      DISP "something is wrong with S-parameter read"
5378      PAUSE
5379      RETURN
5380!
5381! ////////////////////////////////////////////////////////////
5382      GCLEAR
5383   SUBEND   !Load_sparms
5384 !
5385 ! ****************************************************************
5386 !
5387   SUB Select_disk(OPTIONAL Prompt$)
5388 Select_disk: ! Original: 04 Jul 1987
5389              ! Revision: 06 Aug 1987
5390      COM /Files/ Diskdrive$[20],Filename$[10]
5391      INTEGER Local_prty,Dd,Pt,Choose(1)
5392      DIM Disc$(30)[60],Title$[40],Displ$[80]
5393      Local_prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
5394      OFF KEY
5395      !
5396      ! Define the disk drives available for this system, reserve the
5397      ! first characters for the drive address and the characters after
5398      ! the - for a description of the drive.
5399      !
5400      ! Example:
5401      ! Disc$(1)=":,700,0,0      HP 9133H HARD disk, volume 0."
5402      !
5403      !
5404      IF NPAR THEN
5405         Displ$=" SELECT DISK DRIVE "&Prompt$&" ... Abort will cancel. "
5406         Displ$=Displ$[1,80]
5407      ELSE
5408         Displ$=" SELECT DISK DRIVE  ... Abort will cancel. "
5409      END IF
5410      Title$=" Available disk drives for this system. "
5411      Pt=1   ! allow only one select
5412      !
5413      IF Diskdrive$[1,1]<>":" THEN Diskdrive$=""
5414      IF Msi_id$[1,1]<>":" THEN Msi_id$=SYSTEM$("MSI")
5415      Diskdrive$=TRIM$(Diskdrive$)
5416      Msi_id$=TRIM$(Msi_id$)
5417      IF LEN(Diskdrive$)>0 AND LEN(Msi_id$)>0 THEN
5418         Disc$(1)=Diskdrive$&RPT$(" ",17-LEN(Diskdrive$))
5419         Disc$(1)=Disc$(1)&"- Last selected disk drive."
5420         Dd=1
5421         IF Diskdrive$<>Msi_id$ THEN
5422            Disc$(2)=Msi_id$&RPT$(" ",17-LEN(Msi_id$))
5423            Disc$(2)=Disc$(2)&"- Start-up mass storage unit specifier."
5424            Dd=Dd+1
5425         ELSE
5426            Disc$(1)=Disc$(1)&"  Start-up MSUS."
5427         END IF
5428      ELSE
5429         IF LEN(Msi_id$)>0 THEN
```

```
5430               Disc$(1)=Msi_id$&RPT$(" ",17-LEN(Msi_id$))
5431               Disc$(1)=Disc$(1)&"- Start-up mass storage unit specifier."
5432               Dd=1
5433           ELSE
5434               Dd=0
5435           END IF
5436       END IF
5437 Disk: !
5438       ! ................ customize system drives here ..................
5439       ! Follow format with - after unit specifier, description is
5440       ! optional but recommended.
5441       ! .............................................................
5442       !
5443       Disc$(Dd+1)=":,707,0          - HP 9122 dual microfloppy left drive"
5444       Disc$(Dd+2)=":,707,1          - HP 9122 dual microfloppy right drive"
5445       Disc$(Dd+3)=":,1404,0         - HP 9127 single 5.25 floppy drive"
5446       Disc$(Dd+4)=":,1400,1         - HP 9153B single microfloppy"
5447       Disc$(Dd+5)=":,1400,0,2       - HP 9153H hard disk ANA volume"
5448       Disc$(Dd+6)=":,1400,0,3       - HP 9153B hard disk PROG volume"
5449       !
5450       Dd=Dd+6    ! add the number of drive specifiers above
5451       !
5452       IF Sys_id$[1,4]<>"S300" THEN
5453           Disc$(Dd+1)=":,4,1              - LEFT internal series 200"
5454           Disc$(Dd+2)=":,4,0              - RIGHT internal series 200"
5455           Dd=Dd+2
5456       END IF
5457       ! .............................................................
5458       !
5459       CALL Menu_scroll(Displ$,Title$,Disc$(*),Dd,Pt,Choose(*))
5460       IF Pt=0 THEN
5461           Diskdrive$="NO DISK"
5462       ELSE
5463           Dd=POS(Disc$(Choose(Pt)),"-")-1 ! find -
5464           IF Dd>5 THEN  !  valid msus
5465               Diskdrive$=TRIM$(Disc$(Choose(Pt))[1,Dd])
5466           ELSE
5467               DISP " ERROR in reading MSUS from string, - chr not found. "
5468               BEEP
5469               PAUSE
5470               Diskdrive$="NO DISK"
5471           END IF
5472       END IF
5473 Diskselected:OFF KEY
5474       SUBEXIT
5475   SUBEND
5476   !
5477   ! *******************************************************
5478   !
5479   SUB Enterfilename(Ac$,OPTIONAL Prompt$)
5480 Enterfilename:      !
5481       COM /Files/ Diskdrive$,Filename$
5482       INTEGER I,Ascii_num
5483       DIM Test$[160]
5484       SELECT NPAR
5485       CASE 1
5486           DISP " ENTER the FILE NAME ";
5487       CASE 2
5488           DISP " ENTER the FILE NAME for ";Prompt$;
5489       END SELECT
5490       SELECT Ac$
5491       CASE "CAT"
5492           DISP " ... (ENTER alone to CAT) ";
5493       CASE "ABORT"
5494           DISP " ... (ENTER alone to ABORT) ";
5495       CASE "VALID"
5496       END SELECT
5497       LINPUT Test$
5498       Test$=TRIM$(Test$)
5499       IF LEN(Test$)=0 THEN
5500           SELECT Ac$
5501           CASE "VALID"
5502               DISP "You MUST enter the FILE NAME now."
5503               BEEP
5504               WAIT 1.8
5505               GOTO Enterfilename
5506           CASE "ABORT","CAT"
5507               GOTO Abortline
5508           CASE ELSE
5509               DISP "Ac$=";Ac$;" in SUB Enterfilename"
5510               BEEP
5511               PAUSE
5512           END SELECT
5513       END IF
5514       IF LEN(Test$)>10 THEN
5515           BEEP
5516           DISP "ERROR in NAME ENTRY--up to 10 chars, you have ";
5517           DISP LEN(Test$);" "
5518           WAIT 1.8
5519           OUTPUT 2 USING "#,K,K";" #";Test$
5520           GOTO Enterfilename
5521       END IF
5522       Filename$=Test$
5523       FOR I=1 TO LEN(Filename$)
5524           Ascii_num=NUM(Filename$[I])
5525           SELECT Ascii_num
5526           CASE 65 TO 90,95,97 TO 122,48 TO 57
5527               !Allowed characters
5528           CASE ELSE
5529               BEEP
5530               DISP "ERROR in NAME ENTRY--ILLEGAL CHARACTERS, TRY AGAIN."
5531               WAIT 1.8
5532               OUTPUT 2 USING "#,K,K";" #";Filename$
5533               GOTO Enterfilename
5534           END SELECT
5535       NEXT I
5536       SUBEXIT
5537 Abortline:Filename$=""
5538       SUBEXIT
5539   SUBEND
5540   !
5541   ! ***************************************************************
5542   !
5543   SUB Errortrap
5544 Errortrap:  ! Original: 13 Nov 1984
5545             ! Revision: 02 Dec 1987
5546       ! Trap most errors here
5547       COM /Files/ Diskdrive$[20],Filename$[10]
5548       DIM File$[20],Test$[160],What$[20],Ac$[5]
5549       SELECT ERRN
5550       CASE 54
5551           DISP "DUPLICATE FILE NAME:  ";Filename$;
5552           DISP "....PURGE old one? (Y/N)";
5553           LINPUT What$
5554           What$=TRIM$(What$)
5555           SELECT What$[1,1]
5556           CASE "Y","y"
5557               PURGE Filename$&Diskdrive$
5558           CASE ELSE
5559               Ac$="VALID"
5560               CALL Enterfilename(Ac$)
5561           END SELECT
```

```
5562        CASE 52,53
5563            DISP "Improper FILE NAME --- ENTER NEW FILE NAME";
5564            OUTPUT 2 USING "#,K,K";" #";Filename$
5565            LINPUT Filename$
5566            Filename$=TRIM$(Filename$)
5567        CASE 56
5568            DISP "FILE:  ";Filename$;" is not on this disk, please insert";
5569            DISP " correct disk"
5570            CALL Pause_key_on
5571        CASE 64
5572            DISP "This disk is full, PLEASE insert clean disk"
5573            CALL Pause_key_on
5574        CASE 56
5575            DISP "DATA INPUT disk must be in drive!!  ";
5576            DISP "...CONTINUE when ready."
5577            CALL Pause_key_on
5578        CASE 72,73,76
5579            DISP Diskdrive$;
5580            DISP " is not available, type correct";
5581            DISP " unit specifier (ie. ':,707,0').";
5582            OUTPUT 2 USING "K,#";Diskdrive$
5583            LINPUT Diskdrive$
5584        CASE 80
5585            DISP "CHECK DISK drive door!"
5586            CALL Pause_key_on
5587        CASE ELSE
5588            DISP ERRM$;" 'CONTINUE' when fixed"
5589            CALL Pause_key_on
5590        END SELECT
5591        DISP CHR$(12)
5592        SUBEXIT
5593    SUBEND
5594    !
5595    ! *****************************************************************
5596    !
5597    SUB Menu_scroll(D$,T$,Items$(*),INTEGER Item_cnt,To_select,Choose(*))
5598 Menu_scroll: ! Original: 22 Jun 1987, Galen Koepke, NBS 723.04
5599            ! Revision: 30 Jun 1987, 13:35
5600            !
5601            ! A general purpose menu utility for scrolling items and
5602            ! selecting a given number of them.
5603            ! The items are arranged in screens of 15 items each and
5604            ! the user may access screens via softkeys. There may be
5605            ! up to 10 screens or 150 items to choose from.
5606            ! Items$(*) contains the item descriptions
5607            ! Item_cnt is the number of items in Items$(*)
5608            ! Choose(*) is dimensioned to the number of required choices
5609            !          and will be filled with the item numbers chosen.
5610            ! To_select is the number of required choices.
5611            !
5612        OPTION BASE 1
5613        PRINTER IS CRT
5614        DEG
5615        GOSUB Def_variables
5616        GOSUB Define_screens
5617        GOSUB Make_selections
5618        IF Null_file THEN  ! reset to zero
5619            Item_cnt=0
5620            Items$(1)=""
5621            To_select=0    ! no valid selections
5622        END IF
5623        SUBEXIT
5624        I
5625        I ////////////////////////////////////////////////////
5626        I
5627 Def_variables:!
5628        !
5629        INTEGER Screen_cnt,Items_per_scn,First_item(10),Last_item(10)
5630        INTEGER I,J,K,First_line,Last_line,Active_screen,Pointer,Last_pt
5631        INTEGER Local_prty,Skips,Knobcount,Pointeractive,K0,Null_file
5632        INTEGER Exit_flag
5633        DIM Marker$[8],Test$[160]
5634        !
5635        ! initialize parameters
5636        !
5637        Local_prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
5638        IF Local_prty<1 THEN Local_prty=10
5639        IF LEN(Sys_id$)=0 THEN Sys_id$=SYSTEM$("SYSTEM ID")
5640        IF Item_cnt<1 THEN
5641            Null_file=1
5642            Item_cnt=1
5643            To_select=0
5644            Items$(1)="*** Empty ***"
5645        ELSE
5646            Null_file=0
5647        END IF
5648        IF To_select>Item_cnt THEN To_select=Item_cnt
5649        Skips=0
5650        Knobcount=0
5651        Doneflag=0
5652        Marker$="===>"&RPT$(CHR$(8),4)
5653        RETURN
5654        !
5655        ! ////////////////////////////////////////////////////
5656        !
5657 Define_screens:! Set up screens of 15 items each.
5658                !
5659        Items_per_scn=15    ! Maximum number of displayable items
5660        IF INT(Item_cnt/Items_per_scn)=Item_cnt/Items_per_scn THEN
5661            Screen_cnt=INT(Item_cnt/Items_per_scn)
5662        ELSE
5663            Screen_cnt=INT(Item_cnt/Items_per_scn)+1
5664        END IF
5665        J=1
5666        FOR I=1 TO Screen_cnt   ! set up each screen
5667            First_item(I)=J
5668            IF J+Items_per_scn-1<Item_cnt THEN
5669                Last_item(I)=J+Items_per_scn-1
5670                J=J+Items_per_scn
5671            ELSE
5672                Last_item(I)=Item_cnt
5673            END IF
5674        NEXT I
5675        RETURN
5676        !
5677        ! ////////////////////////////////////////////////////
5678        !
5679 Make_selections:! MENU setup and use.
5680        Active_screen=1   ! first screen is active
5681        First_line=2      ! first printed line on screen = 2 or greater.
5682        GOSUB Write_screen  ! activate screen at Active_screen
5683                            ! and set First_line and Last_line for Pointer
5684                            ! write Marker$ to first non-selected line.
5685        K0=0                ! Keys start at zero
5686        Exit_flag=0         ! allow ENTER key to exit when selections filled.
5687 Key_loop: !
5688        DISP D$
5689        ON KBD,Local_prty GOSUB Process_kbd
5690        ON KNOB .01,Local_prty GOSUB Move_pointer
5691        IF Skips<To_select THEN
5692            IF To_select>1 THEN
5693                Test$=" Select  "&VAL$(Skips+1)&" of "&VAL$(To_select)
```

```
5694          ELSE
5695              Test$=" Select"
5696          END IF
5697          ON KEY K0 LABEL Test$,Local_prty GOSUB Select_item
5698      ELSE
5699          ON KEY K0 LABEL " Exit",Local_prty GOTO Exit_line
5700      END IF
5701      IF Active_screen<Screen_cnt THEN
5702          ON KEY K0+1 LABEL " Next   Screen",Local_prty GOSUB Next_screen
5703      ELSE
5704          OFF KEY K0+1
5705      END IF
5706      IF Active_screen>1 THEN
5707          ON KEY K0+2 LABEL " Last   Screen",Local_prty GOSUB Last_screen
5708      ELSE
5709          OFF KEY K0+2
5710      END IF
5711      IF Skips>0 THEN
5712          ON KEY K0+3 LABEL " Select  Reset",Local_prty GOSUB Select_reset
5713      ELSE
5714          OFF KEY K0+3
5715      END IF
5716      ON KEY K0+4 LABEL " ABORT ",Local_prty GOTO Escape_line
5717      IF Exit_flag THEN Exit_line
5718      GOTO Key_loop
5719 Escape_line:Skips=0
5720      MAT Choose= (0)
5721      To_select=0
5722 Exit_line:OFF KEY
5723      OFF KNOB
5724      OFF KBD
5725      OUTPUT KBD;CHR$(255)&CHR$(75);
5726      PRINT CHR$(128);
5727      ! everything cleared, now go back to work.
5728      RETURN
5729      !
5730      ! ////////////////////////////////////////////////////
5731      !
5732 Next_screen:      !
5733      OFF KBD
5734      OFF KNOB
5735      OFF KEY
5736      IF Active_screen=Screen_cnt THEN RETURN
5737      Active_screen=Active_screen+1
5738      GOSUB Write_screen
5739      RETURN
5740      !
5741      ! ////////////////////////////////////////////////////
5742      !
5743 Last_screen:      !
5744      OFF KBD
5745      OFF KNOB
5746      OFF KEY
5747      IF Active_screen=1 THEN RETURN
5748      Active_screen=Active_screen-1
5749      GOSUB Write_screen
5750      RETURN
5751      !
5752      ! ///////////////////////////////////////////////////
5753      !
5754 Select_item:!
5755      OFF KBD
5756      OFF KNOB
5757      OFF KEY
5758      IF NOT Pointeractive THEN
5759          DISP "NO additional selections for this screen."
5760          BEEP
5761          WAIT 2
5762          DISP CHR$(12);
5763          RETURN
5764      END IF
5765      IF Skips=To_select THEN
5766          IF To_select=0 THEN
5767              DISP "This menu is for information only,";
5768              DISP " no selection allowed."
5769          ELSE
5770              DISP "All selections have been filled,";
5771              DISP " 'Select Reset' to repeat."
5772          END IF
5773          BEEP
5774          WAIT 2
5775          DISP CHR$(12);
5776          RETURN
5777      END IF
5778      Skips=Skips+1
5779      Choose(Skips)=First_item(Active_screen)+Pointer-First_line
5780      PRINT CHR$(129);  ! inverse video
5781      PRINT TABXY(10,Pointer);Items$(Choose(Skips))
5782      PRINT CHR$(128);
5783      PRINT TABXY(1,Pointer);
5784      SELECT Pointer
5785      CASE First_line
5786          GOSUB Point_forward
5787      CASE Last_line
5788          GOSUB Point_backward
5789      CASE ELSE
5790          ! move forward unless it requires wrapping to beginning.
5791          IF Skips-1>0 THEN  ! check for selected items.
5792              I=Pointer-First_line
5793              LOOP
5794                  K=0
5795                  FOR J=1 TO Skips
5796                      IF First_item(Active_screen)+I=Choose(J) THEN K=1
5797                  NEXT J
5798              EXIT IF K=0
5799                  I=I+1
5800                  IF I+First_line>Last_line THEN K=-1
5801              EXIT IF K=-1
5802              END LOOP
5803              IF K=0 THEN
5804                  GOSUB Point_forward
5805              ELSE
5806                  GOSUB Point_backward
5807              END IF
5808          ELSE
5809              GOSUB Point_forward
5810          END IF
5811      END SELECT
5812      RETURN
5813      !
5814      ! ///////////////////////////////////////////////////
5815      !
5816 Select_reset:      !Clear Choose file
5817      OFF KBD
5818      OFF KNOB
5819      OFF KEY
5820      Skips=0
5821      MAT Choose= (0)
5822      GOSUB Write_screen
5823      RETURN
5824      !
5825      ! //////////////////////////////////////////////////
```

```
      !
Process_kbd:! Allow use of arrows and enter key in addition to soft.
      Test$=KBD$
      IF LEN(Test$)=1 AND Test$[1,1]<>CHR$(32) THEN
          BEEP 80.,.1
          RETURN
      END IF
      IF Test$[1,1]=CHR$(32) THEN GOSUB Point_forward
      IF Test$[1,1]<>CHR$(255) THEN RETURN
      SELECT Test$[2,2]
      CASE CHR$(255)
                 ! do nothing
      CASE "V","T"
          GOSUB Point_forward
      CASE "^","W"
          GOSUB Point_backward
      CASE "E"
          IF Skips<To_select THEN
              GOSUB Select_item
          ELSE
              ! exit routine
              Exit_flag=1
          END IF
      CASE ELSE
          BEEP 80.,.1
      END SELECT
      Test$=""
      RETURN
      !
      ! ///////////////////////////////////////////////////
      !
Point_forward:Knobcount=5
      GOSUB Move_pointer
      RETURN
Point_backward:Knobcount=-5
      GOSUB Move_pointer
      RETURN
      !
      ! ///////////////////////////////////////////////////
      !
Jog_pointer:! Move the selection pointer on the active screen.
            ! without regard to selected values
      IF Knobcount>0 THEN  ! Move forward
          Pointer=Pointer+1
      ELSE                 ! Move backward
          Pointer=Pointer-1
      END IF
      IF Pointer<First_line THEN Pointer=Last_line
      IF Pointer>Last_line THEN Pointer=First_line
      RETURN
      !
      ! /////////////////////////////////////////////////////////
      !
Move_pointer:! Control pointer to avoid re-selection of items
      IF NOT Pointeractive THEN RETURN  ! No selections to be made.
      Knobcount=Knobcount+KNOBX+KNOBY
      IF ABS(Knobcount)<4 THEN RETURN
      Last_pt=Pointer
      GOSUB Jog_pointer
      IF Skips>0 THEN
          LOOP
              J=Pointer-First_line
              FOR I=1 TO Skips
                  IF First_item(Active_screen)+J=Choose(I) THEN J=999
              NEXT I
              IF J=999 AND Pointer=Last_pt THEN Pointeractive=0
          EXIT IF Pointeractive=0
              IF J=999 THEN GOSUB Jog_pointer
          EXIT IF J<>999
          END LOOP
      END IF
      Knobcount=0
      OUTPUT KBD;CHR$(255)&CHR$(84);  ! Bring screen home
      IF Last_pt=Last_line THEN PRINT CHR$(132);
      PRINT "    ";
      IF Pointeractive THEN ! Pointer active
          IF Pointer=Last_line THEN
              PRINT CHR$(132);
          ELSE
              PRINT CHR$(128);
          END IF
          PRINT TABXY(1,Pointer);Marker$;CHR$(128);
      END IF
      RETURN
      !
      ! ///////////////////////////////////////////////////
      !
Write_screen:! Write the screen pointed to by Active_screen
      ! home and clear screen
      OUTPUT KBD;CHR$(255)&CHR$(84)&CHR$(255)&CHR$(75);
      !
      PRINT TABXY(1,First_line-1);CHR$(132);" Item #| Screen #";
      PRINT USING "#,2D,3A,2D,3A";Active_screen," of ";Screen_cnt;" | "
      PRINT T$;RPT$(" ",52-LEN(T$));CHR$(128);
      J=0
      REPEAT
          IF J=Last_item(Active_screen)-First_item(Active_screen) THEN
              PRINT CHR$(132);
              PRINT TABXY(1,First_line+J);RPT$(" ",80)
          ELSE
              PRINT CHR$(128);
          END IF
          PRINT TABXY(5,First_line+J);
          PRINT USING "3D,A,#";First_item(Active_screen)+J,"|"
          IF Skips>0 THEN  ! make this line inverse video
              FOR I=1 TO Skips
                  IF First_item(Active_screen)+J=Choose(I) THEN
                      PRINT CHR$(129);
                  END IF
              NEXT I
          END IF
          PRINT TABXY(10,First_line+J);Items$(First_item(Active_screen)+J)
          J=J+1
      UNTIL J>=(Last_item(Active_screen)-First_item(Active_screen)+1)
      Last_line=Last_item(Active_screen)-First_item(Active_screen)
      Last_line=Last_line+First_line
      !
      ! set marker to first non-selected item.
      !
      Pointeractive=0
      IF To_select>0 THEN Pointeractive=1
      IF Skips>0 AND Pointeractive=1 THEN ! find first non-selected item
          J=0
          LOOP
              Pointer=First_line+J
              FOR I=1 TO Skips
                  IF First_item(Active_screen)+J=Choose(I) THEN Pointer=0
              NEXT I
          EXIT IF Pointer<>0
              J=J+1
              IF First_line+J>Last_line THEN
                  Pointeractive=0
```

```
5958                 Pointer=First_line
5959             END IF
5960         EXIT IF Pointer<>0
5961         END LOOP
5962     ELSE
5963         Pointer=First_line
5964     END IF
5965     IF Pointeractive THEN
5966         IF Pointer=Last_line THEN
5967             PRINT CHR$(132);
5968         ELSE
5969             PRINT CHR$(128);
5970         END IF
5971         PRINT TABXY(1,Pointer);Marker$;CHR$(128);
5972     END IF
5973     RETURN
5974   SUBEND
5975   !
5976   ! ***********************************************************
5977   !
5978   SUB Edit_data(Prompt$,Variable,OPTIONAL Multiplier,Uvariable)
5979 Edit_data:OFF KEY
5980     IF NPAR>2 THEN
5981         Test=Variable*Multiplier
5982         IF NPAR=4 THEN Uvariable=Uvariable*Multiplier
5983     ELSE
5984         Test=Variable
5985     END IF
5986     ON ERROR GOTO Test_again
5987 Test_again:!
5988     OUTPUT 2 USING "K,#";Test
5989     DISP "Enter the value of ";Prompt$;
5990     INPUT Variable
5991     OFF ERROR
5992     IF NPAR=4 THEN
5993         Utest=Uvariable
5994         ON ERROR GOTO Utest_again
5995 Utest_again:!
5996         OUTPUT 2 USING "K,#";Utest
5997         DISP "Enter the uncertainty in ";Prompt$;
5998         INPUT Uvariable
5999         OFF ERROR
6000     END IF
6001     IF NPAR>2 THEN
6002         Variable=Variable/Multiplier
6003         IF NPAR=4 THEN Uvariable=Uvariable/Multiplier
6004     END IF
6005   SUBEND
6006  !
6007  ! ****************************************************************
6008  !
6009   SUB Phase_delay
6010 Phase_delay:  !
6011 Init_com:  !
6012     OPTION BASE 1
6013     COM /Lab_info/ REAL Rel_humidity,Temperature,Pressure,C
6014 !  COM /Sparms/ REAL Freq(801),COMPLEX S11(801),S21(801),S12(801),S22(801)
6015 !  COM /Sparms/ Mag_s11_id$[40],Ang_s11_id$[40],Mag_s21_id$[40],Ang_s21_id$
 => [40]
6016 !  COM /Sparms/ Mag_s22_id$[40],Ang_s22_id$[40],Mag_s12_id$[40],Ang_s12_id$
 => [40]
6017 !  COM /Sparm/ Frequency(801),COMPLEX Sparm(801),Sparm$[3],INTEGER Num_poin
 => ts
6018     COM /S_array/ INTEGER Dcount,Svalid,REAL S(801,5,2)
6019 Init_variables:  !
6020     RAD
6021     B=.02286
6022     Coaxial_len=1.15
6023     Waveguide_len=.36
6024 Calc_delay:  !
6025     Cutoff_freq=C/(2*B)  ! X-BAND WAVEGUIDE
6026     FOR J=1 TO Num_points
6027         Lambda=C/Frequency(J)
6028         Lambda_wg=Lambda/SQRT(1-(Cutoff_freq/Frequency(J))^2)
6029         Phase_delay_wg=4*PI*Waveguide_len/Lambda_wg
6030         Phase_delay_cx=4*PI*Coaxial_len/Lambda
6031         Total_delay=Phase_delay_wg+Phase_delay_cx
6032         Real_delay=COS(Total_delay)
6033         Imag_delay=-SIN(Total_delay)
6034         IF BIT(Svalid,0) THEN S(J,1,1)=S(J,1,1)*Real_delay
6035         IF BIT(Svalid,0) THEN S(J,1,2)=S(J,1,2)*Imag_delay
6036         IF BIT(Svalid,1) THEN S(J,2,1)=S(J,2,1)*Real_delay
6037         IF BIT(Svalid,1) THEN S(J,2,2)=S(J,2,2)*Imag_delay
6038         IF BIT(Svalid,2) THEN S(J,3,1)=S(J,3,1)*Real_delay
6039         IF BIT(Svalid,2) THEN S(J,3,2)=S(J,3,2)*Imag_delay
6040         IF BIT(Svalid,3) THEN S(J,4,1)=S(J,4,1)*Real_delay
6041         IF BIT(Svalid,3) THEN S(J,4,2)=S(J,4,2)*Imag_delay
6042     NEXT J
6043     DEG
6044   SUBEND
6045 !
6046  ! ****************************************************************
6047 !
6048   SUB Enter_id(Id$,OPTIONAL Return_test$)
6049 Enter_id:!
6050   !
6051  !LAST REVISION  30/SEPT/86
6052     OPTION BASE 1
6053     COM /Bugs/ INTEGER Bug1,Bug2,Bug3,Printer
6054   !
6055     DIM Test$[160]
6056     INTEGER N
6057     N=LEN(Id$)
6058     Test$=Id$
6059     SELECT Id$
6060     CASE ""
6061         !OUTPUT NOTHING
6062     CASE ELSE
6063         OUTPUT 2 USING "K,#";Test$
6064     END SELECT
6065     SELECT NPAR
6066     CASE 1    !NO Return_test$ given
6067         DISP CHR$(129);"Please ENTER a description (<= 40 chrs).";
6068         DISP CHR$(128);
6069     CASE ELSE
6070         DISP CHR$(129);"Please ENTER a description (<= 40 chrs) ";
6071         DISP CHR$(128);
6072         SELECT Return_test$
6073         CASE Id$
6074             DISP " for THIS ID";
6075         CASE "ABORT"
6076             DISP " CLR LN/ ENTER to ABORT."
6077         CASE ELSE
6078             DISP " for ";Return_test$;
6079         END SELECT
6080     END SELECT
6081     LINPUT Test$
6082     DISP ""
6083     Test$=TRIM$(Test$)
6084     N=LEN(Test$)
6085     SELECT N
6086     CASE >40
```

