Design and Automation of MEDUSA
(Materials and Electronic Device Universal System Analyzer)

by

Phillip Johnson

Thesis submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Masters of Science

in

Electrical Engineering

APPROVED:

Dr. L. C. Burton

Dr. F. W. Stephenson

Dr. S. Onishi

February 1990
Blacksburg, VA
Design and Automation of MEDUSA

(Materials and Electronic Device Universal System Analyzer)

by

Phillip L. Johnson

Dr. L.C. Burton, Chairman

Electrical Engineering

(ABSTRACT)

MEDUSA (Materials and Electronic Devices Universal System Analyzer) is a computer controlled automated workstation capable of conducting eight different experiments, under different independent parameters, and plotting twenty-eight different graphs representing basic semiconductor diode and transport characteristics. This thesis discusses the methodology of computer automation, and the development of the MEDUSA experimental test station.

MEDUSA is divided into four different sections: a controlling batch file, a parameter selection routine (PARAMETER), an experimental running routine (RUNIT), and a data manipulation/plotting routine (GRAPHICS). MEDUSA conducts these eight experiments (capacitance and conductance versus time, voltage, current versus voltage, van der Pauw, and four-point resistivity) over a temperature range of 10-600K, with minimal operator interaction.

The graphics routines, using elemental semiconductor equations, process the data, and plots high quality graphs suitable for publication. Device and material results are shown to substantiate the validity of this automated system.
ACKNOWLEDGEMENTS

The author would like to thank Dr. L. C. Burton, who supported my work for more years than he should have, and for giving me much valuable information which will stand me in good stead in the future.

Also, much gratitude to Dr. F. W. Stephenson and Dr. S. Onishi for allowing me to panic in the last days and making time for my defense in their busy schedules. Eric Cole deserves more than thanks for coming up with the MEDUSA concept, and giving me much direction in software development. Also, Sidhartha Sen deserves a second Master's degree for all his help, advise, and friendship. Thanks to Chris Turman for bringing me back to the real world (yeah, thanks Chris!) and keeping things in perspective with his "wit". Also, Gary Kunselman, thanks for your support, editing skills and especially your sense of humor in the early morning.

I appreciate the backing of everyone else who has supported me in the past years (but not to the detractors), including Ashok Vaseashta, Eric Ellis, Monty Hayes, Chen, Keith Kasprak, Lynn Hellbaum, and In Yoo.

Finally, special thanks to Jana Dajani, for her complete support in this time of trauma, and her persistent nudging.
# TABLE OF CONTENTS

ACKNOWLEDGEMENTS ................................................................. iii  

TABLE OF CONTENTS .................................................................. iv  

LIST OF FIGURES ......................................................................... vii  

LIST OF TABLES ........................................................................... x  

CHAPTER 1: INTRODUCTION ......................................................... 1  
  1.1 Introduction ................................................................. 1  
  1.2 Literature Review ....................................................... 2  
  1.3 Thesis Objective ......................................................... 4  

CHAPTER 2: DEVICE PARAMETERS AND HARDWARE OVERVIEW ....... 6  
  2.1 Semiconductor Characteristics ....................................... 6  
    2.1.1 Transport Characteristics .................................. 6  
    2.1.2 Diode Characteristics ...................................... 11  
    2.1.3 Trap Parameters ............................................. 16  
  2.2 Experimental Test Station ............................................. 17  
  2.3 DUT Hardware .......................................................... 17  
  2.4 IEEE Interface .......................................................... 20  

CHAPTER 3: EXPERIMENTAL TECHNIQUES .................................. 21  
  3.1 IEEE-488 Interface ...................................................... 21  
    3.1.1 IEEE BUS Lines ............................................. 21  
    3.1.2 General Interface Protocol ............................. 24  
    3.1.3 Running the IEEE-488 Interface .................. 26  
  3.2 PROGRAMMING LANGUAGE ............................................. 26  
    3.2.1 Language Requirements .................................. 26  
    3.2.2 Language Chosen............................................. 27  
  3.3 MEDUSA'S SOFTWARE .................................................... 28  
    3.3.2 PARAMETER .................................................. 30  
    3.3.3 RUNIT .......................................................... 34  
    3.2.4 GRAPHICS ...................................................... 52  

CHAPTER 4: EXPERIMENTAL RESULTS AND DISCUSSION ............ 73  
  4.1 Capacitance Experiments ............................................. 73  
    4.1.1 Capacitance vs. Time ....................................... 73  
    4.1.2 Capacitance vs. Voltage ................................... 73  
    4.1.3 \(1/C^2\) vs. Voltage ........................................ 76  
    4.1.4 Capacitance vs. Temperature .......................... 76  
    4.1.5 DLTS Spectrum .............................................. 76  
    4.1.6 Barrier Height versus Temperature .................. 80  
    4.1.7 Dopant Profile ................................................ 80  
  4.2 Conductance Measurements ............................................ 80  
    4.2.1 Conductance vs. Time ...................................... 80  
    4.2.2 Conductance vs. Voltage .................................. 84  
    4.2.3 Conductance vs. Temperature ......................... 84  

iv
4.3 Current Measurements .................................................. 84
  4.3.1 Current vs. Voltage ................................................. 84
  4.3.2 Natural Log Current vs. Voltage .............................. 88
  4.3.3 Ln(I/T^2) vs. qV/kT ............................................... 88
  4.3.4 \( I_{sat} \) vs. Temperature ................................ 88
  4.3.5 Ideality Factor vs. Temperature .............................. 88
  4.3.6 Barrier Height vs. Temperature ............................... 88
4.4 Four Point Resistivity .............................................. 94
4.5 van der Pauw Measurements ........................................ 94

CHAPTER 5: CONCLUSIONS AND FUTURE RECOMMENDATIONS ............. 99

REFERENCES ........................................................................ 101

APPENDIX ........................................................................... 103

Vita .................................................................................... 242
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Three connector layouts for the room temperature van der Pauw Measurements</td>
<td>7</td>
</tr>
<tr>
<td>2.2</td>
<td>The function ( f(R_a/R_b) )</td>
<td>9</td>
</tr>
<tr>
<td>2.3</td>
<td>Energy diagram for metal n-type semiconductor contact</td>
<td>12</td>
</tr>
<tr>
<td>2.4</td>
<td>Ideal 1/C(^2) versus Voltage Plot</td>
<td>14</td>
</tr>
<tr>
<td>2.5</td>
<td>MEDUSA hardware layout</td>
<td>18</td>
</tr>
<tr>
<td>2.6</td>
<td>TO-8 can with mounted DUT</td>
<td>19</td>
</tr>
<tr>
<td>3.1</td>
<td>IEEE-488 BUS layout</td>
<td>23</td>
</tr>
<tr>
<td>3.2</td>
<td>IEEE-488 timing diagram</td>
<td>25</td>
</tr>
<tr>
<td>3.3</td>
<td>Flow chart MEDUSA.BAT</td>
<td>29</td>
</tr>
<tr>
<td>3.4</td>
<td>First flow chart for PARAMETER</td>
<td>31</td>
</tr>
<tr>
<td>3.5</td>
<td>Second flow chart for PARAMETER</td>
<td>33</td>
</tr>
<tr>
<td>3.6</td>
<td>First flow chart for RUNIT</td>
<td>38</td>
</tr>
<tr>
<td>3.7</td>
<td>Time run flow chart</td>
<td>39</td>
</tr>
<tr>
<td>3.8</td>
<td>Room temperature flow chart</td>
<td>39</td>
</tr>
<tr>
<td>3.9</td>
<td>Flow chart for temperature run</td>
<td>41</td>
</tr>
<tr>
<td>3.10</td>
<td>Flow chart for experiment section</td>
<td>42</td>
</tr>
<tr>
<td>3.11</td>
<td>Flow chart for C-t and G-t experiments</td>
<td>44</td>
</tr>
<tr>
<td>3.12</td>
<td>Flow chart for C-G-V experiment</td>
<td>46</td>
</tr>
<tr>
<td>3.13</td>
<td>Flow chart for C-G experiment</td>
<td>47</td>
</tr>
<tr>
<td>3.14</td>
<td>Flow chart for I-V experiment</td>
<td>48</td>
</tr>
<tr>
<td>3.15</td>
<td>Flow chart for C-V experiment</td>
<td>50</td>
</tr>
<tr>
<td>3.16</td>
<td>Flow chart for van der Pauw experiment</td>
<td>51</td>
</tr>
<tr>
<td>3.17</td>
<td>Flow chart for 4-Point resistance</td>
<td>53</td>
</tr>
<tr>
<td>3.18</td>
<td>Flow chart for RUNIT end</td>
<td>54</td>
</tr>
</tbody>
</table>
Figure 4.15  Ideality Factor versus Temperature ........................................... 92
Figure 4.16  I-V Barrier Height versus Temperature ........................................ 93
Figure 4.17  4-Point Resistivity versus Temperature ......................................... 95
Figure 4.18  Resistivity versus Temperature .................................................... 96
Figure 4.19  Mobility versus Temperature ...................................................... 97
Figure 4.14  Carrier Concentration versus Temperature .................................... 98
LIST OF TABLES

Table 3.1 IEEE-488 line connections ........................................ 22
Table 3.2 Group experiments ..................................................... 32
Table 3.3 Group I parameters ..................................................... 32
Table 3.4 Group II parameters .................................................... 35
Table 3.5 Group III parameters ................................................... 35
Table 3.6 Group IV parameters ................................................... 35
Table 3.7 Experiments and their related instruments ....................... 36
Table 3.8 Standard choice sequence for GRAPHICS ....................... 56
Table 3.9 Available graphs in the program GRAPHICS ...................... 56
Table 3.10 Graphs and their related parameters ............................. 59
Table 3.11 MODIFY data options .................................................. 64
Table 3.12 PLOTTER function key definitions ............................... 67
Table 3.13 Line types ............................................................... 69
Table 3.14 Symbol types ............................................................ 69
Table 3.15 Color types .............................................................. 69
Table 3.16 PLOTTER information block ....................................... 70
CHAPTER 1: INTRODUCTION

1.1 Introduction

Solid state electronics is based largely upon semiconductor technology. Certain design principles and semiconductor electrical characterization form the basis for current electronic devices. Device quality and electrical parameters are found by electrical testing. Electrical characterization of materials and devices often entail tedious repetitive measurements. Often data points must be taken over a wide temperature range, increasing the time necessary for an experiment. Sometimes, a temperature scan may take a week or more to complete. Another important factor is experimental error. As electrical data frequently changes over time, a measurement taken at different times may yield conflicting results. So, if the time between measurements changes, inconsistencies may be expected. In order to minimize these errors and improve the data acquisition rate, an experimental test station was conceived, developed and implemented. This environment, nicknamed MEDUSA (Materials and Electronic Device Universal System Analyzer) allows the scientist or engineer to concentrate on the more important tasks of analyses and interpretation. The researcher inserts the device under test (DUT), enters which set of experiments to conduct, chooses the various experimental parameters, and then allows the test to run. After concluding the experiments, the controlling software enters a graphics package. This package, depending on the electrical characteristics relevant to the experiment, processes the raw data and converts this data into a graphic display.

This experimental test station is able to conduct eight different electrical experiments, and produce twenty-eight different graphs. The experiments were chosen specifically to produce information necessary to explore the electrical characteristics of both doped and undoped semiconductor regions. These characteristics are divided into two major sections; material and device characteristics. Material characteristics involve ion bombardment damage, doping and trap densities, mobility, and resistivity information, whereas diode characteristics address built-in potential, barrier
height, diode ideality factor, and saturation current. Such characteristic information is vital for quality control considerations and in making "good" and reproducible devices. Ion implantation damage information is especially important for GaAs. The majority of GaAs devices employ ion implantation techniques. When GaAs is implanted, the accelerated ions cause lattice disruptions resulting in an amorphous layer. Even though annealing restores the crystal lattice and activates the dopant, some damage remains [1]. Characterizing this damaged layer allows the researcher to probe the ion implanted layer. This characterization may be accomplished utilizing Deep Level Transient Spectroscopy (DLTS), which is further discussed in Chapter 2.

Doping profiles, mobility, resistivity and trap information provide a designer the parameters necessary for device optimization. The research described is aimed towards obtaining these parameters in a reliable, efficient and timely manner.

1.2 Literature Review

MEDUSA requires intensive data manipulation. An in-depth examination of the equations involved is given in Chapter 2. The resistivity, mobility, and carrier concentration measurements utilize van der Pauw's seminal work [2]. The equations for the average resistivity, mobility and carrier concentration for an arbitrary shape were obtained from Hemenger [3]. Carrier profiling requires expanding van der Pauw's work to take into account the different concentrations throughout the implanted layer [4].

Various diode characteristics can be uncovered from experimental measurements utilizing equations found in the literature [5-9]. The C-V barrier height, doping density, and the build-in potential all emerge from the $1/C^2$ versus voltage characteristic [5-7]. The C-V barrier height is estimated from the slope of the $1/C^2$ versus voltage slope [5-7]. The doping density is also derived from the same slope [5]. The built-in potential is the zero capacitance point of the $1/C^2$ versus voltage curve. The ideality factor, saturation current, and the I-V barrier height emerge from the current voltage equation [5-7]. Finally, the trap parameters originate from capacitance transient measurements at different temperatures [8,9].
Literature is quite extensive concerning laboratory automation in its various guises, both large factory and laboratory layouts. Now, with the advent of an affordable computer system, an automated laboratory test facility is a practical alternative to manual measurements. Various system layouts are presented in the literature [10-13]. The most complete step-by-step automated laboratory design is given by Dessy [12], where an overview is presented of the entire process of building an automated laboratory from computer selection to system design.

The central instrument of any automated work station is a computer. Currently, a wide choice between computer types (mini-computer, micro-computer, or mainframe) allows more flexibility in the design of a work station. Several informative articles explain the differences between the computers and their architectures [14,15]. The articles explain in great detail the differences between central processing units (CPUs), and their utilization of memory. The architecture most frequently used in smaller automation configurations features either the Intel 8088-80X86 micro-processor employed by the IBM compatible computers, or the MC68000 micro-processor series utilized by Apple. MEDUSA employs an IBM-AT, since it is one of the easiest computers to interface with an IEEE-488 bus.

Most interface manufacturers have a profusion of communications boards, including IEEE-488, RS-232C, and A/D-D/A (Analog/Digital-Digital/Analog) converters [11,16-19]. Several papers give an overview of these three interface schemes [20-22]. The first paper explains serial and parallel communications and their major automation off-shoots, RS232C and IEEE-488, respectively [20]. An explanation of A/D-D/A converter functions and a fairly comprehensive list of commercial vendors for digital interface equipment is given by Dessy [21]. Also, one paper gives examples of working configurations utilizing all three types of interfacing [22]. A more detailed look at the differences between the interfaces and an explanation of the IEEE-488 interface is presented in Chapter 3.

The operating systems (OS) and languages available for a computer are varied and wide ranging. The main OS for micro-computers are UNIX, RT-11, and MS/DOS [23-25]. These articles explain the differences between the three OS and several other types, while suggesting RT-11 is the
best for multi-tasking (doing more than one operation at a time), UNIX for multiple users, and
MS/DOS being a simple, user friendly OS. MS/DOS was chosen as the OS for MEDUSA because
it is easy to use and is the standard system for IBM micro-computers.

The languages available for these systems are even more plentiful than the operating systems
themselves. The computer languages found most often in the laboratory are: BASIC, FORTRAN-77,
C, Pascal, and Forth [25-31]. The differences are elaborated well in Dessy [25], allowing one to make
an informed language selection. References 27-31 are invaluable for showing the syntax of each
language and giving the beginner a chance to learn the language. A BASIC compatible language was
chosen for MEDUSA. The justification is given in Chapter 3.

1.3 Thesis Objective

The main objective of this research was to build a user-friendly automated experiment
workstation. Some of the design considerations are:

• Make the system simple enough to operate so a person unfamiliar with the layout can use
  it with ease in a relatively short time

• Design a hardware configuration capable of conducting a variety of experiments including:
  Capacitance versus Time
  Capacitance versus Voltage
  Conductance versus Voltage
  Current versus Voltage
  Four-point Resistance

• Conduct the above experiments under the following conditions:
  Different experiment dependent parameters (bias, pulse voltage, delay time, hold
time)
  Over a temperature range (10-600K)
  Over an extended time period (life testing)
  At room temperature

• Store the data obtained from the above experiments in a form acceptable to a written data
  manipulation program and understandable to the researcher.

• Convert the data files into graphs displaying different semiconductor characteristics

• Plot the graphs in a format suitable for publication

How these goals were attained is covered in greater detail in the next four chapters. The second
chapter explains the semiconductor equations utilized by MEDUSA in the graphing program. Also an overview of the hardware layout and the basics of digital interfacing are explained in greater detail. Chapter 3 covers IEEE-488 interfacing, the computer language chosen and the programs necessary to implement MEDUSA in the mode described in the paragraph above. Chapter 4 displays the most important semiconductor plots. Finally Chapter 5 presents the conclusions and future recommendations.
CHAPTER 2: DEVICE PARAMETERS AND HARDWARE OVERVIEW

This chapter provides basic information integral to the design and automation of MEDUSA. Basic semiconductor equations, which are used in the graphing procedures are presented, along with an overview of the MEDUSA hardware layout. Also introduced, is a brief introduction to computer interfacing.

2.1 Semiconductor Characteristics

Semiconductor characteristics can be classified under two broad categories: transport and diode characteristics. Transport characteristics yield carrier density (electrons or holes), resistivity, mobility, and their profiles. Diode characteristics provide a quantitative method of describing the quality of the diode, it's breakdown, forward characteristics, and reverse saturation. Diode capacitance transient measurements at different temperatures provide deep level information about the semiconductor.

2.1.1 Transport Characteristics

Transport characteristics, derived from van der Pauw measurements, may be broken into four sections, (i) resistivity, (ii) mobility, (iii) carrier concentration, and (iv) electrical profiles.

(i) Average Resistivity

Resistivity gives the opposition to a voltage or current through a device. This parameter is extremely important, not only for itself, but for other parameters (mobility and carrier concentration) derived from it. Resistivity is found from the van der Pauw experiment, which is a variation of a four-point resistance measurement. Figure 2.1 shows the three configurations necessary for the experiment. When the experiment is conducted in this manner, the four contact positions are inconsequential, provided they are close to the sample periphery. There are three four-point resistance measurements done immediately for calibration (\(R_x\), \(R_m\), and \(R_y\)). These are shown in Figures 2.1 (a)-(c). The
Figure 2.1 Three connector layouts for the room temperature van der Pauw measurements [3]
measurements in (a) and (b) give an average resistivity ($\rho$) using the equation [3]:

$$\rho = \frac{\pi t}{\ln 2} \left( \frac{R_a + R_b}{2} \right) f \left( \frac{R_a}{R_b} \right)$$  \hspace{1cm} (2.1)

where:

$$R_a = \frac{V_a}{I_a}$$
$$R_b = \frac{V_b}{I_b}$$

$t$ = sample thickness

$f(R_a/R_b)$ = function defined in Figure 2.2

(ii) Average Mobility

Mobility ($\mu$) gives a measure of device "speed", a good indication of how quickly the device switches from one state to another. Average $\mu$ may be found utilizing the resistivity calculated above, coupled with another resistance measurement [3]:

$$\mu = 10^8 \left( \frac{\Delta R_c t}{B \rho} \right)$$  \hspace{1cm} (2.2)

where:

$$\Delta R_c = \left| \frac{V_c(\text{on})}{I_c(\text{on})} - \frac{V_c(\text{off})}{I_c(\text{off})} \right|$$  \hspace{1cm} (2.3)

$\Delta R_c$ represents the change in resistance of the configuration shown in Figure 2.1 (c) with both the magnet on and off, and $B$ is the magnetic Field in Gauss.

(iii) Average Carrier Concentration

The average carrier concentration ($n$) gives the number of electrons or holes per unit volume, that are active at the temperature probed. Even though this characteristic is found ultimately from the resistivity in section 2.1 (i), it is the origin from which both mobility and resistivity emerge. Average carrier concentration in the active layer is obtained once the the mobility (using equation 2.2)
Figure 2.2 The function $f(R_u/R_b)$ [2]
is known, using the relation [3]:

\[ n = \frac{1}{\rho q \mu} \quad (2.4) \]

\[ n = \left( \frac{B}{\Delta R_c t} \right) 10^{-8} \quad (2.5) \]

where \( q \) is the electronic charge

(iv) Electrical Profiling

A carrier profile can be obtained by a combination of wet etching and van der Pauw measurements done repeatedly until the whole active layer is removed. The etching removes the surface of the material, exposing the material underneath. The amount of material etched from the surface depends on the etchant used and the etching time. The measurement is usually taken at room temperature because most of the dopants are activated. It may also be done as a function of temperature.

For depth profiling, these measurements are taken at various depths \( (d_n) \) over the thickness desired. The equations are modified for depth profiling by [4]:

\[ \sigma_n = \frac{\bar{\sigma}_{n+1} d_{n+1} - \bar{\sigma}_n d_n}{\Delta d} \quad (2.6) \]

\[ \mu_n = \frac{\bar{\mu}_{n+1} \bar{\sigma}_{n+1} d_{n+1} - \bar{\mu}_n \bar{\sigma}_n d_n}{\bar{\sigma}_{n+1} d_{n+1} - \bar{\sigma}_n d_n} \quad (2.7) \]

\[ \frac{n_n}{\mu_n \sigma_n q} \quad (2.8) \]
where:
\[ \Delta d = d_{n+1} - d_n \text{ etched away thickness} \]
\[ \sigma_n, \mu_n, \text{ and } n_n \text{ are the conductivity, mobility, and} \]
\[ \text{carrier concentration of the } n^{th} \text{ layer, respectively.} \]

2.1.2 Diode Characteristics

Diode characteristics can be divided into the following five sections, (i) barrier height, (ii) doping densities, (iii) built-in potential, (iv) ideality factor, and (v) saturation current.

(i) Barrier Height

The barrier height is indicated in Figure 2.3 as \( q\Psi_B \) [6]. This potential is the difference between the conduction band in the semiconductor at the surface and the Fermi level of the metal. Barrier height (defined for metal-semiconductor junctions, not pn junctions), gives an indication of the quality of the rectifying contact. As the barrier height increases, the reverse current decreases (in the ideal case). This is especially important for Metal-Semiconductor Field Effect Transistors (MESFETS) where many of the transistor characteristics depend on the reverse biased metal-semiconductor gate.

MEDUSA uses two methods for calculating the barrier height: current-voltage and capacitance-voltage. The first method is more accurate for moderately doped semiconductors. The second is better for highly doped semiconductors, but it assumes a perfect contact with no damage or interfacial layer (this layer gives two capacitors in series, leading to an erroneous barrier height). The forward current-voltage equation is [6]:

\[
I_f = A A^* T^2 \exp \left[ \frac{q(V_f - \Psi_B)}{kT} \right]
\]

where:
\( V_f \) is the forward bias in Volts
\( I_f \) is the forward current at \( V_f \)
\( T \) is the temperature in Kelvin
\( A \) is the diode area
\( A^* \) is the Richardson constant
Figure 2.3 Energy band diagram of a metal n-type semiconductor contact [6]
\( \Phi_B \) is the barrier height
\( k \) is Boltzmann's constant

The barrier height \( \Phi_B \) can be obtained from the slope of \( \ln(I/V^2) \) versus \( q/kT \) at a given forward bias.

The capacitance-voltage equation is [6,7]:

\[
\frac{1}{C^2} = \frac{2}{qe_s N_d A^2} (V_R + V_{bi}) \tag{2.10}
\]

\[
\Phi_B = V_{bi} + V_n + \frac{kT}{q} \tag{2.11}
\]

\[
V_n = \left( \frac{kT}{q} \right) \left( \frac{N_c}{N_d} \right) \tag{2.12}
\]

where:

- \( V_R \) is the reverse bias
- \( V_n \) is the built-in potential
- \( N_d \) is the doping density

(ii) Doping Densities

The doping density is one of the most important semiconductor characteristics. It can be related to mobility, carrier lifetime, breakdown voltage, transconductance, barrier height and a plethora of other characteristics. The doping density for a uniformly doped semiconductor is found from the slope of the \( 1/C^2 \) versus voltage plot (see Figure 2.4). The doping density \( (N_d) \) is found from [5-7]:

\[
N_d = \frac{2}{qe_s} \left[ -\frac{1}{\partial(A^2/C^2)/\partial V} \right] \tag{2.13}
\]

where:

- \( C \) is the diode capacitance
- \( e_s \) is the semiconductor permittivity
- \( \partial(A^2/C^2)/\partial V \) is the slope of the \( A^2/C^2 \) curve versus reverse bias
Figure 2.4 Ideal $1/C^2$ versus voltage plot
(iii) **Built-In Potential**

The built-in potential is shown in Figure 2.4 as \( V_b \). This voltage is the difference between the conduction band at the interface and the bulk. The built-in potential is found from the capacitance versus voltage experiment. This voltage is found at the voltage (zero capacitance point) intercept of the \( 1/C^2 \) versus \( V \) curve (See Figure 2.4).

(iv) **Ideality Factor**

The ideality factor gives an indication of how well the diode follows the thermionic emission characteristic. This factor is given by [6,7]:

\[
I = I_s \left[ e^{\frac{qV}{kT}} - 1 \right] \tag{2.14}
\]

Measuring the ideality factor is accomplished by conducting a current-voltage experiment. The slope of the natural log current versus forward bias curve determines the ideality factor. The ideality factor should be close to one for a good diode at low doping levels [6].

(v) **Saturation Current**

Saturation current is defined in the thermionic emission model. It gives the reverse current before breakdown and how well the diode behaves at forward voltages. A good diode has a low saturation current. The ideal diode equation (with an ideality factor of unity) is [5-7]:

\[
I = I_s \left[ e^{\frac{qV}{kT}} - 1 \right] \tag{2.15}
\]

or

\[
\ln(I_s) = \frac{qV}{kT} - \ln(I) \tag{2.15}
\]
where $I_s$ is the saturation current.

The saturation current is found from the current intercept of a natural log current versus voltage plot.

2.1.3 Trap Parameters

Trap energy ($E_t$) is the position of the trap below the conduction band edge. The energy level and the trap density can be found from Deep Level Transient Spectroscopy (DLTS). A DLTS spectrum, at a fixed rate window, is the plot of a transient capacitance magnitude versus temperature. A peak in the spectrum occurs when the trap emission rate equals the rate window. The sign of the transient will denote the type of trap, electron or hole, and the magnitude of the peak is approximately related to the trap density as [8]:

$$N_t = 2N_d \left| \frac{\Delta C}{C} \right|$$  \hspace{1cm} (2.17)

where

$$\Delta C = \frac{C(t_1) - C(t_2)}{\exp^{-\alpha t_1} - \exp^{-\alpha t_2}}$$  \hspace{1cm} (2.18)

and

$C$ is the reversed biased capacitance
$C(t_1) - C(t_2)$ are diode capacitances at times $t_1$ and $t_2$, respectively
$N_d$ is the donor concentration
$\alpha$ is the trap emission rate at the DLTS peak temperature

A trap emission rate, or rate window, for an exponential transient, can be obtained as [9]:

$$e_n = \frac{\ln \left( \frac{t_1}{t_2} \right)}{t_1 - t_2}$$  \hspace{1cm} (2.19)
If the rate window is chosen to be equal to 50 s\(^{-1}\), then the trap energy can be estimated as:

\[ E_c = 23.7 kT_m \]  \hspace{1cm} (2.20)

where \( T_m \) is the temperature of a DLTS peak.

This is accurate to within \( \pm 10\% \).

2.2 Experimental Test Station

The hardware layout for MEDUSA is shown in Figure 2.5. The instruments on the left side are connected to the computer via the IEEE-488 bus. The capacitance-time (C-t/HP 4280A) meter with the assistance of the pulse voltage generator (HP 8112A) controls the first group of experiments, the capacitance/conductance measurements. The picoammeter (pA meter/HP 4140B) controls the second set of experiments, comprised of current-voltage and capacitance-voltage measurements. The pA meter in conjunction with the Keithley 195A multimeter, is used in the four-point resistance measurement. When combined with the Varian magnet, the Van der Pauw/mobility experiment is enabled. The cryostat, consisting of a IEEE-488 temperature controller, the cold head, compressor, and vacuum pump control the temperature setting for the device under test (DUT). On the right side of the figure, the parallel printer port and serial port are connected to the Okidata 192 printer and HP 7475A plotter, respectively.

2.3 DUT Hardware

The most important piece of hardware is the TO-8 header shown in Figure 2.6, which holds the DUT in place during the experiment. Four out of the twelve possible header leads are used. The eight unused leads are removed. For a diode, only two leads are necessary, one for the ohmic and the other for a Schottky contact. Usually, more than one Schottky contact (up to three) are made, to allow a typical diode to be chosen for the capacitance and forward voltage characteristics. Both the
Figure 2.5 MEDUSA hardware layout
Figure 2.6 TO-8 Can with mounted DUT
van der Pauw and four-point resistance measurements utilize all available leads. An alumina substrate employed as an insulator is connected to the header base utilizing thermally conductive grease, insuring against a common ground through the wafer back. The DUT is then placed on top of the alumina, once again secured with thermally conductive grease. The bridge between the header and the DUT is made with 1 mil (1/1000 inch) gold wire, utilizing Epo-Tek H20E electrically conductive epoxy. The epoxy is subsequently cured at 150°C under nitrogen for 5 minutes. The header cap in Figure 3.2 has a small vent hold drilled through it. This vent allows outgassing of the thermal grease and epoxy at higher temperature (above approximately 290K). The TO-8 header with sample mounted is then placed inside the coldhead and secured with a strap to insure proper electrical and thermal contact during the experiment.

2.4 IEEE Interface

A digital interface is necessary for the computer automation of any laboratory experimental layout. This interface connects the various measurement taking instruments to the computer, providing a pathway for data to flow in both directions. Data from the computer provides the various instruments with the information related to experimental parameters. The instruments send the computer the data acquired from the actual experiment. The interface provides the protocol and pathways necessary for device communication. A standard which has become very popular is the IEEE-488 standard. This standard is further explored in the following chapter.
CHAPTER 3: EXPERIMENTAL TECHNIQUES

This chapter covers two main areas; how the computer interface operates, and MEDUSA's programming layout. The computer interface which works with all the measurement instruments is the IEEE-488 interface. The program layout, broken into four sections, controls the experiments and analyzes the data.

3.1 IEEE-488 Interface

The IEEE-488 interface supplies the connection between the controlling program and the instrumentation responsible for the data acquisition. The following three sections provide information on: (i) IEEE bus lines, (ii) general interface protocol, and (iii) running the IEEE interface.

3.1.1 IEEE BUS Lines

The interface protocol between the computer and the various pieces of electronic equipment is based upon the IEEE interface standard 488-1975 [5]. The IEEE standard also conforms to an ANSI standard (MC1.1) and HP publications. This representation is generally accepted for virtually all digital interface apparatus. Another term frequently used in literature is GPIB (General Purpose Interface Bus). Both notations will be used in this manuscript.

The IEEE standard is based on parallel bus architecture, which implies that one byte (8 bits) is sent along eight lines simultaneously. The cable is connected to 24-pin IEEE-488 connectors. The pin layout, following the IEEE-488 standard, is shown in Table 3.1. It should be noted that lines 1-4 and 13-16 are data lines, which makes up one byte. The bus is designed to accept the following line layout (see Figure 3.1). Five management lines, allow the controller to oversee all bus operations. The handshake lines supervise all data transfer, and data lines allow parallel communications. Finally, the remaining eight lines are grounds.

The management lines consist of EOI (End Or Identify), IFC (Interface Clear), SRQ (Service
<table>
<thead>
<tr>
<th>Contact Number</th>
<th>IEEE-488 Designation</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DIO1</td>
<td>Data</td>
</tr>
<tr>
<td>2</td>
<td>DIO2</td>
<td>Data</td>
</tr>
<tr>
<td>3</td>
<td>DIO3</td>
<td>Data</td>
</tr>
<tr>
<td>4</td>
<td>DIO4</td>
<td>Data</td>
</tr>
<tr>
<td>5</td>
<td>EOI</td>
<td>Management</td>
</tr>
<tr>
<td>6</td>
<td>DAV</td>
<td>Handshake</td>
</tr>
<tr>
<td>7</td>
<td>NRFD</td>
<td>Handshake</td>
</tr>
<tr>
<td>8</td>
<td>NDAC</td>
<td>Handshake</td>
</tr>
<tr>
<td>9</td>
<td>IFC</td>
<td>Management</td>
</tr>
<tr>
<td>10</td>
<td>SRQ</td>
<td>Management</td>
</tr>
<tr>
<td>11</td>
<td>ATN</td>
<td>Management</td>
</tr>
<tr>
<td>12</td>
<td>SHIELD</td>
<td>Ground</td>
</tr>
<tr>
<td>13</td>
<td>DIO5</td>
<td>Data</td>
</tr>
<tr>
<td>14</td>
<td>DIO6</td>
<td>Data</td>
</tr>
<tr>
<td>15</td>
<td>DIO7</td>
<td>Data</td>
</tr>
<tr>
<td>16</td>
<td>DIO8</td>
<td>Data</td>
</tr>
<tr>
<td>17</td>
<td>REN</td>
<td>Management</td>
</tr>
<tr>
<td>18</td>
<td>Gnd</td>
<td>Ground</td>
</tr>
<tr>
<td>19</td>
<td>Gnd</td>
<td>Ground</td>
</tr>
<tr>
<td>20</td>
<td>Gnd</td>
<td>Ground</td>
</tr>
<tr>
<td>21</td>
<td>Gnd</td>
<td>Ground</td>
</tr>
<tr>
<td>22</td>
<td>Gnd</td>
<td>Ground</td>
</tr>
<tr>
<td>23</td>
<td>Gnd</td>
<td>Ground</td>
</tr>
<tr>
<td>24</td>
<td>Gnd</td>
<td>Ground</td>
</tr>
</tbody>
</table>
Figure 3.1 IEEE-488 BUS layout
Request), ATN (Attention), and REN (Remote Enable). The EOI identifies the end of a multi-byte transfer. The IFC line takes the addressed instrument(s) out of both talk and listen modes. When the SRQ line is made active by an instrument, it requires some type of service. The ATN line regulates whether the information placed on the bus is data or a controller command. When REN line is set low, either all the devices on the bus are sent to remote, or the one device immediately addressed.

The three handshake lines consist of DAV (Data Valid), NRFD (Not Ready For Data), and NDAC (Not Data Accepted). These handshake lines work together to assure data reliability. Figure 3.2 shows the relative states of the lines for proper data transfer. When the data is placed on the bus by the source, the source checks to see if the NRFD line is high. Concurrently, the NDAC line should be low from previous data transfers. When both of these lines are properly set, the source sets the DAV line low. The NDAC line remains low until the slowest listener has received the data and sets the line high. After all devices are cleared, the NDAC line is sent low. Finally, the eight data lines allow transfer of all information over the bus, one byte at a time.

A typical configuration is shown in Figure 3.1. This configuration can be either serial or parallel, depending on application needs. Parallel connection of all devices enhances data transfer speed, while reducing the total number of allowed devices and cable length.

3.1.2 General Interface Protocol

Figure 3.1 shows a main controller (the computer) with all other instruments in series with it. There are three main types of devices; controller, talker, and listener. The controller supervises all other bus instruments. The talker sends data through the bus, either to other instruments on the bus and/or to the controller. The listener(s) receive any data which has been sent by the talker. Almost any instrument can be a talker, listener or both. A typical listener is a plotter or printer.

At any one time there can only be one operative controller. The general limit for instruments is fifteen. The limit is also controlled by the total cable length between the instruments. This length is dependent on the necessary bit rate, but a maximum length of twenty meters is usually observed.
Figure 3.2 IEEE-488 timing diagram [34,35]
Only one talker exists at a time. The other devices are usually addressed as listeners. The talker is chosen by the controller who addresses the instrument and then tells the talker to send the data onto the bus.

A Scientific Solutions IEEE-488 controller was chosen. It was picked for its ease of use, and cost.

3.1.3 Running the IEEE-488 Interface

To get the IEEE-488 interface to run properly, the computer must first initialize the IEEE controller board. This board resides in one of the computer expansion slots. When the controller wants to open the instrument(s) for communications it must send a remote enable signal (REN) through the bus to place each device into a computer controlled mode. Next, the computer will address the device it needs to communicate with. This allows the controller to either give information to (such as instrument experimental parameters), or get information from the device (data). Once the device is addressed, it responds to the controller commands. After finishing the controller instructions, the instrument sends a clear message (NDAC) over the bus, then the controller goes onto the next function [32-34]. All these functions are controlled by a software program which tells the controller, through machine language routines, exactly what steps to take to successfully run an experiment. Once the operations have been concluded, the controller then sends an clear signal over the bus (IFC) and removes the devices from the bus (sends the REN line high).

3.2 PROGRAMMING LANGUAGE

MEDUSA requires a programming language which can fulfill a wide range of objectives. These requirements and the language which satisfies the criteria are detailed in the next two sections.

3.2.1 Language Requirements

MEDUSA places many demands on a programming language. These demands are: large available memory (greater than 256 Kbytes), Scientific Solutions (IEEE-488) compatible (in this case BASIC adaptable), ability to use Enhanced graphics (EGA), ease of programming, and expense.
MEDUSA is a very involved program, so a large amount of memory is necessary to allow utilization of arrays, memory intensive graphics, and data processing sections. Also, Scientific Solution's software is written in BASIC and uses about 40 Kbytes of memory on its own. This precludes using BASIC or many of the other BASIC compatible software packages available at the time, because they allowed a maximum utilization of 64 Kbytes of memory, not nearly enough. Also, in order to get reasonable detail on screen graphics, and to utilize the Paradise EGA card and monitor, a program is necessary which allows enhanced graphics programming.

3.2.2 Language Chosen[27]

The language chosen was BetterBASIC™. It supports complete utilization up to 640 Kbytes of memory, is BASIC compatible, allows enhanced graphics, and has a low cost. It also expands BASIC's limited programming abilities. Some of these enhancements include procedures (or subroutines), local and global variables, recursion, pointers, allows math co-processor programming (speeding up any math calculations), expanded math accuracy (for both math co-processor and decimal math), and program block structures which include IF-THEN and DO-LOOP statements. Another advantage of BetterBASIC™ is the ability to reduce the size of procedures, further decreasing memory usage.

BetterBASIC™ allows three types of program saves. The most memory consuming, yet least likely to have errors, is a file listing. This is an ASCII file (a file readable outside of the BetterBASIC™ environment). The drawback to this method of saving is the lengthy period it takes to reload and compile the program when reentering BetterBASIC™. The second is considered a normal save in BASIC. It is an object code save, which takes less memory and is quick to reload. It is also more likely to have errors in the saved file. The final method is a compressed file. This takes the procedures and removes the symbol and source tables, making the files much smaller. The two drawbacks to compressed file are, the compressed procedures cannot be edited, and the file is the most likely to fail when saved. The advantage is a much smaller file, which allows greater array
allocation because one of the main concerns was memory availability. MEDUSA uses compressed files.

Throughout the programming period, it was necessary to reduce memory requirements at all stages. This was accomplished using BetterBASIC's procedure reduction features, creating three software modules and linking the various routines through a DOS batch file. This is further explained in the next section.

3.3 MEDUSA'S SOFTWARE

MEDUSA is broken into a controlling batch file and 3 programming sections (a complete program listing is given in the appendix). The batch file controls the linking between various programs. The primary sections include a parameter setting program (PARAMETER), an experimental measurement program (RUNIT), and a data processing/plotting program (GRAPHICS). These programs are discussed more fully in the following four sections.

3.3.1 MEDUSA.BAT

MEDUSA.BAT (shown in Figure 3.3) is a DOS batch file which links between the three programs and configures BetterBASIC with the correct library modules. These modules configure BetterBASIC with the proper math and graphics routines. When the user types MEDUSA <ENTER> while in DOS, the batch file is implemented sending the user into PARAMETER. Once the user is finished with the parameter setting program, the program exits back to DOS where the batch file either links to RUNIT or GRAPHICS, depending upon previously supplied information. If GRAPHICS is chosen, MEDUSA.BAT will find the file "C:\DATA\END", and the user is immediately transferred to the plotting program, bypassing all experimental routines. Selecting RUNIT executes the experimental data acquisition program. After departing RUNIT, MEDUSA.BAT searches for the file "C:\DATA\REDO". If this file is located, the user returns to PARAMETER, otherwise continuing to GRAPHICS. Once the user is done with the graphics section, the batch file clears the screen and returns the user to the main directory.
Figure 3.3 Flow chart for MEDUSA.BAT
3.3.2 PARAMETER

The flow charts for PARAMETER are shown in Figures 3.4 and 3.5. Figure 3.4 depicts the program start, where characteristic information about the DUT and rudimentary experimental information is entered. Figure 3.5 concludes the flow chart, giving an outline of each experimental group's parameter setting routine.

When the batch file initiates PARAMETER, the program immediately displays an introductory screen welcoming the user to MEDUSA and requesting instruments to be turned on. Next the program checks for the file "C:\DATA\REDO". If the program finds this record, signifying a return from the program RUNIT, the user is asked if he wants to use the same DUT information block. If a different information block is required, the program continues on to the run type. If the same DUT information block is desired, the user enters the directory for the new experimental run. If this directory has not been previously specified, the data is immediately transferred from the old directory to the new one. Otherwise, the user must authorize clearing old data from this directory. Finally, after the information has been transferred, the program continues on to the run type.

If the program does not find the file "C:\DATA\REDO", the above procedure is bypassed. At this point, the program gives a choice between a time, temperature, room temperature run, or an exit to the graphics routine. If GRAPHICS is chosen, PARAMETER makes a file "C:\DATA\END" then immediately exits to DOS, allowing MEDUSA.BAT to resume control. Otherwise, the program sets the appropriate run type flag. If the program entered from RUNIT and the DUT information copied to a new directory, the program bypasses the DUT information routine and continues to the section choosing the experimental group. Otherwise, the user enters device information, i.e., the directory, user's name, sample number, cryostat reading, and any desired comments. Upon making an error, the program allows changes to the above information. Now the actual experiments are chosen. Table 3.2 gives the four experimental groups and their individual choices.

After choosing the experimental group, the program breaks off into the specified experimental group (Figure 3.5) where each individual experiment is selected. If the user does not choose any
Figure 3.4 First flow chart for Parameter
### Table 3.2 Group Experiments

<table>
<thead>
<tr>
<th>GROUP I</th>
<th>GROUP II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacitance vs. Time</td>
<td>Current vs. Voltage</td>
</tr>
<tr>
<td>Conductance vs. Time</td>
<td>Capacitance vs. Voltage</td>
</tr>
<tr>
<td>Capacitance and Conductance vs. Voltage</td>
<td></td>
</tr>
<tr>
<td>Capacitance and Conductance</td>
<td></td>
</tr>
<tr>
<td>GROUP III</td>
<td>GROUP IV</td>
</tr>
<tr>
<td>Van der Pauw/Mobility</td>
<td>4-point Resistivity</td>
</tr>
</tbody>
</table>

### Table 3.3 Group I Parameters

<table>
<thead>
<tr>
<th>GROUP I</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacitance vs. Time and Conductance vs. Time</td>
</tr>
<tr>
<td>Temperature Range/Increment</td>
</tr>
<tr>
<td>Bias Voltage Range/Increment</td>
</tr>
<tr>
<td>Number of Samples</td>
</tr>
<tr>
<td>Pulse Times and increments</td>
</tr>
<tr>
<td>Pulse Voltage Range/Increment</td>
</tr>
<tr>
<td>Step Delay Time</td>
</tr>
<tr>
<td>Capacitance and Conductance vs. Voltage</td>
</tr>
<tr>
<td>Temperature Range/Increment</td>
</tr>
<tr>
<td>Voltage Range/Increment</td>
</tr>
<tr>
<td>Hold Time</td>
</tr>
<tr>
<td>Step Delay Time</td>
</tr>
<tr>
<td>Capacitance and Conductance</td>
</tr>
<tr>
<td>Temperature Range/Increment</td>
</tr>
<tr>
<td>Bias Voltages/Increment</td>
</tr>
</tbody>
</table>
Figure 3.5  Second flow chart for PARAMETER
experiments, the program loops back to the TEMP/TIME/ROOM TEMP/GRAPHICS section (see Figure 3.4 - NONE). Following experimental selection, the user sets the temperature range, time range, or room temperature. If the temperature run has been selected, the program allows the user to enter the overall temperature range and its increment, then checks to ensure these temperatures are within the cryostat's abilities. If a time run is chosen, both the temperature entered and time parameters are checked to confirm that they are within program specifications and temperature controller limits. The room temperature setting defaults to 290K, but allows the user to enter a different temperature.