```
            DISP "Length of data_id$ too long. You entered ";N;
            DISP " characters. Try again."
            BEEP
            WAIT 1.5
            IF NPAR=2 THEN
                IF Id$<>Return_test$ THEN
                    OUTPUT 2 USING "#,K";Test$
                END IF
            END IF
            GOTO Enter_id
        CASE =0
            IF NPAR>1 THEN
                IF Return_test$="ABORT" THEN
                    Id$=Test$   !=""
                    SUBEXIT
                END IF
            END IF
            DISP "You must ENTER SOMETHING or you'll ";
            DISP "never get out of this."
            BEEP
            WAIT 1.8
            GOTO Enter_id
        CASE ELSE
            !Everything ok
        END SELECT
        Id$=Test$
    SUBEND    !Enter_id
 !
 ! ************************************************************
 !
    SUB Save_file(T_f(*),INTEGER Datacount,Id$)
  Save_file:  !
        OPTION BASE 1
        COM /Files/ Diskdrive$,Filename$
        ON ERROR CALL Errortrap
        Diskspace=INT((3500+(Datacount*16))/256)+1
        CREATE BDAT Filename$&Diskdrive$,Diskspace,256
!       CREATE ASCII Filename$&Diskdrive$,Diskspace*2
        ASSIGN @Datapath TO Filename$&Diskdrive$
        OUTPUT @Datapath;"N"
        OUTPUT @Datapath;TRIM$(Id$)
        OUTPUT @Datapath;Datacount
        OUTPUT @Datapath;Datacount
        OUTPUT @Datapath;T_f(*)
        ASSIGN @Datapath TO *
        OFF ERROR
    SUBEND     !Save_file
!
! ****************************************************************
!
    SUB Enter_real(Low,High,Prompt$,Result)
        OPTION BASE 1
        INTEGER Bad_number
        DIM Test$[160]
        REAL Test_real
  Enter_real:!
        DISP "Please ENTER the desired value for";Prompt$;
        OUTPUT 2 USING "K,#";Result
        LINPUT Test$
        Bad_number=0
        ON ERROR GOSUB Trap_bad_number
        Test_real=VAL(TRIM$(Test$))
        OFF ERROR
        IF Bad_number THEN
            IF Test$="" THEN
                SUBEXIT
            ELSE
                GOTO Enter_real
            END IF
        END IF
        SELECT Test_real
        CASE <Low
            BEEP 1000,.3
            DISP " Number entered is TOO LOW. ";
            DISP " LOWEST allowable number is ";Low
            WAIT 2.1
            GOTO Enter_real
        CASE >High
            BEEP 1000,.3
            DISP " Number entered is TOO HIGH. ";
            DISP " HIGHEST allowable number is ";High
            WAIT 2.1
            GOTO Enter_real
        CASE ELSE
            Result=Test_real
            ! Number within limits
        END SELECT
        SUBEXIT
!
! ///////////////////////////////////////////////////////////
!
  Trap_bad_number:!
        SELECT ERRN
        CASE 15,32
            DISP CHR$(129);" What you ENTERED is not a number! Try again. ";
            DISP CHR$(128)
            Bad_number=1
            WAIT 1.7
            LINPUT "Please ENTER the number you wish for",Test$
        CASE ELSE
            DISP ERRN,ERRM$
            BEEP 850,.5
            Bad_number=1
            PAUSE
        END SELECT
        RETURN
 !
 ! ///////////////////////////////////////////////////////////
 !
    SUBEND
 !
 ! ************************************************************
 !
    SUB New_save_sparms(INTEGER Ssave)
 ! This subprogram saves the chosen S-parameters that are passed in common
 ! blocks.  The BITs 0 through 3 of the pass parameter Ssave are used to
 ! select which S-parameters are to be saved.
 ! The S(I,J,K) array is (801,5,2) big.  I=1:Dcount .. K=REAL/IMAG
 ! J=1:S11 .. J=2:S21 .. J=3:S12 .. J=4:S22 .. J=5,K=1:Frequency
 ! This array construct is the same as the NWA automated measurement program
 ! To save disk space when only interested in one S-parameter, other
 ! programs save only the frequency and the S-parameter.  That is why the
 ! Basket array is a variable size that is determined by Num_sparms.
 ! If Basket contained all four S-parameters, its dimension would be
 ! (I,J)=(Dcount,9), where J=1:Freqeuncy, 2/3:REAL/IMAG(S11), 4/5:REAL/IMAG(
   => S21), 6/7:REAL/IMAG(S12), 8/9:REAL/IMAG(S22)
 ! If Basket contained just S22 then (I,J)=(Dcount,3) where J=1:Frequency,
   => J=2/3:REAL/IMAG(S22), which saves diskspace.
 !
        OPTION BASE 1
        COM /Files/ Diskdrive$[20],Filename$[10]
        COM /S_array/ INTEGER Dcount,Svalid,REAL S(801,5,2)
```

```
6217       INTEGER Num_sparms,I
6218       Num_sparms=BIT(Ssave,0)+BIT(Ssave,1)+BIT(Ssave,2)+BIT(Ssave,3)
6219       ALLOCATE Basket(1:Dcount,Num_sparms*2+1)
6220 New_save_sparms:OFF KEY
6221       MAT Basket(1:Dcount,1)= S(1:Dcount,5,1)   !Frequency
6222       Column=2
6223       FOR I=1 TO 4
6224           IF BIT(Ssave,I-1) THEN          !SAVE S11,S21,S12,S22
6225               MAT Basket(1:Dcount,Column)= S(1:Dcount,I,1)
6226               MAT Basket(1:Dcount,Column+1)= S(1:Dcount,I,2)
6227               Column=Column+2
6228           END IF
6229       NEXT I
6230       DISP CHR$(131);" Saving: ";CHR$(128);Filename$;Diskdrive$
6231       Diskspace=INT(Dcount*16.*(Num_sparms*2+1)/256)+20
6232       ON ERROR GOSUB Trap_error
6233 !     CREATE BDAT Filename$&Diskdrive$,Diskspace,256
6234       CREATE ASCII Filename$&Diskdrive$,Diskspace
6235       OFF ERROR
6236       ASSIGN @Datapath TO Filename$&Diskdrive$
6237       OUTPUT @Datapath;Dcount,Ssave
6238       OUTPUT @Datapath;Basket(*)
6239       ASSIGN @Datapath TO *
6240       DEALLOCATE Basket(*)
6241       DISP
6242       SUBEXIT
6243 Trap_error:       DISP ERRM$
6244                    BEEP
6245                    PAUSE
6246                    RETURN
6247 Purge_old:PURGE Filename$&Diskdrive$
6248       BEEP
6249       PRINT "I just purged ";Filename$;Diskdrive$
6250       RETURN
6251   SUBEND
6252 !
6253 ! ****************************************************************
6254 !
6255   SUB Fit_sparms
6256 !     Fit_lorentz(Kx,Ky,Q,F0,Ukx,Uky,Uq,Uf0)           Sss=1
6257 ! or  Fit_lorentz(Q0,Ql,F0,Phase,Uq0,Uql,Uf0,Uphase)  Sss=2 or 3
6258  !Written by Eric J. Vanzura
6259  !Levenberg-Marquart algorithm CSUB taken from:
6260  !Numerical Recipes : the art of Scientific Computing
6261  !W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling
6262  !Cambridge Press 1987
6263 Fit_sparms:OFF KEY
6264       GOSUB Init
6265       GOSUB Init_est
6266       GOSUB Initial_plot             !Plot meas'd S-parms & initial estimate
6267       REPEAT
6268 Do_mrq:!
6269           GOSUB Printlatestvals
6270           Ochisq=Chisq
6271           MAT Old_a= A
6272           DISP "Performing Levenberg-Marquart Least-square-fit"
6273           OFF KEY
6274           CALL Mrqmin(X(*),Y(*),Sig(*),Ndata(*),A(*),Ma(*),Lista(*),Mfit(*)
  => ,Covar(*),Alpha(*),Nca(*),Chisq,Alamda,Sss,Debug)
6275           GOSUB Convergencekeys
6276           MAT Old_est= Sgen
6277           GOSUB Gen_estimate
6278           IF NOT Speed THEN
6280               GOSUB Plot_estimate
6281           END IF
6282           GOSUB Ask_converge
6283       UNTIL Itst>=Itst_limit OR Force_converge
6284 Mrq_out:OFF KEY
6285       Alamda=0
6286       CALL Mrqmin(X(*),Y(*),Sig(*),Ndata(*),A(*),Ma(*),Lista(*),Mfit(*),Cov
  => ar(*),Alpha(*),Nca(*),Chisq,Alamda,Sss,Debug)
6287       IF NOT Speed THEN
6288       GOSUB Plot_residuals
6289       END IF
6291       GOSUB Calc_sig2
6292       GOSUB Calc_uncert
6293       GOSUB Print_results
6294       GOTO Subend
6295 !
6296 ! ////////////////////////////////////////////////////////////////
6297 !
6298 Calc_uncert:!
6299       SELECT Sss
6300       CASE =1
6301           Ua(1)=SQRT(Sig2*Covar(1,1)) !Kx
6302           Ua(2)=SQRT(Sig2*Covar(2,2)) !Ky
6303           Ua(3)=SQRT(Sig2*Covar(3,3)) !Q
6304           Ua(4)=SQRT(Sig2*Covar(4,4)) !F0
6305           Ua(5)=SQRT(Sig2*Covar(5,5)) !
6306       CASE =2,=3
6307           Ua(1)=SQRT(Sig2*Covar(1,1)) !Uq0
6308           Ua(2)=SQRT(Sig2*Covar(2,2)) !Uql
6309           Ua(3)=SQRT(Sig2*Covar(3,3)) !Uf0
6310           Ua(4)=SQRT(Sig2*Covar(4,4)) !Uphase
6311           Ua(4)=Ua(4)*360/2/PI            !in degrees
6312           Ua(5)=SQRT(Sig2*Covar(5,5)) !Umag (beta)
6313       CASE ELSE
6314           BEEP
6315           DISP "Sss=";Sss;" in Calc_uncert. PAUSED"
6316           PAUSE
6317           GOTO Calc_uncert
6318       END SELECT
6319       RETURN
6320 !
6321 ! ////////////////////////////////////////////////////////////////
6322 !
6323 Calc_sig2:!
6324       FOR I=1 TO Fitcount
6325           GOSUB Gen_cnum
6326           N=Start_index+I-1
6327           Sig2=Sig2+(ABS(CMPLX(S(N,Ns,1),S(N,Ns,2))-Cnum))^2
6328       NEXT I
6329       Sig2=Sig2/(Fitcount-Mfit(2))
6330       RETURN
6331 !
6332 ! ////////////////////////////////////////////////////////////////
6333 !
6334 Print_results:!
6335       PRINTER IS 1
6336       PRINT Description$;" fit to ";Sparm$;" values."
6337       PRINT USING "K,3D,2(3X,K,MD.3DE)";"Iteration #",K,"Chi_squared=",Chis
  => q,"Alamda=",Alamda
6338       PRINT USING "3D,K,3D,K,3D";Dcount;" data points in file. Start_index=
  => ",Start_index,"  Stop_index=",Stop_index
6339       GOSUB Print_a
6340       GOSUB Print_ua
6341       PRINT USING "K,D.4DE";"Chi-squared=",Chisq
6342       IF Printer_on THEN
6343           PRINTER IS Print_addr
6344           GOSUB Print_a
6345           GOSUB Print_ua
6346           PRINT USING "K,D.4DE";"Chi-squared=",Chisq
```

```
6347          PRINTER IS 1
6348       END IF
6349       Reset_keys=1
6350       RETURN
6351 !
6352 ! ////////////////////////////////////////////////////////////
6353 !
6354 Initial_plot:!
6355       ALLOCATE File(Fitcount*3,2),Old_est(Fitcount,2),Orig_sparm(Fitcount,2
  => )
6356       ALLOCATE Sgen(Fitcount,2)
6357       MAT Orig_sparm(*,1:2)= S(Start_index:Stop_index,Ns,1:2)
6358       GOSUB Gen_estimate
6359       MAT Old_est= Sgen
6360       IF NOT Speed THEN GOSUB Plot_estimate
6361       RETURN
6362 !
6363 ! ////////////////////////////////////////////////////////////
6364 !
6365 Gen_cnum:!
6366       SELECT Sss
6367       CASE 1   !S21
6368           Cnum=-X(I)^2*CMPLX(A(1),-A(2))/CMPLX(X(I)^2-A(4)^2,-X(I)*A(4)/A(3
  => ))
6369       CASE 2   !S11
6370           Degrees=360*A(4)/2/PI
6371           P=CMPLX(COS(Degrees),SIN(Degrees))
6372           Ql2f=2*A(2)*(X(I)-A(3))/A(3)
6373           Cnum=A(5)*CMPLX(A(2)/A(1)+Ql2f*Ql2f,Ql2f*(1-A(2)/A(1)))*P/(1+Ql2f
  => *Ql2f)
6374       CASE 3   !S21
6375           Degrees=360*A(4)/2/PI
6376           P=CMPLX(COS(Degrees),SIN(Degrees))
6377           G0gl=1-A(1)/A(2)
6378           Dum=CMPLX(1,2*(X(I)-A(3))/A(2))
6379           Cnum=A(5)*G0gl*P/Dum
6380       CASE ELSE
6381           BEEP
6382           DISP "Sss=";Sss;" is not valid in Gen_cnum."
6383           PAUSE
6384       END SELECT
6385       RETURN
6386 !
6387 ! ////////////////////////////////////////////////////////////
6388 !
6389 Gen_estimate:!
6390       DISP "Generating latest estimate values"
6391       MAT Sgen= (0)
6392       SELECT Sss
6393       CASE =1  !Nussenzveig Lorentzian
6394           FOR I=1 TO Fitcount
6395               GOSUB Gen_cnum
6396               Sgen(I,1)=REAL(Cnum)
6397               Sgen(I,2)=IMAG(Cnum)
6398           NEXT I
6399       CASE =2  !Schulten S11
6400           FOR I=1 TO Fitcount
6401               GOSUB Gen_cnum
6402               Sgen(I,1)=REAL(Cnum)
6403               Sgen(I,2)=IMAG(Cnum)
6404           NEXT I
6405       CASE =3  !Schulten S21
6406           FOR I=1 TO Fitcount
6407               GOSUB Gen_cnum
6408               Sgen(I,1)=REAL(Cnum)
6409               Sgen(I,2)=IMAG(Cnum)
6410           NEXT I
6411       CASE ELSE
6412           DISP "Sss=";Sss;" is not valid in Gen_estimate."
6413           BEEP
6414           PAUSE
6415           GOTO Gen_estimate
6416       END SELECT
6417       RETURN
6418 !
6419 ! ////////////////////////////////////////////////////////////
6420 !
6421 Plot_estimate:!
6422       Pack_data(File(*))
6423       Id$=Sparm$
6424       Pen=2
6425       Pack_data(File(*),Orig_sparm(*),Fitcount,Pen,Id$)
6426 !
6427       Id$="Old estimate"
6428       Pen=4
6429       Pack_data(File(*),Old_est(*),Fitcount,Pen,Id$)
6430 !
6431       Id$="Latest estimate"
6432       Pen=6
6433       Pack_data(File(*),Sgen(*),Fitcount,Pen,Id$)
6434 !
6435       Test$="Real Part"
6436       Prompt$="Imaginary Part"
6437       Id$="Results from file:"&Rawdatafile$
6438       Init_graphics(Test$,Prompt$,Id$)
6439       Ax$="XY"
6440       Autoscale(File(*),Ax$)
6441       Refresh_labels
6442       DISP "S-parameters are: Red=measured data, Green=Old estimate, Blue=L
  => atest Estimate"
6443       Plot_all(File(*))
6444       RETURN
6445 !
6446 ! ////////////////////////////////////////////////////////////
6447 !
6450 Plot_residuals:!
6451       Pack_data(File(*))
6452       ALLOCATE Reres(Fitcount,2),Imres(Fitcount,2)
6453       FOR I=1 TO Fitcount
6455       Reres(I,1)=S(Start_index+I-1,5,1)/1.E+9
6456       Imres(I,1)=S(Start_index+I-1,5,1)/1.E+9
6458       Reres(I,2)=Sgen(I,1)-Orig_sparm(I,1)
6459       Imres(I,2)=Sgen(I,2)-Orig_sparm(I,2)
6460       NEXT I
6461       Id$="real residuals"
6462       Pen=2
6463       Pack_data(File(*),Reres(*),Fitcount,Pen,Id$)
6464 !
6465       Id$="imag residuals"
6466       Pen=4
6467       Pack_data(File(*),Imres(*),Fitcount,Pen,Id$)
6468 !
6472       Test$="Frequency (GHz)"
6473       Prompt$="residual"
6474       Id$="red=real green=imaginary"
6475       Init_graphics(Test$,Prompt$,Id$)
6476       Ax$="XY"
6477       Autoscale(File(*),Ax$)
6478       Refresh_labels
6480       Plot_all(File(*))
6481       RETURN
6482 !
```

```
6483 Init:!
6484       DEG
6485       OPTION BASE 1
6486       INTEGER I,Prty,Itst,Itst_limit,Ns
6487       INTEGER Pen,Done,Fitcount
6488       COMPLEX Cnum,P,Dum
6489       REAL Debug
6490       DIM Test$[160],Prompt$[40],Id$[40],Rawdatafile$[10],Sparm$[3]
6491       COM /History/ Status$[1],Time_orgn$[8],Date_orgn$[11]
6492       COM /History/ Time_chng$[8],Date_chng$[11],Description$[160]
6493       COM /Labels/ Labels$(30)[60],INTEGER Lbl_count,REAL Lbl_addr(30,6)
6494       COM /Data_param/ INTEGER Data_count,Filesize,Curvecount,Roster(17,4)
6495       COM /Data_param/ REAL Sym_size,Symbol$(17)[2],Curve_id$(17)[40]
6496       COM /Data_param/ REAL Xmin_data,Xmax_data
6497       COM /Data_param/ REAL Ymin_data,Ymax_data
6498       COM /Graphics/ INTEGER Speed
6499       COM /Background/ Graphtype$[12],Margins$(2)[10],Papersize$[1]
6500       COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
6501       COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
6502       COM /Background/ Xgrid_tick$[4],INTEGER Xmajor,Xminor
6503       COM /Background/ Ygrid_tick$[4],INTEGER Ymajor,Yminor
6504       COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
6505       COM /Axes_labels/ Print_xlabels$[3],Print_ylabels$[3]
6506       COM /Axes_labels/ Sig_digits$[2],REAL Xlcsize,Ylcsize
6507       COM /Windowspace/ REAL Xmin,Xmid,Xmax,Ymin,Ymid,Ymax! graph edges UDU
6508       COM /Windowspace/ REAL Xleft,Xright,Ybottom,Ytop! paper edges UDUs
6509       COM /Log_scale/ REAL Xcycles,Xbegin,Ycycles,Ybegin
6510       COM /Plot_device/ Plot_lang$[10],INTEGER Plot_addr
6511       COM /Hard_space/ REAL Xview_lft,Xview_rt,Yview_btm,Yview_top
6512       COM /Hard_space/ REAL Viewscale,Aspect_ratio
6513       COM /Frame_onoff/ INTEGER Frame_flag
6514       COM /Clear_space/ REAL Space_lft,Space_rt,Space_btm,Space_top
6515 !     !
6516 Init_com:  !
6517       COM /Fittype/ INTEGER Sselect,Start_index,Stop_index
6518       COM /S_array/ INTEGER Dcount,Svalid,REAL S(801,5,2)
6519       COM /Cdata/ INTEGER Mfit(2),Lista(5,2),Ma(2),REAL A(5),Ua(5),INTEGER
  => Ndata(2)
6520       COM /Cstats/ Alamda,Chisq,INTEGER Nca(2),REAL Alpha(5,5),Covar(5,5)
6521       COM /Output/ INTEGER Print_addr,Plotter,Printer_on,Plotter_on
6522 Init_vals:!
6523           Speed=0
6524       K=0
6525       Itst=0
6526       Itst_limit=3
6527       Plot_addr=3
6528       Plot_lang$="INTERNAL"
6529       M=5       !Dimension of Covar,Alpha,Lista & A matricies
6530       Debug=0
6531 !
6532       IF NOT BIT(Svalid,Sselect) THEN
6533           BEEP
6534           DISP "The S(*) array for ";Sparm$;" is not valid (doesn't have da
  => ta?) PAUSED"
6535           PRINT "The S(*) array for ";Sparm$;" is not valid (doesn't have d
  => ata?) PAUSED"
6536           PAUSE
6537       END IF
6538 !Sss=1 S21-Nussenzvieg    Sss=2 S11-Schulten    Sss=3 S21-Schulten
6539       SELECT Sselect
6540       CASE 0       !S11
6541           Sss=2
6542           Sparm$="S11"
6543       CASE 1       !S21
6544           Sss=1
6545 !         Sss=3
6546           Sparm$="S21"
6547       CASE 2       !S12
6548           Sss=1
6549           Sss=3
6550           Sparm$="S12"
6551       CASE 3       !S22
6552           Sss=2
6553           Sparm$="S22"
6554       CASE ELSE
6555           BEEP
6556           DISP "Sselect is not valid in Fit_sparms"
6557           PAUSE
6558       END SELECT
6559       Ns=Sselect+1
6560       RETURN
6561 !
6562 ! ////////////////////////////////////////////////////////////
6563 !
6564 Init_est:  !
6565       Fitcount=Stop_index-Start_index+1
6566       Start_freq=S(Start_index,5,1)
6567       Stop_freq=S(Stop_index,5,1)
6568       SELECT Sss
6569       CASE =1  ! Nussenzveig Lorentzian
6570           Ix=INT(Start_index+Fitcount/2)
6571           Angle=ARG(CMPLX(S(Ix,Ns,1),S(Ix,Ns,2)))
6572           Mag=ABS(CMPLX(S(Ix,Ns,1),S(Ix,Ns,2)))
6573           A(1)=-2*10^(-5)*Mag*SIN(Angle)
6574           A(2)=-2*10^(-5)*Mag*COS(Angle)
6575           A(3)=1.5*S(Ix,5,1)/(Stop_freq-Start_freq)
6576           A(4)=S(Ix,5,1)
6577       CASE =2,=3  ! Schulten S11 or S21
6578           Ix=INT(Start_index+Fitcount/2)
6579 !         A(1)=S(Ix,5,1)/(Stop_freq-Start_freq)*1.8
6580 !         A(2)=S(Ix,5,1)/(Stop_freq-Start_freq)*2.0
6584           A(1)=S(Ix,5,1)/90000
6585           A(2)=S(Ix,5,1)/79000
6586           A(3)=S(Ix,5,1)
6590           A(4)=ARG(CMPLX(S(Ix,Ns,1),S(Ix,Ns,2)))*2*PI/360
6591 !         A(4)=0
6592 !         A(5)=ABS(CMPLX(S(Ix,Ns,1),S(Ix,Ns,2)))
6593           A(5)=1
6594       CASE ELSE
6595           DISP "Sss=";Sss;" is not valid in Init_est"
6596           BEEP
6597           PAUSE
6598           GOTO Init_est
6599       END SELECT
6600       Ndata(1)=0
6601       Ndata(2)=Fitcount*2
6602       Ma(1)=0              !size of A array (unknowns)
6603       Ma(2)=M
6604       Chisq=1              !measure of fit
6605       Mfit(1)=0            !number of unknowns to be fitted
6606       Mfit(2)=M
6607       IF Sss=1 THEN Mfit(2)=M-1
6608       MAT Lista= (0)
6609       Lista(1,2)=1         !A array index of the unknowns to be fitted
6610       Lista(2,2)=2
6611       Lista(3,2)=3
6612       Lista(4,2)=4
6613       Lista(5,2)=5
6614       IF Sss=1 THEN Lista(5,2)=0
6615       MAT Covar= (0)
6616       MAT Alpha= (0)
6617       Nca(1)=0             !Size of Alpha & Covar arrays >= Ma
```