The next section involves entering each of the chosen experimental parameters (see Tables 3.3-3.6). The program asks for each parameter, one at a time, allowing reentry of miskeyed values. After entering all the parameters, the program goes to the IEEE-488 interface and asks the instruments necessary to the experiment (see Table 3.7) if the given parameters are within each instrument's capability. If an instrument cannot handle these values, it returns a service request (SRQ). The program queries the instrument, reads the error message, and prints it to the screen. After pausing five seconds, the program returns to the parameter entry routine and allows the user to reenter their parameters. After the instruments accept the given parameters, the computer proceeds to the next chosen experiment and repeats this procedure. After finishing all the chosen experiments, the program directs the DUT information to a file called "C:\DATA\###\INFO", where ### is the requested directory. This is a permanent file, where the user can later retrieve the device information. Two other files are temporarily saved, "C:\DATA\EXPT", and "C:\DATA\PARAM". The "EXPT" file holds the experimental directory, chosen experimental group, type of temperature/time run, and the selected individual experiments. The "PARAM" file contains all the experimental parameters. RUNIT utilizes both files. After saving these files, the program exits to DOS, and MEDUSA.BAT proceeds to the experimental program, RUNIT.

3.3.3 RUNIT

The program RUNIT performs the experimental measurements and saves them in
Table 3.4 Group II parameters

<table>
<thead>
<tr>
<th>GROUP II</th>
<th>Capacitance vs. Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Current vs. Voltage</strong></td>
<td><strong>Temperaure Range/Increment</strong></td>
</tr>
<tr>
<td>Temperature Range/Increment</td>
<td>Start Voltage</td>
</tr>
<tr>
<td>Start Voltage</td>
<td>Stop Voltage</td>
</tr>
<tr>
<td>Stop Voltage</td>
<td>Step Voltage</td>
</tr>
<tr>
<td>Step Voltage</td>
<td>Hold Time</td>
</tr>
<tr>
<td>Hold Time</td>
<td>Step Delay Time</td>
</tr>
</tbody>
</table>

Table 3.5 Group III parameters

<table>
<thead>
<tr>
<th>GROUP III</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Van der Pauw/Mobility</strong></td>
</tr>
<tr>
<td>Temperature Range/Increment</td>
</tr>
<tr>
<td>Current Bias</td>
</tr>
</tbody>
</table>

Table 3.6 Group IV parameters

<table>
<thead>
<tr>
<th>GROUP IV</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Four Point Resistance</strong></td>
</tr>
<tr>
<td>Temperature Range/Increment</td>
</tr>
<tr>
<td>Current Bias</td>
</tr>
</tbody>
</table>
Table 3.7 Experiments and their related instruments

<table>
<thead>
<tr>
<th>Capacitance vs. Time and Conductance vs. Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-t Meter HP 4280A</td>
</tr>
<tr>
<td>Pulse Generator HP 8112A</td>
</tr>
<tr>
<td>Cryophysics Thermometer/Controller Model 4075</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Capacitance and Conductance vs. Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>and</td>
</tr>
<tr>
<td>Capacitance and Conductance</td>
</tr>
<tr>
<td>C-t Meter HP 4280A</td>
</tr>
<tr>
<td>Cryophysics Thermometer/Controller Model 4075</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Current vs. Voltage and Capacitance vs. Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>pA meter/DC Voltage Source HP 4140B</td>
</tr>
<tr>
<td>Cryophysics Thermometer/Controller Model 4075</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>van der Pauw/Mobility</th>
</tr>
</thead>
<tbody>
<tr>
<td>pA meter/DC Voltage Source HP 4140B</td>
</tr>
<tr>
<td>Keithley 228A Voltage/Current Source</td>
</tr>
<tr>
<td>Digital Multimeter Keithley 195A</td>
</tr>
<tr>
<td>Cryophysics Thermometer/Controller Model 4075</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Four Point Resistance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital Multimeter Keithley 195A</td>
</tr>
<tr>
<td>pA meter/DC Voltage Source HP 4140B</td>
</tr>
<tr>
<td>Keithley 228A Voltage/Current Source</td>
</tr>
<tr>
<td>Alpha Model 45-30 Power Supply/Varian Magnet</td>
</tr>
<tr>
<td>C-t Meter HP 4280A</td>
</tr>
<tr>
<td>Cryophysics Thermometer/Controller Model 4075</td>
</tr>
</tbody>
</table>
GRAPHICS compatible ASCII files. The two major blocks of RUNIT are the temperature setting section, and instrument setup with data taking/storing routines. Figure 3.6 shows the first section of RUNIT. When the program enters from PARAMETER, the computer immediately inquires if the device is correctly connected to the instruments. When the user finishes the configuration and hits any key on the keyboard, the computer initializes the IEEE-488 interface. The program now loads two files, "EXPT" and "PARAM", which specify the experiments to conduct and their associated parameters. Once these files have been read, the program searches for a van der Pauw experiment. This experiment requires a room temperature four-point resistance measurement coupled with a magnetic field measurement (see chapter 2.2.1 (i)). The room temperature readings are now taken. The experiment requires different cabling to achieve the desired results. The program accomplishes this by pausing and asking the user to switch the coaxial cables to the configuration displayed on the screen. After finishing these measurements, the program returns to the main program flow. At this point the program branches to one of three sections, (i) time run, (ii) room temperature, or (iii) temperature run.

(i) Time Run

Figure 3.7 gives the flow chart for the TIME run. First, the time is initialized to zero and displayed on the screen. Next, the program goes to the temperature setting routine (Figure 3.9) which sets the proper DUT temperature. After completing the first experimental set, the program loops back to A', where the time between experimental runs is added to the original time. If this time is greater than the maximum time given for the experiment, the program exits to a finishing routine (Figure 3.19). If the current time is less than the maximum time given by the user, the computer pauses until reaching the next experimental run time. At this point the program prints the new time and again runs the chosen experiment set (Figure 3.10).

(ii) Room Temperature Run

The room temperature flow chart is given in Figure 3.8. The room temperature is simply printed to the screen, and then the program proceeds to the experimental sections (Figure 3.10).
Figure 3.6 First flow chart for RUNIT
Figure 3.7  Time run flow chart

Figure 3.8  Room temperature flow chart
After completing the experimental set, the program continues to the finishing routine (Figure 3.19).

(iii) Temperature Run

The temperature setting routine (Figure 3.9) is the most vital section of the RUNIT program. It consists of setting the correct DUT temperature while ensuring the low temperature silicon sensor does not overheat. The first step prints the sample temperature set-point and determines if the temperature is greater than 290K. If this is true, the computer displays a warning message to the screen, advising the user to turn on the vacuum pump to remove any outgassing from the sample.

The routine now determines the correct temperature-dependent controller parameters. These parameters are implemented by sending them to the temperature controller. The current temperature is read from the controller and compared to the sample set-point. If the temperature varies from the set-point by more than \( \pm 1 \) K, the program checks the silicon sensor for catastrophic failure which occurs at temperatures greater than 325K. If the coldfinger fails this test, the program prints a coldhead failure message and shuts down the system. Upon passing the test, the above routine repeats until reaching the set-point. Once this occurs, the program pauses for one minute to allow for DUT temperature stabilization. After which, the computer continues to check the DUT's temperature until it once again reaches the set-point. At this time the temperature routine exits to the experiments to be run (Figure 3.10).

After completing the experiment set, the program loops back to the temperature setting routine and increases the temperature by its increment. It then checks the new sample set-point against the maximum temperature. If the set-point is above the maximum temperature, the program exits to the finishing routines (Figure 3.19). If not, the program returns to the beginning of the temperature routine (point B, Figure 3.9) and sets the next temperature.

After executing one of the three above routines, the program branches to the correct experiment(s) (Figure 3.10). First, all IEEE-488 bus instruments are set to their default states. The interface accomplishes this by sending an "all device clear" command over the bus to ensure each experiment group works in the same manner. The routine cycles through all possible experiments.
Figure 3.9 Flow chart for temperature run
Figure 3.10 Flow chart for experiment selection
If one of these experiments were chosen in PARAMETER, the program branches to that experiment. The program then checks if the experiment is to be run at that temperature (assuming a temperature run), and if so, runs the experiment. After completing the data acquisition, the experiment's new temperature is set by increasing the current temperature by its increment. The program then returns to Figure 3.10. Upon completing all the experiments in the set, the program returns to the correct run type routine.

The experiments are broken up into seven sections as depicted in Table 3.1, with the capacitance versus time and conductance versus time experiments being combined. Each experiment is outlined below.

(i) Capacitance vs. Time or Conductance vs. Time

Figure 3.11 gives the C-t/G-t flow chart. The experiment is specified by the routine in Figure 3.10 in which the computer sets either the capacitance or conductance experiment. After making this decision, the experiment-dependent parameters for the capacitance meter are also set. Following the opening of the output file (either "C:\DATA\###\CT" or "C:\DATA\###\GT"), the experiment name is printed to the screen. The program sends the initial pulse voltage information to the HP 8112A pulse generator. Depending on the specified parameters, the computer decides whether the experimental measurements require block mode which is where the capacitance meter saves all data until completing the experiment, and then sends the entire array to the computer via the IEEE-488 interface. Experiments with short time intervals between readings (less than ten milliseconds) require block mode. Once making this decision, the program sends the setup information to the capacitance meter at which time the experiment is actually performed. If not in block mode, the computer simultaneously reads the data from the capacitance meter while the meter takes data readings. If block mode is enabled, the computer pauses ten seconds and then sends the entire data set to the computer. The program then checks for another hold time. If one exists, the program loops back to point S* in the flow chart and repeats the experiment.

After completing all the hold times, the computer checks for more than one bias voltage. If
Figure 3.11  Flow chart for C-t and G-t experiments
this occurs, the voltage is incremented, and the program loops back to $S^*$ and repeats the experiments until exceeding the maximum bias point. Finally, the pulse generator's voltages are checked. If another voltage is discovered, it is incremented and the program loops back to $S^*$, and the experiment is repeated until all pulse voltages are exhausted. Now, the program returns to Figure 3.10.

(ii) Capacitance and Conductance vs. Voltage

The flow chart for the C-G-V experiment is shown in Figure 3.12. The first step saves pertinent parameters in the file "C:\DATA\##\CGV". The routine after which prints the experiment name and sends the necessary data to the HP 4280A capacitance meter. After turning on the bias voltage, the computer reads the data from the C-t meter and stores them in the data file while checking if all the C-t readings have been achieved. If the experiment requires more data, the program continues reading from the C-t meter until completing all measurements.

After taking all the readings, the program determines if another voltage set is required. If another set exists, the computer loops back and repeats the entire experimental run. Once completing all specified bias voltages, the program continues to Figure 3.10.

(iii) Capacitance and Conductance

The capacitance and conductance experiment is a derivative of the preceding experiment. The flow chart is shown in Figure 3.13. The experimental parameters are saved in "C:\DATA\##\CG" and the experiment name is displayed on the screen. After sending the parameters to the C-t meter and turning on the bias voltage, the program reads the one data point from the capacitance meter and saves this data to disk. The program checks for another bias voltage, if found, the program loops back and runs another capacitance group. After exhausting the voltages, the program continues on to Figure 3.10.

(iv) Current vs. Voltage

The current versus voltage flow chart (Figure 3.14) matches the capacitance and conductance versus voltage flow chart with the only differences being the save file name (C:\DATA\##\IV*) and the communications to the picoammeter, instead of the capacitance meter. The computer sends the
Figure 3.12 Flow Chart for C-G-V Experiment
Figure 3.13 Flow Chart for C-G Experiment
Figure 3.14 Flow Chart for I-V Experiment
proper setup parameters to the pA meter, then reads the current and voltage from the IEEE-488 interface bus while conducting the experiment. After completing the first set of voltages, the experiment checks for a second voltage set, repeating the experiment if these voltages are located. After this second set, the program proceeds to Figure 3.10.

(v) Capacitance vs. Voltage

The capacitance versus voltage experiment is shown in Figure 3.15. This experiment saves the pertinent parameters in "C:\DATA\###\CV" and prints the experiment name on the screen. The routine then sends the setup parameters to the pA meter and proceeds to read the data from the bus for the voltage range taken from the data file. After scanning all the voltages, the program branches to Figure 3.10.

(vi) van der Pauw/ Mobility

As mentioned earlier (at the beginning of 3.2.4), four room temperature resistance measurements are taken for the van der Pauw experiment (refer to Figure 3.6). The program reads a set of current/voltages with and without the magnetic field. The data is saved in a file named "C:\DATA\###\MOB".

The flow chart shown in Figure 3.16 gives the basic outline of the experiment. After saving the parameters necessary for the GRAPHICS program, the experiment name is displayed on the screen. The applicable parameters are sent to the pA meter, C-t meter, and the Keithley digital multi-meter. The C-t meter controls the magnet via a relay through an analog output. The pA meter steps the voltage in a positive manner until achieving the required current. The multi-meter takes the voltage reading once reaching this requested current. After saving the current/voltage data for the magnet-off state, the C-t meter turns on the magnet and repeats the experiment until the current is once again attained. The magnet-on data is then saved, the magnet turned off, and the program continues to Figure 3.11.

(vii) Four-point Resistance

The four-point resistance experiment requires the user to manually set the current using a
Figure 3.15 Flow Chart for C-V Experiment
Figure 3.16 Flow Chart for van der Pauw Experiment
current source (Figure 3.18). The current should remain constant throughout the temperature/time scan. This alleviates any temperature fluctuation caused by Joule heating. The experiment saves the pertinent parameters, displays the experiment's name, and sends the parameters necessary to Keithley multi-meter, placing it into voltage reading mode. After zeroing the multi-meter, the experiment reads the voltage and saves it to the file "C:\DATA\###\RES". Once completed, the program branches back to the appropriate run type, (Figure 3.10).

Returning to the Run type

After completing all the experiments for a given temperature/time, the program branches back to the appropriate run type, the TIME/TEMP RUN/ROOM TEMP section (Figure 3.11).

Finishing RUN

After all of the experimental sets are completed as required by the temperature scan, time run, or room temperature run, the program branches to Figure 3.18. First, the program sends default parameters to the cryostat which set the temperature to 2.1K and turn off the heater rapidly cooling the sample. Next, the program clears the IEEE-488 bus (IFC command). Finally, the program asks if another experimental run is required. If one is desired, the program creates a file named "C:\DATA\REDO" which MEDUSA.BAT reads and returns the user to PARAMETER. Otherwise the program exits to DOS where MEDUSA.BAT proceeds to GRAPHICS.

3.2.4 GRAPHICS

When the batch file MEDUSA.BAT transfers operation to the program GRAPHICS, a welcome screen and the main menu are displayed in order (Figure 3.19). From the main menu the user can access any of the sub-programs within GRAPHICS, utilizing the ten function keys. This menu is also the return point from all other locations in the program. The standard outline for retrieving data and plotting the results is shown in Table 3.7. which gives each step with its explanation and reference to the suitable flow chart.

Choosing a Graph

The first step allows the user to select the graph (see Table 3.8). Figure 3.20 furnishes the
Figure 3.17 Flow Chart for 4-Point Resistance Experiment
Figure 3.18  Flow Chart for RUNIT End
Figure 3.19 MAIN MENU Flow Chart

Figure 3.20 SYSTEM and GRAPH MENU Flow Charts
### Table 3.8 Standard Choice Sequence for GRAPHICS

<table>
<thead>
<tr>
<th>FKEYS</th>
<th>Notes:</th>
<th>Figures:</th>
</tr>
</thead>
<tbody>
<tr>
<td>F5</td>
<td>shows possible graphs</td>
<td>Figure 3.21</td>
</tr>
<tr>
<td>F7, F8</td>
<td>loads data into graph matrices</td>
<td>Figure 3.XX</td>
</tr>
<tr>
<td>F9</td>
<td>allows manipulation of extrema</td>
<td>Figure 3.XX</td>
</tr>
<tr>
<td>F10</td>
<td>displays graph on screen</td>
<td>Figure 3.XX</td>
</tr>
<tr>
<td>F3</td>
<td>allows manipulation of data</td>
<td>Figure 3.XX</td>
</tr>
<tr>
<td>F6</td>
<td>finds fit over specified X range</td>
<td>Figure 3.XX</td>
</tr>
<tr>
<td>F4</td>
<td>plots graph on HP 7475A</td>
<td>Figure 3.XX</td>
</tr>
<tr>
<td>F2</td>
<td>saves in GRAPHICS compatible form</td>
<td>Figure 3.XX</td>
</tr>
<tr>
<td>F1</td>
<td>exits to DOS</td>
<td>Figure 3.21</td>
</tr>
</tbody>
</table>

### 3.9 Available Graphs in the program GRAPHICS

1. Hand Entry Graph
2. Capacitance vs. Time
3. Conductance vs. Time
4. Capacitance vs. Voltage (HP 4280A)
5. Conductance vs. Voltage
6. Capacitance vs. Temperature
7. Conductance vs. Temperature
8. 1/C^2 vs. Voltage (HP 4280A)
9. Dopant Profile (HP 4280A)
10. Barrier vs. Temperature (1/C^2)
11. DLTS Spectrum
12. Current vs. Voltage
13. Capacitance vs. Voltage (HP 4140B)
14. 1/C^2 vs. Voltage (HP 4140B)
15. log(I) vs. Voltage
16. ln(I) vs. qV/kT
17. ln(I/T^2) vs. q/kT
18. I_{sat} vs. Temperature
19. Ideality Factor vs. Temperature
20. Barrier vs. Temperature (HP 4140B)
21. Resistivity vs. Temperature
22. Mobility vs. Temperature
23. Carrier Concentration vs. Temperature
24. Activation Percentage vs. Temperature
25. 1/C^2 vs. Voltage
26. Barrier vs. Temperature (T/C)
27. ln(I_{sat}) vs. Temperature
28. Four-Point Resistivity vs. Temperature
flow chart. Screen constraints allow a maximum of twenty-four experiments to be shown at a time. Since more than twenty-four graphs exist in MEDUSA, and to allow for expansion into different graphs, the program allows multiple screens. Once the user has chosen the graph, the program returns to the main menu.

**Load Data into a Graph**

The user must choose between loading a graph into either matrix one or two (Figure 3.21). Two plots on the same axes allow easier comparison between graphs of the same type. After the user selects the graph matrix, the program sets a flag telling the program where to load the experimental data. The program, with the exception of the hand entry graph (explained in the following paragraph), now transfers to the data acquisition section, allowing input of the drive directory and the necessary experimental parameters. After the user enters the drive and directory, the program checks to confirm the existence of the directory and the file conforming to the experiment chosen. If this check fails, the routine loops back and allows the user to reenter the information. After locating the correct data file, the user enters the type of run: room temperature, time or temperature scan. After the program sets the corresponding flag, the user keys in the experimental and graph-dependent information (shown in Table 3.10). After the program finds the proper data, the program loads it into the specified graph matrix, and exits to the main menu. If the specified data is not located, the program returns to the selection of drive and directory, where the user reenters the parameters.

The exception to the above flow chart is the hand entry graph. After the user enters the graph matrix, they are, in this case, transferred to the menu shown in Figure 3.22. This menu is where graph loading occurs for the SAVE DATA point given in the main menu (refer to Figure 3.20). Also, the graph may be saved here. Other options allow the user to enter the graph and axes titles. Data may also be entered and changed at the user's discretion. Finally, the last function returns the user to the main directory.

(iii) Scale

The flow chart scale (Figure 3.23) gives the user an opportunity to adjust the plot axes of which default to their minimum and maximum values. The routine allows modification of the axes.
Figure 3.21 LOAD GRAPH 1 and 2 Flow Chart
<table>
<thead>
<tr>
<th>No.</th>
<th>Graph Description</th>
<th>Area of Device</th>
<th>Bias Dir</th>
<th>Contact Factor</th>
<th>Drain Current</th>
<th>Electrical Thickness</th>
<th>Hold Time</th>
<th>Leaktage Ft</th>
<th>Pull Voltage</th>
<th>Reflection Coefficient</th>
<th>Run Type</th>
<th>Temperature</th>
<th>Type of Material</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hand Entry Graph</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Capacitance vs. Time</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Conductance vs. Time</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Capacitance vs. Voltage (HP 4280A)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Conductance vs. Voltage</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Capacitance vs. Temperature</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Conductance vs. Temperature</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>$1/C^2$ vs. Voltage (HP 4280A)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Dopant Profile (HP 4280A)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>10</td>
<td>Barrier vs. Temperature (1/C^2)</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>DLTS Spectrum</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Current vs. Voltage</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>13</td>
<td>Capacitance vs. Voltage (HP 4140B)</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>14</td>
<td>$1/C^2$ vs. Voltage (HP 4140B)</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>15</td>
<td>log(I) vs. Voltage</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>16</td>
<td>Ln(I) vs. $q/kT$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>17</td>
<td>Ln(I) vs. $1/kT$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>18</td>
<td>$I_{sat}$ vs. Temperature</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>19</td>
<td>Ideality Factor vs. Temperature</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>20</td>
<td>Barrier vs. Temperature (HP 4140B)</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>21</td>
<td>Resistivity vs. Temperature</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>22</td>
<td>Mobility vs. Temperature</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>23</td>
<td>Carrier Concentration vs. Temperature</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>24</td>
<td>Activation Percentage vs. Temperature</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>25</td>
<td>$1/C^2$ vs. Voltage</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>26</td>
<td>Barrier vs. Temperature (1/C)</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>27</td>
<td>Ln($I_{sat}$) vs. Temperature</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>28</td>
<td>Four-Point Resistivity vs. Temperature</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

* see Figure 3.22
Figure 3.22 Hand Entry Flow Chart

Figure 3.23 SCALE Flow Chart
or an exit back to the main menu. If changes are desired, the program prompts the user to change each value, defaulting to the original values. After entering the four values, the routine loops back and prints the new extrema allowing multiple chances to change the minimum/maximum values for the graph. If the user decides not to change these values, the program returns to the main menu.

(iv) Screen Plot

When the program enters the screen plot menu (Figure 3.24), the routine gives a choice between a dot or line graph. After entering the choice, the program prints the axis and the existing graphs. Following the display of these points, the program adds a line between points if a line graph was previously selected. The computer then waits until any key has been hit to return to the main menu.

(v) Modify Data

This section allows experimental data alteration (Figure 3.25). Table 3.11 shows the basic types of modification. First, the user selects an option or choosing F1 exits the routine and returns to the main menu. Upon selecting an option, the program asks which graph to modify and the routine separates into the individual sections.

For the add, multiply, and power function keys, the user enters the axes to modify and the scalar value, followed by the operation which is performed on either the X- or Y-range, depending on the previous choice. The natural log, exponential function, or absolute value option asks the user for a range before executing the required operation.

For both add and delete points options, the routine prints up to thirteen data points, allowing the user to modify any one of them. If none of these are changed, the program scrolls to the next thirteen data points continuing until a value is either added or deleted or all data points have been shown. The program then loops back to the beginning of the modify routine. Finally, the clear graph function erases the specified graph.

(vi) Least-square Fit

The least-square fit is shown in Figure 3.26. The program queries for an X-axis range and then computes the calculations shown below [17]:

61
Figure 3.24 SCREEN PLOT flow chart
Figure 3.25  MODIFY data flow chart
Table 3.11  MODIFY data options

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>MAIN MENU</td>
</tr>
<tr>
<td>F2</td>
<td>ADD A CONSTANT TO A RANGE</td>
</tr>
<tr>
<td>F3</td>
<td>MULTIPLY A RANGE BY A CONSTANT</td>
</tr>
<tr>
<td>F4</td>
<td>TAKE RANGE TO A POWER</td>
</tr>
<tr>
<td>F5</td>
<td>NATURAL LOG OF A RANGE</td>
</tr>
<tr>
<td>F6</td>
<td>EXPONENTIATE A RANGE</td>
</tr>
<tr>
<td>F7</td>
<td>ABSOLUTE VALUE OF A RANGE</td>
</tr>
<tr>
<td>F8</td>
<td>DELETE A POINT IN A RANGE</td>
</tr>
<tr>
<td>F9</td>
<td>ADD A POINT IN A RANGE</td>
</tr>
<tr>
<td>F10</td>
<td>CLEAR A GRAPH</td>
</tr>
</tbody>
</table>

Figure 3.26  LEAST-SQUARE FIT flow chart
\[ m = \frac{S_{xy}}{S_{xx}} \]  

(3.1)

\[ y_{int} = \bar{y} - \frac{m}{x} \]  

(3.2)

\[ x_{int} = -\frac{y_{int}}{x} \]  

(3.3)

\[ \rho = \frac{S_{xy}}{\sqrt{S_{xx} S_{yy}}} \]  

(3.4)

\[ \bar{x} = \frac{\sum x}{n} \]  

(3.5)

\[ \bar{y} = \frac{\sum y}{n} \]  

(3.6)

\[ S_{xx} = \sum x^2 - (\sum x)^2 \]  

(3.7)

\[ S_{yy} = \sum y^2 - (\sum y)^2 \]  

(3.8)

\[ S_{xy} = \sum xy - \frac{\sum x \sum y}{n} \]  

(3.9)
where:

\[ m \] is the slope
\[ \rho \] is the correlation

The results, X-axis range, X and Y intercepts, slope and correlation, are shown on the screen in two columns (the first for graph one, the second for graph two) and the program pauses until any key is hit. The program returns to the main menu.

(vii) Plotter

The plotter routine utilizes a different set of function keys as defined in Table 3.12. The flow chart is presented in Figures 3.27 and 3.28. A return to the main menu is accomplished by pressing F1. F2 controls the plotter pen speed which defaults to the correct speed for paper plots. The speed should be reduced for transparencies for better resolution and to reduce smearing. F3, F4, and F5 control default options (see Figure 3.27 (a)). Tables 3.12-3.15 give the options for each key respectively. F3 controls the line type, F4 the symbol style, and F5 the pen color. The user selects the graph and the appropriate line/symbol/color. Finally, the program either returns to the PLOTTER menu or allows another change, depending on the choice given by the user.

F6 controls the plotting of the axes, information block (assuming the information block option has been set), and titles. Figure 3.27 (b) gives the flow chart. First, an option for plotting a grid is given. After this the program plots the axes and titles. Next, if a grid has been chosen, this is plotted. Depending on the F9 choice, the information block may be plotted. Finally, the program returns to the PLOTTER menu.

F6 and F7 controls the first and second plot. After one of these is chosen, the graph is plotted with the appropriate pen color, symbol, and line type (chosen earlier).

F9 controls the information block. Table 3.16 gives the material plotted if this key is enabled. The key acts as a toggle switch, enabling/disabling the information block as the key is pressed.

F10 allows the plotting of a caption. The user simply types in the caption and the plotter prints the caption below the graph.

After the plotting routine is completed, the user presses F1 and the program returns to the
Table 3.12 PLOTTER function key definitions

F1 RETURN TO MAIN MENU
F2 PLOTTER PEN SPEED
F3 LINE TYPE
F4 SYMBOL TYPE
F5 PEN COLOR
F6 PLOT GRAPH AXES
F7 PLOT FIRST GRAPH
F8 PLOT SECOND GRAPH
F9 INFORMATION BLOCK
F10 CAPTION

Figure 3.27 First PLOTTER flow chart
Figure 3.27 Second PLOTTER flow chart
Table 3.13  Line types

(1) NO LINE
(2) SOLID LINE
(3) DASHED LINE
(4) DASH-DOT LINE

Table 3.14  Symbol types

(1) NO SYMBOL
(2) PLUS (+)
(3) STAR (*)
(4) AMPERSAND (&)

Table 3.15  Color types

(1) THICK BLACK
(2) THIN BLACK
(3) DARK BLUE
(4) GREEN
(5) LIGHT BLUE
(6) RED
Table 3.16 PLOTTER information block

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SYMBOL</td>
</tr>
<tr>
<td>2</td>
<td>DIRECTORY</td>
</tr>
<tr>
<td>3</td>
<td>TEMPERATURE</td>
</tr>
<tr>
<td>4</td>
<td>TIME</td>
</tr>
<tr>
<td>5</td>
<td>BIAS VOLTAGE</td>
</tr>
<tr>
<td>6</td>
<td>PULSE HIGH VOLTAGE</td>
</tr>
<tr>
<td>7</td>
<td>HOLD TIME</td>
</tr>
<tr>
<td>8</td>
<td>RATE WINDOW</td>
</tr>
<tr>
<td>9</td>
<td>GRAPH RANGE</td>
</tr>
<tr>
<td>10</td>
<td>SLOPE</td>
</tr>
<tr>
<td>11</td>
<td>X-INTERCEPT</td>
</tr>
<tr>
<td>12</td>
<td>Y-INTERCEPT</td>
</tr>
<tr>
<td>13</td>
<td>RANGE FOR FIT</td>
</tr>
<tr>
<td>14</td>
<td>CORRELATION</td>
</tr>
</tbody>
</table>
main menu.

(vii) Save Data

The GRAPHICS program allows a graph to be saved in an ASCII compatible format by striking F2 (see Figure 3.29). Upon entering the save data routine, the user keys in a graph name, which defaults to the graph title. After choosing the file name, the program queries whether graph one or two is to be stored. Finally, the program saves the requested graph and returns to the main menu.

(viii) Exit To DOS

The program can be exited by striking F1. This is shown in Figure 3.21 (a) where the program thanks the user for using MEDUSA, clears the screen, and exits to DOS.
Figure 3.29 SAVE DATA flow chart
CHAPTER 4: EXPERIMENTAL RESULTS AND DISCUSSION

The MEDUSA system was tested and improved by doing a variety of measurements on both Schottky diodes and thick-film superconductors. The graphs shown below, while not inclusive, give the most important plots for a researcher. The sections are broken into five parts: 4.1 Capacitance experiments, 4.2 Conductance Experiments, 4.3 Current Measurements, 4.4 four-point Resistivity, and 4.5 van der Pauw Measurements.

4.1 Capacitance Experiments

The capacitance experiments consist of both capacitance versus time and capacitance versus voltage measurements. All readings were taken from the C-t meter. The pA meter (HP 4140B) can measure capacitances, but these measurements were not included because the C-t meter gives more accurate capacitance values. GaAs Schottky diodes were the devices tested in this section.

4.1.1 Capacitance vs. Time

Figure 4.1 gives the capacitance versus time at 330K after the application of a majority trap-filling pulse of 4 volts with a hold time of 10 mS. Thermal emission from traps causes an exponential increase in capacitance values[16]. The tested device does not conform to this theory. There are many factors that may contribute to a non-exponential decay rate [18]:

(i) electric field dependent emission rate
(ii) large deep-level defects may overwhelm the smaller net shallow dopant density
(iii) a non-ideal step junction
(iv) presence of distributed trap levels (transient is the sum of individual transients)
(v) partial trap charging occurring only in a section of the depletion region

4.1.2 Capacitance vs. Voltage

Figure 4.2 shows a capacitance versus voltage plot at room temperature which presents the desired trend with reverse bias. According to the Shockley model, for a uniformly doped sample, capacitance should have an inverse square relationship with voltage. The ideal capacitance versus voltage equation for a Schottky barrier is the same as a one-sided pn junction or, in a slightly
Figure 4.1 Capacitance versus Time
Figure 4.2 Capacitance versus Voltage
different form than equation 2.11 [12],

\[ C = \left[ \frac{\frac{Qe \varepsilon_0 N_d}{2 (V_0 + V_R)}}{2 (V_0 + V_R)} \right]^{\frac{1}{2}} \]  \hspace{1cm} (4.2)

The easiest method for proving that the graph follows an inverse square relationship is by observing the 1/C^2 versus V plot given in the next section.

4.1.3 1/C^2 vs. Voltage

Figure 4.3 shows the graph for the 1/C^2 versus voltage plot at room temperature which shows good linear behavior as predicted by 2.11 and 4.2. The slope of the plot yields a doping density of 7.6x10^{16}/cm^3, as compared with the vendor specification of 6x10^{16}/cm^3. The X-intercept gives a barrier height of 1.10 volts.

4.1.4 Capacitance vs. Temperature

Figure 4.4 gives a plot of capacitance dependence on temperature at a -4 volt bias. Depending upon the material, the plot is expected to either increase or remain invariant with temperature. The steps in Figure 4.4 can be seen as a correlation to the traps obtained from the DLTS spectrum where the trap switches from mostly full to mostly empty [16].

4.1.5 DLTS Spectrum

Figure 4.5 demonstrates a DLTS spectrum for a n-type GaAs sample from 100 to 400K with a bias of -4 volts, a trap filling pulse of 4 volts, a hold time of 10 ms, and a 50.29 second^{-1} rate window. There are three peaks evident at 155K, 240K, and 335K which correspond to trap energy levels of .312, .490, and .684 eV, respectively, based on equation 2.20. The peak height in the spectrum relates to the trap density, through equations 2.17 and 2.18. Accordingly, the trap density for .312, .490, and .684 eV are 2.57x10^{16}, 1.07x10^{16}, and 4.72x10^{16}/cm^3, respectively. These trap densities are close to the doping density of 7.6x10^{16}/cm^3 found in section 4.1.3 which may explain the behavior of the capacitance versus time graph (Figure 4.1).
Figure 4.3  $1/C^2$ versus Voltage
Figure 4.4 Capacitance versus Temperature
Figure 4.5 DLTS Spectrum
4.1.6 Barrier Height versus Temperature

Figure 4.6 displays a barrier height’s (\(\phi_n\)) temperature dependence. This spectrum reveals a high \(\phi_n\) value essentially constant over the given temperature range. The barrier height should increase with temperature as shown in equation 2.12. Some possible reasons for this deviation are [13]:

(i) interface corruption
(ii) an insulating layer between the contact and n-region
(iii) edge leakage current, and deep impurity levels.

Also, the tested wafer was only moderately doped (6X10^{14}/cm^3); hence a barrier height deduced from a current-voltage plot would be more reliable (see Chapter 2.1.2 (i)).

4.1.7 Dopant Profile

The room temperature dopant profile shown in Figure 4.7 depicts an essentially flat relationship out to .35 micron depth with an error of +/-10%, correct for the tested uniformly doped sample. Beyond .35 micrometers, the diode must be biased near breakdown causing erroneous C-V measurements.

4.2 Conductance Measurements

The conductance experiments are comprised of both conductance versus time and conductance versus voltage measurements. These experiments are useful in determining both the lossy nature of Schottky diodes and the calculation of trap densities (equation 2.18). The series resistance between Schottky and ohmic contacts constitute a large part of the experimental error in the determination of actual capacitance. A reverse-biased diode’s low conductance is of vital importance to the quality of a device, and subsequently to the authenticity of DLTS measurements. The conductance measurements were executed using the C-t meter at different reverse voltages on a GaAs schottky diode.

4.2.1 Conductance vs. Time

The conductance versus time plot (Figure 4.8) at 330K, with a hold time of 10 mS, shows an almost linear increase over the range of 40 mS. This suggests a reverse biased diode becoming
Figure 4.6 C-V Barrier Height
DOPANT PROFILE (HP 4280A)

Dopant Density (cm^{-3}) \times 10^{10}

Depth (Micrometers)

Figure 4.7 Dopant Profile
Figure 4.8 Conductance versus Time
increasingly leaky with time. This may explain the poor results of Figure 4.1 (capacitance versus time), because one possible reason for a non-exponential characteristic is a series resistance.

### 4.2.2 Conductance vs. Voltage

Figure 4.9 gives the room temperature conductance versus voltage graph over a range of -8 to 0 volts at room temperature. The increasing conductance correlates well with a reverse biased diode as it slowly moves toward forward bias. The more rapid increase near zero bias indicates an increase in the dissipation factor of the depletion layer capacitor. The best dissipation factor is found at moderate reverse biases where the saturation current is the smallest. Although the conduction remains below 3 msiemens, the capacitor quality degrades as reverse bias decreases.

### 4.2.3 Conductance vs. Temperature

Figure 4.10 gives the conductance versus temperature graph at -4 volts, showing an almost exponential increase at higher temperatures. The low temperature noisy behavior may be attributed to the accuracy limitations of the C-t meter. The conductance shows the proper trend because as the temperature increases, the electrons become more active due to thermal effects, causing an increase in the conductivity, i.e., the diode becomes leakier at higher temperatures. The conductivity at room temperature corresponds well with the temperature scan. The room temperature reading in Figure 4.9 at -4 volts is approximately .01 msiemens, while Figure 4.10 gives approximately .012 mmhos. These measurements concur well, within experimental error limits.

### 4.3 Current Measurements

All the current related graphs are derived from the current versus voltage experiment. The measurements within the group were taken from the pA meter (HP 4140B). The tested devices were GaAs Schottky diodes.

#### 4.3.1 Current vs. Voltage

Figure 4.11, a current versus voltage plot at room temperature, shows diode breaking down at a reverse bias of approximately 10 volts. The forward voltage turn-on appears correct, with the ideality factor found to be about 1.15. The +/- 10 mA flat region shown in the curve is due to the
Figure 4.9 Conductance versus Voltage
Figure 4.10  Conductance versus Temperature
CURRENT vs VOLTAGE

Figure 4.11 Current versus Voltage
current limitation of pA meter (10 mA).

4.3.2 Natural Log Current vs. Voltage

Figure 4.12 gives the Ln(I) versus qV/kT graph at room temperature. The plot shows the correct trend: current decreasing as the voltage approaches zero, a minimum at zero voltage, and approximately an exponential increase at forward biases (the linear slope in this graph). The ideality factor is given as the inverse slope of the forward characteristic, and from the least-square fit explained in the chapter 3.4, the slope is found to be .874 which corresponds to an ideality factor of 1.144. The reverse saturation current at room temperature is found to be 9.26X10^10 amps from the Y-intercept of the forward characteristic.

4.3.3 Ln(I/T^2) vs. qV/kT

Figure 4.13 gives an Ln(I/T^2) versus qV/kT plot taken at a forward bias of .15 volts. The curve is approximately linear over the indicated range and provides a range over which to take the least-square fit for the following three graphs. This is especially useful because the graph is derived from equation 2.10 which allows the barrier height to be calculated from the flat section of this graph. The slope of the curve gives a barrier height value of 1.02 eV.

4.3.4 I_sat vs. Temperature

Figure 4.14 gives a typical I_sat versus temperature graph. The saturation current is basically zero until about 260K where the electrons begin to activate. The saturation current increases approximately exponentially above 290K as predicted by the thermionic emission model (equation 2.16).

4.3.5 Ideality Factor vs. Temperature

Figure 4.15 displays an ideality factor's dependence on temperature. The ideality factor is seen to increase with temperature because the leakage in the diode becomes greater and the current transport deviates from the thermionic model at higher temperatures.

4.3.6 Barrier Height vs. Temperature

The barrier height shown in Figure 4.16 follows an approximately linear increase as predicted
Figure 4.12 Natural Log Current versus Voltage
Figure 4.13 \( \ln(1/T^2) \) versus \( qV/kT \)
Figure 4.14 $I_{\text{sat}}$ versus Temperature
Figure 4.15 Ideality Factor versus Temperature
Figure 4.16  I-V Barrier Height versus Temperature
by the equation [13]

\[ \Phi_b = \frac{kT}{q} \ln \left( \frac{\lambda^* T^2}{J_s} \right) \]  

(4.2)

The first-order approximation is roughly linear as shown by the kT/q term. The natural log term should have a second-order impact.

4.4 Four Point Resistivity

The four-point resistivity experiment was implemented using a thick film superconducting sample. Figure 4.17 gives a graph which conforms to the superconductor temperature spectrum. The thick film sample has a small resistance at temperatures above 90K, while the resistance drops rapidly to approximately zero ohms below that.

4.5 van der Pauw Measurements

van der Pauw measurements are used to evaluate transport parameters like resistivity, mobility and carrier concentration in a semiconductor. The resistivity dependence on temperature for a 2 MeV silicon implanted sample is shown in Figure 4.18. The corresponding behavior of mobility and carrier concentration is shown in Figures 4.19 and 4.20 where the mobility values peak at temperatures of around 100K. Below 100K impurity scattering is seen to be dominating, while above 100K lattice scattering governs carrier mobility. Carrier concentration marginally increases over the temperature range investigated.
Figure 4.17 Resistivity versus Temperature
Figure 4.18 Resistivity versus Temperature
Figure 4.19  Mobility versus Temperature
Figure 4.20 Carrier Concentration versus Temperature
CHAPTER 5  CONCLUSIONS AND FUTURE RECOMMENDATIONS

The MEDUSA program has been in operation for over two years. The software has been extensively tested and debugged. Except for minor programming errors and one hardware failure (the cryostat temperature controller failed), the system has been virtually trouble free.

The IEEE-488 parallel interface scheme was chosen over both RS-232C and A/D-D/A converters because the IEEE-488 bus allows simple programming to conduct the experiments. Also, the IEEE-488 advantage in speed helped increase the allowed measurement speed. Finally, the IEEE-488 interface is commonly added to most types of new measurement equipment at a low cost.

The hardware instruments chosen to be utilized by MEDUSA required an IEEE-488 interface board as the digital communications board between the computer and various instruments. The Scientific Solution card was chosen over the competitor’s for both it’s price and the ease of use.

The Scientific Solution’s IEEE-488 controller required a type a BASIC to be employed. BetterBASIC™ was chosen as the best BASIC compatible language available at the time. The major consideration for choosing the language was two-fold: the memory capabilities of the software had to be greater than 256Kbytes, and subroutines were required to make the programming more modular, and therefore easier to both program and debug.

MEDUSA was seperated into four sections: a controlling batch file and programs, PARAMETER, RUNIT, and GRAPHICS. PARAMETER provides the experimental parameter setting and checking abilities, RUNIT conducts the experiments, and GRAPHICS manipulates the data utilizing the correct equations to produce the required plots.

MEDUSA is extremely user friendly and the error checking routines are extensive, allowing errors to be made in the entering of variables. For example, if any unsuitable experimental parameters are entered, the program via the IEEE-488 bus, checks them with the instruments and returns the correct error message. It then allows the parameters to be recentered. Also, the configuration has been operated by several people, each learning to use MEDUSA in a matter of hours.
The four experimental groups have been exhaustively tested, especially the Group I, capacitance vs. time and capacitance vs. voltage experiments. All three modes of operation, time, room temperature, and temperature range, along with the various parameters particular to an experiment have been extensively utilized.

Various plots are shown in Chapter 4. Given the limitation of both the instruments and the devices tested, all experiments shown in Chapter 4 yield correct results. The only exceptions are the capacitance versus time and C-V barrier height versus temperature plots. One possible reason for the discrepancy between the plots and literature is a surface insulating layer, causing a non-ideal step junction. More reasons are given in sections 4.1.1 and 4.1.6.

Future Recommendations

The possible improvements to MEDUSA are designed to ease the researcher’s difficulty in running experiments and analyzing the data. Some recommended changes are listed below:

- Switch to another IEEE-488 interface card which allows machine language calls rather than BASIC calls, increasing data transfer rates and reducing memory requirements.

- Rewrite the software in QuickBASIC\textsuperscript{\textregistered}, which allows executable files (as opposed to BASIC interpreted files). Additionally, QuickBASIC\textsuperscript{\textregistered} is a current language, whereas BetterBASIC\textsuperscript{\textregistered} is now defunct.

- Incorporate a scanner into the hardware setup allowing all four experiment groups to be conducted at once. This would be accomplished by switching between the different meters (C-t and pA) in the same temperature sweep, increasing accuracy by alleviating a build-up of thermal stress over several temperature scans and quickening the data acquisition phase.

- Add an option to the C-t experiment allowing more than one step delay time. This would allow widely different rate windows during the same temperature scan.

- Add the necessary software/hardware modules to conduct FET characterization.
REFERENCES


7. A.A. Elshiabini-Riad, Class Notes, Fall, 1989.


29. IBM BASIC Reference.


APPENDIX
PARAMETER
SOURCE
PRECISION= 7
AUTODEF=ON
OPTION BASE=0
ERL=ON
ERRORMODE=GLOBAL
RESUME=STATEMENT
FORMODE=GW
SCOPE=OFF
PROCS=18

STRING ARRAY(?) = RECS,FLDS
STRING: PARAMS[?],IEEE_FCTNS[?]
INTEGER: CTROLR%,TRUE%,COMM%,PORT4%,PORT5%,PORT9%
INTEGER: PORT0%,FUM%,FALSE%,MY_FLAG%,MAX_TIME%
REAL: STROFF
INTEGER: STROFF%
REAL: DSPTR
INTEGER: MY_ADDR%,BD_ADDR%,PORT1%,PORT2%
INTEGER: PORT3%,PORT6%,PORT7%,PORT8%
STRING: CHARIS[?]
INTEGER: TCIMODE%,MS%,INTR%,INTVECTOR%,INTENABLE%
INTEGER:
INTMASK%,SUBLIB%,TIMEOUT%,INTSETUP%,POLLOBYTE%,WRSTR%,WRFUL%,CB_FLAG%
INTEGER: HCSR%,VCSR%,RDSTR%,RDFUL%
REAL: CSEG
INTEGER: SAVESTAT%,SYC%
STRING: DUMMS[?]
INTEGER: NEOS%,TERM%,N1%,LAST_INT1%
INTEGER: INT1STAT%,CTR%
STRING: DATA.Strings[?]
INTEGER: CHAR%,STRLEN%,CB%,DAT%,LAST%,POLLOBYTE%,STATUS%
INTEGER: SRQ%,SYS%,BIT%,SENSE%,X%,N2%,Y%,ADSTAT%
STRING: END_Seqs[?],LAST_CHARS[?],SEPS[?]
STRING ARRAY(?) = SARS
STRING: TEMPS[?]
INTEGER: Runnumber%,Redo%,M%,Number%,Rownumber%,B%
INTEGER: Col%,Choice%,Row%,Time%,N%,Delay%
STRING: FileS[5],ChkS[3],NameS[60]
STRING: TitleS[?]