```
6618        Nca(2)=M
6619        Alamda=-2
6620        ALLOCATE Old_a(M)
6621        ALLOCATE REAL X(Fitcount*2),Y(Fitcount*2),Sig(Fitcount*2)
6622        MAT Sig= (1)
6623        MAT X(1:Fitcount)= S(Start_index:Stop_index,5,1)
6624        MAT X(Fitcount+1:Fitcount*2)= S(Start_index:Stop_index,5,1)
6625        MAT Y(1:Fitcount)= S(Start_index:Stop_index,Ns,1)
6626        MAT Y(Fitcount+1:Fitcount*2)= S(Start_index:Stop_index,Ns,2)
6627        RETURN !Init_est
6628 !
6629 ! ///////////////////////////////////////////////////////////////
6630 !
6631 Done_return:OFF KEY
6632        Done=1
6633        RETURN
6634 !
6635 ! ///////////////////////////////////////////////////////////////
6636 !
6637 Force_converge:PRINT "Convergence will be forced."
6638        Force_converge=1
6639        RETURN
6640 !
6641 ! ///////////////////////////////////////////////////////////////
6642 !
6643 Preventconverge:PRINT "Convergence will be prevented."
6644        Itst=0
6645        RETURN
6646 !
6647 ! ///////////////////////////////////////////////////////////////
6648 !
6649 Graph_on: !
6650        Speed=0
6651        OFF KEY 4
6652 !      ON KEY 4 LABEL "GRAPH  OFF",Prty GOSUB Graph_off
6653        RETURN
6654 !
6655 ! ///////////////////////////////////////////////////////////////
6656 !
6657 Graph_off: !
6658        Speed=1
6659        OFF KEY 4
6660 !      ON KEY 4 LABEL "GRAPH  ON",Prty GOSUB Graph_on
6661        RETURN
6662 !
6663 ! ///////////////////////////////////////////////////////////////
6664 !
6666 Debug_on:PRINT "Debug flag is turned on."
6667        Debug=1
6668        OFF KEY 6
6669        ON KEY 6 LABEL "DEBUG  OFF",Prty GOSUB Debug_off
6670        RETURN
6671 !
6672 ! ///////////////////////////////////////////////////////////////
6673 !
6674 Debug_off:PRINT "Debug flag is turned off."
6675        Debug=0
6676        OFF KEY 6
6677        ON KEY 6 LABEL "DEBUG  ON",Prty GOSUB Debug_on
6678        RETURN
6679 !
6680 ! ///////////////////////////////////////////////////////////////
6681 !
6682 Ask_converge:!
6683        SELECT Chisq
6684        CASE >Ochisq
6685            Itst=0
6686            PRINT USING "K,D.DE";"New Chisq is greater than old Chisq by:";Ch
  => isq-Ochisq
6687        CASE =Ochisq
6688            PRINT "Chisq is equal to old Chisq"
6689        CASE <Ochisq
6690            PRINT USING "K,D.DE";"New Chisq is LESS than old Chisq by:";Ochis
  => q-Chisq
6691            IF ABS((Ochisq-Chisq)/Chisq)<.003 THEN Itst=Itst+1
6692            IF ABS((Ochisq-Chisq)/Chisq)<.00003 THEN Itst=Itst+1
6693            PRINT USING "K,D";"Itst=";Itst
6694            PRINT USING "K,D.DE";"ABS((Ochisq-Chisq)/Chisq)=";ABS((Ochisq-Chi
  => sq)/Chisq)
6695        END SELECT
6696        RETURN
6697 !
6698 ! ///////////////////////////////////////////////////////////////
6699 !
6700 Convergencekeys:!
6701        Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
6702        ON KEY 0 LABEL "FORCE  CONVERG",Prty GOSUB Force_converge
6703        ON KEY 2 LABEL "PREVENTCONVERG",Prty GOSUB Preventconverge
6704        IF Speed THEN
6705            ON KEY 4 LABEL "GRAPH  ON",Prty GOSUB Graph_on
6706        ELSE
6707            ON KEY 4 LABEL "GRAPH  OFF",Prty GOSUB Graph_off
6708        END IF
6709        IF Debug THEN
6710            ON KEY 6 LABEL "DEBUG  OFF",Prty GOSUB Debug_off
6711        ELSE
6712            ON KEY 6 LABEL "DEBUG  ON",Prty GOSUB Debug_on
6713        END IF
6714        RETURN
6715 !
6716 ! ///////////////////////////////////////////////////////////////
6717 !
6718 Printlatestvals:!
6719        PRINT USING "K,MD.3DE";"Chisq=",Chisq
6720        K=K+1
6721        PRINT USING "K,3D,L,K,MD.3DE";"Iteration #",K,"Alamda=",Alamda
6722        GOSUB Print_a
6723        RETURN
6724 !
6725 ! ///////////////////////////////////////////////////////////////
6726 !
6727 Print_a:!
6728        SELECT Sss
6729        CASE =1
6730            IF A(4)<1.E+10 THEN
6731                PRINT USING A_flt10g;"QL=",A(3),"FO =",A(4)/1.E+9,"F0+(Span/2
  => )=",(A(4)+A(4)/2/A(3))/1.E+9,"Span=",A(4)/A(3)/1000
6732            ELSE
6733                PRINT USING A_fgt10g;"QL=",A(3),"FO =",A(4)/1.E+9,"F0+(Span/2
  => )=",(A(4)+A(4)/2.E+9/A(3))/1.E+9,"Span=",A(4)/A(3)/1000
6734            END IF
6735        CASE =2,=3
6736            PRINT USING "2(K,MD.4DE,2X),K,MD.9DE,2X,K,M3D.DD,2X,K,MD.3DE";"Q0
  => =",A(1),"Ql=",A(2),"F0=",A(3),"Degrees=",A(4)*360/2/PI,"Mag=",A(5)
6737        CASE ELSE
6738            DISP "Sss=";Sss;" is not valid in print_a. program paused"
6739            BEEP
6740            PAUSE
6741        END SELECT
6742        RETURN
6743 A_flt10g:IMAGE K,6D,X,K,X,D.9D,X,K,X,D.9D,X,K,3D.3D
6744 A_fgt10g:IMAGE K,6D,X,K,DD.9D,X,K,DD.9D,X,K,3D.3D
```

```
!
! ///////////////////////////////////////////////////////////////
!
Print_ua:!
      SELECT Sss
      CASE =1
          PRINT USING "2(K,MD.3DE,4X),K,M5D,3X,K,3X,MD.3DE";"Ukx=",Ua(1),"U
ky=",Ua(2),"UQ=",Ua(3),"Uf0=",Ua(4)
      CASE =2,=3
          PRINT USING "2(K,MD.3DE,2X),K,MD.9DE,2X,K,M3D.3D,2X,K,MD.3DE";"UQ
0=",Ua(1),"UQl=",Ua(2),"UF0=",Ua(3),"Udegrees=",Ua(4),"Umag=",Ua(5)
      CASE ELSE
          DISP "Sss=";Sss;" is not valid in Print_ua. program paused"
          BEEP
          PAUSE
      END SELECT
      RETURN
Subend:!
  SUBEND
!
! ****************************************************************
!
  SUB New_load_sparms
! This subprogram loads S-parameters into an allocated basket which then
! puts the S-parameters into the appropriate S_parm according to the
! 0 through 3 BIT of the parameter Ssave.
! The S(I,J,K) array is (801,5,2) big.  I=1:Dcount .. K=REAL/IMAG
! J=1:S11 .. J=2:S21 .. J=3:S12 .. J=4:S22 .. J=5,K=1:Frequency
! This array construct is the same as the NWA automated measurement program
! To save disk space when only interested in one S-parameter, other
! programs save only the frequency and the S-parameter.  That is why the
! Basket array is a variable size that is determined by Num_sparms.
! If Basket contained all four S-parameters, its dimension would be
! (I,J)=(Dcount,9), where J=1:Freqeuncy, 2/3:REAL/IMAG(S11), 4/5:REAL/IMAG(
S21), 6/7:REAL/IMAG(S12), 8/9:REAL/IMAG(S22)
! If Basket contained just S22 then (I,J)=(Dcount,3) where J=1:Frequency,
 J=2/3:REAL/IMAG(S22), which saves diskspace.
      OPTION BASE 1
      COM /Files/ Diskdrive$[20],Filename$[10]
      COM /S_array/ INTEGER Dcount,Svalid,REAL S(*)
      INTEGER Num_sparms,I
New_load_sparms:OFF KEY
      DISP CHR$(131);" Loading: ";CHR$(128);Filename$;Diskdrive$
      ASSIGN @Datapath TO Filename$&Diskdrive$
      ENTER @Datapath;Dcount,Svalid
      Num_sparms=BIT(Svalid,0)+BIT(Svalid,1)+BIT(Svalid,2)+BIT(Svalid,3)
      ALLOCATE Basket(Dcount,Num_sparms*2+1)
      ENTER @Datapath;Basket(*)
      ASSIGN @Datapath TO *
      MAT S(1:Dcount,5,1)= Basket(*,1)
      Column=2
      FOR I=1 TO 4
          IF BIT(Svalid,I-1) THEN        !TRANSFER TO S11,S21,S12,S22
              MAT S(1:Dcount,I,1)= Basket(1:Dcount,Column)
              MAT S(1:Dcount,I,2)= Basket(1:Dcount,Column+1)
              Column=Column+2
          END IF
      NEXT I
      DEALLOCATE Basket(*)
      DISP
  SUBEND
!
! ****************************************************************
!
  SUB Pause_key_on
Pause_key_on: ! Make sure that CONTINUE key exists.
              ! Original: 02 Dec 1987
              ! Revision: 02 Dec 1987
      COM /Sys/ Sys_id$[10]
      IF Sys_id$[1,4]="S300" THEN ! reset to S300 system keys
          CONTROL KBD,15;0
          CONTROL CRT,12;2
          LOAD KEY
      END IF
      PAUSE
      IF Sys_id$[1,4]="S300" THEN  ! set to S200 compatible keys
          OUTPUT KBD USING "K,#";"SCRATCH KEY X"
          CONTROL KBD,15;1
          CONTROL CRT,12;0
      END IF
      SUBEXIT
  SUBEND
!
! ****************************************************************
!
```

Appendix E

CAVITYPROG Program Listing

```
4  !    PURGE "CAVITYPROG"
6  !    RE-STORE "CAVITYPROG"
8  ! Written by Eric J. Vanzura & Michael D. Janezic
10!
12     GOSUB Init
14     GOSUB Main_menu
16     GOTO End
18   !
20   ! ////////////////////////////////////////////////////////////
22   !
24 Init:  !
26     OPTION BASE 1
28 Init_com: !
30     COM /Environment/ Degc,Mbar,Relh,C
32     COM /Environment/ Udegc,Umbar,Urelh,Uc
34     COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
36     COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
38     COM /Sample/ Erreps(12),Errtand(12)
40     COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
42     COM /Cavity_dims/ Udiameter,Ulength
44     COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
46     COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
48     COM /Constants/ Bessel_root
50     COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,Prom
 => pt$[80]
52     COM /History/ Status$[1],Time_orgn$[8],Date_orgn$[11]
54     COM /History/ Time_chng$[8],Date_chng$[11],Description$[160]
56     COM /Labels/ Labels$(30)[60],INTEGER Lbl_count,REAL Lbl_addr(30,6)
58     COM /Data_param/ INTEGER Data_count,Filesize,Curvecount,Roster(17,4)
60     COM /Data_param/ REAL Sym_size,Symbol$(17)[2],Curve_id$(17)[40]
62     COM /Data_param/ REAL Xmin_data,Xmax_data
64     COM /Data_param/ REAL Ymin_data,Ymax_data
66     COM /Background/ Graphtype$[12],Margins$(2)[10],Papersize$[1]
68     COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
70     COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
72     COM /Background/ Xgrid_tick$[4],INTEGER Xmajor,Xminor
74     COM /Background/ Ygrid_tick$[4],INTEGER Ymajor,Yminor
76     COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
78     COM /Axes_labels/ Print_xlabels$[3],Print_ylabels$[3]
80     COM /Axes_labels/ Sig_digits$[2],REAL Xlcsize,Ylcsize
82     COM /Windowspace/ REAL Xmin,Xmid,Xmax,Ymin,Ymid,Ymax     ! graph edges UDU
84     COM /Windowspace/ REAL Xleft,Xright,Ybottom,Ytop    ! paper edges UDUs
86     COM /Log_scale/ REAL Xcycles,Xbegin,Ycycles,Ybegin
88     COM /Plot_device/ Plot_lang$[10],INTEGER Plot_addr
90     COM /Hard_space/ REAL Xview_lft,Xview_rt,Yview_btm,Yview_top
92     COM /Hard_space/ REAL Viewscale,Aspect_ratio
94     COM /Frame_onoff/ INTEGER Frame_flag
96     COM /Clear_space/ REAL Space_lft,Space_rt,Space_btm,Space_top
98   !      !
100    COM /Ana_data/ INTEGER Prog_id,Sweep_type,Sweep_mode,Datacount
102    COM /Ana_data/ INTEGER S11_valid,S21_valid,S12_valid,S22_valid,Ports
104    COM /Ana_data/ REAL Z0,Start,Stop
106    COM /Ana_data/ Title$[55],Company$[30],Operator_name$[30],Measure_time$[3
 => 0]
108    COM /Fittype/ INTEGER Sselect,Start_index,Stop_index
110    COM /S_array/ INTEGER Dcount,Svalid,REAL S(801,5,2)
112    COM /Cdata/ INTEGER Mfit(2),Lista(5,2),Ma(2),REAL A(5),Ua(5),INTEGER Ndat
 => a(2)
114    COM /Cstats/ Alamda,Chisq,INTEGER Nca(2),REAL Alpha(5,5),Covar(5,5)
116    COM /Output/ INTEGER Print_addr,Plotter,Printer_on,Plotter_on
118    COM /Sys/ Sys_id$[10]
120    COM /Sys_msi/ Msi_id$[20]
122    COM /Files/ Diskdrive$[20],Filename$[10]
124    COM /Enlarge_file/ INTEGER Overflow
126  ! COM /Fitsparm/ COMPLEX Sparm(801),Sparm$[3]
128    INTEGER Prty
130  !
132  ! ////////////////////////////////////////////////////////////
134  !
136 Init_vals: ! All values in SI (MKS) units
138    Bessel_root=3.8317059702
140    IF Diameter=0 OR Length0=0 THEN
142        Udiameter=.002/1.E+3
144        Ulength=.013/1.E+3
146        Diameter=.06002996
148        Length0=.4303531
150    END IF
152    IF Samplelen=0 OR Eps_re_guess=0 THEN
154        Eps_re_guess=3.83
156        Samplelen=.0144854
158        Usamplelen=.0000001
160    END IF
162    IF Freq(2)=0 THEN
164        Freq(2)=1.002389381E+10
166        Ufreq(2)=5.E+3
168    END IF
170    IF Degc=0 OR Relh=0 OR Mbar=0 THEN
172        Degc=23
174        Relh=40
176        Mbar=830
178        Udegc=.05
180        Urelh=1
182        Umbar=1
184    END IF
186    MAT Spectrum= (0)
188    FOR I=1 TO SIZE(Spectrum,1)
190        Spectrum(I,1)=1000
192    NEXT I
194    Num_modes=0
196    CALL Speed_of_light
198    CALL Uncert_c
199    Printer_on=1
200    Print_addr=9
202    RETURN
203  !
204  ! ////////////////////////////////////////////////////////////
206  !
208 Main_menu:!
210    GOSUB Menu_main
212    Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
214    ON KEY 1 LABEL "End    Program",Prty GOSUB End_program
216    ON KEY 2 LABEL "VarableLength",Prty CALL Var_length
218    ON KEY 3 LABEL "VarableFreqncy",Prty CALL Var_freq
220  ! ON KEY 4 LABEL "Calc   Dimens",Prty CALL Calc_dimens
222    ON KEY 4 LABEL "Cavity Dimens",Prty CALL Cavity_dimens
224    ON KEY 5 LABEL "Calc   C",Prty CALL Calc_c
226    ON KEY 6 LABEL "Calc   Modes",Prty CALL Calc_modes
228    ON KEY 7 LABEL "Calc   Freqs",Prty CALL Calc_freqs
230    ON KEY 8 LABEL "Fit    S parms",Prty CALL Fit_sparms
232  ! ON KEY 9 LABEL "Error  Curves",Prty CALL Error_curves
234    Done=0
236    Prior_menu=0
238    LOOP
240        IF Done THEN GOTO Main_menu
242    EXIT IF Prior_menu
244    END LOOP
246    RETURN
248  !
250  ! ////////////////////////////////////////////////////////////
252  !
254 End:OFF KEY
256    DISP "Program finished. Enter 'RUN' to start again."
```

```
257    SELECT TRIM$(Sys_id$)
258    CASE "PC300"
260        LOAD KEY "EDITKEYS:DOS,C"
261    CASE "S300"
262        LOAD KEY "/BASIC/EDITKEYS:,1400,0"
263    END SELECT
264    STOP
265  !
266  ! ////////////////////////////////////////////////////////////
267  !
268 Menu_main: !
270    CLEAR SCREEN
272    PRINT USING Image6;Sample_id$
274    PRINT USING Image7;Samplelen*1.E+3,Usamplelen*1.E+3
276    PRINT USING Image8;Eps_re_guess
278    PRINT USING Image1;Degc,Udegc,Relh,Urelh
280    PRINT USING Image2;Mbar,Umbar
282    PRINT USING Image3;C,Uc
284    PRINT
286    PRINT USING Image4;Diameter*1.E+3,Udiameter*1.E+3
288    PRINT USING Image5;Length0*1.E+3,Ulength*1.E+3
290    DISP "Main program menu (select from options given below)"
292    RETURN
294  !
296 Image1:IMAGE "Temperature = ",2D.2D," +/- ",D.2D," deg C   Relative Humidity
 =>  = ",2D.D," +/- ",D.D," %"
298 Image2:IMAGE "Barometric Pressure = ",3D.D," +/- ",D.D,"   mbar"
300 Image3:IMAGE "Speed of Light      = ",D.7DESZ," +/- ",D.3DESZ," meters/secon
 => d"
302 Image4:IMAGE "Resonator Diameter             =  ",2D.5D," +/- ",D.5D,"  mm"
304 Image5:IMAGE "          Length (micr. zero) = ",3D.5D," +/- ",D.5D,"  mm"
306 Image6:IMAGE "Sample    Description: ",K,"                 "
308 Image7:IMAGE "          Length     = ",2D.4D," +/- ",D.4D," mm"
310 Image8:IMAGE "       Epsilon Guess = ",3D.4D
312 !
314 ! ////////////////////////////////////////////////////////////
316 !
318 End_program:OFF KEY
320    Prior_menu=1
322    RETURN
324    END
326 !
328 ! ************************************************************
330 !
332    SUB Var_length
334 Var_length:OFF KEY
336        OPTION BASE 1
338        COM /Environment/ Degc,Mbar,Relh,C
340        COM /Environment/ Udegc,Umbar,Urelh,Uc
342        COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
344        COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
346        COM /Sample/ Erreps(12),Errtand(12)
348        COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
350        COM /Cavity_dims/ Udiameter,Ulength
352        COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
354        COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
356        COM /Constants/ Bessel_root
358        COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
 => Prompt$[80]
360    !
362        GOSUB Menu_var_len
364    !
366        DISP CHR$(129);" This is the Fixed-frequency/Variable-length menu ";C
 => HR$(128)
368        Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
370        ON KEY 1 LABEL "Prior  Menu",Prty CALL Prior_menu
372        ON KEY 2 LABEL "Sample Info",Prty CALL Sample_info
374        ON KEY 3 LABEL "Resnce Freq",Prty CALL Resnce_freq
376        ON KEY 4 LABEL "Micr.  Readngs",Prty CALL Micr_readings
378        ON KEY 5 LABEL "Q      Values",Prty CALL Q_values
380        ON KEY 6 LABEL "Band   Widths",Prty CALL Band_widths
382        ON KEY 7 LABEL "Calc   Epsilon",Prty GOSUB Calc_eps_len
384        ON KEY 8 LABEL "Calc   Errors ",Prty GOSUB Calc_errs_len
386        Done=0
388        Prior_menu=0
390        LOOP
392            IF Done THEN GOTO Var_length
394        EXIT IF Prior_menu
396        END LOOP
398        OFF KEY
400        Prior_menu=0
402        Done=T
404        SUBEXIT
406 !
408 Calc_eps_len:OFF KEY
410        Do_var_freq=0
412        CALL Calc_eps_re
414        CALL Calc_tand
416        CALL Calc_eps_im
418        CALL Calc_errors(1)
420        Done=1
422        RETURN
424  !
426  ! ////////////////////////////////////////////////////////////
428  !
430 Calc_errs_len:OFF KEY
432        Do_var_freq=0
434        CALL Calc_errors
436        Done=1
438        RETURN
440  !
442  ! ////////////////////////////////////////////////////////////
444  !
446 Menu_var_len:  !
448        CLEAR SCREEN
450        PRINT USING Image6;Sample_id$
452        PRINT USING Image6a;Eps_re_guess,"Method:Variable Length"
454        PRINT USING Image7;Samplelen*1.E+3,Usamplelen*1.E+3
456        PRINT
458        PRINT USING Image1;Degc,Udegc,Relh,Urelh
460        PRINT USING Image2;Mbar,Umbar
462        PRINT USING Image3;C,Uc
464        PRINT
466        PRINT USING Image4;Diameter*1.E+3,Udiameter*1.E+3
468        PRINT USING Image5;Length0*1.E+3,Ulength*1.E+3
470        PRINT
472        PRINT USING Image10ab
474        PRINT USING Image11b;Freq(1)/1.E+9,Ufreq(1)/1.E+9,Freq(2)/1.E+9,Ufreq
 => (2)/1.E+9
476        PRINT USING Image11a;Micr(1)*1.E+3,Umicr(1)*1.E+3,Micr(2)*1.E+3,Umicr
 => (2)*1.E+3
478        PRINT USING Image12ab;Q(1),Uq(1),Q(2),Uq(2)
480        PRINT USING Image13ab;Bw(1)/1.E+3,Ubw(1)/1.E+3,Bw(2)/1.E+3,Ubw(2)/1.E
 => +3,8
482        PRINT
484        PRINT USING Image15ab;Eps_re,Ueps_re,Eps_im,Ueps_im
486        PRINT USING Image16ab;Tand,Utand
488        RETURN
490 !
492 Image1:IMAGE "Temperature = ",2D.2D," +/- ",D.2D," deg C   Relative Humidity
 =>  = ",2D.D," +/- ",D.D," %"
494 Image2:IMAGE "Barometric Pressure = ",3D.D," +/- ",D.D,"   mbar"
```