STRING ARRAY(6)[5] = FileS
STRING ARRAY(11)[256]: InformationS
INTEGER ARRAY(31): Expt%
INTEGER ARRAY(11): Group%
REAL ARRAY(26,31): Parameter!
STRING: ExptS[20]
REAL: Tyme,Min!,Max!,Delta!,Time!
INTEGER: L%
REAL: Timer!
INTEGER: Startnoerror%
STRING: Function[5],FunctionS[?],MonikerS[13]
REAL: Pulsevolt!,L
REAL: M,Bias!,Minholdtime!,Mindelaytime!,Data_string,Pulse!,Maxtemp!,Mintemp!
REAL: TEMP!,I,V,Experiment,Parameters
INTEGER: DATA_STRINGS%
REAL: Mintime!,Maxtime!,L!
REAL ARRAY(?) : PARAMETER!
INTEGER: DLEY%
REAL: PARAM
INTEGER: Maxloop%
STRING ARRAY(?) : Inofrmation$
REAL: Timer!
INTEGER: EXPT%,Space%
INTEGER: SER_POLL%
STRING ARRAY(?) : Ch$
REAL: O
STRING: FILE$[8],BSS[34]
REAL: M%8,Hello
STRING: XS[?]

PROCEDURE: Border()
  INTEGER ARG: Rownumber%/VAR
END PROCEDURE

PROCEDURE: Title()
  STRING ARG: Title$
END PROCEDURE

PROCEDURE: Menu()
  INTEGER ARG: Number%/VAR,Rownumber%/VAR
  STRING ARG: Name$
  INTEGER ARG: Row%/VAR
END PROCEDURE

PROCEDURE: Finish()
  INTEGER ARG: Rownumber%/VAR,Number%/VAR
  STRING ARG: Name$
END PROCEDURE

PROCEDURE: Filecheck()
  STRING ARG: File$
  INTEGER ARG: Redo%/VAR
END PROCEDURE

PROCEDURE: Placeampersand()
  INTEGER ARG: Number%,Choice%
END PROCEDURE

PROCEDURE: Removeampersand()
  INTEGER ARG: Number%,Choice%
END PROCEDURE
PROCEDURE: Settempparam()
    INTEGER ARG: Time%,Col%
    REAL ARG: Minimumtemp!/VAR,Maximumtemp!/VAR,Incrementtemp!/VAR
    STRING ARG: Expt$,$Name$
END PROCEDURE

PROCEDURE: Timeparam()
    REAL ARG: Finaltime!/VAR,Deltatime!/VAR
END PROCEDURE

PROCEDURE: Svolt()
    REAL ARG: Start!/VAR,Stop1!/VAR,Step1!/VAR,Stop2!/VAR,Step2!/VAR
END PROCEDURE

PROCEDURE: DVdt()
    REAL ARG: DVdt!/VAR,Start!/VAR,Stop!/VAR,Step!/VAR
END PROCEDURE

PROCEDURE: Settime()
    REAL ARG: Stepdelay!/VAR,initialhold!/VAR,Finalhold!/VAR,Deltahold!/VAR
    REAL ARG: Minholdtime!,Mindelaytime!
    INTEGER ARG: Type%
END PROCEDURE

PROCEDURE: Setbias()
    REAL ARG: Initialbias!/VAR,Finalbias!/VAR,Deltabias!/VAR
END PROCEDURE

PROCEDURE: Setsamples()
    REAL ARG: SAMPLES!/VAR
END PROCEDURE

PROCEDURE: Setpulse()
    REAL ARG: Initialpulse!/VAR,Finalpulse!/VAR,Deltapulse!/VAR,Lowpulse!/VAR
END PROCEDURE

PROCEDURE: Copytodisk()
    STRING ARG: FILES
END PROCEDURE

PROCEDURE: Timedelay()
    INTEGER ARG: Delay%/OPT=5
END PROCEDURE

PROCEDURE: Clearscreen()
    INTEGER ARG: Row%
END PROCEDURE

PROCEDURE: Border
    INTEGER: M%
9 REM
10 REM ************************************************************
11 REM *
12 REM *   The Procedure BORDER prints '*' around the menu.      *
13 REM *
14 REM *   to call type BORDER (ROWNUMBER%) <RETURN>             *
15 REM *
16 REM *   INTEGER: M% - used in a FOR-NEXT loop                 *
17 REM *
18 REM *   INTEGER ARGUMENT: Rownumber%/VAR - used to delineate   *
19 REM *   the maximum row for the '*'                         *
20 REM *
21 REM ************************************************************
22 REM
23 REM
100 COLOR 2,0
110 LOCATE 3,1
120 PRINT "*****************************************************"
130 PRINT "*******************"
140 Rownumber% = Rownumber% + 1
150 FOR M% = 4 TO Rownumber%
160 LOCATE M%,1
170 PRINT "***
180 LOCATE M%,79
190 PRINT "***
200 NEXT M%
210 Rownumber% = Rownumber% + 1
220 LOCATE Rownumber%,1
230 PRINT "*****************************************************"
240 PRINT "*******************"
250 Rownumber% = 0
260 COLOR 7,0
270 EXIT

END PROCEDURE

PROCEDURE: Title
INTEGER: Column%

9 REM
10 REM ************************************************************
11 REM *
12 REM *   The Procedure TITLE prints the title of the menu       *
13 REM *   transferred from the main program by TITLE$ in the     *
14 REM *   location 1,col%                                        *
15 REM *
16 REM *   to call: TITLE (COL%,TITLE$)                          *
17 REM *
18 REM *   INTEGER ARGUMENT: Col% - gives the 1-80 column number  *
19 REM *
20 REM *   STRING ARGUMENT: Title$ - gives the title string      *
21 REM *
22 REM ************************************************************
23 REM
100 Column% = CINT((79 - LEN(Title$))/2)
110 COLOR 5,0:CLS
120 LOCATE 1,Column%
130 PRINT Title$
140 COLOR 7,0
150 EXIT
END PROCEDURE

PROCEDURE: Menu
INTEGER: Col%
STRING: Number$[4]
INTEGER: Row1%
9 REM
10 REM ********************************************************************************
11 REM * 
12 REM * The Procedure MENU takes the names transferred by the * 
13 REM * calling program and prints them as a screen. * 
14 REM * 
15 REM * to call: MENU (NUMBER%,ROWNUMBER%,NAMES,ROW%) * 
16 REM * 
17 REM * INTEGER: Col% * 
18 REM * 
30 REM * INTEGER ARGUMENT: Number%/VAR,Rownumber%/VAR,Row%/VAR * 
32 REM * 
40 REM * STRING: Number$[4] * 
43 REM * 
50 REM * STRING ARGUMENT: Name$ * 
59 REM * 
60 REM ********************************************************************************
61 REM
100 COLOR 3,0
110 Number% = Number% + 1
120 Number$ = STRS(Number%) 
130 DELS(Number$,1,1)
140 IF Number% < 13 THEN Col% = 5 ELSE Col% = 42
150 IF Number% = 13 THEN Row% = 1 ELSE Row% = Row% + 1
160 LOCATE Row%+4,Col%
170 PRINT "(\:Number$\:" ",Name$
180 IF Rownumber% < CSRLIN THEN Rownumber% = CSRLIN
190 COLOR 7,0
200 EXIT
END PROCEDURE

PROCEDURE: Finish
STRING: Number$[4]
INTEGER: Column%
9 REM
10 REM ********************************************************************************
11 REM * 
12 REM * The Procedure FINISH prints the message which allows * 
13 REM * the user to exit the menu. *
14 REM * 
15 REM * to call: FINISH (ROWNUMBER%,NUMBER%,COL%,NAME$) 
16 REM * 
20 REM * INTEGER ARGUMENT: Rownumber%/VAR,Number%/VAR,col% 
22 REM * 
30 REM * STRING: Number$[4] 
32 REM * 
40 REM * STRING ARGUMENT: Name$ 
42 REM * 
50 REM ................................................................. 
51 REM 
100 Column% = CINT((77 - LEN(Name$))/2) 
110 COLOR 3,0 
120 Rownumber% = Rownumber% + 1 
130 Number% = Number% + 1 
140 Number$ = STR$(Number%) 
150 DEL$(Number$[1,1]) 
160 LOCATE Rownumber%,Column% 
170 PRINT "(;Number$;:) ";Name$ 
180 COLOR 7,0 
190 EXIT 
END PROCEDURE 

PROCEDURE: Filecheck 
STRING: Chk$[16] 
INTEGER: T%,1% 
STRING: AS[?] 
INTEGER: Maxloop% 
9 REM 
10 REM ................................................................. 
11 REM * 
12 REM * The Procedure FILECHECK sees if the directory specified 
13 REM * has been previously used. If it has then it asks the 
14 REM * user if they want to erase all previous files. If the 
15 REM * directory has not been used, it then creates the asked 
16 REM * for directory. 
17 REM * to call: FILECHECK (FILES,REDO%) 
18 REM * 
20 REM * INTEGER ARGUMENT: Rownumber%/VAR, Number%/VAR, col% 
23 REM * 
30 REM * STRING: Number$[4] 
32 REM * 
40 REM * STRING ARGUMENT: Name$ 
43 REM * 
50 REM ................................................................. 
51 REM 
99 CLOSE 
100 ON ERROR GOTO 10000 
110 OPEN "DATA" + File$ + "INFO" FOR INPUT AS #1 
120 IF ERR = 1001 OR ERR=1007 THEN GOTO 180 'If no file goto erase routine 
130 CLOSE #1
140 LOCATE 20,5:COLOR 7,0
150 PRINT * "THE FILE NUMBER YOU CHOSE HAS ALREADY BEEN USED."
160 INPUT * "DO YOU WANT TO OVERWRITE THE EXISTING FILES (Defaults to NO)"; Chk$ 
170 IF INSTR(Chk$,*"y"* ) <> 0 OR INSTR(Chk$,*"Y"*) <> 0 THEN GOTO 180 ELSE GOTO 1000
180 RESTORE, 50000
190 READ Maxloop%
210 I% = I% + 1
220 READ A$
230 KILL "\DATA\" + File$ + "\" + A$
265 IF I% < Maxloop% THEN GOTO 210
270 RMDIR "\data\" + File$
280 MKDIR "\data\" + File$
290 Redo% = 0
300 GOTO 1010
1000 Redo% = 1
1010 ON ERROR 0
1020 COLOR 7, 0
1030 LOCATE 19, 5: PRINT SPC(79); PRINT SPC(79); PRINT SPC(79); PRINT SPC(79)
1040 CLEAR
1050 EXIT
10000 REM
10001 REM * * * * * * * * * * * * * * * * * * * * * * * * * * * *
10002 REM * These are the ERROR handlers which allows the Procedure *
10003 REM * to check to see if the files or directory have been *
10004 REM * used or exist. *
10005 REM *
10006 REM * * * * * * * * * * * * * * * * * * * * * * * * * * * *
10007 REM
10010 IF ERR = 1001 THEN RESUME NEXT
10020 IF ERR = 18 THEN RESUME, 270
10030 IF ERR = 1007 THEN RESUME NEXT
10040 IF ERR = 1022 THEN RESUME, 280
10050 PRINT "I’m sorry but the FILECHECK procedure still doesn’t work."
10060 PRINT "This is error number "; ERR; " from line "; ERL
10070 END
50000 REM
50001 REM * * * * * * * * * * * * * * * * * * * * * * * * * * * *
50002 REM *
50003 REM * The DATA statements are for erasing the possible files *
50004 REM * from the hard disk. *
50005 REM *
50006 REM * * * * * * * * * * * * * * * * * * * * * * * * * * * *
50007 REM
50010 DATA 9
50020 DATA info,C-t,CGV,CG,G-t,IV,CV,Mob,Res
END PROCEDURE

PROCEDURE: Placecampersand
INTEGER: Col%, Pos%
9 REM
10 REM
11 REM * The Procedure PLACEAMPERSAND takes the numbers passed
12 REM * by CHOICE% and NUMBER% and arranges them so that an
13 REM * "@" is placed in the appropriate place to show that
14 REM * the item chosen has actually been chosen.
15 REM *
16 REM *
17 REM * to call: PLACEAMPERSAND (NUMBER%,CHOICE%)
18 REM *
19 REM * INTEGER: Pos%, Col%
20 REM *
21 REM *
22 REM * INTEGER ARGUMENT: NUMBER%, CHOICE%
23 REM *
24 REM *
25 REM
26 REM
27 REM IF Number% < 12 THEN Col% = 3
28 REM IF Number% > 12 THEN Col% = 40
29 REM IF Choice% >= 12 THEN Pos% = Choice% - 12 ELSE Pos% = Choice%
30 REM LOCATE Pos%+4,Col%
31 PRINT "@"
32 EXIT
33 END PROCEDURE

PROCEDURE: Removeampersand
INTEGER: Col%,Pos%
100 IF Number% < 12 THEN Col% = 3 ELSE Col% = 40
110 IF Choice% > 12 THEN Pos% = 12 - Choice% ELSE Pos% = Choice%
120 LOCATE Pos%+4,Col%
130 PRINT ""*
140 EXIT
150 END PROCEDURE

PROCEDURE: Settempparam
REAL: Initialtemp!,Finaltemp!,Deltatemp!,Chk!,Chk1!,Min!,Max!,Test!
INTEGER: Error%,Line%
STRING: Chk$[4]
REAL: Test
INTEGER: Delay%
REAL: Tyme
STRING: Sign$[16]
REAL: Increment!,Incrementtemp!,Maximumumtemp!

9 REM
10 REM **
11 REM * The Procedure SETTEMPPARAM allows the user to 1) set
12 REM * the initial,final and increment temperature for each
13 REM * experiment, 2) set the temperature for a TIME run, or
14 REM * 3) choose a room temperature run. This procedure also
15 REM * has a variety of error checks to make sure the chosen
16 REM * temperatures are possible.
17 REM *
18 REM *
19 REM *
20 REM * to call: SETTEMPPARAM (TIME%,COL%,INITIALTEMP,
21 REM *  FINALTEMP,DELTATEMP,EXPTS,NAMES) *
22 REM *
30 REM * INTEGER: error%,line%
32 REM *
40 REM * INTEGER ARG: time%,col%
41 REM *  min!,max!,test!
42 REM *
50 REM * REAL: initialtemp!,finaltemp!,deltatemp!,chk!,chk1!
51 REM *  min!,max!,test!
52 REM *
60 REM * REAL ARG: minimumtemp!/VAR,maximumtemp!/VAR,
61 REM *  incrementtemp!/VAR
62 REM *
70 REM * STRING: chk$  
72 REM *
80 REM * STRING ARG: expt$,name$  
82 REM *
85 REM ........................................................................
86 REM
90 ON ERROR GOTO 300000
95 IF Time% = 3 THEN GOTO 800
100 IF Time% = 1 THEN GOTO 600
105 IF Expt$ <> "Cryostat" THEN GOTO 130
110 COLOR 2,0:CLS:LOCATE 1,Col%
120 PRINT "This sets the temperature for the ";Expt$
130 COLOR 4,0:LOCATE 3,5
140 PRINT "These are the Cryostat temperature ";Name$":"  
150 COLOR 3,0:LOCATE 5,16
160 PRINT "Initial";SPC(15);"Final";SPC(15);"Increment"
170 COLOR 9,0:LOCATE 6,17
180 PRINT Minimumtemp!;SPC(17);Maximumtemp!;SPC(18);Incrementtemp!
190 COLOR 5,0:LOCATE 10,5
250 LOCATE 10,1:PRINT SPC(79)
260 LOCATE 10,5:INPUT "Enter the initial temperature ";Initialtemp!
270 IF Initialtemp! >= Minimumtemp! THEN GOTO 290  
280 Error% = 1:Line% = 1:GOTO 2000
290 IF Initialtemp! <= Maximumtemp! THEN GOTO 310
300 Error% = 2:Line% = 1:GOTO 2000
310 LOCATE 12,5:INPUT "Enter the final temperature ";Finaltemp!
320 IF Finaltemp! <= Maximumtemp! THEN GOTO 340
340 IF Finaltemp! >= Initialtemp! THEN GOTO 360
350 Error% = 8:Line% = 2:GOTO 2000
360 LOCATE 14,5:INPUT "Enter the temperature increment ";Deltatemp!
370 IF Deltatemp! > Incrementtemp! THEN GOTO 410
380 IF Deltatemp! = 0 AND Initialtemp! = Finaltemp! THEN GOTO 410
390 IF Deltatemp! >= Incrementtemp! THEN GOTO 410
400 Error% = 3:Line% = 3:GOTO 2000
410 GOSUB 1000
420 COLOR 7,0:LOCATE 20,1
430 INPUT "Do you want to change any of the above (Defaults to No)"; Chk$ 
440 Chk$ = MIDS(Chk$,1,1)
450 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 110
460 IF Expt$ = "Cryostat" THEN GOTO 560 "If for cryostat goto var. set
470 IF SGN(Increment$!) = SGN(Deltatemp!) THEN GOTO 560
480 IF SGN(Increment$!) = SGN(Deltatemp!) THEN GOTO 560
490 Minimumtemp! = Finaltemp!
500 Maximumtemp! = Initialtemp!
510 Increment$! = SGN(Increment$!) * ABS(Deltatemp!)
520 EXIT
560 Minimumtemp! = Initialtemp!
570 Maximumtemp! = Finaltemp!
580 Increment$! = Deltatemp!
590 EXIT
599 REM
600 REM "---------------------------------------------"
601 REM *  
602 REM * This section is used for a time run. It allows the  
603 REM * user to enter the temperature which will be used.  
604 REM *  
605 REM "---------------------------------------------"
606 REM
610 COLOR 2,0:CLS:LOCATE 12,5
620 PRINT "At what temperature do you want to do the time run (must be ",
630 PRINT "between ":Minimumtemp!:;" and ":Maximumtemp!:;"
640 INPUT "K";Min!
650 IF Min! >= Minimumtemp! THEN GOTO 670
660 Min! = 10:Error% = 1:Line% = 4:GOTO 2000
670 IF Min! <= Maximumtemp! THEN GOTO 690
680 Max! = 600:Error% = 2:Line% = 4:GOTO 2000
690 Minimumtemp! = Min!
700 Maximumtemp! = Min!
710 Increment$! = 0
720 EXIT
799 REM
800 REM
810 CLS:COLOR 2,0:LOCATE 12,5
820 INPUT "Enter The Room's Temperature(defaults to 290K)"; Minimumtemp!
825 IF Minimumtemp! = 0 THEN Minimumtemp! = 290
830 Maximumtemp! = Minimumtemp!
840 Increment$! = 400.0
850 EXIT
899 STOP
999 REM
1000 REM "---------------------------------------------"
1001 REM *  
1002 REM * This section checks to see if the temperatures set  
1003 REM * intersect those of the cryostat.  
1004 REM *  
1005 REM "---------------------------------------------"
1006 REM
1009 GOTO 1030
1010 IF Initialtemp! + ABS(Deltatemp!) < Finaltemp! THEN GOTO 1030
1020 Error% = 5:Line% = 1:GOTO 2000
1030 Chk! = (Initialtemp! - Minimumtemp!)/Incrementtemp!
1040 Chk!! = FIX(Chk!)
1050 Test! = Chk1! - Chk!
1060 IF Test < 0.0001 OR Test > 0.0001 THEN GOTO 1080
1070 Error% = 6:Line% = 1:GOTO 2000
1080 Chk! = Deltatemp!/Incrementtemp!
1090 Chk1! = FIX(Chk!)
1100 Test! = Chk1! - Chk!
1110 IF Test < 0.0001 OR Test > 0.0001 THEN GOTO 1130
1120 Error% = 7:Line% = 3:GOTO 2000
1130 RETURN
1199 REM
2000 REM
2001 REM ****************************
2002 REM * This is the section which prints up the error messages *
2003 REM * for temperature settings which are not allowed. *
2004 REM *
2005 REM ****************************
2006 REM
2010 COLOR 7,0:LOCATE 20,5
2020 ON Error% GOSUB 2100,2150,2200,2250,2300,2350,2400,2450
2030 Delay%= 3:GOSUB 20000
2040 COLOR 5,0:LOCATE 20,1;PRINT SPC(79);PRINT SPC(79)
2050 ON Line% GOTO 2500,2550,2600,2650
2099 REM
2100 REM -----------This is for a temperature less than the minimum--------
2101 REM -----------cryostat temperature-----------
2102 REM
2110 PRINT "The temperature must be greater than ";Minimumtemp!;"!
2120 RETURN
2129 REM
2150 REM -----------This is for a temperature greater than the maximum--------
2151 REM -----------cryostat temperature-----------
2152 REM
2160 PRINT "The temperature must be less than ";Maximumtemp!;"!"
2170 RETURN
2199 REM
2200 REM -----------This is for an increment less than the cryostat--------
2201 REM -----------temperature---------
2202 REM
2210 PRINT "The temperature increment must be greater than";Incrementtemp!;"!"
2220 RETURN
2249 REM
2250 REM -----------This is for an increment which is going in--------
2251 REM -----------the wrong direction--------
2252 REM
2255 IF SGN(Finaltemp! - Initialtemp!) = 1 THEN Sign$ = "positive"
2257 IF SGN(Finaltemp! - Initialtemp!) = -1 THEN Sign$ = "negative"
2258 IF SGN(Finaltemp! - Initialtemp!) = 0 THEN Sign$ = "no slope"
2260 PRINT "The temperature increment must be ";Sign$;"!"
2270 RETURN
2299 REM
2300 REM ------This is for an intersection of less than 2---------
2301 REM
2310 PRINT "The parameters you have set has only 1 experimental reading in"
2320 PRINT "it. It must have at least 2. ;"
2340 RETURN
2349 REM
2350 REM -------This is for a wrong initial temperature--------
2351 REM
2360 PRINT "The cryostat temperatures will never intersect with the initial"
2370 PRINT "temperature set"
2380 RETURN
2389 REM
2400 REM -------For an impossible temperature increment--------
2401 REM
2410 PRINT "The increment you want is not possible!"
2420 RETURN
2450 REM
2460 PRINT "The Final Temperature must be greater than ";Initialtemp!
2470 RETURN
2499 REM
2500 REM -------This sends the program back to the initial temp--------
2501 REM
2510 GOTO 250
2549 REM
2550 REM -----This takes the experiment to the final temperature----
2551 REM
2560 LOCATE 12,:PRINT SPC(79)
2570 GOTO 310
2599 REM
2600 REM -------This takes the experiment back to the increment--------
2601 REM
2610 LOCATE 14,1,:PRINT SPC(79)
2620 GOTO 360
2649 REM
2650 REM -----This takes the experiment back to the time segment------
2651 REM
2660 GOTO 610
19999 REM
20000 REM "*****************************************************************
20001 REM * 
20002 REM * This allows a timed delay for error messages          *
20003 REM * 
20004 REM "*****************************************************************
20005 REM
20010 Tyme = TIMER + Delay%
20020 IF Tyme > TIMER THEN GOTO 20020
20030 RETURN
30000 PRINT "The error is ";ERR;" and the line number is ";:ERL
END PROCEDURE