```
496 Image3:IMAGE "Speed of Light      = ",D.7DESZ," +/- ",D.3DESZ," meters/secon
 => d"
498 Image4:IMAGE "Resonator Diameter              =  ",2D.5D," +/- ",D.5D,"  mm"
500 Image5:IMAGE "         Length (micr. zero) = ",3D.5D," +/- ",D.5D,"  mm"
502 Image6:IMAGE "Sample Description: ",K,"               "
504 Image6a:IMAGE "Initial guess for epsilon':",2D.3D,10X,K
506 Image7:IMAGE "       Length    = ",2D.4D," +/- ",D.4D," mm"
508 Image10ab:IMAGE 28X,"Empty",20X,"with Sample"
510 Image11a:IMAGE "Micrometer reading = ",2(S2D.4D,"   +/- ",D.4D,6X),"mm"
512 Image11b:IMAGE "Resonance frequency= ",2(2D.7D," +/- ",D.7D,3X),"GHz"
514 Image12ab:IMAGE "Q value          =   ",2(5D,"    +/- ",3D,11X)
516 Image13ab:IMAGE "3 dB Bandwidth     =  ",2(5D.2D,"   +/- ",3D.2D,6X),B,"kHz"
518 Image15ab:IMAGE "Epsilon'  = ",3D.4D," +/- ",D.4D,5X,"Epsilon''  = ",2D.5D,"
 =>  +/- ",D.5D
520 Image16ab:IMAGE "Tan delta = ",D.6D," +/- ",D.6D
522 !
524 ! ///////////////////////////////////////////////////////////////
526 !
528    SUBEND
530 !
532 ! ****************************************************************
534 !
536    SUB Var_freq
538 Var_freq:OFF KEY
540       OPTION BASE 1
542       COM /Environment/ Degc,Mbar,Relh,C
544       COM /Environment/ Udegc,Umbar,Urelh,Uc
546       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
548       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
550       COM /Sample/ Erreps(12),Errtand(12)
552       COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
554       COM /Cavity_dims/ Udiameter,Ulength
556       COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
558       COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
560       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
 => Prompt$[80]
562       COM /Constants/ Bessel_root
564    !
566       GOSUB Menu_var_freq
568    !
570       DISP CHR$(129);" This is the Fixed-length/Variable-frequency menu ";C
 => HR$(128)
572       Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
574       ON KEY 1 LABEL "Prior  Menu",Prty CALL Prior_menu
576       ON KEY 2 LABEL "Sample Info",Prty CALL Sample_info
578       ON KEY 3 LABEL "Resnce Freqs",Prty CALL Resnce_freqs
580       ON KEY 4 LABEL "Micr.  Readng",Prty CALL Micr_reading
582       ON KEY 5 LABEL "Q      Values",Prty CALL Q_values
584       ON KEY 6 LABEL "Band   Widths",Prty CALL Band_widths
586       ON KEY 7 LABEL "Calc   Epsilon",Prty GOSUB Calc_eps_freq
588       ON KEY 8 LABEL "Calc   Errors ",Prty GOSUB Calc_errs_freq
590       Done=0
592       Prior_menu=0
594       LOOP
596          IF Done THEN GOTO Var_freq
598       EXIT IF Prior_menu
600       END LOOP
602       OFF KEY
604       Prior_menu=0
606       Done=1
608       SUBEXIT
610 !
612 Calc_eps_freq:OFF KEY
614       Do_var_freq=1
616       CALL Calc_eps_re
618       CALL Calc_tand
620       CALL Calc_eps_im
622       CALL Calc_errors(1)
624       CALL Calc_err_anal
626       Done=1
628       RETURN
630  !
632  ! //////////////////////////////////////////////////////////////
634  !
636 Calc_errs_freq:OFF KEY
638       Do_var_freq=1
640       CALL Calc_errors
642       Done=1
644       RETURN
646  !
648 ! //////////////////////////////////////////////////////////////
650 !
652 Menu_var_freq: !
654       CLEAR SCREEN
656       PRINT USING Image6;Sample_id$
658       PRINT USING Image6a;Eps_re_guess,"Method:Variable Frequency"
660       PRINT USING Image7;Samplelen*1.E+3,Usamplelen*1.E+3
662       PRINT
664       PRINT USING Image1;Degc,Udegc,Relh,Urelh
666       PRINT USING Image2;Mbar,Umbar
668       PRINT USING Image3;C,Uc
670       PRINT
672       PRINT USING Image4;Diameter*1.E+3,Udiameter*1.E+3
674       PRINT USING Image5;Length0*1.E+3,Ulength*1.E+3
676       PRINT
678       PRINT USING Image10ab
680       PRINT USING Image11b;Freq(1)/1.E+9,Ufreq(1)/1.E+9,Freq(2)/1.E+9,Ufreq
 => (2)/1.E+9
682       PRINT USING Image11a;Micr(1)*1.E+3,Umicr(1)*1.E+3,Micr(2)*1.E+3,Umicr
 => (2)*1.E+3
684       PRINT USING Image12ab;Q(1),Uq(1),Q(2),Uq(2)
686       PRINT USING Image13ab;Bw(1)/1.E+3,Ubw(1)/1.E+3,Bw(2)/1.E+3,Ubw(2)/1.E
 => +3,8
688       PRINT
690       PRINT USING Image15ab;Eps_re,Ueps_re,Eps_im,Ueps_im
692       PRINT USING Image16ab;Tand,Utand
694       RETURN
696 !
698 Image1:IMAGE "Temperature = ",2D.2D," +/- ",D.2D," deg C   Relative Humidity
 =>  = ",2D.D," +/- ",D.D," %"
700 Image2:IMAGE "Barometric Pressure = ",3D.D," +/- ",D.D,"   mbar"
702 Image3:IMAGE "Speed of Light      = ",D.7DESZ," +/- ",D.3DESZ," meters/secon
 => d"
704 Image4:IMAGE "Resonator Diameter              =  ",2D.5D," +/- ",D.5D,"  mm"
706 Image5:IMAGE "         Length (micr. zero) = ",3D.5D," +/- ",D.5D,"  mm"
708 Image6:IMAGE "Sample Description: ",K,"               "
710 Image6a:IMAGE "Initial guess for epsilon':",3D.3D,10X,K
712 Image7:IMAGE "       Length    = ",2D.4D," +/- ",D.4D," mm"
714 Image10ab:IMAGE 28X,"Empty",20X,"with Sample"
716 Image11a:IMAGE "Micrometer reading = ",2(S2D.4D,"   +/- ",D.4D,6X),"mm"
718 Image11b:IMAGE "Resonance frequency= ",2(2D.7D," +/- ",D.7D,3X),"GHz"
720 Image12ab:IMAGE "Q value          =   ",2(5D,"    +/- ",4D,10X)
722 Image13ab:IMAGE "3 dB Bandwidth     =  ",2(5D.2D,"   +/- ",3D.2D,6X),B,"kHz"
724 Image15ab:IMAGE "Epsilon'  = ",3D.4D," +/- ",D.4D,5X,"Epsilon''  = ",2D.5D,"
 =>  +/- ",D.5D
726 Image16ab:IMAGE "Tan delta = ",D.6D," +/- ",D.6D
728    SUBEND
730  !
732  ! ***************************************************************
734  !
736    SUB Calc_dimens
738       OPTION BASE 1
```

```
740       COM /Environment/ Degc,Mbar,Relh,C
742       COM /Environment/ Udegc,Umbar,Urelh,Uc
744       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
746       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
748       COM /Sample/ Erreps(12),Errtand(12)
750       COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
752       COM /Cavity_dims/ Udiameter,Ulength
754       COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
756       COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
758       COM /Constants/ Bessel_root
760       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
 => Prompt$[80]
762       COM /Output/ INTEGER Print_addr,Plotter,Printer_on,Plotter_on
764       COM /Sys/ Sys_id$[10]
766       COM /Sys_msi/ Msi_id$[20]
768       COM /Files/ Diskdrive$[20],Filename$[10]
770       INTEGER Xpos,Ypos,N
772       REAL C0
774       C0=2.99792456E+8    !vacuum speed of light in meters/second
776   !
778       Diameter_temp=Diameter
780       Length0_temp=Length0
782 Calc_dimens:OFF KEY
784       GOSUB Menu_calc_dimen
786       DISP CHR$(129);" This is the menu for resonator dimensions calculatio
 => ns ";CHR$(128)
788       Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
790       ON KEY 0 LABEL "Print  Dimens",Prty GOSUB Print_dimens
792       ON KEY 1 LABEL "Cancel Results",Prty GOSUB Cancel_results
794       ON KEY 2 LABEL "Keep   Results",Prty CALL Prior_menu
796       ON KEY 3 LABEL "Add Del Mode",Prty GOSUB Add_mode
798       ON KEY 13,Prty GOSUB Delete_mode
800       ON KEY 4 LABEL "Chng Tog Mode",Prty GOSUB Change_mode
802       ON KEY 14,Prty GOSUB Toggle_mode
804       ON KEY 6 LABEL "Init   Modes",Prty GOSUB Init_modes
806       ON KEY 7 LABEL "Root/ Micr.",Prty GOSUB Change_root
808       ON KEY 17,Prty GOSUB Micr_reading
810       ON KEY 8 LABEL "Disk   Modes",Prty GOSUB Disk_modes
812       ON KEY 9 LABEL "Find   Dimens",Prty GOSUB Find_dimens
814       Done=0
816       Prior_menu=0
818       LOOP
820           IF Done THEN GOTO Calc_dimens
822       EXIT IF Prior_menu
824       END LOOP
826       OFF KEY
828       Prior_menu=0
830       Done=1
832       SUBEXIT
834   !
836   ! ////////////////////////////////////////////////////////////
838   !
840 Cancel_results:OFF KEY
842       Diameter=Diameter_temp
844       Length0=Length0_temp
846       Prior_menu=1
848       RETURN
850   !
852   ! ////////////////////////////////////////////////////////////
854   !
856 Print_dimens:!
858       IF Printer_on THEN
860           PRINTER IS Print_addr
862           GOSUB Print_dimens2
864           PRINTER IS 1
866       END IF
868       GOSUB Print_dimens2
870       RETURN
872   !
874 Print_dimens2: !
876       PRINT "Filename$:";Filename$;
878       First=1
880       FOR I=1 TO Num_modes
882           IF Spectrum(I,5) THEN
884               IF First THEN
886                   PRINT USING "#,K,DD";"Modes:",Spectrum(I,1)
888                   First=0
890               ELSE
892                   PRINT USING "#,K,DD";",",Spectrum(I,1)
894               END IF
896           END IF
898       NEXT I
900       PRINT
902       PRINT USING "K,2D.4D,K,D.4D,K,3D.4D,K,D.4D";"     Diameter:",Diameter
 => *1.E+3,"+/-",Udiameter*1.E+3,"   Length:",Length0*1.E+3,"+/-",Ulength*1.E+3
904       RETURN
906   !
908   ! ////////////////////////////////////////////////////////////
910   !
912 Menu_calc_dimen: !
914       CLEAR SCREEN
916       PRINT USING Image1;Degc,Udegc,Relh,Urelh
918       PRINT USING Image2;Mbar,Umbar
920       PRINT USING Image3;C,Uc
922       PRINT USING Image4;Diameter*1.E+3,Udiameter*1.E+3
924       PRINT USING Image5;Length0*1.E+3,Ulength*1.E+3
926       PRINT USING Image11a;Micr(2)*1.E+3,Umicr(2)*1.E+3
928       PRINT USING Image7c;Bessel_root
930       PRINT USING Image8c
932 Men_calc_dimen2: !
934       Xpos=0
936       Ypos=8
938       FOR I=1 TO Num_modes
940           Ypos=Ypos+1
942           PRINT TABXY(Xpos,Ypos);
944           IF Spectrum(I,5) THEN PRINT CHR$(129);
946           PRINT USING Image9c;Spectrum(I,1),Spectrum(I,2)/1.E+9
948           PRINT CHR$(128);
950           IF I=9 THEN
952               Xpos=35
954               Ypos=8
956               PRINT TABXY(Xpos,Ypos);
958               PRINT USING Image8c
960           END IF
962       NEXT I
964       RETURN
966 Image1:IMAGE "Temperature = ",2D.2D," +/- ",D.2D," deg C   Relative Humidity
 =>  = ",2D.D," +/- ",D.D," %"
968 Image2:IMAGE "Barometric Pressure = ",3D.D," +/- ",D.D,"   mbar"
970 Image3:IMAGE "Speed of Light      = ",D.7DESZ," +/- ",D.3DESZ," meters/secon
 => d"
972 Image4:IMAGE "Resonator Diameter             =  ",2D.5D," +/- ",D.5D,"  mm"
974 Image5:IMAGE "          Length (micr. zero) = ",3D.5D," +/- ",D.5D,"  mm"
976 Image7c:IMAGE "Bessel function root: ",2D.7D
978 Image8c:IMAGE "Axial mode  Frequency (GHz)"
980 Image9c:IMAGE X,2D,9X,2D.9D
982 Image11a:IMAGE "Micrometer reading = ",2(S2D.4D,"   +/- ",D.4D,6X),"mm"
984   !
986   ! ////////////////////////////////////////////////////////////
988   !
990 Add_mode:OFF KEY
992       Num_modes=Num_modes+1
```

```
994        INPUT "Enter the value of the axial mode number",Spectrum(Num_modes,1
  => )
996        INPUT "Enter the value of the frequency (GHz).",Spectrum(Num_modes,2)
998        Spectrum(Num_modes,2)=Spectrum(Num_modes,2)*1.E+9
1000       INPUT "Enter the value of resonance Q",Spectrum(Num_modes,3)
1002       Spectrum(Num_modes,4)=.05   !Resonance level
1004       INPUT "Will this mode be included in calculations (y/n)",A$
1006       IF A$[1]="y" OR A$[1]="Y" THEN
1008           Spectrum(Num_modes,5)=1
1010       ELSE
1012           Spectrum(Num_modes,5)=0
1014       END IF
1016 !     Sort_modes(Spectrum(*), integer num_modes)
1018       MAT SORT Spectrum(*,1)
1020       Done=1
1022       RETURN
1024 !
1026 ! ////////////////////////////////////////////////////////////
1028 !
1030 Change_mode:OFF KEY
1032       INPUT "Enter the value of the axial mode you wish to change",Edit_num
1034       J=1
1036       WHILE (Spectrum(J,1)<>Edit_num) AND (J<Num_modes)
1038           J=J+1
1040       END WHILE
1042       Test=Spectrum(J,1)
1044       OUTPUT 2 USING "K,#";Test
1046       INPUT "Enter the value of the axial mode number",Spectrum(J,1)
1048       Test=Spectrum(J,2)/1.E+9
1050       OUTPUT 2 USING "K,#";Test
1052       INPUT "Enter the value of the frequency (GHz).",Spectrum(J,2)
1054       Spectrum(J,2)=Spectrum(J,2)*1.E+9
1056       Test=Spectrum(J,3)
1058       OUTPUT 2 USING "K,#";Test
1060       INPUT "Enter the resonance Q value",Spectrum(J,3)
1062       INPUT "Do you want to include this mode in calculations",A$
1064       IF A$[1]="y" OR A$[1]="Y" OR A$="" THEN
1066           Spectrum(J,5)=1
1068       ELSE
1070           Spectrum(J,5)=0
1072       END IF
1074       MAT SORT Spectrum(*,1)
1076 !     CALL Sort_modes(Spectrum(*), integer num_modes)
1078       Done=1
1080       RETURN
1082 !
1084 ! ////////////////////////////////////////////////////////////
1086 !
1088 Toggle_mode:!
1090       INPUT "Enter the mode number you wish to toggle",J
1092       FOR I=1 TO Num_modes
1094           IF Spectrum(I,1)=J THEN
1096               IF Spectrum(I,5) THEN
1098                   Spectrum(I,5)=0
1100               ELSE
1102                   Spectrum(I,5)=1
1104               END IF
1106           END IF
1108       NEXT I
1110       Done=1
1112       RETURN
1114 !
1116 ! ////////////////////////////////////////////////////////////
1118 !
1120 Delete_mode:OFF KEY
1122       INPUT "Enter the value of the axial mode you wish to delete",Del_num
1124       FOR J=1 TO Num_modes
1126           IF Spectrum(J,1)=Del_num THEN
1128               Spectrum(J,1)=1000
1130               Spectrum(J,2)=0
1132               Spectrum(J,3)=0
1134               Spectrum(J,4)=0
1136               Spectrum(J,5)=0
1138               Hit=Hit+1
1140           END IF
1142 !             FOR K=J TO Num_modes
1144 !                 Spectrum(K,1)=Spectrum(K+1,1)
1146 !                 Spectrum(K,2)=Spectrum(K+1,2)
1148 !             NEXT K
1150 !             J=J-1
1152 !             Num_modes=Num_modes-1
1154 !         END IF
1156       NEXT J
1158       Num_modes=Num_modes-1
1160       MAT SORT Spectrum(*,1)
1162       Done=1
1164       RETURN
1166 !
1168 ! ////////////////////////////////////////////////////////////
1170 !
1172 Init_modes:OFF KEY
1174       DISP "Do you really wish to initialize the spectrum modes?"
1176       Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1178       ON KEY 1 LABEL "Yes",Prty GOSUB Continue
1180       ON KEY 2 LABEL "No",Prty CALL Prior_menu
1182       Done=0
1184       Prior_menu=0
1186       LOOP
1188           IF Done THEN Quit
1190       EXIT IF Prior_menu
1192       END LOOP
1194       OFF KEY
1196       Prior_menu=0
1198       Done=1
1200 Quit: !
1202       RETURN
1204 Continue:MAT Spectrum= (0)
1206       FOR I=1 TO SIZE(Spectrum,1)
1208           Spectrum(I,1)=1000
1210       NEXT I
1212       Num_modes=0
1214       Done=1
1216       RETURN
1218 !
1220 ! ////////////////////////////////////////////////////////////
1222 !
1224 Change_root:OFF KEY
1226       DISP "Enter the value of the Bessel function root.";
1228       INPUT Bessel_root
1230       PRINT TABXY(0,7);
1232       PRINT USING Image7c;Bessel_root
1234       Done=1
1236       RETURN
1238 !
1240 ! ////////////////////////////////////////////////////////////
1242 !
1244 Disk_modes:OFF KEY
1246       CALL Select_disk("File with frequency/Q data")
1248       IF Diskdrive$="NO DISK" THEN GOTO Adm
1250       CALL Enterfilename("CAT","File with frequency/Q data")
1252       IF Filename$="" THEN
1254           CAT Diskdrive$
```

```
1256          CALL Enterfilename("ABORT","File with frequency/Q data")
1258       END IF
1260       ALLOCATE A(40,2),Id$[40]
1262       CALL Enter_file(A(*),Num_modes,Id$)
1264       IF A(1,1)>8.23E+9 THEN
1266           Ix=15
1268       ELSE
1270           Ix=14
1272       END IF
1273       Ic=0
1275       FOR I=1 TO Num_modes
1276           IF I>1 THEN
1277               IF PROUND(Spectrum(Ic,2),4)=PROUND(A(I,1),4) THEN GOTO Skip_s
  => pec_fill
1278           END IF
1279           Ic=Ic+1
1280           Spectrum(Ic,1)=Ic+Ix
1281           Spectrum(Ic,2)=A(I,1)
1282           Spectrum(Ic,3)=A(I,2)
1283 !         IF I+Ix=27 THEN
1284 !             Spectrum(I,5)=0
1286 !         ELSE
1288           Spectrum(Ic,5)=1
1290 !         END IF
1291 Skip_spec_fill:!
1293       NEXT I
1294       Num_modes=Ic
1296       DEALLOCATE Id$,A(*)
1297 Adm:  Done=1
1298       RETURN
1300 !
1302 ! ////////////////////////////////////////////////////////////
1304 !
1306 Find_dimens:OFF KEY
1308       J=0
1310       FOR I=1 TO Num_modes
1312           IF Spectrum(I,5) THEN J=J+1
1314       NEXT I
1316       ALLOCATE Spec(1:J,2)
1318       J=0
1320       FOR I=1 TO Num_modes
1322           IF Spectrum(I,5) THEN J=J+1
1324           IF Spectrum(I,5) THEN MAT Spec(J,*)= Spectrum(I,1:2)
1326       NEXT I
1328       CALL Find_dims(Spec(*),Length0,Diameter,Ulength,Udiameter,Micr(2),C,B
  => essel_root,J)
1330       DEALLOCATE Spec(*)
1332       PRINT TABXY(0,4);
1334       PRINT USING Image4;Diameter*1.E+3,Udiameter*1.E+3
1336       PRINT TABXY(0,5);
1338       PRINT USING Image5;Length0*1.E+3,Ulength*1.E+3
1340 !     PRINT TABXY(0,15);B(1,1);B(2,1)
1342 !     PRINT "COVARIANCE MATRIX"
1344 !     PRINT Cvm(1,1),Cvm(1,2)
1346 !     PRINT Cvm(2,1),Cvm(2,2)
1348       Done=1
1350       RETURN
1352 !
1354 ! ////////////////////////////////////////////////////////////
1356 !
1358 Micr_reading:OFF KEY
1360       CALL Edit_data("the micrometer reading of the cavity (mm).",Micr(2),1
  => .E+3,Umicr(2))
1362       PRINT TABXY(0,6);
1364       PRINT USING Image11a;Micr(2)*1.E+3,Umicr(2)*1.E+3
1366       Done=1
1368       RETURN
1370   SUBEND
1372 !
1374 ! ************************************************************
1376 !
1378   SUB Calc_c
1380 !
1382 ! Equations from H. Liebe in "An Updated Model for Millimeter Wave
1384 ! Propagation in Moist Air
1386 ! RADIO SCIENCE, Vol. 20, Num 5, pp.1069-1089, Sept/Oct 1985
1388 !
1390 Calc_c:OFF KEY
1392       COM /Environment/ Degc,Mbar,Relh,C
1394       COM /Environment/ Udegc,Umbar,Urelh,Uc
1396       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
1398   !
1400       GOSUB Menu_calc_c
1402   !
1404       Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1406       ON KEY 1 LABEL "Prior  Menu",Prty CALL Prior_menu
1408       ON KEY 2 LABEL "Change Temp",Prty CALL Temp
1410       ON KEY 3 LABEL "Change Rel Hum",Prty CALL Rel_hum
1412       ON KEY 4 LABEL "Change Presure",Prty CALL Pressure
1414       Prior_menu=0
1416       Done=0
1418       LOOP
1420           IF Done THEN GOTO Calc_c
1422       EXIT IF Prior_menu
1424       END LOOP
1426       Prior_menu=0
1428       Done=1
1430       SUBEXIT
1432 !
1434 Menu_calc_c:!
1436       CLEAR SCREEN
1438       PRINT USING Image1;Degc,Udegc,Relh,Urelh
1440       PRINT USING Image2;Mbar,Umbar
1442       PRINT USING Image3;C,Uc
1444       DISP "Menu for speed of light calculation"
1446       RETURN
1448 Image1:IMAGE "Temperature = ",2D.2D," +/- ",D.2D," deg C   Relative Humidit
  => y = ",2D.D," +/- ",D.D," %"
1450 Image2:IMAGE "Barometric Pressure = ",3D.D," +/- ",D.D,"   mbar"
1452 Image3:IMAGE "Speed of Light      = ",D.7DESZ," +/- ",D.3DESZ," meters/seco
  => nd"
1454   SUBEND
1456 !
1458 ! ************************************************************
1460 !
1462   SUB Calc_modes
1464 Calc_modes:OFF KEY
1466   !
1468       OPTION BASE 1
1470       COM /Environment/ Degc,Mbar,Relh,C
1472       COM /Environment/ Udegc,Umbar,Urelh,Uc
1474       COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
1476       COM /Cavity_dims/ Udiameter,Ulength
1478       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
1480   !
1482       LOADSUB Modefreqs2 FROM "MODEFREQS2"
1484       CALL Modefreqs2(C,Diameter/2,Length0)
1486       DELSUB Modefreqs2
1488       Prior_menu=0
1490       Done=1
```

```
1492   SUBEND
1494 !
1496 ! ****************************************************************
1498 !
1500   SUB Calc_freqs
1502 Calc_freqs:OFF KEY
1504   !
1506       OPTION BASE 1
1508       COM /Environment/ Degc,Mbar,Relh,C
1510       COM /Environment/ Udegc,Umbar,Urelh,Uc
1512       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
1514       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
1516       COM /Sample/ Erreps(12),Errtand(12)
1518       COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
1520       COM /Cavity_dims/ Udiameter,Ulength
1522       COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
1524       COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
1526       COM /Constants/ Bessel_root
1528       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
1530   !
1532       GOSUB Menu_calc_freqs
1534   !
1536       Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1538       ON KEY 1 LABEL "Prior  Menu",Prty CALL Prior_menu
1540       ON KEY 2 LABEL "Sample Thickns",Prty CALL Sample_thickns
1542       ON KEY 3 LABEL "Sample Epsilon",Prty CALL Epsilon
1544       ON KEY 4 LABEL "Cavity Diameter",Prty CALL Diameter
1546       ON KEY 5 LABEL "Cavity Length ",Prty CALL Length
1548       ON KEY 6 LABEL "Micr.  Reading",Prty CALL Micr_readings
1550       ON KEY 7 LABEL "Freq.  Empty",Prty CALL Freq_empty
1552       ON KEY 8 LABEL "Freq.  Loaded",Prty CALL Freq_loaded
1554       Prior_menu=0
1556       Done=0
1558       LOOP
1560           IF Done THEN GOTO Calc_freqs
1562       EXIT IF Prior_menu
1564       END LOOP
1566       OFF KEY
1568       Prior_menu=0
1570       Done=1
1572       SUBEXIT
1574 !
1576 ! ////////////////////////////////////////////////////////////////
1578 !
1580 Menu_calc_freqs: !
1582       CLEAR SCREEN
1584       PRINT USING Image6;S_id$
1586       PRINT USING Image7;Samplelen*1.E+3,Usamplelen*1.E+3
1588       PRINT USING Image17;Eps_re,Ueps_re
1590       PRINT
1592       PRINT USING Image1;Degc,Udegc,Relh,Urelh
1594       PRINT USING Image2;Mbar,Umbar
1596       PRINT USING Image3;C,Uc
1598       PRINT
1600       PRINT USING Image4;Diameter*1.E+3,Udiameter*1.E+3
1602       PRINT USING Image5;Length0*1.E+3,Ulength*1.E+3
1604       PRINT USING Image5a;(Length0+Micr(1))*1.E+3,Ulength
1606       PRINT
1608       PRINT USING Image10ab
1610       PRINT USING Image11a;Micr(1)*1.E+3,Umicr(1)*1.E+3,Micr(2)*1.E+3,Umicr
  => (2)*1.E+3
1612       PRINT
1614       RETURN
1616 Image1:IMAGE "Temperature = ",2D.2D," +/- ",D.2D," deg C   Relative Humidit
  => y = ",2D.D," +/- ",D.D," %"
1618 Image2:IMAGE "Barometric Pressure = ",3D.D," +/- ",D.D,"   mbar"
1620 Image3:IMAGE "Speed of Light      = ",D.7DESZ," +/- ",D.3DESZ," meters/seco
  => nd"
1622 Image4:IMAGE "Resonator Diameter            =  ",2D.5D," +/- ",D.5D,"  mm"
1624 Image5:IMAGE "          Length (micr. zero) = ",3D.5D," +/- ",D.5D,"  mm"
1626 Image5a:IMAGE"          Length (total)      = ",3D.5D," +/- ",D.5D,"  mm"
1628 Image6:IMAGE "Sample Description: ",K,"                   "
1630 Image7:IMAGE "       Length    = ",2D.4D," +/- ",D.4D," mm"
1632 Image10ab:IMAGE 28X,"Empty",20X,"with Sample"
1634 Image11a:IMAGE "Micrometer reading = ",2(3D.4D,"   +/- ",D.4D,6X),"mm"
1636 Image17:IMAGE "       Epsilon'  =",3D.4D," +/- ",D.4D
1638   SUBEND
1640 !
1642 ! ****************************************************************
1644 !
1646   SUB Sample_descr
1648 Sample_descr:OFF KEY
1650   !
1652       OPTION BASE 1
1654       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
1656       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
1658       COM /Sample/ Erreps(12),Errtand(12)
1660       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
1662   !
1664       LOADSUB Enter_id FROM "BAS_SUBS:,1400,0,3"
1666       Enter_id(Sample_id$,"Sample")
1668       DELSUB Enter_id
1670       Done=1
1672   SUBEND
1674 !
1676 ! ****************************************************************
1678 !
1680   SUB Sample_thickns
1682 Sample_thickns:OFF KEY
1684   !
1686       OPTION BASE 1
1688       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
1690       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
1692       COM /Sample/ Erreps(12),Errtand(12)
1694       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
1696   !
1698       CALL Edit_data("the sample length (mm).",Samplelen,1.E+3,Usamplelen)
1700       Done=1
1702   SUBEND
1704 !
1706 ! ****************************************************************
1708 !
1710   SUB Resnce_freq
1712 Resnce_freq:OFF KEY
1714   !
1716       OPTION BASE 1
1718       COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
1720       COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
1722       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
1724   !
1726       CALL Edit_data("the resonance frequency (GHz)",Freq(2),1.E-9)
1728       CALL Edit_data("the uncertainty in the resonance frequency (kHz)",Ufr
  => eq(2),1.E-3)
1730       Freq(1)=Freq(2)
1732       Ufreq(1)=Ufreq(2)
1734       IF Bw(1)>0 THEN
1736           Q(1)=Freq(1)/Bw(1)
1738           Uq(1)=Q(1)-Freq(1)/(Bw(1)+Ubw(1))
```

```
1740        END IF
1742        IF Bw(2)>0 THEN
1744            Q(2)=Freq(2)/Bw(2)
1746            Uq(2)=Q(2)-Freq(2)/(Bw(2)+Ubw(2))
1748        END IF
1750        Done=1
1752    SUBEND
1754 !
1756 ! ****************************************************************
1758 !
1760    SUB Micr_readings
1762 Micr_readings:OFF KEY
1764    !
1766        OPTION BASE 1
1768        COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
1770        COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
1772        COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
1774    !
1776        Prompt$="Do you want to change the EMPTY or LOADED cavity micrometer
  => reading?"
1778        CALL Empty_or_loaded
1780        IF Prior_menu THEN
1782            Prior_menu=0
1784            Done=T
1786            SUBEXIT
1788        END IF
1790 !
1792        IF Empty_cavity THEN
1794            CALL Edit_data("the micrometer reading for the empty cavity (mm).
  => ",Micr(1),1.E+3,Umicr(1))
1796        ELSE
1798            CALL Edit_data("the micrometer reading for the loaded cavity (mm)
  => .",Micr(2),1.E+3,Umicr(2))
1800        END IF
1802        GOTO Micr_readings
1804    SUBEND
1806 !
1808 ! ****************************************************************
1810 !
1812    SUB Q_values
1814 Q_values:OFF KEY
1816    !
1818        OPTION BASE 1
1820        COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
1822        COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
1824        COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
1826    !
1828        Prompt$="Do you want to change the EMPTY or LOADED cavity Q-value?"
1830        CALL Empty_or_loaded
1832        IF Prior_menu THEN
1834            Prior_menu=0
1836            Done=T
1838            SUBEXIT
1840        END IF
1842        IF Empty_cavity THEN
1844 !          CALL Edit_data("Q for the empty cavity.",Q(1),1,Uq(1))
1845            CALL Edit_data("Q for the empty cavity.",Q(1),1)
1846            Uq(1)=Q(1)/100
1848            Bw(1)=Freq(1)/Q(1)
1849            Ubw(1)=Bw(1)-Freq(1)/(Q(1)+Uq(1))
1850        ELSE
1852 !          CALL Edit_data("Q for the loaded cavity.",Q(2),1,Uq(2))
1853            CALL Edit_data("Q for the loaded cavity.",Q(2),1)
1854            Uq(2)=Q(2)/100
1856            Bw(2)=Freq(2)/Q(2)
1857            Ubw(2)=Bw(2)-Freq(2)/(Q(2)+Uq(2))
1858        END IF
1860        GOTO Q_values
1862    SUBEND
1864 !
1866 ! ****************************************************************
1868 !
1870    SUB Band_widths
1872 Band_widths:OFF KEY
1874    !
1876        OPTION BASE 1
1878        COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
1880        COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
1882        COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
1884    !
1886        Prompt$="Do you want to change the bandwidth of the EMPTY or LOADED c
  => avity?"
1888        CALL Empty_or_loaded
1890        IF Prior_menu THEN
1892            Prior_menu=0
1894            Done=T
1896            SUBEXIT
1898        END IF
1900 !
1902        IF Empty_cavity THEN
1904            CALL Edit_data("of the 3dB bandwidth for the empty cavity (kHz)."
  => ,Bw(1),1.E-3,Ubw(1))
1906            Q(1)=Freq(1)/Bw(1)
1908            Uq(1)=ABS(Q(1)-Freq(1)/(Bw(1)+Ubw(1)))
1910        ELSE
1912            CALL Edit_data("of the 3dB bandwidth for the loaded cavity (kHz).
  => ",Bw(2),1.E-3,Ubw(2))
1914            Q(2)=Freq(2)/Bw(2)
1916            Uq(2)=ABS(Q(2)-Freq(2)/(Bw(2)+Ubw(2)))
1918        END IF
1920        GOTO Band_widths
1922    SUBEND
1924 !
1926 ! ****************************************************************
1928 !
1930    SUB Resnce_freqs
1932    !
1934        OPTION BASE 1
1936        COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
1938        COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
1940        COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
1942    !
1944 Resnce_freqs:OFF KEY
1946        Prompt$="Do you want to change the EMPTY or LOADED cavity resonance f
  => requency?"
1948        CALL Empty_or_loaded
1950        IF Prior_menu THEN
1952            Prior_menu=0
1954            Done=T
1956            SUBEXIT
1958        END IF
1960        IF Empty_cavity THEN
1962            CALL Edit_data("the empty cavity resonance frequency (GHz).",Freq
  => (1),1.E-9)
1964            CALL Edit_data("the uncertainty in the empty cavity res. freq. (k
  => Hz)",Ufreq(1),1.E-3)
1966            IF Bw(1)>0 THEN
1968                Q(1)=Freq(1)/Bw(1)
```

```
            Uq(1)=Q(1)-Freq(1)/(Bw(1)+Ubw(1))
          END IF
      ELSE
          CALL Edit_data("the loaded cavity resonance frequency (GHz).",Freq(2),1.E-9)
          CALL Edit_data("the uncertainty in the loaded cavity res. freq. (kHz).",Ufreq(2),1.E-3)
          IF Bw(2)>0 THEN
              Q(2)=Freq(2)/Bw(2)
              Uq(2)=Q(2)-Freq(2)/(Bw(2)+Ubw(2))
          END IF
      END IF
      GOTO Resnce_freqs
  SUBEND
!
! ***************************************************************
!
  SUB Micr_reading
Micr_reading:OFF KEY
  !
      OPTION BASE 1
      COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
      COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
      COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,Prompt$[80]
  !
      CALL Edit_data("the micrometer reading (mm).",Micr(1),1.E+3,Umicr(1))
      Micr(2)=Micr(1)
      Umicr(2)=Umicr(1)
      Done=1
  SUBEND
!
! ***************************************************************
!
  SUB Calc_eps_re
!
! Equations from F. Horner, T.A. Taylor, R. Dunsmuir in "Resonance
! Methods of Dielectric Measurement at Centimetre Wavelengths"
! J. IEE, 1946, 93, Pt.III, pp.53-68
!
Calc_eps_re:OFF KEY
  !
      RAD
      OPTION BASE 1
      COM /Environment/ Degc,Mbar,Relh,C
      COM /Environment/ Udegc,Umbar,Urelh,Uc
      COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
      COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
      COM /Sample/ Erreps(12),Errtand(12)
      COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
      COM /Cavity_dims/ Udiameter,Ulength
      COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
      COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
      COM /Constants/ Bessel_root
  !
      Y=Length0+Micr(2)-Samplelen
      Mu=(4*PI)*10^(-7)
      Eps_air=1/(Mu*(C^2))
      Eps0=8.854188E-12
      W=2*PI*Freq(2)
      K=Bessel_root/(Diameter/2)
      Eps_re=Eps_re_guess
      B0=SQRT(W^2*Mu*Eps_air-K^2)
      B1=SQRT(W^2*Mu*Eps_re*Eps0-K^2)
      REPEAT
          I=I+1
          Funct_b1=TAN(B1*Samplelen)/B1+TAN(B0*Y)/B0
          Funct_b1_p=(B1*Samplelen*(1/COS(B1*Samplelen)^2)-TAN(B1*Samplelen))/(B1^2)
          Deltab1=Funct_b1/Funct_b1_p
          IF ABS(Deltab1)<1.E-2 THEN Deltab1=Deltab1/10
          B1=B1-Deltab1
!         PRINT Deltab1
      UNTIL (I=100) OR (ABS(Deltab1)<1.E-4)
      Eps_re=(B1^2+K^2)/(B0^2+K^2)
!     PRINT Eps_re
!     PAUSE
      IF I=100 THEN Eps_re=0
      DEG
  SUBEND
 !
 ! ***************************************************************
 !
  SUB Calc_tand
!
! Equations from E. Ni and U. Stumper "Permittivity Measurements Using a
! Frequency-tuned Microwave TE01 Cavity Resonator"
! IEE PROCEEDINGS, Vol. 132, Pt. H, No. 1, February 1985 p27-32.
!
Calc_tand:OFF KEY
  !
      RAD
      OPTION BASE 1
      COM /Environment/ Degc,Mbar,Relh,C
      COM /Environment/ Udegc,Umbar,Urelh,Uc
      COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
      COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
      COM /Sample/ Erreps(12),Errtand(12)
      COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
      COM /Cavity_dims/ Udiameter,Ulength
      COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
      COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
      COM /Constants/ Bessel_root
      COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,Prompt$[80]
  !
      IF Do_var_freq THEN      !Variable frequency/ fixed length
          L0=Length0+Micr(1)
      ELSE                     !Variable length/ fixed frequency
          L0=Length0+Micr(2)
      END IF
      L=Length0+Micr(2)
      A=Diameter/2
      Fc=(Bessel_root*C)/(PI*Diameter)
      Q0=Fc/Freq(2)
      B0=2*PI*Freq(2)*SQRT(1-Q0^2)/C
      Be=2*PI*Freq(2)*SQRT(Eps_re-Q0^2)/C
      B=(Be/B0)^2
      X=Be*Samplelen
      P=(SIN(X)^2)+(B*(COS(X)^2))
      G=(2*Be*A*((Eps_re-Q0^2)+(1-Q0^2)*P))/(2*X-SIN(2*X))
      F=(Q0^2*(2*X+(B-1)*SIN(X)+(2*Be*L-2*X)*P))/(2*X-SIN(2*X))
      E=Eps_re*(Q0^2+2*A*(1-Q0^2)/L0)
      D=(F+G)/E
      A=(2*X*Eps_re+(B-Eps_re)*SIN(2*X)+(2*Be*L-2*X)*P)/(Eps_re*(2*X-SIN(2*X)))
      Tand=(A*Bw(2)-D*Bw(1))/Freq(2)
      DEG
  SUBEND
!
! ***************************************************************
!
```

```
2222   SUB Calc_eps_im
2224 Calc_eps_im:OFF KEY
2226   !
2228       RAD
2230       OPTION BASE 1
2232       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
2234       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
2236       COM /Sample/ Erreps(12),Errtand(12)
2238   !
2240       Eps_im=Tand*Eps_re
2242       DEG
2244   SUBEND
2246 !
2248 ! ************************************************************
2250 !
2252   SUB Edit_data(Prompt$,Variable,OPTIONAL Multiplier,Uvariable)
2254 Edit_data:OFF KEY
2256       IF NPAR>2 THEN
2258           Test=Variable*Multiplier
2260           IF NPAR=4 THEN Uvariable=Uvariable*Multiplier
2262       ELSE
2264           Test=Variable
2266       END IF
2268       ON ERROR GOTO Test_again
2270 Test_again:!
2272       OUTPUT 2 USING "K,#";Test
2274       DISP "Enter the value of ";Prompt$;
2276       INPUT Variable
2278       OFF ERROR
2280       IF NPAR=4 THEN
2282           Utest=Uvariable
2284           ON ERROR GOTO Utest_again
2286 Utest_again:!
2288           OUTPUT 2 USING "K,#";Utest
2290           DISP "Enter the uncertainty in ";Prompt$;
2292           INPUT Uvariable
2294           OFF ERROR
2296       END IF
2298       IF NPAR>2 THEN
2300           Variable=Variable/Multiplier
2302           IF NPAR=4 THEN Uvariable=Uvariable/Multiplier
2304       END IF
2306   SUBEND
2308  !
2310  ! ************************************************************
2312  !
2314   SUB Speed_of_light
2316 Speed_of_light: !
2318   !
2320       COM /Environment/ Degc,Mbar,Relh,C
2322       COM /Environment/ Udegc,Umbar,Urelh,Uc
2324   !
2326       Theta=300/(273.15+Degc)
2328       Esaturated=Theta^5/(.4151*10^(9.834*Theta-10))
2330       E=Relh*Esaturated/100
2332       Kpa=Mbar/10
2334       Pdryair=Kpa-E
2336       N=1+((((41.6*Theta)+2.39)*E*Theta)+(2.588*Pdryair*Theta))*10^(-6)
2338       C=2.99792458*10^8/N
2340   SUBEND
2342 !
2344 ! ************************************************************
2346 !
2348   SUB Temp
2350 Temp:   !
2352   !
2354       COM /Environment/ Degc,Mbar,Relh,C
2356       COM /Environment/ Udegc,Umbar,Urelh,Uc
2358       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
2360   !
2362       CALL Edit_data("the temperature in the lab (C).",Degc,1,Udegc)
2364       CALL Speed_of_light
2366       CALL Uncert_c
2368       Done=1
2370   SUBEND
2372 !
2374 ! ************************************************************
2376 !
2378   SUB Rel_hum
2380 Rel_hum: !
2382   !
2384       COM /Environment/ Degc,Mbar,Relh,C
2386       COM /Environment/ Udegc,Umbar,Urelh,Uc
2388       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
2390   !
2392       CALL Edit_data("the relative humidity in the lab. (%)",Relh,1,Urelh)
2394       CALL Speed_of_light
2396       CALL Uncert_c
2398       Done=1
2400   SUBEND
2402 !
2404 ! ************************************************************
2406 !
2408   SUB Pressure
2410 Pressure: !
2412   !
2414       COM /Environment/ Degc,Mbar,Relh,C
2416       COM /Environment/ Udegc,Umbar,Urelh,Uc
2418       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
2420   !
2422       CALL Edit_data("the pressure in the lab (millibars).",Mbar,1,Umbar)
2424       CALL Speed_of_light
2426       CALL Uncert_c
2428       Done=1
2430   SUBEND
2432 !
2434 ! ************************************************************
2436 !
2438   SUB Uncert_c
2440 Uncert_c: !
2442   !
2444       COM /Environment/ Degc,Mbar,Relh,C
2446       COM /Environment/ Udegc,Umbar,Urelh,Uc
2448   !
2450       C0=2.99792458*10^(8)
2452       T=Degc+273.15
2454       P=Mbar/10
2456       H=Relh
2458       Ut=Udegc
2460       Up=Umbar/10
2462       Uh=Urelh
2464       A=(300/T)^7
2466       B=(300/T)^6
2468       D=10^((9.834*300/T)-10)
2470       N=1+10^(-6)*((41.6*A*H+2.39*B*H)/(41.51*D)+2.588*P*(300/T)-((2.588*B*
  => H)/(41.51*D)))
2472       Temp1=41.51*D*9.834*300*LOG(10)*(-1/(T^2))*(41.6*A*H-.198*B*H)
2474       Temp2=41.51*D*(41.6*(300^7)*(-7)*(T^(-8))*H-.198*(300^6)*(-6)*(T^(-7)
  => )*H)
```

```
2476       Temp3=2.588*P*300*(-1/(T^2))
2478       Par_n_res_t=(((Temp1-Temp2)/((41.51*D)^2))+Temp3)*10^(-6)
2480       Par_n_res_p=(300/T)*2.588*10^(-6)
2482       Par_n_res_h=((41.6*A-.198*B)/(41.51*D))*10^(-6)
2484       Uncert_n=SQRT(((Par_n_res_t*Ut)^2)+((Par_n_res_p*Up)^2)+((Par_n_res_h
   => *Uh)^2))
2486       Uc=C0*(1/(N^2))*Uncert_n
2488   SUBEND
2490 !
2492 ! ****************************************************************
2494 !
2496   SUB Freq_empty
2498 Freq_empty:OFF KEY
2500   !
2502       OPTION BASE 1
2504       COM /Environment/ Degc,Mbar,Relh,C
2506       COM /Environment/ Udegc,Umbar,Urelh,Uc
2508       COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
2510       COM /Cavity_dims/ Udiameter,Ulength
2512       COM /Constants/ Bessel_root
2514       COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
2516       COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
2518       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
   => Prompt$[80]
2520   !
2522       Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
2524       ON KEY 1 LABEL "Prior  Menu",Prty CALL Prior_menu
2526       ON KEY 2 LABEL "Start  Freq",Prty GOSUB Start_freq
2528       ON KEY 3 LABEL "Stop   Freq",Prty GOSUB Stop_freq
2530       ON KEY 4 LABEL "Calc   Modes",Prty GOSUB Calc_freq_empty
2532       Prior_menu=0
2534       Done=0
2536       LOOP
2538           IF Done THEN GOTO Freq_empty
2540       EXIT IF Prior_menu
2542       END LOOP
2544       OFF KEY
2546       Prior_menu=0
2548       Done=T
2550       SUBEXIT
2552 !
2554 Start_freq: !
2556       Edit_data("the value of the starting frequency",Start_freq,1.E-9)
2558       Done=1
2560       RETURN
2562 !
2564 Stop_freq: !
2566       Edit_data("the value of the stopping frequency",Stop_freq,1.E-9)
2568       Done=1
2570       RETURN
2572 !
2574 Calc_freq_empty: !
2576       PRINT TABXY(0,16);
2578 Image11b:IMAGE "TE01(",2D,") mode occurs at ",2D.7D,"GHz"
2580       B=Bessel_root/(PI*Diameter)
2582       FOR My=1 TO 50
2584           E=My/(2*(Length0+Micr(1)))
2586           E=E^2+B^2
2588           E=SQR(E)
2590           Frequency=C*E
2592           IF Frequency>Stop_freq THEN GOTO Resend
2594           IF Frequency<Start_freq THEN GOTO Respaths
2596           PRINT USING Image11b;My,Frequency/1.E+9
2598 Respaths: !
2600       NEXT My
2602 Resend:   !
2604       Done=1
2606       RETURN
2608   SUBEND
2610 !
2612 ! ****************************************************************
2614 !
2616   SUB Freq_loaded
2618 Freq_loaded:OFF KEY
2620   !
2622       RAD
2624       OPTION BASE 1
2626       COM /Environment/ Degc,Mbar,Relh,C
2628       COM /Environment/ Udegc,Umbar,Urelh,Uc
2630       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
2632       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
2634       COM /Sample/ Erreps(12),Errtand(12)
2636       COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
2638       COM /Cavity_dims/ Udiameter,Ulength
2640       COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
2642       COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
2644       COM /Constants/ Bessel_root
2646   !
2648       Edit_data("the initial guess for the resonance frequency (GHz).",Freq
   => (2),1.E-9)
2650       B=Samplelen
2652       Y=Length0+Micr(2)-B
2654       Mu=(4*PI)*10^(-7)
2656       Eps0=1/(Mu*(C^2))
2658       K=Bessel_root/(Diameter/2)
2660       REPEAT
2662           I=I+1
2664           W=2*PI*Freq(2)
2666           B0=SQRT(W^2*Mu*Eps0-K^2)
2668           B1=SQRT(W^2*Mu*Eps_re*Eps0-K^2)
2670           Fun_f=TAN(B1*B)/B1+TAN(B0*Y)/B0
2672           Fun_f_p=4*PI^2*Mu*Eps0*Freq(2)*(Eps_re*(B*SQR(B1)*(1/COS(B*B1)^2)
   => -TAN(B*B1)*(1/SQR(B1)))/(B1^2)+(Y*SQR(B0)*(1/COS(Y*B0)^2)-TAN(Y*B0)*(1/SQR(
   => B0)))/(B0^2))
2674           Freq(2)=Freq(2)-(Fun_f/Fun_f_p)
2676       UNTIL (I=100000) OR (ABS(Fun_f/Fun_f_p)<1.E-5)
2678       W=2*PI*Freq(2)
2680       B0=SQRT(W^2*Mu*Eps0-K^2)
2682       B1=SQRT(W^2*Mu*Eps_re*Eps0-K^2)
2684       Halfwavelens1=B1*B/PI
2686       Halfwavelens0=B0*Y/PI
2688       Halfwavelens=Halfwavelens0+Halfwavelens1
2690 Image11b:IMAGE "Resonant mode ",2D.2D," occurs at ",2D.7D," +/- ",D.7D,3X,"
   => GHz"
2692       PRINT TABXY(0,16);
2694       PRINT USING Image11b;Halfwavelens,Freq(2)/1.E+9,Ufreq(2)/1.E+9
2696       PRINT "                                                              "
2698       PRINT "                                                              "
2700       PRINT "                                                              "
2702       DEG
2704   SUBEND
2706 !
2708 ! ****************************************************************
2710 !
2712   SUB Epsilon
2714 Epsilon:OFF KEY
2716   !
2718       OPTION BASE 1
2720       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
2722       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
2724       COM /Sample/ Erreps(12),Errtand(12)
2726       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
```

```
  => Prompt$[80]
2728   !
2730       CALL Edit_data("the relative epsilon of the sample.",Eps_re,1)
2732       Done=1
2734   SUBEND
2736 !
2738 ! ****************************************************************
2740 !
2742   SUB Diameter
2744 Diameter:OFF KEY
2746   !
2748       OPTION BASE 1
2750       COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
2752       COM /Cavity_dims/ Udiameter,Ulength
2754       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
2756   !
2758       CALL Edit_data("the diameter of the cavity (mm).",Diameter,1.E+3,Udia
  => meter)
2760       Done=1
2762   SUBEND
2764 !
2766 ! ****************************************************************
2768 !
2770   SUB Length
2772 Length:OFF KEY
2774   !
2776       OPTION BASE 1
2778       COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
2780       COM /Cavity_dims/ Udiameter,Ulength
2782       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
2784   !
2786       CALL Edit_data("the length of the zeroed cavity (mm).",Length0,1.E+3,
  => Ulength)
2788       Done=1
2790   SUBEND
2792 !
2794 ! ****************************************************************
2796 !
2798   SUB Sort_modes(Array(*),N)
2800       IF N=1 THEN SUBEXIT
2802       FOR J=2 TO N
2804           Dummy1=Array(J,1)
2806           Dummy2=Array(J,2)
2808           FOR I=J-1 TO 1 STEP -1
2810               IF Array(I,1)<=Dummy1 THEN GOTO Mark
2812               Array(I+1,1)=Array(I,1)
2814               Array(I+1,2)=Array(I,2)
2816           NEXT I
2818           I=0
2820 Mark:  !
2822           Array(I+1,1)=Dummy1
2824           Array(I+1,2)=Dummy2
2826       NEXT J
2828   SUBEND
2830 !
2832 ! ****************************************************************
2834 !
2836   SUB Empty_or_loaded
2838 Empty_or_loaded:OFF KEY
2840   !
2842       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
2844   !
2846       DISP Prompt$
2848       Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
2850       ON KEY 1 LABEL "PRIOR  MENU",Prty CALL Prior_menu
2852       ON KEY 2 LABEL "EMPTY  CAVITY",Prty GOSUB Empty
2854       ON KEY 3 LABEL "LOADED CAVITY",Prty GOSUB Loaded
2856       Done=0
2858       Prior_menu=0
2860       LOOP
2862       EXIT IF Prior_menu OR Done
2864       END LOOP
2866       SUBEXIT
2868 !
2870 ! ////////////////////////////////////////////////////////////
2872 !
2874 Empty:OFF KEY
2876       Empty_cavity=1
2878       Done=1
2880       RETURN
2882 !
2884 ! ////////////////////////////////////////////////////////////
2886 !
2888 Loaded:OFF KEY
2890       Empty_cavity=0
2892       Done=1
2894       RETURN
2896   SUBEND
2898 !
2900 ! ****************************************************************
2902 !
2904   SUB Prior_menu
2906       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
2908       Prior_menu=1
2910   SUBEND
2912 !
2914 ! ****************************************************************
2916 !
2918   SUB Calc_errors(OPTIONAL INTEGER Dont_print)
2920 !
2922 !
2924 Calc_errors:OFF KEY
2926   !
2928       OPTION BASE 1
2930       COM /Environment/ Degc,Mbar,Relh,C
2932       COM /Environment/ Udegc,Umbar,Urelh,Uc
2934       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
2936       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
2938       COM /Sample/ Erreps(12),Errtand(12)
2940       COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
2942       COM /Cavity_dims/ Udiameter,Ulength
2944       COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
2946       COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
2948       COM /Constants/ Bessel_root
2950       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
2951       Print_addr=9
2953 ! !
2954       DIM Eps(12),Tandel(12)
2956       Diameter=Diameter+Udiameter
2958       CALL Calc_eps_re
2960       Eps(1)=Eps_re
2962       Diameter=Diameter-Udiameter
2964   !
2966       Length0=Length0+Ulength
2968       CALL Calc_eps_re
2970       Eps(2)=Eps_re
2972       Length0=Length0-Ulength
```