PROCEDURE: Timeparam
STRING: Chk$[4]
  9 REM
  10 REM *******************************************************
  11 REM *
  12 REM * The Procedure TIMEPARAM sets the final and incremental *
  13 REM * times for a constant temperature time run. *
  14 REM *
  15 REM * to call: TIMEPARAM (FINALTIME!,DELTATIME!)
  16 REM *
  20 REM * REAL ARG: Finaltime!/VAR, Deltatime!/VAR
  22 REM *
  30 REM * STRING: Chk$[4]
  32 REM *
  40 REM *******************************************************
  42 REM
100 ON ERROR GOTO 10000
110 COLOR 2,0;CLS;LOCATE 1,25
120 PRINT "This sets the TIME parameters"
130 COLOR 5,0
140 LOCATE 4,3;INPUT "Enter the amount of time this program is to run (> 5 minutes);Finaltime!"
150 LOCATE 6,3;PRINT "Enter the length of time between experiment";
160 COLOR 3,0;PRINT " set ";
170 COLOR 5,0;INPUT "runs ( > 5 minutes);Deltatime!"
180 COLOR 7,0;LOCATE 20,5
190 INPUT "Do you want to change any of the above (Defaults to No);Chk$"
200 Chk$ = MIDS(Chk$[,1,1]
210 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 110
220 Finaltime! = 60 * Finaltime!
230 Deltatime! = 60 * Deltatime!
240 EXIT
10000 CLS;LOCATE 15,5;PRINT "The error is ";:ERR;" in line ";:ERL
10010 STOP
END PROCEDURE

PROCEDURE: Svolt
STRING: Chk$[3]
  9 REM
  10 REM *******************************************************
  11 REM *
  12 REM * The Procedure SVOLT allows the user to enter the *
  13 REM * voltages (including a break voltage and two different *
  14 REM * increments. It does not include any error checking *
  15 REM * for entering incorrect voltages. *
  16 REM *
  17 REM * to call SVOLT (START!,STOP1!,STEP1!,STOP2!,STEP2!)
18 REM *
30 REM * REAL ARGUMENTS: Start1/VAR, Stop1/VAR, Stop2/VAR
32 REM * Step1/VAR, Step2/VAR
40 REM *
41 REM *******************************************************
42 REM
100 ON ERROR GOTO 10000
110 LOCATE 3,1
120 DO 20 TIMES
130 PRINT SPC(79)
140 REPEAT
145 LOCATE 4,24:COLOR 4,0
147 PRINT "This sets the Voltage settings"
150 COLOR 5,0
160 LOCATE 6,5:INPUT "Enter the Start Voltage (-100 to 100 V):";Start!
170 LOCATE 8,5:INPUT "Enter the first Stop Voltage (-100 to 100 V):";Stop1!
180 LOCATE 10,5:INPUT "Enter the first Step Voltage (> abs[0.01] V):";Step1!
190 LOCATE 12,5:INPUT "Enter the second Stop Voltage (-100 to 100 V):";Stop2!
200 LOCATE 14,5:INPUT "Enter the second Step Voltage (> abs[0.01] V):";Step2!
210 COLOR 7,0:LOCATE 20,1
220 INPUT "Do you want to change any of the above (Defaults to No)";ChkS
230 ChkS = MIDS(ChkS,1,1)
250 IF ChkS = "Y" OR ChkS = "y" THEN GOTO 110
260 ON ERROR 0
270 Step1! = VAL(STRS(Step1!))
280 Step2! = VAL(STRS(Step2!))
290 EXIT
10000 REM
10010 IF ERR = 1008 THEN RESUME
10020 COLOR 7,0:LOCATE 18,3:PRINT "Sorry the procedure ";
10030 COLOR 20,0:PRINT "SVOLT ";
10040 COLOR 7,0:PRINT "is bombing. This is error ";ERR;" and line number ";ERL
END PROCEDURE

PROCEDURE: DVdt
STRING: ChkS[3]
9 REM
10 REM *******************************************************
11 REM *
12 REM * The Procedure dVdt allows the user set the slope *
13 REM * for the C-V experiment in group 2. *
14 REM *
15 REM * to call: DVDT (Dvdt!,start!,stop!,step!) *
16 REM *
30 REM * real arg: dVdt!/VAR *
32 REM *
40 REM *******************************************************
41 REM
47 REM * real arg:start!/var,stop!/var,step!/var
100 ON ERROR GOTO 10000
110 LOCATE 3,1
120 DO 20 TIMES
130 PRINT SPC(79)
140 REPEAT
150 COLOR 4:LOCATE 4,23
160 PRINT "This sets the dV/dt slope"
170 COLOR 5,0
180 LOCATE 6,5:INPUT "Enter the dV/dt setting (0.001 to 1 V/s)";DVdt
185 LOCATE 8,5:INPUT "Enter the Start Voltage (-100 to 100 V)";Start!
187 LOCATE 10,5:INPUT "Enter the Stop Voltage (-100 to 100 V)";Stop!
188 LOCATE 12,5:INPUT "Enter the Step Voltage (-10 to 10 V)";Step!
190 LOCATE 20,5:COLOR 7,0
200 INPUT "Do you want to change any of the above (Defaults to No)";Chk$ 
210 Chk$ = MID$(Chk$,1,1)
220 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 110
230 EXIT
1000 REM
10001 REM *****************************************************
10002 REM *                                                    *
10003 REM * This is the error procedures. It corrects the expected *
10004 REM * errors and kills the program and writes which error  *
10005 REM * and line number for an unexpected error            *
10006 REM *                                                    *
10007 REM *****************************************************
10008 REM
10010 IF ERR = 1008 THEN RESUME
10020 COLOR 7,0:LOCATE 20,5:PRINT "Sorry the procedure ";
10030 COLOR 20,0:PRINT "dVdt ";
10040 COLOR 7,0:PRINT "bombing. The error is ";ERR;" in line number";ERL
10050 STOP
END PROCEDURE

PROCEDURE: Settime
STRING: Chk$[3]
REAL: Tyme!
INTEGER: M%
10 REM
11 REM *****************************************************
12 REM *
13 REM * The Procedure SETTIME allows the user to enter the *
14 REM * Step Delay Time, Initial Hold Time, Final Hold Time *
15 REM * and Delta Hold Time in seconds.                     *
16 REM *
17 REM * to call: SETTIME (STEPDELAY!,INITIALHOLD!,FINALHOLD!, *
18 REM * DELTAHOLD!,MINHOLDTIME!,MINDELAYTIME!,TYPE%)         *
19 REM *
20 REM * REAL ARG: Stepdelay!/VAR, Initialhold!/VAR,        *
21 REM * Deltahold!/VAR, Finalhold!/VAR                      *
22 REM *
23 REM *****************************************************
24 REM
100 ON ERROR GOTO 10000
110 LOCATE 3,1
120 DO 21 TIMES
130 PRINT SPC(79)
140 REPEAT
150 LOCATE 4,25:COLOR 4.0
160 PRINT "This sets the Time Parameters"
170 COLOR 5.0
180 IF Type% = 3 THEN GOTO 210
190 LOCATE 6,5:PRINT "Enter the Step Delay Time, in seconds (t > ";,Min Delay Time!;
200 INPUT " seconds")\Stepdelay!
210 LOCATE 8,5:PRINT "Enter the Initial Hold Time, in seconds (t > ";,Min Hold Time!;
220 INPUT " seconds")\Initial Hold!
230 IF Type% = 1 THEN GOTO 290
240 IF Type% = 3 THEN GOTO 290
250 LOCATE 10,5:PRINT "Enter the Final Hold Time, in seconds (t > ";,Min Hold Time!;
260 INPUT " seconds")\Final Hold!
270 LOCATE 12,5:PRINT "Enter the Delta Hold Time, in seconds (t > ";,Min Hold Time!;
280 INPUT " seconds")\Delta Hold!
290 IF Steptdelay! < Min Delay Time! OR Initial Hold! < Min Hold Time! THEN GOSUB 10100
300 LOCATE 20,5:COLOR 7.0
310 INPUT "Do you need to change any of the above (Defaults to No)";Chk$ 320 Chk$ = MIDS(Chk$,1,1)
330 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 110
340 EXIT
9999 STOP
10000 REM
10001 REM ************************************************************
10002 REM *
10003 REM * This error routine takes care of any expected errors. *
10004 REM * If an unexpected error occurs, the program prints the  *
10005 REM * procedure name, error number and line number to screen *
10006 REM * then ends the program.   *
10007 REM *
10008 REM ************************************************************
10009 REM
10010 IF ERR = 1008 THEN RESUME
10020 LOCATE 23,5:COLOR 7.0
10030 PRINT "Sorry the Procedure ";
10040 COLOR 20,0:PRINT "settime";
10050 COLOR 7,0:PRINT " has bombed. The error number is ";ERR;" from line ";ERL
10099 STOP
10100 REM
10101 REM ************************************************************
10102 REM *
10103 REM * This error is given if either the initial hold time or *
10104 REM * the step delay time is less than the asked for values. *
10105 REM * It then returns them to the beginning of this routine. *
10106 REM *
10107 REM ************************************************************
10108 REM
10110 LOCATE 20,5:COLOR 7.0
10120 PRINT "The times you have chosen are not within the meter's resolution."
10130 LOCATE 21,5:PRINT "Please choose different parameters."
10140 Tyme! = TIMER + 5
10150 IF TIMER < Tyme! THEN GOTO 10150
10160 RETURN,100
10170 STOP
END PROEDURE

PROCEDURE: Setbias
STRING: Chk$[3]

9 REM
10 REM *********************************************************
11 REM *
12 REM * The Procedure SETBIAS sets the different bias voltages *
13 REM *
14 REM * to call: SETBIAS (INITIALBIAS!,FINALBIAS!,DELTABIAS! *
15 REM *
16 REM * real arg:initialbias!/VAR,deltabias!/VAR,finalbias!/VAR *
17 REM *
18 REM *********************************************************
19 REM
20 ON ERROR GOTO 10000
21 LOCATE 3,1
22 DO 21 TIMES
23 PRINT SPC(79)
24 REPEAT
25 LOCATE 4,25:COLOR 4,0
26 PRINT 'This sets the Bias Voltages'
27 COLOR 5,0
28 LOCATE 6,5:INPUT "Enter the Initial Bias Voltage (-99 to 99 V);";Initialbias!
29 LOCATE 8,5:INPUT "Enter the Final Bias Voltage (-99 to 99 V);";Finalbias!
30 LOCATE 10,5:INPUT "Enter the Delta Bias Voltage ( > abs(0.01) V);";Deltabi
31 LOCATE 20,5:COLOR 7,0
32 INPUT "Do you need to change any of the above (Defaults to No)";Chk$
33 Chk$ = MID$(Chk$,1,1)
34 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 110
35 EXIT
36 REM
37 REM *********************************************************
38 REM *
39 REM * These are the error messages which take care of the *
40 REM * expected errors. If an unexpected error takes place *
41 REM * an error messages is given along with the line number *
42 REM * and the program stops. *
43 REM *
44 REM *********************************************************
45 REM
46 IF ERR = 10008 THEN RESUME
47 LOCATE 23,5:COLOR 7,0
48 PRINT "The procedure ";
49 COLOR 20,1:PRINT "SETBIAS";
COLOR 7,0:PRINT "has bombed. The error number is ";ERR;" and the line number ";:ERL
STOP
END PROCEDURE

PROCEDURE: Setsamples
STRING: Chk$[3]
REM * to call: SETSAMPLES (samples!)
30 REM * REAL ARG: SAMPLES!Var
100 ON ERROR GOTO 10000
110 LOCATE 3,1
120 DO 21 TIMES
130 PRINT SPC(79)
140 REPEAT
150 LOCATE 4,25:COLOR 4,0
160 PRINT "This sets the Number of Samples"
170 COLOR 5,0
180 LOCATE 6,5:INPUT "Enter the number of Samples needed ( > 1)";SAMPLES!
190 LOCATE 20,5:COLOR 7,0
200 INPUT "Do you want to change any of the above (Defaults to No)";Chk$
210 Chk$ = MID$(Chk$,1,1)
220 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 110
230 SAMPLES! = INT(SAMPLES!)
240 EXIT
10000 STOP
10001 REM ***********************************************************************
10002 REM *
10003 REM * This corrects any expected errors. If an error is *
10004 REM * unexpected the program bombs and an error message with *
10005 REM * its line number. *
10006 REM *
10007 REM ***********************************************************************
10008 REM
10010 IF ERR = 1008 THEN RESUME
10020 LOCATE 1,5:COLOR 7,0
10030 PRINT "The procedure ";
10040 COLOR 20,0:PRINT "SETSAMPLES ";
10050 COLOR 7,0:PRINT "has bombed. The error is ";:ERR;" and line number ";:ERL
10060 STOP
END PROCEDURE

PROCEDURE: Setpulse
STRING: Chk$[3]
9 REM
10 REM ***********************************************************************
11 REM *
12 REM * The Procedure SETPULSE allows the user to enter the *
13 REM * Initial High Pulse, Final High Pulse, Delta High Pulse *
14 REM * and Low Pulses for the voltages in the C-t program *
15 REM *
16 REM * to call: SETPULSE (INITIALPULSE!,FINALPULSE!, *
17 REM * DELTAPULSE!,LOWPULSE! *
18 REM *
20 REM * REAL ARG: Initialpulse!/VAR, Finalpulse!/VAR, *
21 REM * DeltaPulse!/VAR *
22 REM *
30 REM * STRING: Chk$[3] *
32 REM *
40 REM *------------------------------------------------------------------------*
42 REM
110 LOCATE 3,1
120 DO 21 TIMES
130 PRINT SPC(79)
140 REPEAT
150 LOCATE 4,25:COLOR 4,0
160 PRINT "This sets the Pulse Voltages"
165 COLOR 5,0
170 LOCATE 6,5:INPUT "Enter the Initial High Pulse (-7 to +7 V)";initialpulse!
180 LOCATE 8,5:INPUT "Enter the Final High Pulse (-7 to +7 V)";Finalpulse!
190 LOCATE 10,5:INPUT "Enter the Delta High Pulse ( > .01 V)";DeltaPulse!
200 LOCATE 12,5:INPUT "Enter the Low Pulse (-7 to +7 V)";Lowpulse!
210 LOCATE 20,5:COLOR 7,0
220 INPUT "Do you want to change any of the above (Defaults to No)";Chk$
230 Chk$ = MIDS(Chk$,1,1)
240 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 110
250 IF DeltaPulse! = 0 THEN Finalpulse! = 0
260 EXIT
END PROCEDURE

PROCEDURE: Copytodisk
INTEGER: Drive%
STRING: Chk$[3],AS[?]
20 REM * integer:Drive%
30 REM * string:chk$[4]
40 REM * STRING ARG:FILE$
100 Drive% = 1
110 CLS:COLOR 3,0:LOCATE 12,5
120 PRINT "Do you want to copy the files to a ";
130 COLOR 26,0
140 ON Drive% GOSUB 1000,2000
150 LOCATE 12,44:COLOR 3,0:INPUT " floppy diskette (Defaults to No)";Chk$
160 Chk$ = MIDS(Chk$,1,1)
170 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 180 ELSE GOTO 280
180 LOCATE 15,5:COLOR 6,0
190 PRINT "Please insert a BLANK FORMATTED ";
200 COLOR 24,0
210 ON Drive% GOSUB 1000,2000
220 LOCATE 15,7:COLOR 5,0:PRINT " floppy diskette into drive";
230 ON Drive% GOSUB 1050,2050
240 PRINT " and press any key to begin copying."
250 AS = INKEY$:IF AS = "" THEN GOTO 250
260 CLS

123
270 ON Drive% GOSUB 1100,2100
280 Drive% = Drive% + 1
290 IF Drive% < 3 THEN GOTO 110 ELSE EXIT
1000 REM
1010 PRINT "1.2 Mbyte";
1020 RETURN
1050 PRINT "A";
1060 RETURN
1100 SHELL "SEM\COPYTOABAT" + FILES
1110 RETURN
2000 REM
2010 PRINT "360 Kbyte";
2020 RETURN
2050 PRINT "B";
2060 RETURN
2100 SHELL "SEM\COPYTOB.BAT" + FILES
2110 RETURN
END PROCEDURE

PROCEDURE: Timedelay
REAL: Tyme!
    22 REM * integer arg:delay%/opt = 5
    32 REM * real:tyme!
    100 Tyme! = TIMER
    110 IF TIMER < Tyme! + Delay%/ THEN GOTO 110
    120 EXIT
END PROCEDURE

PROCEDURE: Clearscren
INTEGER: M%
    1 REM
    2 REM **************************************************************
    3 REM *
    5 REM * to call: CLEARSCREEN (ROW%)  *
    19 REM *
    20 REM * INTEGER ARG:row%  *
    21 REM *
    31 REM **************************************************************
    32 REM
    100 FOR M% = Row% TO 23
    110 LOCATE M%,1:PRINT SPC(79)
    120 NEXT M%
    130 EXIT
END PROCEDURE

'MAIN Program:

200 REM
201 REM **************************************************************
202 REM *
203 REM * This section prints a welcome message to the screen, *
204 REM * clears all variables, changes the drive and directory *
205 REM * to "C:\", sets the appropriate devices to remote and *
206 REM * initializes the error handler. *
207 REM *
208 REM *******************************************************
209 REM
210 CLS:STATUSLINE OFF
220 COLOR 2,0,8:LOCATE 12,30
230 PRINT "MEDUSA welcomes you"
240 LOCATE 14,12:PRINT "Materials and Electronic Device, Ultimate System Analyzer"
250 Timedelay (5)
260 CLEAR
270 ON ERROR GOSUB 9500
280 DRIVES$ = "C:\":DIRS$ = "%"
290 PARAMS = "INIT/1/6310/P":GOSUB 50000
300 PARAMS = "SDR/5,12,16,17,8":GOSUB 50000
310 REM
311 REM *******************************************************
312 REM *
313 REM * This checks to see if the batch file has come from *
314 REM * the program "RUNIT" and if it has it branches to *
315 REM * see if the user wants to use the same information *
316 REM * block. If the batch file hasn't come from "RUNIT" then *
317 REM * a message about turning on the printer is sent to *
318 REM * screen and the printer is sent a code to configure it. *
319 REM *
320 REM *******************************************************
321 REM
330 OPEN "DATA:REDO" FOR INPUT AS #1
340 GOSUB 9000
350 COLOR 2,0,0:CLS:LOCATE 12,25
360 PRINT "Please turn on the printer"
390 Timedelay(3)
400 GOTO 1210
401 REM *******************************************************
402 REM *
404 REM * information to correctly identify the sample. *
405 REM *
406 REM *******************************************************
407 REM
410 Redo% = 0
420 COLOR 2,0:CLS:LOCATE 3,5
430 INPUT *Enter the Directory number (between 1 and 999)*File$
440 Information$(1) = File$
450 IF VAL(File$) < 1 OR VAL(File$) > 999 THEN GOTO 420
460 Filecheck (File$,Redo%) 
470 IF Redo% = 1 THEN GOTO 410
480 Information$(2) = DATES$
490 COLOR 2,0
500 LOCATE 5,5
505 LINE INPUT "Enter Experimenter's Name. ";Information$(3)
510 LOCATE 7,5
515 LINE INPUT "Enter the sample number. ";Information$(4)
520 LOCATE 9,5
525 LINE INPUT "Enter the Cryostat chronometer reading. ";Information$(5)
530 LOCATE 11,5
535 LINE INPUT "Enter any comments. ";Information$(6)
760 LOCATE 20.5:COLOR 7,0
770 INPUT "Do you want to change any of the above (Defaults to No)";Chk$
780 Chk$ = MID$(Chk$,1,1)
790 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 800 ELSE GOTO 1510
800 REM
801 REM "***********************************************************************"
802 REM *
803 REM * This allows the user to change any part of the 
804 REM * information block, using the various Procedures. 
805 REM *
806 REM "***********************************************************************"
807 REM
808 Number% = 0:Row% = 0
809 Title$ = "INFORMATION CHANGES"
810 Title (Title$)
812 RESTORE,60010
814 READ Maxloop%
815 DO Maxloop% TIMES
817 READ Name$
818 Menu (Number%,Rownumber%,Name$,Row%) REM
819 REPAT
900 Name$ = "Finished changing information"
910 Finish (Rownumber%,Number%,Name$)
920 Border (Rownumber%)
930 DO
940 Clearscreen (20)
950 COLOR 6,0:LOCATE 20,5
960 INPUT "Enter which number you want to change";Choice%
970 IF Choice% < 1 OR Choice% > Number% THEN GOTO 940
980 IF Choice% = Number% THEN EXIT TO,1510
990 Placeamperand (Number%,Choice%)
995 Clearscreen (20)
1000 LOCATE 20,3:COLOR 4,0
1010 PRINT "The current information is: ";Information$(Choice%)
1020 LOCATE 22,1:LINE INPUT "What is the new information? ";Information$(Choice%)
1030 IF Choice% <= 1 THEN GOTO 1100
1040 Redo% = 0
1050 Clearscreen (20)
1060 Filecheck (Information$(1),Redo%)
1070 IF Redo% = 1 THEN GOTO 1090
1080 File$ = Information$(1)
1090 Information$(1) = File$
1100 Placeamperand (Number%,Choice%)
1110 REPEAT
1200 REM
1201 REM *-------------------------------------------------------------*
1202 REM *
1203 REM * This section allows the user to choose which type of *
1204 REM * temperature run he wishes. The choices are a 1)Time run *
1205 REM * 2)Temp run; 3)Room Temp; 4)Exit to graphs *
1206 REM *
1207 REM *-------------------------------------------------------------*
1208 REM
1210 Number%= 0:Row%= 0
1220 Title$ = "Menu for choosing TIME/TEMPERATURE Run"
1230 Title (Title$)
1240 RESTORE,60410
1250 READ Maxloop%
1260 DO Maxloop% TIMES
1270 READ Name$
1280 Menu (Number%,Rownumber%,Name$,Row%)
1290 REPEAT
1300 Name$ = "EXIT to Graphing Routines"
1310 Finish (Rownumber%,Number%,Name$)
1320 Border (Rownumber%)
1330 Cl earscreen (20)
1340 COLOR 6,0:LOCATE 20,5
1350 INPUT "Choose which type of run";Choice%
1360 IF Choice% < 1 OR Choice% > Number% THEN GOTO 1330
1370 IF Choice% = Number% THEN GOTO 8910
1380 Expt%(26) = Choice%
1390 RESTORE,60010
1400 READ M%
1410 M% = M% + 1
1420 RESTORE,60600
1430 DO Choice% TIMES
1440 READ Information$:M%()
1450 REPEAT
1455 IF INSTR(UPPERS$(Chk$),"Y")=0 THEN GOTO 410
1500 REM
1501 REM *-------------------------------------------------------------*
1502 REM *
1503 REM * This section allows the user to choose between groups. *
1504 REM *
1505 REM *-------------------------------------------------------------*
1506 REM
1510 Number% = 0:Row% = 0
1520 RESTORE,60100
1530 Title$ = "GROUP TYPES"
1540 Title (Title$)
1550 READ Maxloop%
1560 DO Maxloop% TIMES
1570 READ Name$
1580 Menu (Number%,Rownumber%,Name$,Row%)
1590 REPEAT
1600 Border (Rownumber%)  
1610 Clearscreen (20)  
1620 COLOR 6,0;LOCATE 20,5  
1630 INPUT "Which group of experiments do you want to perform";Choice%  
1640 IF Choice% < 1 OR Choice% > Number% THEN GOTO 1610  
1650 Exp1%(25) = Choice%  
1700 REM  
1701 REM ===>  
1702 REM *  
1703 REM * This section allows the user to pick which experiments  
1704 REM * he wants to run from the group chosen previously  
1705 REM *  
1706 REM ===>  
1707 REM  
1710 Number% = 0;Row% = 0  
1720 RESTORE,60200  
1730 Title$ = "EXPERIMENTAL MENU"  
1740 Title (Title$)  
1750 READ B%  
1760 FOR M% = 1 TO B%  
1770 READ Group%(M%)  
1780 NEXT M%  
1790 GOSUB 41000  'Used to set Read Pointer  
1800 DO Group%(Exp1%(25)) TIMES  
1810 READ Name$  
1820 Menu (Number%,Rownumber%,Name$,Row%)  
1830 REPEAT  
1840 Name$ = "Finished choosing experiments"  
1850 Finish (Rownumber%,Number%,Name$)  
1860 Border (Rownumber%)  
1870 DO  
1880 Clearscreen (20)  
1890 LOCATE 20,5;COLOR 6,0  
1900 INPUT "Which experiment do you want to run";Choice%  
1910 IF Choice% < 1 OR Choice% > Number% THEN GOTO 1880  
1920 IF Choice% = Number% THEN EXIT  
1930 Placecampersand (Number%,Choice%)  
1940 Exp1%(Choice%) = Choice%  
1950 REPEAT  
1960 REM  
1961 REM -------This asks if any experiments are to be deleted-------  
1962 REM  
1970 DO  
1980 Clearscreen (20)  
1990 LOCATE 20,1;COLOR 7,0  
2000 INPUT "Do you need to delete any of the above (Defaults to No)";Chk$  
2010 Chk$ = MID$(Chk$,1,1)  
2020 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 2030 ELSE EXIT  
2030 Clearscreen (20)  
2040 LOCATE 20,1;COLOR 6,0  
2050 INPUT "Which of the above do you need to delete";Choice%
2060 IF Choice% < 1 OR Choice% > Number% - 1 THEN GOTO 2090
2070 Removampersand (Number%,Choice%)
2080 Expt%(Choice%) = 0
2090 REPEAT
2100 REM
2102 REM -----This asks if any experiments are to be added-----
2103 REM
2110 DO
2120 Clearscreen (20)
2130 LOCATE 20,1:COLOR 7,0
2140 INPUT *Do you want to add any to the above (Defaults to NO)*;Chk$
2150 Chk$ = MID$(Chk$,1,1)
2160 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 2170 ELSE EXIT
2170 Clearscreen (20)
2180 COLOR 6,0;LOCATE 20,5;INPUT *Which experiment do you want to add*;Choice%
2190 IF Choice% < 1 OR Choice% > Number% - 1 THEN GOTO 2220
2200 Expt%(Choice%) = Choice%
2210 Placeampersand(Number%,Choice%)
2220 REPEAT
2230 REM
2231 REM *********************************************************
2232 REM *
2233 REM * This section returns the program to the Temperature *
2234 REM * run type choosing section if no experiments were chosen.*
2235 REM *
2236 REM *********************************************************
2237 REM
2240 FOR M% = 1 TO 24
2250 IF Expt%(M%) <> 0 THEN EXIT TO,2500
2260 NEXT M%
2270 CLS;COLOR 7,0;LOCATE 12,36
2280 PRINT "NO experiments were chosen"
2290 Timedelay (4)
2300 GOTO 1210
2500 REM
2501 REM *********************************************************
2502 REM *
2503 REM * This allows the user to set the Cryostat temperature(s) *
2504 REM * and the times if it is a TIME run. *
2505 REM *
2506 REM *********************************************************
2507 REM
2510 ON Expt%(26) GOSUB 2600,2700,2900
2599 STOP
2600 REM
2601 REM *********************************************************
2602 REM *
2603 REM * This subroutine allows the user to set the time *
2604 REM * variables and the 1 temperature for a time run. *
2605 REM *
2606 REM *********************************************************
2607 REM
2610 Parameter!(25,1) = 30.0
2620 Parameter!(25,2) = 600.0
2630 Parameter!(25,3) = 1.0
2640 GOSUB 3500
2650 Timeparam(Parameter!(25,5),Parameter!(25,6))
2660 RETURN,4000
2699 STOP
2700 REM
2701 REM *****************************************************
2702 REM *
2703 REM * This subroutine allows the user to set the temperatures *
2704 REM * for a temp run. *
2705 REM *
2706 REM *****************************************************
2707 REM
2710 Parameter!(25,1) = 30.0
2720 Parameter!(25,2) = 600.0
2730 Parameter!(25,3) = 1.0
2740 GOSUB 3500
2750 RETURN,4000
2799 STOP
2900 REM
2901 REM *****************************************************
2902 REM *
2903 REM * This subroutine allows the user to set the room *
2904 REM * temperature. *
2905 REM *
2906 REM *****************************************************
2907 REM
2910 GOSUB 3500
2930 RETURN,4000
3499 STOP
3500 REM
3501 REM *****************************************************
3502 REM *
3503 REM * This section goes to the procedure SETTEMPPARAM to *
3504 REM * allow the user to set what temperatures for the *
3505 REM * cryostat. *
3506 REM *
3507 REM *****************************************************
3508 REM
3510 Col% = 20
3520 Expt$ = "Cryostat"
3530 Name$ = "limits"
3540 Settempparam
3550 (Expt!(26),Col%,Parameter!(25,1),Parameter!(25,2),Parameter!(25,3),Expt$,Name$)
3550 Parameter!(25,4) = Parameter!(25,1)
3560 RETURN
3999 STOP
4000 REM

130
4001 REM ***************************************************************
4002 REM *
4003 REM * This section branches to the parameter setting *
4004 REM * subroutines selected by the user. *
4005 REM *
4006 REM ***************************************************************
4007 REM
4010 ON Expt\%(25) GOTO 4100,4200,4300,4400
4020 STOP
4100 REM
4101 REM ***************************************************************
4102 REM *
4103 REM * This is to branch to the set-up routine for the *
4104 REM * experiments in group I. *
4105 REM *
4106 REM ***************************************************************
4107 REM
4110 L% = 1
4120 ON Expt\%(L\%) GOSUB 10000,10000,12000,13000
4130 L% = L% + 1
4140 IF L% <= Group\%(Expt\%(25)) THEN GOTO 4120
4150 GOTO 8010
4199 STOP
4200 REM
4201 REM ***************************************************************
4202 REM *
4203 REM * This is to branch to the set-up routine for the *
4204 REM * experiments in group II. *
4205 REM *
4206 REM ***************************************************************
4207 REM
4210 L% = 1
4220 ON Expt\%(L\%) GOSUB 14000,15000
4230 L% = L% + 1
4240 IF L% <= Group\%(Expt\%(25)) THEN GOTO 4220
4250 GOTO 8010
4299 STOP
4300 REM
4301 REM ***************************************************************
4302 REM *
4303 REM * This is the branch to Group III. Since no parameters *
4304 REM * need to be set it goes immediately to the lprint *
4305 REM * statements. *
4306 REM *
4307 REM ***************************************************************
4308 REM
4310 CLS:LOCATE 1,20:COLOR 2,0
4320 PRINT "This sets the";
4330 COLOR 18,0:PRINT " MOBILITY";
4340 COLOR 2,0:PRINT " Parameter Settings"
4350 COLOR 14,0 : SET CURSOR 13,6 : INPUT 'Enter the Current Bias in mA ( <9 mA ).
PARAMETER!(1,17)
4360 IF Parameter!(1,17) <= 0 OR Parameter!(1,17) > 9 THEN GOTO 4310
4370 Parameter!(1,17) = Parameter!(1,17) * 1E-03
4380 GOTO 8010
4400 REM
4401 REM **********************************************
4402 REM *
4403 REM * This is the branch to Group IV. Since no parameters *
4404 REM * need to be set it goes immediately to the lprint *
4405 REM * statements. *
4406 REM *
4407 REM **********************************************
4408 REM
4410 CLS:LOCATE 1.15:COLOR 2,0
4420 PRINT "This sets the";
4430 COLOR 18,0:PRINT " 4-point Resistivity";
4440 COLOR 2,0:PRINT " Parameter Settings"*
4450 COLOR 14,0 : SET CURSOR 13,6 : INPUT "Enter the Current Bias in mA ( <100 mA )."
4460 IF Parameter!(1,17) <= 0 OR Parameter!(1,17) > 100 THEN GOTO 4410
4480 GOTO 8010
8000 REM
8001 REM **********************************************
8002 REM *
8003 REM * This section saves all the information necessary to *
8004 REM * running the experiments. It also prints out a list *
8005 REM * of the information block, the cryostat settings, the *
8006 REM * time settings and all the parameters for each *
8007 REM * experiment chosen. *
8008 REM *
8009 REM **********************************************
8010 REM
8011 REM ------This prints and saves the information block------
8012 REM
8014 LPRINT CHR$(24);CHR$(27);CHR$(58)
8015 LPRINT
CHR$(27);CHR$(68);CHR$(35);CHR$(45);CHR$(55);CHR$(65);CHR$(75);CHR$(85);CHR$(0)
8020 OPEN "DATA" + File$ + "INFO" FOR OUTPUT AS #1
8030 RESTORE,60010
8040 READ M%
8050 Mloop% = M% + 1
8060 LPRINT CHR$(9);CHR$(9);CHR$(9);CHR$(9);CHR$(9);CHR$(9);DATE$
8070 FOR M% = 1 TO Mloop%
8080 READ Name$
8090 PRINT #1,Name$
8100 PRINT #1,Information$(M%)
8110 IF M% = 2 THEN GOTO 8140
8120 LPRINT SPC(10);Name$
8130 LPRINT SPC(10);Information$(M%)
8140 NEXT M%
8150 CLOSE #1

132
8160 REM
8161 REM ------This prints and saves the Cryostat and Time Settings-------
8162 REM
8170 OPEN "DATA\PARAM" FOR OUTPUT AS #1
8180 RESTORE,60320
8190 LPRINT CHR$(10);CHR$(10)
8200 LPRINT SPC(10);"These are the Cryostat and Time settings"*
8210 LPRINT
8220 FOR M% = 1 TO 6
8230 READ Name$
8240 PRINT #1,Parameter$(25,M%)  
8250 IF M% = 4 THEN GOTO 8270
8260 LPRINT SPC(10);Name$,Parameter$(25,M%)
8270 NEXT M%
8280 CLOSE #1
8290 LPRINT CHR$(10)
8300 REM
8310 REM ------This prints the parameter settings-------
8320 REM
8310 LPRINT SPC(10);"These are the parameter settings for the chosen experiments"*
8320 GOSUB 40000
8330 LPRINT
8340 LPRINT SPC(10);"PARAMETER";CHR$(09);
8350 FOR M% = 1 TO Group%(Expt%(25))
8360 READ Name$
8370 IF Expt%(M%) = 0 THEN GOTO 8390
8380 LPRINT Name$,CHR$(09);
8390 NEXT M%
8400 LPRINT CHR$(10)
8410 RESTORE,60310
8420 READ Maxloop%
8430 FOR M% = 1 TO Maxloop%
8440 READ Name$
8450 IF M% > 3 AND M% < 7 THEN GOTO 8520
8460 LPRINT SPC(10);Name$,CHR$(09);
8470 FOR N% = 1 TO Group%(Expt%(25))
8480 IF Expt%(N%) = 0 THEN GOTO 8500
8490 LPRINT Parameter$(N%,M%);CHR$(09);
8500 NEXT N%
8510 LPRINT CHR$(13);
8520 NEXT M%
8530 LPRINT CHR$(12)
8540 REM
8550 REM ------This saves which experiments are to be run-------
8560 REM
8550 OPEN "DATA\EXPT" FOR OUTPUT AS #1
8560 PRINT #1,File$
8570 FOR M% = 1 TO 30
8580 PRINT #1,Expt%(M%)
8590 NEXT M%
8600 CLOSE #1

133
8610 REM
8611 REM ------This saves the parameters for the experiments------
8612 REM
8620 OPEN "DATA\PARAM" FOR APPEND AS #1
8630 RESTORE,60310
8640 READ Maxloop%
8650 FOR N% = 1 TO Group%(Expt%(25))
8660 IF Expt%(N%) = 0 THEN GOTO 8700
8670 FOR M% = 1 TO Maxloop%
8680 PRINT #1, Parameter!(N%,M%)
8690 NEXT M%
8700 NEXT N%
8710 CLOSE #1
8800 REM
8801 REM ******************************************************
8802 REM *
8803 REM * This exits the "PARAMETER" program and goes to the *
8804 REM * program "RUNIT". *
8805 REM *
8806 REM ******************************************************
8807 REM
8808 REM
8809 CLS
8810 SYSTEM
8899 STOP
8900 REM
8901 REM ******************************************************
8902 REM *
8903 REM * This allows the user to go directly to the graphics *
8904 REM * routines if he has chosen to do so in the TEMP RUN *
8905 REM * section. *
8906 REM *
8907 REM ******************************************************
8908 REM
8910 OPEN "DATA\END" FOR OUTPUT AS #1
8920 PRINT #1,"Howdy there pardner"
8930 CLOSE
8940 CLS
8950 SYSTEM
8999 STOP
9000 REM
9001 REM ******************************************************
9002 REM *
9003 REM * This subroutine allows the user to reuse the *
9004 REM * information block used in the previous experiment. *
9005 REM *
9006 REM ******************************************************
9007 REM
9010 CLOSE
9013 LPRINT CHR$(24)
9020 CLS:COLOR 10,0,0:LOCATE 12,5
9030 INPUT "Do you want to use the same information block (Defaults to No)"; Chk$
9040 Chk$ = MID$(Chk$,1,1)
9050 IF Chk$ = "Y" OR Chk$ = "y" THEN GOTO 9090 ELSE GOTO 9260
9090 OPEN "DATA\REDO" FOR INPUT AS #1
9100 INPUT #1,File$
9110 CLOSE #1
9160 RESTORE,60010
9170 READ Maxloop%
9180 OPEN "DATA\" + FILE$ + "INFO" FOR INPUT AS #1
9190 INPUT #1,BS$,BS$
9200 FOR M% = 2 TO Maxloop%
9210 INPUT #1,BS$,Information$(M%)
9220 NEXT M%
9230 CLOSE #1
9233 CLEARSCREEN (17):COLOR 10,0,0
9234 LOCATE 17,5:INPUT "Enter the directory number (between 1 and 999)");File$
9235 IF VAL(File$) < 1 OR VAL(File$) > 999 THEN GOTO 9233
9236 Information$(1) = File$
9237 Redo% = 0
9238 Filecheck (File$,Redo%)
9239 IF Redo% = 1 THEN GOTO 9233
9240 KILL "DATA\REDO"
9250 GOTO 1210
9260 KILL "DATA\REDO"
9270 GOTO 1210
9499 STOP
9500 REM    **********************************************************************
9501 REM     *
9502 REM    *   This is the ERROR handling routines. It checks to see   *
9503 REM    *   what error is and attempts to fix it.                  *
9504 REM     *
9505 REM    **********************************************************************
9506 REM
9510 IF ERR = 1008 THEN RESUME NEXT
9520 IF ERR = 1001 THEN RESUME,350
9530 IF ERR = 1007 THEN RESUME,350
9540 CLS:COLOR 7,0:LOCATE 11,5
9550 PRINT"Sorry the main program is bombing. This is the error ";ERR
9560 LOCATE 12,5:PRINT "The line number is ";ERL
9570 STOP
9999 STOP
10000 REM
10001 REM    **********************************************************************
10002 REM     *
10003 REM     *   This is the Capacitance vs. Time and Conductance vs. *
10004 REM     *   Time Subroutine. It sets the parameters and then sends *
10005 REM     *   them to the C-V meter and Pulse Generator for checking. *
10006 REM     *
10007 REM    **********************************************************************
10008 REM
10010 CLS:LOCATE 1,19:COLOR 2,0
10020 IF L% = 1 GOTO 10060      'If C-t skip next 2 instructions
10030 Function$ = "FN6"
10040 Name$ = "G-t"
10050 GOTO 10080 'Since G-t skip C-t set-up
10060 Function$ = "FNS"
10070 Name$ = "C-t"
10080 COLOR 2,0:PRINT "This sets the ";
10090 COLOR 18,0:PRINT Name$;
10100 COLOR 2,0:PRINT " Experiment Parameters"
10110 Parameter!(L%,1) = Parameter!(25,1)
10120 Parameter!(L%,2) = Parameter!(25,2)
10130 Parameter!(L%,3) = Parameter!(25,3)
10140 Parameter!(L%,4) = Parameter!(25,4)
10150 IF Expt%(26) = 3 OR Expt%(26) = 1 THEN GOTO 10220
10160 REM
10161 REM -----Sets Temperature Parameters-----
10162 REM
10170 Expt$ = Name$ + " Experiment"
10180 Name$ = "Settings"
10190 Col% = 20
10200 Settempparam
(Time%,Col%,Parameter!(L%,1),Parameter!(L%,2),Parameter!(L%,3),Expt$ ,Name$)
10210 REM
10211 REM -----Sets Bias Voltages-----
10212 REM
10220 Setbias (Parameter!(L%,17),Parameter!(L%,18),Parameter!(L%,19))
10230 Parameter!(L%,4) = Parameter!(L%,1)
10240 REM
10241 REM -----Sets Number of Samples-----
10242 REM
10250 Setsamples (Parameter!(L%,20))
10260 REM
10261 REM -----Sets Pulse Times-----
10262 REM
10270 Settime
(Parameter!(L%,13),Parameter!(L%,14),Parameter!(L%,15),Parameter!(L%,16),1.0E-05,1.0E-05,0)
10280 REM
10281 REM -----Sets Pulse Voltages-----
10282 REM
10290 Setpulse (Parameter!(L%,21),Parameter!(L%,22),Parameter!(L%,23),Parameter!(L%,24))
10300 REM
10301 REM******************************************************************************
10302 REM*
10304 REM* This section checks the parameters with the C-V meter *
10305 REM* and Pulse generator. *
10306 REM*
10307 REM******************************************************************************
10308 REM
10310 REM---Checks the Initial High Pulse Voltage with the Pulse Generator----
10311 REM
10320 DO 2 TIMES
10330 PARAMS$ = "SER.POLL/13":GOSUB 50000