```
2974  !
2976      Samplelen=Samplelen+Usamplelen
2978      CALL Calc_eps_re
2980      Eps(3)=Eps_re
2982      Samplelen=Samplelen-Usamplelen
2984  !
2986      Degc=Degc+Udegc
2988      CALL Speed_of_light
2990      CALL Calc_eps_re
2992      Degc=Degc-Udegc
2994      Eps(4)=Eps_re
2996  !
2998      Relh=Relh+Urelh
3000      CALL Speed_of_light
3002      CALL Calc_eps_re
3004      Relh=Relh-Urelh
3006      Eps(5)=Eps_re
3008  !
3010      Mbar=Mbar+Umbar
3012      CALL Speed_of_light
3014      CALL Calc_eps_re
3016      Mbar=Mbar-Umbar
3018      Eps(6)=Eps_re
3020  !
3022      CALL Speed_of_light
3024      C=C+Uc
3026      CALL Calc_eps_re
3028      C=C-Uc
3030      Eps(7)=Eps_re
3032  !
3034      Freq(2)=Freq(2)+Ufreq(2)
3036      CALL Calc_eps_re
3038      Eps(8)=Eps_re
3040      Freq(2)=Freq(2)-Ufreq(2)
3042  !
3044      Micr(1)=Micr(1)+Umicr(1)
3046      CALL Calc_eps_re
3048      Eps(9)=Eps_re
3050      Micr(1)=Micr(1)-Umicr(1)
3052  !
3054      Micr(2)=Micr(2)+Umicr(2)
3056      CALL Calc_eps_re
3058      Eps(10)=Eps_re
3060      Micr(2)=Micr(2)-Umicr(2)
3062  !
3064      CALL Calc_eps_re
3066      Eps(11)=Eps_re
3068      Eps(12)=Eps_re
3070  !
3072      Diameter=Diameter+Udiameter
3074      CALL Calc_tand
3076      Diameter=Diameter-Udiameter
3078      Tandel(1)=Tand
3080  !
3082      Length0=Length0+Ulength
3084      CALL Calc_tand
3086      Length0=Length0-Ulength
3088      Tandel(2)=Tand
3090  !
3092      Samplelen=Samplelen+Usamplelen
3094      CALL Calc_tand
3096      Samplelen=Samplelen-Usamplelen
3098      Tandel(3)=Tand
3100  !
3102      Degc=Degc+Udegc
3104      CALL Speed_of_light
3106      CALL Calc_tand
3108      Degc=Degc-Udegc
3110      Tandel(4)=Tand
3112  !
3114      Relh=Relh+Urelh
3116      CALL Speed_of_light
3118      CALL Calc_tand
3120      Relh=Relh-Urelh
3122      Tandel(5)=Tand
3124  !
3126      Mbar=Mbar+Umbar
3128      CALL Speed_of_light
3130      CALL Calc_tand
3132      Mbar=Mbar-Umbar
3134      Tandel(6)=Tand
3136  !
3138      CALL Speed_of_light
3140      C=C+Uc
3142      CALL Calc_tand
3144      C=C-Uc
3146      Tandel(7)=Tand
3148  !
3150      Freq(2)=Freq(2)+Ufreq(2)
3152      CALL Calc_tand
3154      Freq(2)=Freq(2)-Ufreq(2)
3156      Tandel(8)=Tand
3158  !
3160      Micr(1)=Micr(1)+Umicr(1)
3162      CALL Calc_tand
3164      Micr(1)=Micr(1)-Umicr(1)
3166      Tandel(9)=Tand
3168  !
3170      Micr(2)=Micr(2)+Umicr(2)
3172      CALL Calc_tand
3174      Micr(2)=Micr(2)-Umicr(2)
3176      Tandel(10)=Tand
3178  !
3180      Bw(1)=Bw(1)+Ubw(1)
3182      CALL Calc_tand
3184      Bw(1)=Bw(1)-Ubw(1)
3186      Tandel(11)=Tand
3188  !
3190      Bw(2)=Bw(2)+Ubw(2)
3192      CALL Calc_tand
3194      Bw(2)=Bw(2)-Ubw(2)
3196      Tandel(12)=Tand
3198  !
3200      CALL Calc_tand
3202  !
3204      MAT Erreps= Eps
3206      MAT Errtand= Tandel
3208      FOR I=1 TO 12
3210          Erreps(I)=Erreps(I)-Eps_re
3212          Errtand(I)=Errtand(I)-Tand
3214      NEXT I
3216      Ueps_re=SQRT(Erreps(1)^2+Erreps(2)^2+Erreps(3)^2+Erreps(7)^2+Erreps(8
 => )^2+Erreps(9)^2+Erreps(10)^2)
3218      Utand=SQRT(Errtand(1)^2+Errtand(2)^2+Errtand(3)^2+Errtand(7)^2+Errtan
 => d(8)^2+Errtand(9)^2+Errtand(10)^2+Errtand(11)^2+Errtand(12)^2)
3220      Ueps_im=Utand*Eps_re
3222 !
3224      IF NPAR<1 THEN !Don't Print
3226 Calc_errs_menu: !
3228          CLEAR SCREEN
3230          PRINT "                UNCERTAINTIES                DELTA EPS
 =>         DELTA TAND"
```

```
3232          PRINT
3234          PRINT "   CAVITY:                                      "
3236          PRINT USING Image17;"     DIAMETER (mm)               : ",Udiameter
 => *1.E+3,Erreps(1),Errtand(1)
3238          PRINT USING Image17;"     LENGTH (mm)                 : ",Ulength*1
 => .E+3,Erreps(2),Errtand(2)
3240          PRINT "   SAMPLE:                                      "
3242          PRINT USING Image17;"     THICKNESS (mm)              : ",Usamplele
 => n*1.E+3,Erreps(3),Errtand(3)
3244          PRINT "   ENVIRONMENT:                                 "
3246          PRINT USING Image17;"     TEMPERATURE (C)             : ",Udegc,Err
 => eps(4),Errtand(4)
3248          PRINT USING Image17;"     RELATIVE HUMIDITY (%)       : ",Urelh,Err
 => eps(5),Errtand(5)
3250          PRINT USING Image17;"     PRESSURE (mbar)             : ",Umbar,Err
 => eps(6),Errtand(6)
3252          PRINT USING Image17;"     SPEED OF LIGHT (m/s)        : ",Uc,Erreps
 => (7),Errtand(7)
3254          PRINT "   MEASUREMENT:                                 "
3256          PRINT USING Image17;"     LOADED FREQUENCY (GHz)      : ",Ufreq(2)/
 => 1.E+9,Erreps(8),Errtand(8)
3258          PRINT USING Image17;"     EMPTY MICR. READING (mm)    : ",Umicr(1)*
 => 1.E+3,Erreps(9),Errtand(9)
3260          PRINT USING Image17;"     LOADED MICR. READING (mm)   : ",Umicr(2)*
 => 1.E+3,Erreps(10),Errtand(10)
3262          PRINT USING Image17;"     EMPTY Q                     : ",Uq(1),Err
 => eps(11),Errtand(11)
3264          PRINT USING Image17;"     LOADED Q                    : ",Uq(2),Err
 => eps(12),Errtand(12)
3266 !
3268          Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
3270          ON KEY 1 LABEL "Prior  Menu",Prty CALL Prior_menu
3271          ON KEY 2 LABEL "Print  Errors",Prty GOSUB Print_errors
3273          Done=0
3274          Prior_menu=0
3276          LOOP
3278            IF Done THEN GOTO Calc_errs_menu
3280          EXIT IF Prior_menu
3282          END LOOP
3284       END IF
3286       Prior_menu=0
3288       Done=1
3290       OFF KEY
3291       SUBEXIT
3292 Print_errors:!
3293       CLEAR SCREEN
3294       PRINTER IS 1
3295       GOSUB Print_them_errs
3302       PRINTER IS Print_addr
3303       GOSUB Print_them_errs
3304       PRINTER IS 1
3305       RETURN
3306 !
3307 ! ////////////////////////////////////////////////////////
3308 !
3309 Print_them_errs:!
3311       IF Freq(2)<1.E+10 THEN
3312          PRINT USING "K,2D.9D,K,3D.4D,2(K,7D)";"Frequency: ",Freq(2)/1.E+9
 => ,"    Micr:",Micr(2)*1.E+3,"            Qe:",Q(1),"  Qs:",Q(2)
3313       ELSE
3314          PRINT USING "K,2D.9D,K,3D.4D,2(K,7D)";"Frequency:",Freq(2)/1.E+9,
 => "    Micr:",Micr(2)*1.E+3,"            Qe:",Q(1),"  Qs:",Q(2)
3315       END IF
3316       PRINT USING Giveus;"UNC:d,l,t,c,f,me,ms,q";Udiameter*1.E+6,Ulength*1.
 => E+6,Usamplelen*1.E+6,Uc,Ufreq(2),Umicr(1)*1.E+6,Umicr(2)*1.E+6,Uq(1),Uq(2)
3317       PRINT USING Imageerreps;Eps_re,Ueps_re,Erreps(1),Erreps(2),Erreps(3),
 => Erreps(7),Erreps(8),Erreps(9),Erreps(10)
3318       ALLOCATE Etand(12)
3319       MAT Etand= Errtand*(1.E+6)
3320       PRINT USING Imageerrtand;Tand*1.E+6,Utand*1.E+6,Etand(1),Etand(2),Eta
 => nd(3),Etand(7),Etand(8),Etand(9),Etand(10),Etand(11),Etand(12)
3321       PRINT
3322       DEALLOCATE Etand(*)
3332       RETURN
5000 Image17:IMAGE K,4D.4D,"      ",D.6D,"          ",D.9D
5001 Imageerreps:IMAGE 3D.4D,"+/-",D.4D,7(X,D.4D)
5002 Imageerrtand:IMAGE 4X,4D,"+/-",4D,2X,7(X,6D),3X,3D,X,3D
5003 Giveus:IMAGE K,3D,4X,3D,4X,D.D,3X,4D,X,6D,4X,D.D,4X,D.D,2X,4D,4D
5005   SUBEND
5006 !
5007 ! ***************************************************************
5008 !
5010   SUB T95percent(REAL T,INTEGER V)
5012 !
5014 T95percent:!
5016       SELECT V !find t-distribution value for 95% confidence
5018       CASE 1   !degrees of freedom
5020          T=12.706
5022       CASE 2
5024          T=4.303
5026       CASE 3
5028          T=3.182
5030       CASE 4
5032          T=2.776
5034       CASE 5
5036          T=2.571
5038       CASE 6
5040          T=2.447
5042       CASE 7
5044          T=2.365
5046       CASE 8
5048          T=2.306
5050       CASE 9
5052          T=2.262
5054       CASE 10
5056          T=2.228
5058       CASE 11
5060          T=2.201
5062       CASE 12
5064          T=2.179
5066       CASE 13
5068          T=2.160
5070       CASE 14
5072          T=2.145
5074       CASE 15
5076          T=2.131
5078       CASE 16
5080          T=2.120
5082       CASE 17
5084          T=2.110
5086       CASE 18
5088          T=2.101
5090       CASE 19
5092          T=2.093
5094       CASE 20
5096          T=2.086
5098       CASE 21
5100          T=2.080
5102       CASE 22
5104          T=2.074
5106       CASE 23
5108          T=2.069
```

```
5110       CASE 24
5112            T=2.064
5114       CASE 25
5116            T=2.060
5118       CASE 26
5120            T=2.056
5122       CASE 27
5124            T=2.052
5126       CASE 28
5128            T=2.048
5130       CASE 29
5132            T=2.045
5134       CASE 30
5136            T=2.042
5138       CASE >30,<=40
5140            T=2.021
5142       CASE >40
5144            T=2.
5146       CASE ELSE
5148            BEEP
5150            DISP "CASE SELECT FOR T-DISTRIBUTION DOESNT MATCH"
5152       END SELECT
5154   SUBEND
5156 !
5158 ! ****************************************************************
5160 !
5162   SUB Cavity_dimens
5164       OPTION BASE 1
5166       COM /Environment/ Degc,Mbar,Relh,C
5168       COM /Environment/ Udegc,Umbar,Urelh,Uc
5170       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
5172       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
5174       COM /Sample/ Erreps(12),Errtand(12)
5176       COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
5178       COM /Cavity_dims/ Udiameter,Ulength
5180       COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
5182       COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
5184       COM /Constants/ Bessel_root
5186       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
5188 Cavity_dimens:OFF KEY
5190       Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
5192       GOSUB Menu_cav_dimen
5194       DISP "Do you want to CALCULATE or ENTER from keyboard the cavity dime
  => nsions?"
5196       ON KEY 1 LABEL "Prior  Menu",Prty CALL Prior_menu
5198       ON KEY 2 LABEL "CALC   DIMENS",Prty CALL Calc_dimens
5200       ON KEY 3 LABEL "ENTER  DIMENS",Prty GOSUB Enter_dimens
5202       LOOP
5204       EXIT IF Done OR Prior_menu
5206       END LOOP
5208       Prior_menu=0
5210       Done=T
5212       SUBEXIT
5214 !
5216 ! ////////////////////////////////////////////////////////////
5218 !
5220 Enter_dimens:OFF KEY
5222       DISP "Which dimension would you like to change?"
5224       GOSUB Menu_cav_dimen
5226       Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
5228       ON KEY 1 LABEL "Prior  Menu",Prty CALL Prior_menu
5230       ON KEY 2 LABEL "Change Diamter",Prty GOSUB Change_diameter
5232       ON KEY 3 LABEL "Change Length",Prty GOSUB Change_length
5234       Done=0
5236       Prior_menu=0
5238       LOOP
5240       EXIT IF Prior_menu
5242           IF Done THEN GOTO Enter_dimens
5244       END LOOP
5246       Prior_menu=0
5248       Done=T
5250       RETURN
5252 !
5254 ! ////////////////////////////////////////////////////////////
5256 !
5258 Menu_cav_dimen:!
5260       CLEAR SCREEN
5262       PRINT USING Image1;Degc,Udegc,Relh,Urelh
5264       PRINT USING Image2;Mbar,Umbar
5266       PRINT USING Image3;C,Uc
5268       PRINT USING Image4;Diameter*1.E+3,Udiameter*1.E+3
5270       PRINT USING Image5;Length0*1.E+3,Ulength*1.E+3
5272       PRINT USING Image11a;Micr(2)*1.E+3,Umicr(2)*1.E+3
5274       PRINT USING Image7c;Bessel_root
5276       PRINT USING Image8c
5278 Image1:IMAGE "Temperature = ",2D.2D," +/- ",D.2D," deg C   Relative Humidit
  => y = ",2D.D," +/- ",D.D," %"
5280 Image2:IMAGE "Barometric Pressure = ",3D.D," +/- ",D.D,"   mbar"
5282 Image3:IMAGE "Speed of Light      = ",D.7DESZ," +/- ",D.3DESZ," meters/seco
  => nd"
5284 Image4:IMAGE "Resonator Diameter             =  ",2D.5D," +/- ",D.5D,"  mm"
5286 Image5:IMAGE "          Length (micr. zero) = ",3D.5D," +/- ",D.5D,"  mm"
5288 Image7c:IMAGE "Bessel function root: ",2D.7D
5290 Image8c:IMAGE "Axial mode  Frequency (GHz)"
5292 Image9c:IMAGE X,2D,9X,2D.7D
5294 Image11a:IMAGE "Micrometer reading = ",2(S2D.4D,"   +/- ",D.4D,6X),"mm"
5296       RETURN
5298 !
5300 ! ////////////////////////////////////////////////////////////
5302 !
5304 Change_diameter:OFF KEY
5306       CALL Edit_data("the cavity diameter (mm)",Diameter,1.E+3,Udiameter)
5308       Done=1
5310       RETURN
5312 !
5314 ! ////////////////////////////////////////////////////////////
5316 !
5318 Change_length:OFF KEY
5320       CALL Edit_data("the cavity length at micrometer=zero (mm)",Length0,1.
  => E+3,Ulength)
5322       Done=1
5324       RETURN
5326   SUBEND
5328 !
5330 ! ****************************************************************
5332 !
5334   SUB Find_dims(Spectrum(*),Length0,Diameter,Ulength,Udiameter,Micr,C,Besse
  => l_root,Num_modes)
5336       OFF KEY
5338       OPTION BASE 1
5340       INTEGER N
5342       N=Num_modes
5344       ALLOCATE X(N,2),Y(N,1),B(2,1),Bt(1,2),Xtx(2,2),A(2),Cvm(2,2)
5346       ALLOCATE Xt(2,N),Yt(1,N),Xtxi(2,2),S2(1),Yty(1),Btxty(1),Xty(2,1)
5348 Find_dims: !This routine finds the dimensions of a cylindrical cavity
5350 !using a TEmn frequency spectrum.  The normal application is for the TE01
5352 !spectrum, but by changing the Bessel_root, one may use any other TE mode.
5354 !A matrix approach is used to perform a least squares regression.  The
5356 !uncertainties returned are covariant, (i.e. the confidence interval is
5358 !an ellipse and not a rectangle) so one must be certain to consider the
5360 !length and diameter uncertainties individually and not simultaneously.
```

```
5362 !References: J.D. Jackson "Classical Electrodynamics" 1962 page 255.
5364 !            N. Draper & H. Smith "Applied Regression Analysis, Second
5366 !            Edition" 1981 Chapter 2 and page 532.
5368      FOR I=1 TO Num_modes
5370          X(I,1)=1
5372          X(I,2)=Spectrum(I,1)^2
5374          Y(I,1)=Spectrum(I,2)^2
5376      NEXT I
5378      MAT Xt= TRN(X)
5380      MAT Xtx= Xt*X
5382      MAT Xtxi= INV(Xtx)
5384      MAT Yt= TRN(Y)
5386      MAT Yty= Yt*Y
5388      MAT Xty= Xt*Y
5390      MAT B= Xtxi*Xty
5392      MAT Bt= TRN(B)
5394      MAT Btxty= Bt*Xty
5396      MAT S2= Yty-Btxty
5398      S2(1)=S2(1)/(N-2)
5400      MAT Cvm= Xtxi*(S2(1))
5402      CALL T95percent(T,Num_modes-2)
5404      Length0=C/(2*SQRT(B(2,1)))-Micr
5406      Diameter=Bessel_root*C/(SQRT(B(1,1))*PI)
5408      Ulength=C*(T*SQRT(Cvm(2,2)))/(2*B(2,1)^(1.5))
5410      Udiameter=C*Bessel_root*(T*SQRT(Cvm(1,1)))/(2*PI*B(1,1)^(1.5))
5412      DEALLOCATE X(*),Y(*),B(*),Bt(*),Xtx(*),A(*),Cvm(*)
5414      DEALLOCATE Xt(*),Yt(*),Xtxi(*),S2(*),Yty(*),Btxty(*),Xty(*)
5416   SUBEND
5418   !
5420   ! ****************************************************************
5422   !
5424   SUB Calc_err_anal
5426 Calc_err_anal:OFF KEY
5428   !
5430      OPTION BASE 1
5432      COM /Environment/ Degc,Mbar,Relh,C
5434      COM /Environment/ Udegc,Umbar,Urelh,Uc
5436      COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
5438      COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
5440      COM /Sample/ Erreps(12),Errtand(12)
5442      COM /Cavity_dims/ Diameter,Length0,Spectrum(40,5),INTEGER Num_modes
5444      COM /Cavity_dims/ Udiameter,Ulength
5446      COM /Cavity_vals/ Freq(2),Micr(2),Q(2),Bw(2)
5448      COM /Cavity_vals/ Ufreq(2),Umicr(2),Uq(2),Ubw(2)
5450      COM /Constants/ Bessel_root
5452      COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
5454 Equations: !
5456      F0=Freq(2)
5458      Fe=Freq(2)
5460      L0=Micr(1)
5462      Le=Micr(2)
5464      S=Samplelen
5466      D=Diameter
5468      U=Bessel_root
5470      Eps=Eps_re
5472      Y=Le-S
5474      Fc=(U*C)/(PI*D)
5476      X=Fc/F0
5478      B0=(2*PI*F0*SQRT(1-(X^2)))/C
5480      Be=(2*PI*Fe*SQRT(Eps-(X^2)))/C
5482 Partials:  !
5484      Par_eps_b=1-(X^2)
5486      Par_eps_q=(-2*S*X)+(2*X)
5488      Par_b_be=2*((TAN(Be*S)*(1/(COS(Be*S)^2))*B)/(TAN(B0*Y)^2))
5490      Par_b_b0=2*((-(TAN(Be*S)^2)*(1/(SIN(B0*Y)^2))*Y)/(TAN(B0*Y)))
5492      Par_b_b=2*((-TAN(Be*S)*(1/(COS(Be*S)^2))*Be)/(TAN(B0*Y)^2))
5494      Par_b_y=2*((-(TAN(Be*S)^2)*(1/(SIN(B0*Y)^2))*B0)/(TAN(B0*Y)))
5496      Par_be_fe=(2*PI*SQRT(Eps_re-(X^2)))/C
5498      Par_be_c=(-2*PI*Fe*SQRT(Eps_re-(X^2)))/(C^2)
5500      Par_be_eps=(PI*Fe)/(C*SQRT(Eps-(X^2)))
5502      Par_be_q=(-2*PI*Fe*X)/(C*SQRT(Eps-(X^2)))
5504      Par_b0_f0=(2*PI*SQRT(1-(X^2)))/C
5506      Par_b0_c=(-2*PI*F0*SQRT(1-X^2))/(C^2)
5508      Par_b0_q=(-2*PI*F0*X)/(C*SQRT(1-(X^2)))
5510      Par_q_fc=1/F0
5512      Par_q_f0=-Fc/(F0^2)
5514      Par_fc_c=U/(PI*D)
5516      Par_fc_d=(-U*C)/(PI*(D^2))
5518      Par_y_le=1
5520      Par_y_l0=-1
5522      Par_y_b=-1
5524 Differentials:!
5526      Del_s=Usamplelen
5528      Del_d=Udiameter
5530      Del_le=Umicr(2)
5532      Del_l0=Umicr(1)
5534      Del_f0=Ufreq(1)
5536      Del_fe=Ufreq(2)
5538      Del_c=Uc
5540      Del_y=(Par_y_le*Del_le)+(Par_y_l0*Del_l0)+(Par_y_b*Del_s)
5542      Del_b0=(Par_b0_f0*Del_f0)+(Par_b0_c*Del_c)+(Par_b0_q*Del_q)
5544      Del_fc=(Par_fc_c*Del_c)+(Par_fc_d*Del_d)
5546      Del_q=(Par_q_fc*Del_fc)+(Par_q_f0*Del_f0)
5548      Del_be=(Par_be_eps*Del_eps)+(Par_be_c*Del_c)+(Par_be_fe*Del_fe)+(Par_
  => be_q*Del_q)
5550      Del_b=(Par_b_be*Del_be)+(Par_b_b0*Del_b0)+(Par_b_b*Del_s)+(Par_b_y*De
  => l_y)
5552      Del_eps=(Par_eps_b*Del_b)+(Par_eps_q*Del_q)
5554   SUBEND
5556 !
5558 ! ****************************************************************
5560 !
5562   SUB Sample_info
5564 Sample_info:OFF KEY
5566      OPTION BASE 1
5568      COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
5570      COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
5572      COM /Sample/ Erreps(12),Errtand(12)
5574      COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
5576      DISP CHR$(129);" You may change Sample Information here. ";CHR$(128)
5578      Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
5580      ON KEY 1 LABEL "Prior  Menu",Prty CALL Prior_menu
5582      ON KEY 2 LABEL "Sample Desc",Prty CALL Sample_descr
5584      ON KEY 3 LABEL "Sample Thickns",Prty CALL Sample_thickns
5586      ON KEY 4 LABEL "Sample Init_Eps",Prty CALL Sample_init_eps
5588      Done=0
5590      Prior_menu=0
5592      LOOP
5594          IF Done THEN GOTO Sample_info
5596      EXIT IF Prior_menu
5598      END LOOP
5600      OFF KEY
5602      Prior_menu=0
5604      Done=T
5606      SUBEXIT
5608   SUBEND
5610 !
5612 ! ****************************************************************
5614 !
5616   SUB Sample_init_eps
```

```
5618 Sample_init_eps:OFF KEY
5620   !
5622       OPTION BASE 1
5624       COM /Sample/ Sample_id$[55],Samplelen,Eps_re,Eps_im,Tand
5626       COM /Sample/ Usamplelen,Ueps_re,Ueps_im,Utand,Eps_re_guess
5628       COM /Sample/ Erreps(12),Errtand(12)
5630       COM /System_status/ INTEGER Done,Prior_menu,Do_var_freq,Empty_cavity,
  => Prompt$[80]
5632   !
5634       CALL Edit_data("the sample's initial guess for epsilon.",Eps_re_guess
  => ,1)
5636       Done=1
5638   SUBEND
5640 !
5642 ! ************************************************************
5644 !
5646   SUB Fit_sparms
5648 !     Fit_lorentz(Kx,Ky,Q,F0,Ukx,Uky,Uq,Uf0)         Sss=1
5650 ! or  Fit_lorentz(Q0,Ql,F0,Phase,Uq0,Uql,Uf0,Uphase)  Sss=2 or 3
5652  !Written by Eric J. Vanzura
5654  !Levenberg-Marquart algorithm CSUB taken from:
5656  !Numerical Recipes : the art of Scientific Computing
5658  !W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling
5660  !Cambridge Press 1987
5662 Fit_sparms:OFF KEY
5664       GOSUB Init
5666       GOSUB Init_est
5668       GOSUB Initial_plot            !Plot meas'd S-parms & initial estimate
5670       REPEAT
5672 Do_mrq:!
5674           GOSUB Printlatestvals
5676           Ochisq=Chisq
5678           MAT Old_a= A
5680           DISP "Performing Levenberg-Marquart Least-square-fit"
5682           OFF KEY
5684           CALL Mrqmin(X(*),Y(*),Sig(*),Ndata(*),A(*),Ma(*),Lista(*),Mfit(*)
  => ,Covar(*),Alpha(*),Nca(*),Chisq,Alamda,Sss,Debug)
5686           GOSUB Convergencekeys
5688           MAT Old_est= Sgen
5690           GOSUB Gen_estimate
5692           IF Speed<>1 THEN
5694               GOSUB Plot_estimate
5696           END IF
5698           GOSUB Ask_converge
5700       UNTIL Itst>=Itst_limit OR Force_converge
5702 Mrq_out:OFF KEY
5704       Alamda=0
5706       CALL Mrqmin(X(*),Y(*),Sig(*),Ndata(*),A(*),Ma(*),Lista(*),Mfit(*),Cov
  => ar(*),Alpha(*),Nca(*),Chisq,Alamda,Sss,Debug)
5708       GOSUB Calc_sig2
5710       GOSUB Calc_uncert
5712       GOSUB Print_results
5714       GOTO Subend
5716 !
5718 ! ////////////////////////////////////////////////////////////
5720 !
5722 Calc_uncert:!
5724       SELECT Sss
5726       CASE =1
5728           Ua(1)=SQRT(Sig2*Covar(1,1)) !Kx
5730           Ua(2)=SQRT(Sig2*Covar(2,2)) !Ky
5732           Ua(3)=SQRT(Sig2*Covar(3,3)) !Q
5734           Ua(4)=SQRT(Sig2*Covar(4,4)) !F0
5736           Ua(5)=SQRT(Sig2*Covar(5,5)) !
5738       CASE =2,=3
5740           Ua(1)=SQRT(Sig2*Covar(1,1)) !Uq0
5742           Ua(2)=SQRT(Sig2*Covar(2,2)) !Uql
5744           Ua(3)=SQRT(Sig2*Covar(3,3)) !Uf0
5746           Ua(4)=SQRT(Sig2*Covar(4,4)) !Uphase
5748           Ua(4)=Ua(4)*360/2/PI           !in degrees
5750           Ua(5)=SQRT(Sig2*Covar(5,5)) !Umag (beta)
5752       CASE ELSE
5754           BEEP
5756           DISP "Sss=";Sss;" in Calc_uncert. PAUSED"
5758           PAUSE
5760           GOTO Calc_uncert
5762       END SELECT
5764       RETURN
5766 !
5768 ! ////////////////////////////////////////////////////////////
5770 !
5772 Calc_sig2:!
5774       FOR I=1 TO Fitcount
5776           GOSUB Gen_cnum
5778           N=Start_index+I-1
5780           Sig2=Sig2+(ABS(CMPLX(S(N,Ns,1),S(N,Ns,2))-Cnum))^2
5782       NEXT I
5784       Sig2=Sig2/(Fitcount-Mfit(2))
5786       RETURN
5788 !
5790 ! ////////////////////////////////////////////////////////////
5792 !
5794 Print_results:!
5796       PRINTER IS 1
5798       PRINT Description$;" fit to ";Sparm$;" values."
5800       PRINT USING "K,3D,2(3X,K,MD.3DE)";"Iteration #",K,"Chi_squared=",Chis
  => q,"Alamda=",Alamda
5802       PRINT USING "3D,K,3D,K,3D";Dcount;" data points in file. Start_index=
  => ",Start_index,"  Stop_index=",Stop_index
5804       GOSUB Print_a
5806       GOSUB Print_ua
5808       PRINT USING "K,D.4DE";"Chi-squared=",Chisq
5810       PRINTER IS Print_addr
5812       GOSUB Print_a
5814       GOSUB Print_ua
5816       PRINT USING "K,D.4DE";"Chi-squared=",Chisq
5818       PRINTER IS 1
5820       Reset_keys=1
5822       RETURN
5824 !
5826 ! ////////////////////////////////////////////////////////////
5828 !
5830 Initial_plot:!
5832       ALLOCATE File(Fitcount*3,2),Old_est(Fitcount,2),Orig_sparm(Fitcount,2
  => )
5834       ALLOCATE Sgen(Fitcount,2)
5836       MAT Orig_sparm(*,1:2)= S(Start_index:Stop_index,Ns,1:2)
5838       GOSUB Gen_estimate
5840       MAT Old_est= Sgen
5842       IF NOT Speed THEN GOSUB Plot_estimate
5844       RETURN
5846 !
5848 ! ////////////////////////////////////////////////////////////
5850 !
5852 Gen_cnum:!
5854       SELECT Sss
5856       CASE 1   !S21
5858           Cnum=-X(I)^2*CMPLX(A(1),-A(2))/CMPLX(X(I)^2-A(4)^2,-X(I)*A(4)/A(3
  => ))
5860       CASE 2   !S11
5862           Degrees=360*A(4)/2/PI
5864           P=CMPLX(COS(Degrees),SIN(Degrees))
```

```
5866        Ql2f=2*A(2)*(X(I)-A(3))/A(3)
5868        Cnum=A(5)*CMPLX(A(2)/A(1)+Ql2f*Ql2f,Ql2f*(1-A(2)/A(1)))*P/(1+Ql2f
  => *Ql2f)
5870      CASE 3   !S21
5872        Degrees=360*A(4)/2/PI
5874        P=CMPLX(COS(Degrees),SIN(Degrees))
5876        G0gl=1-A(1)/A(2)
5878        Dum=CMPLX(1,2*(X(I)-A(3))/A(2))
5880        Cnum=A(5)*G0gl*P/Dum
5882      CASE ELSE
5884        BEEP
5886        DISP "Sss=";Sss;" is not valid in Gen_cnum."
5888        PAUSE
5890      END SELECT
5892      RETURN
5894 !
5896 ! ////////////////////////////////////////////////////////////
5898 !
5900 Gen_estimate:!
5902      DISP "Generating latest estimate values"
5904      MAT Sgen= (0)
5906      SELECT Sss
5908      CASE =1   !Nussenzveig Lorentzian
5910        FOR I=1 TO Fitcount
5912          GOSUB Gen_cnum
5914          Sgen(I,1)=REAL(Cnum)
5916          Sgen(I,2)=IMAG(Cnum)
5918        NEXT I
5920      CASE =2   !Schulten S11
5922        FOR I=1 TO Fitcount
5924          GOSUB Gen_cnum
5926          Sgen(I,1)=REAL(Cnum)
5928          Sgen(I,2)=IMAG(Cnum)
5930        NEXT I
5932      CASE =3   !Schulten S21
5934        FOR I=1 TO Fitcount
5936          GOSUB Gen_cnum
5938          Sgen(I,1)=REAL(Cnum)
5940          Sgen(I,2)=IMAG(Cnum)
5942        NEXT I
5944      CASE ELSE
5946        DISP "Sss=";Sss;" is not valid in Gen_estimate."
5948        BEEP
5950        PAUSE
5952        GOTO Gen_estimate
5954      END SELECT
5956      RETURN
5958 !
5960 ! ////////////////////////////////////////////////////////////
5962 !
5964 Plot_estimate:!
5966      Pack_data(File(*))
5968      Id$=Sparm$
5970      Pen=2
5972      Pack_data(File(*),Orig_sparm(*),Fitcount,Pen,Id$)
5974 !
5976      Id$="Old estimate"
5978      Pen=4
5980      Pack_data(File(*),Old_est(*),Fitcount,Pen,Id$)
5982 !
5984      Id$="Latest estimate"
5986      Pen=6
5988      Pack_data(File(*),Sgen(*),Fitcount,Pen,Id$)
5990 !
5992      Test$="Real Part"
5994      Prompt$="Imaginary Part"
5996      Id$="Results from file:"&Rawdatafile$
5998      Init_graphics(Test$,Prompt$,Id$)
6000      Ax$="XY"
6002      Autoscale(File(*),Ax$)
6004      Refresh_labels
6006      DISP "S-parameters are: Red=measured data, Green=Old estimate, Blue=L
  => atest Estimate"
6008      Plot_all(File(*))
6010      RETURN
6012 !
6014 ! ////////////////////////////////////////////////////////////
6016 !
6018 Init:!
6020      DEG
6022      OPTION BASE 1
6024      INTEGER I,Prty,Itst,Itst_limit,Ns
6026      INTEGER Pen,Done,Fitcount
6028      COMPLEX Cnum,P,Dum
6030      REAL Debug
6032      DIM Test$[160],Prompt$[40],Id$[40],Rawdatafile$[10],Sparm$[3]
6034      COM /History/ Status$[1],Time_orgn$[8],Date_orgn$[11]
6036      COM /History/ Time_chng$[8],Date_chng$[11],Description$[160]
6038      COM /Labels/ Labels$(30)[60],INTEGER Lbl_count,REAL Lbl_addr(30,6)
6040      COM /Data_param/ INTEGER Data_count,Filesize,Curvecount,Roster(17,4)
6042      COM /Data_param/ REAL Sym_size,Symbol$(17)[2],Curve_id$(17)[40]
6044      COM /Data_param/ REAL Xmin_data,Xmax_data
6046      COM /Data_param/ REAL Ymin_data,Ymax_data
6048      COM /Graphics/ INTEGER Speed
6050      COM /Background/ Graphtype$[12],Margins$(2)[10],Papersize$[1]
6052      COM /Background/ REAL Pen_speed,INTEGER Backgnd_pen,Auto_time
6054      COM /Background/ INTEGER Auto_file,REAL X_cross_y,Y_cross_x
6056      COM /Background/ Xgrid_tick$[4],INTEGER Xmajor,Xminor
6058      COM /Background/ Ygrid_tick$[4],INTEGER Ymajor,Yminor
6060      COM /Background/ REAL Xmin_graph,Xmax_graph,Ymin_graph,Ymax_graph
6062      COM /Axes_labels/ Print_xlabels$[3],Print_ylabels$[3]
6064      COM /Axes_labels/ Sig_digits$[2],REAL Xlcsize,Ylcsize
6066      COM /Windowspace/ REAL Xmin,Xmid,Xmax,Ymin,Ymid,Ymax! graph edges UDU
6068      COM /Windowspace/ REAL Xleft,Xright,Ybottom,Ytop! paper edges UDUs
6070      COM /Log_scale/ REAL Xcycles,Xbegin,Ycycles,Ybegin
6072      COM /Plot_device/ Plot_lang$[10],INTEGER Plot_addr
6074      COM /Hard_space/ REAL Xview_lft,Xview_rt,Yview_btm,Yview_top
6076      COM /Hard_space/ REAL Viewscale,Aspect_ratio
6078      COM /Frame_onoff/ INTEGER Frame_flag
6080      COM /Clear_space/ REAL Space_lft,Space_rt,Space_btm,Space_top
6082 !    !
6084 Init_com:  !
6086      COM /Fittype/ INTEGER Sselect,Start_index,Stop_index
6088      COM /S_array/ INTEGER Dcount,Svalid,REAL S(801,5,2)
6090      COM /Cdata/ INTEGER Mfit(2),Lista(5,2),Ma(2),REAL A(5),Ua(5),INTEGER
  => Ndata(2)
6092      COM /Cstats/ Alamda,Chisq,INTEGER Nca(2),REAL Alpha(5,5),Covar(5,5)
6094      COM /Output/ INTEGER Print_addr,Plotter,Printer_on,Plotter_on
6096      ASSIGN @Printer TO Print_addr
6098 Init_vals:!
6100      K=0
6102      Itst=0
6104      Itst_limit=3
6106 !    Printer_on=0
6108      Plot_addr=3
6110      Plot_lang$="INTERNAL"
6112      M=5      !Dimension of Covar,Alpha,Lista & A matricies
6114      Debug=0
6116 !
6118      IF NOT BIT(Svalid,Sselect) THEN
6120        BEEP
6122        DISP "The S(*) array for ";Sparm$;" is not valid (doesn't have da
```

```
 => ta?) PAUSED"
6124          PRINT "The S(*) array for ";Sparm$;" is not valid (doesn't have d
 => ata?) PAUSED"
6126          PAUSE
6128       END IF
6130 !Sss=1 S21-Nussenzvieg    Sss=2 S11-Schulten    Sss=3 S21-Schulten
6132       SELECT Sselect
6134       CASE 0      !S11
6136          Sss=2
6138          Sparm$="S11"
6140       CASE 1      !S21
6142          Sss=1
6144 !        Sss=3
6146          Sparm$="S21"
6148       CASE 2      !S12
6150          Sss=1
6152          Sss=3
6154          Sparm$="S12"
6156       CASE 3      !S22
6158          Sss=2
6160          Sparm$="S22"
6162       CASE ELSE
6164          BEEP
6166          DISP "Sselect is not valid in Fit_sparms"
6168          PAUSE
6170       END SELECT
6172       Ns=Sselect+1
6174       RETURN
6176 !
6178 ! ///////////////////////////////////////////////////////////////
6180 !
6182 Init_est:  !
6184       Fitcount=Stop_index-Start_index+1
6186       Start_freq=S(Start_index,5,1)
6188       Stop_freq=S(Stop_index,5,1)
6190       SELECT Sss
6192       CASE =1  ! Nussenzveig Lorentzian
6194          Ix=INT(Start_index+Fitcount/2)
6196          Angle=ARG(CMPLX(S(Ix,Ns,1),S(Ix,Ns,2)))
6198          Mag=ABS(CMPLX(S(Ix,Ns,1),S(Ix,Ns,2)))
6200          A(1)=-2*10^(-5)*Mag*SIN(Angle)
6202          A(2)=-2*10^(-5)*Mag*COS(Angle)
6204          A(3)=1.5*S(Ix,5,1)/(Stop_freq-Start_freq)
6206          A(4)=S(Ix,5,1)
6208       CASE =2,=3  ! Schulten S11 or S21
6210          Ix=INT(Start_index+Fitcount/2)
6212          A(1)=S(Ix,5,1)/(Stop_freq-Start_freq)*1.8
6214          A(2)=S(Ix,5,1)/(Stop_freq-Start_freq)*2.0
6216 !        A(1)=S(Ix,5,1)/90000
6218 !        A(2)=S(Ix,5,1)/79000
6220          A(3)=S(Ix,5,1)
6222          A(4)=ARG(CMPLX(S(Ix,Ns,1),S(Ix,Ns,2)))*2*PI/360
6224 !        A(4)=0
6226 !        A(5)=ABS(CMPLX(S(Ix,Ns,1),S(Ix,Ns,2)))
6228          A(5)=.7
6230       CASE ELSE
6232          DISP "Sss=";Sss;" is not valid in Init_est"
6234          BEEP
6236          PAUSE
6238          GOTO Init_est
6240       END SELECT
6242       Ndata(1)=0
6244       Ndata(2)=Fitcount*2
6246       Ma(1)=0            !size of A array (unknowns)
6248       Ma(2)=M
6250       Chisq=1            !measure of fit
6252       Mfit(1)=0          !number of unknowns to be fitted
6254       Mfit(2)=M
6256       IF Sss=1 THEN Mfit(2)=M-1
6258       MAT Lista= (0)
6260       Lista(1,2)=1       !A array index of the unknowns to be fitted
6262       Lista(2,2)=2
6264       Lista(3,2)=3
6266       Lista(4,2)=4
6268       Lista(5,2)=5
6270       IF Sss=1 THEN Lista(5,2)=0
6272       MAT Covar= (0)
6274       MAT Alpha= (0)
6276       Nca(1)=0           !Size of Alpha & Covar arrays >= Ma
6278       Nca(2)=M
6280       Alamda=-2
6282       ALLOCATE Old_a(M)
6284       ALLOCATE REAL X(Fitcount*2),Y(Fitcount*2),Sig(Fitcount*2)
6286       MAT Sig= (1)
6288       MAT X(1:Fitcount)= S(Start_index:Stop_index,5,1)
6290       MAT X(Fitcount+1:Fitcount*2)= S(Start_index:Stop_index,5,1)
6292       MAT Y(1:Fitcount)= S(Start_index:Stop_index,Ns,1)
6294       MAT Y(Fitcount+1:Fitcount*2)= S(Start_index:Stop_index,Ns,2)
6296       RETURN !Init_est
6298 !
6300 ! ///////////////////////////////////////////////////////////////
6302 !
6304 Done_return:OFF KEY
6306       Done=1
6308       RETURN
6310 !
6312 ! ///////////////////////////////////////////////////////////////
6314 !
6316 Force_converge:PRINT "Convergence will be forced."
6318       Force_converge=1
6320       RETURN
6322 !
6324 ! ///////////////////////////////////////////////////////////////
6326 !
6328 Preventconverge:PRINT "Convergence will be prevented."
6330       Itst=0
6332       RETURN
6334 !
6336 ! ///////////////////////////////////////////////////////////////
6338 !
6340 Graph_on: !
6342       Speed=1
6344       OFF KEY 4
6346       ON KEY 4 LABEL "GRAPH  OFF",Prty GOSUB Graph_off
6348       RETURN
6350 !
6352 ! ///////////////////////////////////////////////////////////////
6354 !
6356 Graph_off: !
6358       Speed=0
6360       OFF KEY 4
6362       ON KEY 4 LABEL "GRAPH  ON",Prty GOSUB Graph_on
6364       RETURN
6366 Debug_on:PRINT "Debug flag is turned on."
6368       Debug=1
6370       OFF KEY 6
6372       ON KEY 6 LABEL "DEBUG  OFF",Prty GOSUB Debug_off
6374       RETURN
6376 !
6378 ! ///////////////////////////////////////////////////////////////
6380 !
6382 Debug_off:PRINT "Debug flag is turned off."
```

```
6384      Debug=0
6386      OFF KEY 6
6388      ON KEY 6 LABEL "DEBUG  ON",Prty GOSUB Debug_on
6390      RETURN
6392 !
6394 ! ////////////////////////////////////////////////////////////
6396 !
6398 Ask_converge:!
6400      SELECT Chisq
6402      CASE >Ochisq
6404          Itst=0
6406          PRINT USING "K,D.DE";"New Chisq is greater than old Chisq by:";Ch
  => isq-Ochisq
6408      CASE =Ochisq
6410          PRINT "Chisq is equal to old Chisq"
6412      CASE <Ochisq
6414          PRINT USING "K,D.DE";"New Chisq is LESS than old Chisq by:";Ochis
  => q-Chisq
6416          IF ABS((Ochisq-Chisq)/Chisq)<.003 THEN Itst=Itst+1
6418          IF ABS((Ochisq-Chisq)/Chisq)<.00003 THEN Itst=Itst+1
6420          PRINT USING "K,D";"Itst=";Itst
6422          PRINT USING "K,D.DE";"ABS((Ochisq-Chisq)/Chisq)=";ABS((Ochisq-Chi
  => sq)/Chisq)
6424      END SELECT
6426      RETURN
6428 !
6430 ! ////////////////////////////////////////////////////////////
6432 !
6434 Convergencekeys:!
6436      Prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
6438      ON KEY 0 LABEL "FORCE  CONVERG",Prty GOSUB Force_converge
6440      ON KEY 2 LABEL "PREVENTCONVERG",Prty GOSUB Preventconverge
6442      IF Speed THEN
6444          ON KEY 4 LABEL "GRAPH  OFF",Prty GOSUB Graph_on
6446      ELSE
6448          ON KEY 4 LABEL "GRAPH  ON",Prty GOSUB Graph_off
6450      END IF
6452      IF Debug THEN
6454          ON KEY 6 LABEL "DEBUG  OFF",Prty GOSUB Debug_off
6456      ELSE
6458          ON KEY 6 LABEL "DEBUG  ON",Prty GOSUB Debug_on
6460      END IF
6462      RETURN
6464 !
6466 ! ////////////////////////////////////////////////////////////
6468 !
6470 Printlatestvals:!
6472      PRINT USING "K,MD.3DE";"Chisq=",Chisq
6474      K=K+1
6476      PRINT USING "K,3D,L,K,MD.3DE";"Iteration #",K,"Alamda=",Alamda
6478      GOSUB Print_a
6480      RETURN
6482 !
6484 ! ////////////////////////////////////////////////////////////
6486 !
6488 Print_a:!
6490      SELECT Sss
6492      CASE =1
6494          IF A(4)<1.E+10 THEN
6496              PRINT USING "2(K,MD.4DE,3X),K,M6D,3X,K,X,MD.9DE";"Kx =",A(1),
  => "Ky =",A(2),"Q =",A(3),"F0 =",A(4)
6498          ELSE
6500              PRINT USING "2(K,MD.4DE,3X),K,M6D,3X,K,M2D.9DE";"Kx =",A(1),"
  => Ky =",A(2),"Q =",A(3),"F0 =",A(4)
6502          END IF
6504      CASE =2,=3
6506          PRINT USING "2(K,MD.4DE,2X),K,MD.9DE,2X,K,M3D.DD,2X,K,MD.3DE";"Q0
  => =",A(3)/A(1),"Ql=",A(3)/A(2),"F0=",A(3),"Degrees=",A(4)*360/2/PI,"Mag=",A(5
  => )
6508      CASE ELSE
6510          DISP "Sss=";Sss;" is not valid in print_a. program paused"
6512          BEEP
6514          PAUSE
6516      END SELECT
6518      RETURN
6520 !
6522 ! ////////////////////////////////////////////////////////////
6524 !
6526 Print_ua:!
6528      SELECT Sss
6530      CASE =1
6532          PRINT USING "2(K,MD.3DE,4X),K,M5D,3X,K,3X,MD.3DE";"Ukx=",Ua(1),"U
  => ky=",Ua(2),"UQ=",Ua(3),"Uf0=",Ua(4)
6534      CASE =2,=3
6536          PRINT USING "2(K,MD.3DE,2X),K,MD.9DE,2X,K,M3D.3D,2X,K,MD.3DE";"UQ
  => 0=",Ua(1),"UQl=",Ua(2),"UF0=",Ua(3),"Udegrees=",Ua(4),"Umag=",Ua(5)
6538      CASE ELSE
6540          DISP "Sss=";Sss;" is not valid in Print_ua. program paused"
6542          BEEP
6544          PAUSE
6546      END SELECT
6548      RETURN
6550 !
6552 ! ////////////////////////////////////////////////////////////
6554 !
6556 Subend:!
6558   SUBEND     !Fit_sparms
6560 !
6562 ! ************************************************************
6564 !
6566   CSUB Mrqmin(X(*),Y(*),Sig(*),INTEGER Ndata(*),REAL A(*),INTEGER Ma(*),Lis
  => ta(*),Mfit(*),REAL Covar(*),Alpha(*),INTEGER Nca(*),REAL Chisq,Alamda,Sss,D
  => ebug)
6568   SUB Enterfilename(Ac$,OPTIONAL Prompt$)
6570      COM /Files/ Diskdrive$[20],Filename$[10]
6572      INTEGER I,Ascii_num
6574      DIM Test$[160]
6576 Enterfilename:!
6578      IF Filename$<>"" THEN
6580          OUTPUT 2 USING "K,#";Filename$
6582      END IF
6584      SELECT NPAR
6586      CASE 1
6588          DISP " ENTER the FILE NAME ";
6590      CASE 2
6592          DISP " ENTER the FILE NAME for ";Prompt$;
6594      END SELECT
6596      SELECT Ac$
6598      CASE "CAT"
6600          DISP " ... (ENTER alone to CAT) ";
6602      CASE "ABORT"
6604          DISP " ... (ENTER alone to ABORT) ";
6606      CASE "VALID"
6608      END SELECT
6610      LINPUT Test$
6612      Test$=TRIM$(Test$)
6614      IF LEN(Test$)=0 THEN
6616          SELECT Ac$
6618          CASE "VALID"
6620              DISP "You MUST enter the FILE NAME now."
6622              BEEP
6624              WAIT 1.8
```

```
6626            GOTO Enterfilename
6628        CASE "ABORT","CAT"
6630            GOTO Abortline
6632        CASE ELSE
6634            DISP "Ac$=";Ac$;" in SUB Enterfilename"
6636            BEEP
6638            PAUSE
6640        END SELECT
6642      END IF
6644      IF LEN(Test$)>10 THEN
6646        BEEP
6648        DISP "ERROR in NAME ENTRY--up to 10 chars, you have ";
6650        DISP LEN(Test$);" "
6652        WAIT 1.8
6654        OUTPUT 2 USING "#,K,K";" #";Test$
6656        GOTO Enterfilename
6658      END IF
6660      Filename$=Test$
6662      FOR I=1 TO LEN(Filename$)
6664        Ascii_num=NUM(Filename$[I])
6666        SELECT Ascii_num
6668        CASE 65 TO 90,95,97 TO 122,48 TO 57
6670            !Allowed characters
6672        CASE ELSE
6674            BEEP
6676            DISP "ERROR in NAME ENTRY--ILLEGAL CHARACTERS, TRY AGAIN."
6678            WAIT 1.8
6680            OUTPUT 2 USING "#,K,K";" #";Filename$
6682            GOTO Enterfilename
6684        END SELECT
6686      NEXT I
6688      SUBEXIT
6690 Abortline:Filename$=""
6692      SUBEXIT
6694   SUBEND
6696   !
6698   ! ***************************************************************
6700   !
6702   SUB Select_disk(OPTIONAL Prompt$)
6704 Select_disk: ! Original: 13 Nov 1984
6706               ! Revision: 02 Dec 1987
6708      COM /Files/ Diskdrive$[20],Filename$[10]
6710      INTEGER Local_prty,Dd,Pt,Choose(1)
6712      DIM Disc$(30)[60],Title$[40],Displ$[80]
6714      Local_prty=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
6716      Sys_id$=SYSTEM$("SYSTEM ID")
6718      OFF KEY
6720      !
6722      ! Define the disk drives available for this system, reserve the
6724      ! first characters for the drive address and the characters after
6726      ! the - for a description of the drive.
6728      !
6730      ! Example:
6732      ! Disc$(1)=":,700,0,0      HP 9133H HARD disk, volume 0."
6734      !
6736      !
6738      IF NPAR>0 THEN
6740        Displ$=" SELECT DISK DRIVE for "&Prompt$&" ... Abort will cancel"
6742        Displ$=Displ$[1,80]
6744      ELSE
6746        Displ$=" SELECT DISK DRIVE  ... Abort will cancel. "
6748      END IF
6750      Title$=" Available disk drives for this system. "
6752      Pt=1   I allow only one select
6754      !
6756      IF Diskdrive$[1,1]<>":" THEN Diskdrive$=""
6758 !      IF Msi_id$[1,1]<>":" THEN Msi_id$=SYSTEM$("MSI")
6760      Msi_id$=SYSTEM$("MSI")
6762      Msi_id$=Msi_id$[POS(Msi_id$,":"),LEN(Msi_id$)]
6764      Diskdrive$=TRIM$(Diskdrive$)
6766      Msi_id$=TRIM$(Msi_id$)
6768      IF LEN(Diskdrive$)>0 AND LEN(Msi_id$)>0 THEN
6770        Disc$(1)=Diskdrive$&RPT$(" ",17-LEN(Diskdrive$))
6772        Disc$(1)=Disc$(1)&"- Last selected disk drive."
6774        Dd=1
6776        IF Diskdrive$<>Msi_id$ THEN
6778          Disc$(2)=Msi_id$&RPT$(" ",17-LEN(Msi_id$))
6780          Disc$(2)=Disc$(2)&"- Start-up mass storage unit specifier."
6782          Dd=Dd+1
6784        ELSE
6786          Disc$(1)=Disc$(1)&"  Start-up MSUS."
6788        END IF
6790      ELSE
6792        IF LEN(Msi_id$)>0 THEN
6794          Disc$(1)=Msi_id$&RPT$(" ",17-LEN(Msi_id$))
6796          Disc$(1)=Disc$(1)&"- Start-up mass storage unit specifier."
6798          Dd=1
6800        ELSE
6802          Dd=0
6804        END IF
6806      END IF
6808 Disk: !
6810      ! ................ customize system drives here ..................
6812      ! Follow format with - after unit specifier, description is
6814      ! optional but recommended.
6816      ! ................................................................
6818      !
6820      SELECT TRIM$(Sys_id$)
6822      CASE "S300:20"
6824        Disc$(Dd+1)=":,1404           - HP 9127 single 5.25 floppy drive"
6826        Disc$(Dd+2)=":,1400,1         - HP 9153 microfloppy drive"
6828        Disc$(Dd+3)=":,1400,0         - HP 9153 hard disk"
6830 !      Disc$(Dd+4)=":,1400,0,3       - HP 9153 hard disk PROGS volume"
6832 !      Disc$(Dd+5)=":,700,0,0        - HP 9133H hard disk volume 0"
6834 !      Disc$(Dd+6)=":,700,0,1        - HP 9133H hard disk volume 1"
6836      !
6838        Dd=Dd+3! add the number of drive specifiers above
6840      !
6842      CASE "PC300"
6844        Disc$(Dd+1)=":,1500,0         - LIF/PC a: drive"
6846        Disc$(Dd+2)=":,1500,1         - LIF/PC b: drive"
6848        Disc$(Dd+3)=":,1500,2         - LIF/PC c: drive PROGS volume"
6850        Disc$(Dd+4)=":,1500,3         - LIF/PC c: drive DATA  volume"
6852        Disc$(Dd+5)=":,1500,4         - LIF/PC c: drive UTILS volume"
6854        Disc$(Dd+6)=":DOS,A           - DOS/PC A: drive"
6856        Disc$(Dd+7)=":DOS,B           - DOS/PC B: drive"
6858        Disc$(Dd+8)=":DOS,C           - DOS/PC C: drive \BLP subdirectory
  => "
6860        Dd=Dd+8
6862      CASE ELSE
6864        BEEP
6866        DISP "You need to define your diskdrives in Select_disk for:";Sys
  => _id$
6868        PAUSE
6870      END SELECT
6872      ! ................................................................
6874      !
6876      CALL Menu_scroll(Displ$,Title$,Disc$(*),Dd,Pt,Choose(*))
6878      IF Pt=0 THEN
6880        Diskdrive$="NO DISK"
6882      ELSE
6884        Dd=POS(Disc$(Choose(Pt)),"-")-1 ! find -
```

```
6886          IF Dd>5 THEN  !  valid msus
6888              Diskdrive$=TRIM$(Disc$(Choose(Pt))[1,Dd])
6890          ELSE
6892              DISP " ERROR in reading MSUS from string, - chr not found. "
6894              BEEP
6896              CALL Pause_key_on
6898              Diskdrive$="NO DISK"
6900          END IF
6902      END IF
6904 Diskselected:OFF KEY
6906      SUBEXIT
6908   SUBEND
6910   !
6912   ! ********************************************************
6914   !
6916   SUB Enter_file(T_f(*),INTEGER Datacount,Id$)
6918 Enter_file:  !
6920      OPTION BASE 1
6922      COM /Files/ Diskdrive$[20],Filename$[10]
6924      ON ERROR CALL Errortrap
6926      ASSIGN @Datapath TO Filename$&Diskdrive$
6928      ENTER @Datapath;Status$
6930      ENTER @Datapath;Id$
6932      ENTER @Datapath;Datacount
6934      ENTER @Datapath;Datacount
6936      ALLOCATE Temp(Datacount,2)
6938      ENTER @Datapath;Temp(*)
6940      ASSIGN @Datapath TO *
6942      IF SIZE(T_f,1)>=Datacount THEN
6944          FOR I=1 TO Datacount
6946              T_f(I,1)=Temp(I,1)
6948              T_f(I,2)=Temp(I,2)
6950          NEXT I
6952      ELSE
6954          DISP "SIZE OF FILE PASSED TO Enter_file=";SIZE(T_f,1);
6956          DISP " IS TOO SMALL. PROCESS ABORTED."
6958          BEEP
6960          WAIT 1.8
6962          Datacount=0
6964          Id$=""
6966      END IF
6968      DEALLOCATE Temp(*)
6970      OFF ERROR
6972   SUBEND
6974 !
6976 ! *****************************************************************
6978 !
6980   SUB Menu_scroll(D$,T$,Items$(*),INTEGER Item_cnt,To_select,Choose(*))
6982 Menu_scroll:! Original: 22 Jun 1987, Galen Koepke, NBS 723.04
6984              ! Revision: 02 Dec 1987
6986             !
6988             ! A general purpose menu utility for scrolling items and
6990             ! selecting a given number of them.
6992             ! The items are arranged in screens of 15 items each and
6994             ! the user may access screens via softkeys. There may be
6996             ! up to 40 screens or 600 items to choose from.
6998             ! Items$(*) contains the item descriptions
7000             ! Item_cnt is the number of items in Items$(*)
7002             ! Choose(*) is dimensioned to the number of required choices
7004             !          and will be filled with the item numbers chosen.
7006             ! To_select is the number of required choices.
7008             !
7010      OPTION BASE 1
7012      PRINTER IS CRT
7014      DEG
7016      GOSUB Def_variables
7018      GOSUB Define_screens
7020      GOSUB Make_selections
7022      IF Null_file THEN  ! reset to zero
7024          Item_cnt=0
7026          Items$(1)=""
7028          To_select=0    ! no valid selections
7030      END IF
7032      SUBEXIT
7034      !
7036      ! ///////////////////////////////////////////////////
7038      !
7040 Def_variables:!
7042 !    COM /Interrupts/ INTEGER Intr_prty
7044      COM /Bugs/ INTEGER Bug1,Bug2,Bug3,Printer
7046      COM /Sys/ Sys_id$[10]
7048      !
7050      INTEGER Screen_cnt,Items_per_scn,First_item(40),Last_item(40)
7052      INTEGER I,J,K,First_line,Last_line,Active_screen,Pointer,Last_pt
7054      INTEGER Local_prty,Skips,Knobcount,Pointeractive,K0,Null_file
7056      INTEGER Exit_flag,Temp
7058      DIM Marker$[8],Test$[160]
7060      !
7062      ! initialize parameters
7064      !
7066      Local_prty=Intr_prty
7068      IF Local_prty<1 THEN Local_prty=10
7070      IF LEN(Sys_id$)=0 THEN Sys_id$=SYSTEM$("SYSTEM ID")
7072      IF Item_cnt<1 THEN
7074          Null_file=1
7076          Item_cnt=1
7078          To_select=0
7080          Items$(1)="*** Empty ***"
7082      ELSE
7084          Null_file=0
7086      END IF
7088      IF To_select>Item_cnt THEN To_select=Item_cnt
7090      Skips=0
7092      Knobcount=0
7094      Doneflag=0
7096      Marker$="===>"&RPT$(CHR$(8),4)
7098      RETURN
7100      !
7102      ! ///////////////////////////////////////////////////
7104      !
7106 Define_screens:! Set up screens of 15 items each.
7108               !
7110      Items_per_scn=15    ! Maximum number of displayable items
7112      IF INT(Item_cnt/Items_per_scn)=Item_cnt/Items_per_scn THEN
7114          Screen_cnt=INT(Item_cnt/Items_per_scn)
7116      ELSE
7118          Screen_cnt=INT(Item_cnt/Items_per_scn)+1
7120      END IF
7122      J=1
7124      FOR I=1 TO Screen_cnt   ! set up each screen
7126          First_item(I)=J
7128          IF J+Items_per_scn-1<Item_cnt THEN
7130              Last_item(I)=J+Items_per_scn-1
7132              J=J+Items_per_scn
7134          ELSE
7136              Last_item(I)=Item_cnt
7138          END IF
7140      NEXT I
7142      RETURN
7144      !
7146      ! //////////////////////////////////////////////////
7148      !
```

```
7150 Make_selections:! MENU setup and use.
7152       Active_screen=1    ! first screen is active
7154       First_line=2       ! first printed line on screen = 2 or greater.
7156       GOSUB Write_screen  ! activate screen at Active_screen
7158                           ! and set First_line and Last_line for Pointer
7160                           ! write Marker$ to first non-selected line.
7162       K0=0                ! Keys start at zero
7164       Exit_flag=0         ! allow ENTER key to exit when selections filled.
7166 Key_loop:  !
7168       ON KBD,Local_prty GOSUB Process_kbd
7170       ON KNOB .01,Local_prty GOSUB Move_pointer
7172       IF Skips<To_select THEN
7174           DISP D$
7176           IF To_select>1 THEN
7178               Test$=" Select  "&VAL$(Skips+1)&" of "&VAL$(To_select)
7180           ELSE
7182               Test$=" Select"
7184           END IF
7186           ON KEY K0 LABEL Test$,Local_prty GOSUB Select_item
7188       ELSE
7190           IF To_select>0 THEN
7192               DISP " Selection process complete ..."
7194           ELSE
7196               DISP " Menu for information only ... "
7198           END IF
7200           ON KEY K0 LABEL "Accept",Local_prty GOTO Exit_line
7202       END IF
7204       IF Active_screen<Screen_cnt THEN
7206           ON KEY K0+1 LABEL " Next   Screen",Local_prty GOSUB Next_screen
7208       ELSE
7210           OFF KEY K0+1
7212       END IF
7214       IF Active_screen>1 THEN
7216           ON KEY K0+2 LABEL " Last   Screen",Local_prty GOSUB Last_screen
7218       ELSE
7220           OFF KEY K0+2
7222       END IF
7224       IF Skips>0 THEN
7226           ON KEY K0+3 LABEL " Reset  Select",Local_prty GOSUB Select_reset
7228       ELSE
7230           OFF KEY K0+3
7232       END IF
7234       IF To_select>0 THEN
7236           ON KEY K0+4 LABEL " Abort ",Local_prty GOTO Escape_line
7238       ELSE
7240           OFF KEY K0+4
7242       END IF
7244       IF Screen_cnt>2 THEN
7246           ON KEY K0+6 LABEL "Jump to Screen",Local_prty GOSUB Jump_to_scn
7248       ELSE
7250           OFF KEY K0+6
7252       END IF
7254       IF Exit_flag THEN Exit_line
7256       GOTO Key_loop
7258 Escape_line:Skips=0
7260       MAT Choose= (0)
7262       To_select=0
7264 Exit_line:OFF KEY
7266       OFF KNOB
7268       OFF KBD
7270       OUTPUT KBD;CHR$(255)&CHR$(75);
7272       PRINT CHR$(128);
7274       ! everything cleared, now go back to work.
7276       RETURN
7278       !
7280       ! ////////////////////////////////////////////////////
7282       !
7284 Next_screen:      !
7286       OFF KBD
7288       OFF KNOB
7290       OFF KEY
7292       IF Active_screen=Screen_cnt THEN RETURN
7294       Active_screen=Active_screen+1
7296       GOSUB Write_screen
7298       RETURN
7300       !
7302       ! ///////////////////////////////////////////////////
7304       !
7306 Last_screen:      !
7308       OFF KBD
7310       OFF KNOB
7312       OFF KEY
7314       IF Active_screen=1 THEN RETURN
7316       Active_screen=Active_screen-1
7318       GOSUB Write_screen
7320       RETURN
7322       !
7324       ! ///////////////////////////////////////////////////
7326       !
7328 Jump_to_errors:DISP " Not a valid screen number ... try again. "
7330       BEEP
7332       WAIT 1.8
7334 Jump_to_scn: !
7336       OFF KBD
7338       OFF KNOB
7340       OFF KEY
7342       DISP " ENTER the screen number desired (1 to ";Screen_cnt;").";
7344       LINPUT Test$
7346       Test$=TRIM$(Test$)
7348       IF LEN(Test$)=0 THEN Jump_to_return
7350       ON ERROR GOTO Jump_to_errors
7352       Temp=INT(VAL(Test$))
7354       OFF ERROR
7356       IF Temp<1 OR Temp>Screen_cnt THEN Jump_to_errors
7358       Active_screen=Temp
7360       GOSUB Write_screen
7362 Jump_to_return: !
7364       DISP CHR$(12)
7366       Test$=""
7368       RETURN
7370       !
7372       ! ///////////////////////////////////////////////////
7374       !
7376 Select_item:!
7378       OFF KBD
7380       OFF KNOB
7382       OFF KEY
7384       IF NOT Pointeractive THEN
7386           DISP "NO additional selections for this screen."
7388           BEEP
7390           WAIT 2
7392           DISP CHR$(12);
7394           RETURN
7396       END IF
7398       IF Skips=To_select THEN
7400           IF To_select=0 THEN
7402               DISP "This menu is for information only,";
7404               DISP " no selection allowed."
7406           ELSE
7408               DISP "All selections have been filled,";
7410               DISP " 'Select Reset' to repeat."
7412           END IF
```

```
7414          BEEP
7416          WAIT 2
7418          DISP CHR$(12);
7420          RETURN
7422      END IF
7424      Skips=Skips+1
7426      Choose(Skips)=First_item(Active_screen)+Pointer-First_line
7428      PRINT CHR$(129);  ! inverse video
7430      PRINT TABXY(10,Pointer);Items$(Choose(Skips))
7432      PRINT CHR$(128);
7434      PRINT TABXY(1,Pointer);
7436      SELECT Pointer
7438      CASE First_line
7440          GOSUB Point_forward
7442      CASE Last_line
7444          GOSUB Point_backward
7446      CASE ELSE
7448          ! move forward unless it requires wrapping to beginning.
7450          IF Skips-1>0 THEN  ! check for selected items.
7452              I=Pointer-First_line
7454              LOOP
7456                  K=0
7458                  FOR J=1 TO Skips
7460                      IF First_item(Active_screen)+I=Choose(J) THEN K=1
7462                  NEXT J
7464              EXIT IF K=0
7466                  I=I+1
7468                  IF I+First_line>Last_line THEN K=-1
7470              EXIT IF K=-1
7472              END LOOP
7474              IF K=0 THEN
7476                  GOSUB Point_forward
7478              ELSE
7480                  GOSUB Point_backward
7482              END IF
7484          ELSE
7486              GOSUB Point_forward
7488          END IF
7490      END SELECT
7492      RETURN
7494      !
7496      ! ////////////////////////////////////////////////////
7498      !
7500 Select_reset:      !Clear Choose file
7502      OFF KBD
7504      OFF KNOB
7506      OFF KEY
7508      Skips=0
7510      MAT Choose= (0)
7512      GOSUB Write_screen
7514      RETURN
7516      !
7518      ! ////////////////////////////////////////////////////
7520      !
7522 Process_kbd:! Allow use of arrows and enter key in addition to soft.
7524      Test$=KBD$
7526      IF LEN(Test$)=1 AND Test$[1,1]<>CHR$(32) THEN
7528          BEEP 80.,.1
7530          RETURN
7532      END IF
7534      IF Test$[1,1]=CHR$(32) THEN GOSUB Point_forward
7536      IF Test$[1,1]<>CHR$(255) THEN RETURN
7538      SELECT Test$[2,2]
7540      CASE CHR$(255)
7542                ! do nothing
7544      CASE "V","T"
7546          GOSUB Point_forward
7548      CASE "^","W"
7550          GOSUB Point_backward
7552      CASE "E","s","t","&"
7554          IF Skips<To_select THEN
7556              GOSUB Select_item
7558          ELSE
7560              ! exit routine
7562              Exit_flag=1
7564          END IF
7566      CASE ELSE
7568          BEEP 80.,.1
7570      END SELECT
7572      Test$=""
7574      RETURN
7576      !
7578      ! ////////////////////////////////////////////////////
7580      !
7582 Point_forward:Knobcount=5
7584      GOSUB Move_pointer
7586      RETURN
7588 Point_backward:Knobcount=-5
7590      GOSUB Move_pointer
7592      RETURN
7594      !
7596      ! ////////////////////////////////////////////////////
7598      !
7600 Jog_pointer:! Move the selection pointer on the active screen.
7602            ! without regard to selected values
7604      IF Knobcount>0 THEN  ! Move forward
7606          Pointer=Pointer+1
7608      ELSE                 ! Move backward
7610          Pointer=Pointer-1
7612      END IF
7614      IF Pointer<First_line THEN Pointer=Last_line
7616      IF Pointer>Last_line THEN Pointer=First_line
7618      RETURN
7620      !
7622      ! ///////////////////////////////////////////////////////////
7624      !
7626 Move_pointer:! Control pointer to avoid re-selection of items
7628      IF NOT Pointeractive THEN RETURN  ! No selections to be made.
7630      Knobcount=Knobcount+KNOBX-KNOBY
7632      IF ABS(Knobcount)<4 THEN RETURN
7634      Last_pt=Pointer
7636      GOSUB Jog_pointer
7638      IF Skips>0 THEN
7640          LOOP
7642              J=Pointer-First_line
7644              FOR I=1 TO Skips
7646                  IF First_item(Active_screen)+J=Choose(I) THEN J=999
7648              NEXT I
7650              IF J=999 AND Pointer=Last_pt THEN Pointeractive=0
7652          EXIT IF Pointeractive=0
7654              IF J=999 THEN GOSUB Jog_pointer
7656          EXIT IF J<>999
7658          END LOOP
7660      END IF
7662      Knobcount=0
7664      OUTPUT KBD;CHR$(255)&CHR$(84);  ! Bring screen home
7666      IF Last_pt=Last_line THEN PRINT CHR$(132);
7668      PRINT "    ";
7670      IF Pointeractive THEN ! Pointer active
7672          IF Pointer=Last_line THEN
7674              PRINT CHR$(132);
7676          ELSE
```

```
7678             PRINT CHR$(128);
7680         END IF
7682         PRINT TABXY(1,Pointer);Marker$;CHR$(128);
7684     END IF
7686     RETURN
7688     !
7690     ! ////////////////////////////////////////////////
7692     !
7694 Write_screen:! Write the screen pointed to by Active_screen
7696     ! home and clear screen
7698     OUTPUT KBD;CHR$(255)&CHR$(84)&CHR$(255)&CHR$(75);
7700     Knobcount=KNOBX-KNOBY   ! Clear knob and keyboard
7702     Knobcount=0
7704     Test$=KBD$
7706     Test$=""
7708     !
7710     PRINT TABXY(1,First_line-1);CHR$(132);" Item #| Screen #";
7712     PRINT USING "#,2D,4A,2D,3A";Active_screen," of ";Screen_cnt;" | "
7714     PRINT T$;RPT$(" ",51-LEN(T$));CHR$(128);
7716     J=0
7718     REPEAT
7720         IF J=Last_item(Active_screen)-First_item(Active_screen) THEN
7722             PRINT CHR$(132);
7724             PRINT TABXY(1,First_line+J);RPT$(" ",80)
7726         ELSE
7728             PRINT CHR$(128);
7730         END IF
7732         PRINT TABXY(5,First_line+J);
7734         PRINT USING "3D,A,#";First_item(Active_screen)+J,"|"
7736         IF Skips>0 THEN  ! make this line inverse video
7738             FOR I=1 TO Skips
7740                 IF First_item(Active_screen)+J=Choose(I) THEN
7742                     PRINT CHR$(129);
7744                 END IF
7746             NEXT I
7748         END IF
7750         PRINT TABXY(10,First_line+J);Items$(First_item(Active_screen)+J)
7752         J=J+1
7754     UNTIL J>=(Last_item(Active_screen)-First_item(Active_screen)+1)
7756     Last_line=Last_item(Active_screen)-First_item(Active_screen)
7758     Last_line=Last_line+First_line
7760     !
7762     ! set marker to first non-selected item.
7764     !
7766     Pointeractive=0
7768     IF To_select>0 THEN Pointeractive=1
7770     IF Skips>0 AND Pointeractive=1 THEN ! find first non-selected item
7772         J=0
7774         LOOP
7776             Pointer=First_line+J
7778             FOR I=1 TO Skips
7780                 IF First_item(Active_screen)+J=Choose(I) THEN Pointer=0
7782             NEXT I
7784         EXIT IF Pointer<>0
7786             J=J+1
7788             IF First_line+J>Last_line THEN
7790                 Pointeractive=0
7792                 Pointer=First_line
7794             END IF
7796         EXIT IF Pointer<>0
7798         END LOOP
7800     ELSE
7802         Pointer=First_line
7804     END IF
7806     IF Pointeractive THEN
7808         IF Pointer=Last_line THEN
7810             PRINT CHR$(132);
7812         ELSE
7814             PRINT CHR$(128);
7816         END IF
7818         PRINT TABXY(1,Pointer);Marker$;CHR$(128);
7820     END IF
7822     RETURN
7824 SUBEND
7826 !
7828 ! ***********************************************************
7830 !
7832 SUB Errortrap
7834 Errortrap: ! Original: 13 Nov 1984
7836           ! Revision: 02 Dec 1987
7838     ! Trap most errors here
7840     COM /Files/ Diskdrive$[20],Filename$[10]
7842     DIM File$[20],Test$[160],What$[20],Ac$[5]
7844     BEEP 400,.6
7846     SELECT ERRN
7848     CASE 54
7850         DISP "DUPLICATE FILE NAME:  ";Filename$;
7852         DISP "....PURGE old one? (Y/N)";
7854         LINPUT What$
7856         What$=TRIM$(What$)
7858         SELECT What$[1,1]
7860         CASE "Y","y"
7862             PURGE Filename$&Diskdrive$
7864         CASE ELSE
7866             Ac$="VALID"
7868             CALL Enterfilename(Ac$)
7870         END SELECT
7872     CASE 52,53
7874         DISP "Improper FILE NAME --- ENTER NEW FILE NAME";
7876         OUTPUT 2 USING "#,K,K";" #";Filename$
7878         LINPUT Filename$
7880         Filename$=TRIM$(Filename$)
7882     CASE 56
7884         DISP "FILE:  ";Filename$;" is not on this disk, please insert";
7886         DISP " correct disk"
7888         CALL Pause_key_on
7890     CASE 64
7892         DISP "This disk is full, PLEASE insert clean disk"
7894         CALL Pause_key_on
7896     CASE 56
7898         DISP "DATA INPUT disk must be in drive!!  ";
7900         DISP "...CONTINUE when ready."
7902         CALL Pause_key_on
7904     CASE 72,73,76
7906         DISP Diskdrive$;
7908         DISP " is not available, type correct";
7910         DISP " unit specifier (ie. ':,707,0').";
7912         OUTPUT 2 USING "K,#";Diskdrive$
7914         LINPUT Diskdrive$
7916     CASE 80
7918         DISP "CHECK DISK drive door!"
7920         CALL Pause_key_on
7922     CASE ELSE
7924         DISP ERRM$;" 'CONTINUE' when fixed"
7926         CALL Pause_key_on
7928     END SELECT
7930     DISP CHR$(12)
7932     SUBEXIT
7934 SUBEND
7936 !
7938 ! ***********************************************************
7940 !
```

```
7942   SUB Pause_key_on
7944 Pause_key_on: ! Make sure that CONTINUE key exists.
7946               ! Original: 02 Dec 1987
7948               ! Revision: 02 Dec 1987
7950       COM /Sys/ Sys_id$[10]
7952       IF Sys_id$[1,4]="S300" THEN ! reset to S300 system keys
7954           CONTROL KBD,15;0
7956           CONTROL CRT,12;2
7958           LOAD KEY
7960       END IF
7962       PAUSE
7964       IF Sys_id$[1,4]="S300" THEN  ! set to S200 compatible keys
7966           OUTPUT KBD USING "K,#";"SCRATCH KEY X"
7968           CONTROL KBD,15;1
7970           CONTROL CRT,12;0
7972       END IF
7974       SUBEXIT
7976   SUBEND
7978   !
7980   ! ***********************************************************
7982   !
```

*U.S. GOVERNMENT PRINTING OFFICE:1993-774-025/89040

NIST *Technical Publications*

Periodical

Journal of Research of the National Institute of Standards and Technology—Reports NIST research and development in those disciplines of the physical and engineering sciences in which the Institute is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Institute's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Institute's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NIST, NIST annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

Applied Mathematics Series—Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NIST under the authority of the National Standard Data Act (Public Law 90-396). NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published bi-monthly for NIST by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements are available from ACS, 1155 Sixteenth St., NW, Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Institute on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NIST under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NIST administers this program in support of the efforts of private sector standardizing organizations.

Consumer Information Series—Practical information, based on NIST research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.

Order the **above** *NIST publications from: Superintendent of Documents, Government Printing Office, Washington, DC 20402.*

Order the **following** *NIST publications—FIPS and NISTIRs—from the National Technical Information Service, Springfield, VA 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NIST pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NIST Interagency Reports (NISTIR)—A special series of interim or final reports on work performed by NIST for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service, Springfield, VA 22161, in paper copy or microfiche form.

U.S. Department of Commerce
National Institute of Standards and Technology
325 Broadway
Boulder, Colorado 80303-3328