136
10340 PARAM$ = "SDR/13":GOSUB 50000
10350 DATA_STRING$ = "M4,CT0,T1,W1,HIL" + STR$(Parameter!(L%,21)) + "V,LOL" +
   STR$(Parameter!(L%,24)) + "V"
10360 PARAM$ = "WR.STR/13/EOS":GOSUB 50000
10370 PARAM$ = "RD.STR/13/EOS":GOSUB 50000
10380 PARAM$ = "SER.POLL/13":GOSUB 50000
10390 REPEAT
10400 IF POLL_RESP% AND &H40 = 64 THEN GOSUB 10740
10410 IF Parameter!(L%,23) = 0 THEN GOTO 10520
10420 REM
10421 REM---Checks the Final High Pulse Voltage with the Pulse Generator----
10422 REM
10430 DO 2 TIMES
10440 PARAM$ = "SER.POLL/13":GOSUB 50000
10450 DATA_STRING$ = "M4,CT0,T1,W1,HIL" + STR$(Parameter!(L%,22)) + "V,LOL" +
   STR$(Parameter!(L%,24)) + "V"
10460 PARAM$ = "WR.STR/13/EOS":GOSUB 50000
10470 PARAM$ = "RD.STR/13/EOS":GOSUB 50000
10480 PARAM$ = "SER.POLL/13":GOSUB 50000
10490 REPEAT
10500 IF POLL_RESP% AND &H40 = 64 THEN GOSUB 10740
10510 PARAM$ = "SDL/13":GOSUB 50000
10520 REM
10521 REM ----Checks the Hold Times vs. Bias Voltage with the C-V meter-----
10522 REM
10530 M% = 14
10540 N% = 17
10550 DATA_STRING$ = Function$ + "CN13TR3LE2SA1PC" + STR$(Parameter!(L%,N%)) +
   ":PN" + STR$(Parameter!(L%,20)) + ":PH" + STR$(Parameter!(L%,M%)) + ":PT" +
   STR$(Parameter!(L%,13)) + ":SW1"
10560 PARAM$ = "WR.STR/17/EOS":GOSUB 50000
10565 DATA_STRING$ = "SW0"
10567 PARAM$ = "WR.STR/17/EOS":GOSUB 50000
10570 PARAM$ = "SER.POLL/17":GOSUB 50000
10580 IF POLL_RESP% AND &H40 = &H40 THEN GOSUB 10870
10590 N% = N% + 1
10600 IF Parameter!(L%,19) = 0 OR Parameter!(L%,18) = 0 THEN GOTO 10620
10610 IF N% = 18 THEN GOTO 10550
10620 IF Parameter!(L%,15) = 0 THEN GOTO 10660
10630 M% = M% + 1
10640 IF M% = 15 THEN GOTO 10540
10650 REM
10651 REM ------This checks to see if delaytime/holdtime is > 200------
10652 REM
10660 IF Parameter!(L%,14) / Parameter!(L%,13) > 200 THEN GOTO 10680
10670 IF Parameter!(L%,16) / Parameter!(L%,13) <= 200 THEN GOTO 10710
10680 COLOR 7,0:CLS:LOCATE 13,5
10690 PRINT "The delay time divided by the pulse times must be less than 200!"
10700 GOTO 10270
10710 PARAM$ = "SDC/13,17":GOSUB 50000
10720 RETURN
10738 STOP
10739 REM
10740 REM
10741 REM *
10742 REM * This section prints the error messages if any of the
10743 REM * parameters evoke an "illegal" call from the generator
10744 REM * or C-V meter.
10745 REM *
10746 REM
10748 REM -----Prints out error message for the Pulse Generator-----
10749 REM
10750 DATA_STRINGS$ = "IERR"
10760 PARAMS = "WR.STR/13//EOS":GOSUB 50000
10770 PARAMS = "RD.STR/13//EOS":GOSUB 50000
10780 Clearscreen (3)
10790 LOCATE 10,5:COLOR 7,0
10800 PRINT "The settings chosen are not viable. You must select new settings."
10810 LOCATE 11,5:PRINT "To aid you the error from the HP8112 A is "
10820 LOCATE 11,47:COLOR 5,0:PRINT MID$(DATA_STRINGS$2,18):COLOR 7,0
10830 LOCATE 12,5:PRINT "Please look in the manual on pages 3-21 to 3-23."
10840 PARAMS = "SDL/13":GOSUB 50000
10850 Timedelay(6)
10860 RETURN,10280
10869 STOP
10870 REM
10871 REM ------Prints out error message for the C-V meter------
10872 REM
10880 DATA_STRINGS$ = "ERR?"
10890 PARAMS = "WR.STR/17//EOS":GOSUB 50000
10900 PARAMS = "RD.STR/17//EOS":GOSUB 50000
10910 Startnoerror% = INSTR(DATA_STRINGS$,"ER00.0")
10920 IF Startnoerror% <> 0 THEN RETURN
10930 Clearscreen (3)
10940 LOCATE 10,5:COLOR 7,0
10950 PRINT "The settings chosen are not viable. You must select new settings."
10960 PRINT "To aid you the error number from the HP4280 A is ";
10970 COLOR 5,0:PRINT DATA_STRINGS$COLOR 7,0
10980 PRINT "Please look in the manual on pages 3-23 to 3-30."
10990 Timedelay(6)
11000 PARAMS = "SDC/17":GOSUB 50000
11010 RETURN,10220
11019 STOP
12000 REM
12001 REM
12002 REM *
12003 REM * This subroutine control the C-G-V experiment and allows *
12004 REM * the user to enter the parameter settings and checks for *
12005 REM * any possible errors.
12006 REM *
12007 REM
12008 REM
12010 CLS:LOCATE 1,20:COLOR 2,0
12020 PRINT "This sets the";
12030 COLOR 18,0:PRINT " C-G-V";
12040 COLOR 2,0:PRINT " Parameter Settings"
12050 Parameter$(3,1) = Parameter$(25,1)
12060 Parameter$(3,2) = Parameter$(25,2)
12070 Parameter$(3,3) = Parameter$(25,3)
12080 Parameter$(3,4) = Parameter$(25,4)
12090 IF Exp$(26) = 3 OR Exp$(26) = 1 THEN GOTO 12150
12100 REM
12101 REM ---------Sets the Temperature Parameters---------
12102 REM
12110 Exp$ = "C-G-V Experiment"
12120 Col% = 19
12130 Name$ = "Settings"
12140 SettempParam (Time%, Col%, Parameter$(3,1), Parameter$(3,2), Parameter$(3,3), Exp$, Name$)
12150 REM
12151 REM ---------Sets the Start, Stop and Delta Voltages---------
12152 REM
12160 Svolt (Parameter$(3,7), Parameter$(3,8), Parameter$(3,9), Parameter$(3,10), Parameter$(3,11))
12170 REM
12171 REM ---------Sets the Hold and Step Delay Time---------
12172 REM
12180 Settime (Parameter$(3,13), Parameter$(3,14), Parameter$(3,15), Parameter$(3,16), 0,1,0,1,1)
12200 REM
12201 REM ******************************************************************************
12202 REM *
12203 REM * This checks the parameters entered with the C-V meter. *
12204 REM *
12205 REM ******************************************************************************
12206 REM
12210 REM ---------This checks the first start, stop and delta voltages------
12211 REM
12220 DATA_STRINGS$ =
"FN11B2TR3PS"+STR$(Parameter$(3,7))+";PP"+STR$(Parameter$(3,8))+";PE"+STR$(Parameter$(3,9))+";PD"+STR$(Parameter$(3,13))+";PL"+STR$(Parameter$(3,14))
12230 PARAMS = "WR,STR/17//EOS";GOSUB 50000
12240 PARAMS = "SER.POLL/17";GOSUB 50000
12250 IF POLL_RESP% AND &H40 = 64 THEN GOSUB 12500
12260 REM
12261 REM ---------This checks the second start, stop and delta voltages------
12262 REM
12270 DATA_STRINGS$ =
"FN11B2TR3PS"+STR$(Parameter$(3,8))+";PP"+STR$(Parameter$(3,10))+";PE"+STR$(Parameter$(3,11))+";PD"+STR$(Parameter$(3,13))+";PL"+STR$(Parameter$(3,14))
12280 PARAMS = "WR,STR/17//EOS";GOSUB 50000
12290 PARAMS = "SER.POLL/17";GOSUB 50000
12300 IF POLL_RESP% AND &H40 = 64 THEN GOSUB 12500
12310 PARAMS = "SDC/17";GOSUB 50000
12320 RETURN
12499 STOP
12500 REM
12501 REM ****************************************************
12502 REM *
12503 REM * This subroutine prints the error found by the C-V *
12504 REM * meter, then returns the program to reenter the *
12505 REM * parameters.
12506 REM *
12507 REM ****************************************************
12508 REM
12510 DATA_STRING$ = "ERR?"
12520 PARAM$ = "WR.STR//17//EOS":GOSUB 50000
12530 PARAM$ = "RD.STR//17//EOS":GOSUB 50000
12540 Startnoerror% = INSTR(DATA_STRING$,"ERR00.0")
12550 IF Startnoerror% <> 0 THEN RETURN
12560 Clearscreen (3)
12570 LOCATE 10,5:COLOR 7,0
12580 PRINT "The settings chosen are not viable. You must select new settings."
12590 PRINT "To aid you the error number from the HP4280 A is ";
12600 COLOR 5,0:PRINT DATA_STRING$:COLOR 7,0
12610 PRINT "Please look in the manual on pages 3-23 to 3-30."
12620 Timedelay(6)
12630 PARAM$ = "SDC/17":GOSUB 50000
12640 RETURN,12150
12999 STOP
13000 REM
13001 REM ****************************************************
13002 REM *
13003 REM * This is the C-G subroutine. It controls the parameter *
13004 REM * settings and also the data collecting and storing. *
13005 REM *
13006 REM ****************************************************
13007 REM
13010 CLS:LOCATE 1,20:COLOR 2,0
13020 PRINT "This sets the";
13030 COLOR 18,0:PRINT " C-G";
13040 COLOR 2,0:PRINT " Experiment Parameters"
13050 Parameter!(4,1) = Parameter!(25,1)
13060 Parameter!(4,2) = Parameter!(25,2)
13070 Parameter!(4,3) = Parameter!(25,3)
13080 Parameter!(4,4) = Parameter!(25,4)
13090 IF Expt!(26) = 3 OR Expt!(26) = 1 THEN GOTO 13150
13100 REM
13101 REM --------Sets the Temperature Parameters--------
13102 REM
13110 Expt$ = "C-G Experiment" 'Sets title and names for the
13120 Col% = 19 'Procedure
13130 Name$ = "Settings"
13140 Settempparam
(Time%,Col%,Parameter!(4,1),Parameter!(4,2),Parameter!(4,3),Expt$,Name$)
13150 REM
13151 REM -----------------Sets the Bias Voltages-----------------
13152 REM
13156 Setbias (Parameter!(4,17),Parameter!(4,18),Parameter!(4,19))
13170 REM
13171 REM ***************************************************************
13172 REM *
13173 REM * This section checks to see if the parameters set are *
13174 REM * allowed by the C-V meter.                             *
13175 REM *
13176 REM ***************************************************************
13177 REM
13178 REM ------This checks the start voltage--------
13179 REM
13180 DATA_STRING$ = "FN1CN101B1RA1MS2SL2TR3PV" + STR$(Parameter!(4,17))
13190 PARAMS = "WR.STR/17//EOS":GOSUB 50000
13200 PARAMS = "SER.POLL/17":GOSUB 50000
13210 IF POLL_RESP% AND &H40 = 64 THEN GOSUB 13500
13220 REM
13221 REM ------This checks the final voltage--------
13222 REM
13230 DATA_STRING$ = "FN1CN101B1RA1MS2SL2TR3PV" + STR$(Parameter!(4,18))
13240 PARAMS = "WR.STR/17//EOS":GOSUB 50000
13250 PARAMS = "SER.POLL/17":GOSUB 50000
13260 IF POLL_RESP% AND &H40 = 64 THEN GOSUB 13500
13270 PARAMS = "SDC/17":GOSUB 50000
13280 RETURN
13299 STOP
13300 REM
13301 REM ***************************************************************
13302 REM *
13303 REM * This subroutine prints the error found by the C-V *
13304 REM * meter, then returns the program to reenter the *
13305 REM * parameters.                                          *
13306 REM *
13307 REM ***************************************************************
13308 REM
13309 DATA_STRING$ = "ERR?"
13310 PARAMS = "WR.STR/17//EOS":GOSUB 50000
13320 PARAMS = "RD.STR/17//EOS":GOSUB 50000
13340 Startnoerror% = INSTR(DATA_STRING$,"ER00.0")
13350 IF Startnoerror% <= 0 THEN RETURN
13356 Clearscreen (3)
13357 LOCATE 10,5:COLOR 7,0
13380 PRINT "The settings chosen are not viable. You must select new settings."
13390 PRINT "To aid you the error number from the HP4280 A is ;"
13399 PRINT DATA_STRING$COLOR 7,0
13400 COLOR 5,0:PRINT DATA_STRING$:COLOR 7,0
13410 PRINT "Please look in the manual on pages 3-23 to 3-30."
13420 Timedelay(6)
13430 PARAMS = "SDC/17":GOSUB 50000
13440 RETURN,13150
13499 STOP
14000 REM
14001 REM
14002 REM *
14003 REM * This is the Current vs. Voltage subroutine. It sets *
14004 REM * up the parameters necessary to run the program. It *
14005 REM * It also checks to see if the parameters are feasible. *
14006 REM *
14007 REM
14008 REM
14010 CLS:LOCATE 1,20:COLOR 2,0
14020 PRINT "This sets the ";
14030 COLOR 18,0:PRINT "I-V";
14040 COLOR 2,0:PRINT " Experiment Parameters"
14050 Parameter!(1,1) = Parameter!(25,1)
14060 Parameter!(1,2) = Parameter!(25,2)
14070 Parameter!(1,3) = Parameter!(25,3)
14080 Parameter!(1,4) = Parameter!(25,4)
14090 IF Expt%(26) = 3 OR Expt%(26) = 1 THEN GOTO 14150
14100 REM
14101 REM -------Sets the Temperature Parameters-------
14102 REM
14110 Expt$ = "I-V Experiment"
14120 Col% = 20
14130 Name$ = "Settings"
14140 Settempparam
(Time%,Col%,Parameter!(1,1),Parameter!(1,2),Parameter!(1,3),Expt$,Name$)
14150 REM
14151 REM -------Sets the Start, Stop and Delta Voltages---------
14152 REM
14160 Svolt (Parameter!(1,7),Parameter!(1,8),Parameter!(1,9),Parameter!(1,10),Parameter!(1,11))
14170 REM
14171 REM -------Sets the Hold and Step Delay Time-------
14172 REM
14180 Settime (Parameter!(1,13),Parameter!(1,14),Parameter!(1,15),Parameter!(1,16),0.7,0.1,1)
14190 REM
14191 REM ***********************************************
14192 REM *
14193 REM * This checks the parameters set with the I-V meter. *
14194 REM *
14195 REM ***********************************************
14196 REM
14200 M% = 7
14210 N% = 8
14215 XS = STR$(VAL(STR$(Parameter!(1,N%+1))))
14220 DATA_STRING$ = "F212L3PS" + STR$(Parameter!(1,M%)) + ";PT" + STR$(Parameter!(1,1,1)) + ";PE" + XS + ";PH" + STR$(Parameter!(1,14)) + ";PD" + STR$(Parameter!(1,13)) + ";W1"
14230 PARAM$ = "WR.STR/5/EOS":"GOSUB 50000
14240 DATA_STRING$ = "W7"
14250 PARAM$ = "WR.STR/5/EOS":"GOSUB 50000
14260 PARAM$ = "SER.POLL/5":"GOSUB 50000
142
14270 IF POLL_RESP% AND &H40 = 64 THEN GOSUB 14500
14280 IF Parameter!(1,11) = 0 THEN GOTO 14320
14290 M% = M% + 1
14300 N% = N% + 2
14310 IF M% = 8 THEN GOTO 14215
14320 PARAM$ = "SDC/5":GOSUB 50000
14330 RETURN
14499 STOP
14500 REM
14501 REM ***********************************************************************
14502 REM *
14503 REM * This prints the error message sent in binary by the       *
14504 REM * I-V meter. It then sends the program back to reenter    *
14505 REM * the parameters.                                        *
14506 REM *
14507 REM ***********************************************************************
14508 REM
14510 Clearscreen (3)
14520 COLOR 7,0:LOCATE 10,5
14530 PRINT "The settings chosen are not viable. You must select new settings."
14540 LOCATE 11,5
14550 PRINT "If you are not sure why this occurred please check the HP 4140B."
14560 LOCATE 12,5:PRINT "The BINARY error code is ",
14570 COLOR 5,0:PRINT MID$(BIN$(POLL_RESP%),8)
14580 TimeDelay(6)
14590 PARAM$ = "SDC/5":GOSUB 50000
14600 RETURN,14160
14999 STOP
15000 REM
15001 REM ***********************************************************************
15002 REM *
15003 REM * This is the Capacitance vs. Voltage subroutine. It      *
15004 REM * sets up the parameters necessary to run the program.     *
15005 REM * It also checks to see if the parameters are feasible.     *
15006 REM *
15007 REM ***********************************************************************
15008 REM
15010 CLS:LOCATE 1,20:COLOR 2,0
15020 PRINT "This sets the";
15030 COLOR 18,0:PRINT " C-V ";
15040 COLOR 2,0:PRINT "Experiment Parameters"
15050 Parameter!(2,1) = Parameter!(25,1)
15060 Parameter!(2,2) = Parameter!(25,2)
15070 Parameter!(2,3) = Parameter!(25,3)
15080 Parameter!(2,4) = Parameter!(25,4)
15090 IF Expt%(26) = 3 OR Expt%(26) = 1 THEN GOTO 15160
15100 REM
15101 REM ..........Sets the Temperature Parameters.........
15102 REM
15110 Expt$ = "C-V Experiment"
15120 Col% = 20
15130 Name$ = "Settings"
15140 Seticempparam
(Time%,Col%,Parameter!(2,1),Parameter!(2,2),Parameter!(2,3),Expt$,Name$)
15150 REM
15151 REM ------Sets the Start, Stop and Delta Voltages------
15152 REM
15160 Dvdt (Parameter!(2,12),Parameter!(2,7),Parameter!(2,8),Parameter!(2,9))
15170 REM
15171 REM ------Sets the Hold Time ----
15172 REM
15180 Settime (Parameter!(2,13),Parameter!(2,14),Parameter!(2,15),Parameter!(2,16),0,1,0,3)
15190 REM
15191 REM ***********************************************
15192 REM *
15193 REM * This checks the parameters set with the I-V meter. *
15194 REM *
15195 REM ***********************************************
15196 REM
15200 DATA_STRING$ = "F312L3PS" + STR$(Parameter!(2,7)-Parameter!(2,9)) + ";PT" +
STR$(Parameter!(2,8)+Parameter!(2,9)) + ";PE" + STR$(Parameter!(2,9)) + ";PV" +
STR$(Parameter!(2,12)) + ";PH" + STR$(Parameter!(2,14)) + ";W1"
15210 PARAM$ = "WR.STR/5/EOS":GOSUB 50000
15215 DATA_STRING$ = "W7"
15218 PARAM$ = "WR.STR/5/EOS":GOSUB 50000
15220 PARAM$ = "SER.POLL/5":GOSUB 50000
15230 IF POLL_RESP% AND &H40 = 64 THEN GOSUB 15500
15240 PARAM$ = "SDC/5":GOSUB 50000
15250 RETURN
15250 REM
15251 REM ***********************************************
15252 REM *
15253 REM * This prints the error message sent in binary by the *
15254 REM * I-V meter. It then sends the program back to reenter *
15255 REM * the parameters. *
15256 REM *
15257 REM ***********************************************
15258 REM
15259 Rem Clearscren (3)
15260 COLOR 7,0:LOCATE 10,5
15260 PRINT "The settings chosen are not viable. You must select new settings."
15260 LOCATE 11,5
15260 PRINT "If you are not sure why this occurred please check the HP 4140B."
15260 LOCATE 12,5:PRINT "The BINARY error code is ";
15260 LOCATE 5,0:PRINT MIDS(BINS(POLL_RESP%),8)
15260 Timedelay(6)
15260 PARAM$ = "SDC/5":GOSUB 50000
15260 RETURN,15150
39999 STOP
40000 REM
40001 REM ***********************************************
40002 REM *
40003 REM * This subroutine places the data pointer to the right *
40004 REM * experimental group, for the print out. *
40005 REM *
40006 REM ****************************************************************************
40007 REM
40010 ON Expt%(25) GOTO 40020,40030,40040,40045
40020 RESTORE,60510:GOTO 40050
40030 RESTORE,60520:GOTO 40050
40040 RESTORE,60530:GOTO 40050
40045 RESTORE,60540:GOTO 40050
40050 RETURN
40059 STOP
41000 REM
41001 REM ****************************************************************************
41002 REM *
41003 REM * This subroutine places the data pointer to the right *
41004 REM * experimental group, for the experiment menu. *
41005 REM *
41006 REM ****************************************************************************
41007 REM
41010 ON Expt%(25) GOTO 41020,41030,41040,41050
41020 RESTORE,60220:GOTO 41060
41030 RESTORE,60230:GOTO 41060
41040 RESTORE,60240:GOTO 41060
41050 RESTORE,60250:GOTO 41060
41060 RETURN
59999 STOP
60000 REM
60001 REM ****************************************************************************
60002 REM *
60003 REM * These DATA statements are used for the information *
60004 REM * block print statements and correction routine. *
60005 REM *
60006 REM ****************************************************************************
60007 REM
60010 DATA 6
60020 DATA "Directory Number","Date ","Experimenters Name","Sample Number","Cryostat Chronometer Reading","Comments","Type of Run"
60099 STOP
60100 REM
60101 REM ****************************************************************************
60102 REM *
60103 REM * These DATA statements allow the user to choose which *
60104 REM * group of experiments to run. *
60105 REM *
60106 REM ****************************************************************************
60110 DATA 4
60120 DATA "GROUP I  C-t, G-t, C-G-V, C-G"
60130 DATA "GROUP II  I-V, C-V"
60140 DATA "GROUP III  Van der Pauw/Mobility"
60150 DATA "GROUP IV  4-point Resistivity"
60199 STOP
60200 REM
60201 REM  
60202 REM  
60203 REM  These DATA statements allow the user to choose which  
60204 REM  experiments he wants to run from the group chosen  
60205 REM  earlier.  
60206 REM  
60207 REM  
60208 REM  
60210 DATA 4,4,2,1,1
60220 DATA "Capacitance (C) vs. Time","Conductance (G) vs. Time","C and G vs. Voltage","Capacitance and Conductance"
60230 DATA "Current vs. Voltage","Capacitance vs. Voltage"
60240 DATA "Van der Pauw/Mobility"
60250 DATA "4-point Resistivity"
60299 STOP
60300 REM
60301 REM  
60302 REM  
60303 REM  These DATA statements are for the PARAMETER (x,y)  
60304 REM  array.  
60305 REM  
60306 REM  
60307 REM  
60310 DATA 24
60320 DATA "Initial Temp","Final Temp","Delta Temp","Current Temp"
60330 DATA "Final Time","Delta Time"
60340 DATA "Start Volt","Stop1 Volt","Step1 Volt","Stop2 Volt","Step2 Volt"
60350 DATA "DV/dt"
60360 DATA "Delay Time","Initial Hold","Final Hold","Delta Hold"
60370 DATA "Initial Bias","Final Bias","Delta Bias"
60380 DATA "Number of Samples"
60390 DATA "Start High Pulse","Final High Pulse","Delta High Pulse","Low Pulse"
60399 STOP
60400 REM
60401 REM  
60402 REM  
60403 REM  This data statement is used for the type of temperature  
60404 REM  run section.  
60405 REM  
60406 REM  
60407 REM  
60410 DATA 3
60420 DATA "Time Run ( > 5 Minutes; 30-600 K)","Temperature Run (30 - 600 K)","Room Temperature Run (290 K)"
60499 STOP
60500 REM
60501 REM  
60502 REM  
60503 REM  These data statements are for the print out of the  

60504 REM * parameter set.
60505 REM *
60506 REM *************************************************
60507 REM
60510 DATA "C-I","G-I","C-G-V","C-G"
60520 DATA "I-V","C-V"
60530 DATA "Van der Pauw"
60540 DATA "4-point Resistivity"
60599 STOP
60600 REM
60601 REM *************************************************
60602 REM *
60603 REM * This data statement gives what type of temperature run *
60604 REM * it is.
60605 REM *
60606 REM *************************************************
60607 REM
60610 DATA "TTEMP","LTEMP","RTEMP"

ENDFILE
RUNIT
PRECISION= 7
AUTODEF=ON
OPTION BASE=0
ERL=ON
ERRORMODE=LOCAL
RESUME=LINE
FORMODE=BB
SCOPE=ON
PROC=2
STRING ARRAY(?) : RECS,FLDS
STRING : PARAMS[?],IEEE_FCTNS[?]
INTEGER : CNTRLR%,TRUE%,COMM%,PORT4%,PORT5%,PORT9%
INTEGER : PORT0%,FUM%,FALSE%,MY_FLAG%,MAX_TIME%
REAL : STROFF
INTEGER : STROFF%
REAL : DSFTR
INTEGER : MY_ADDR%,BD_ADDR%,PORT1%,PORT2%
INTEGER : PORT3%,PORT6%,PORT7%,PORT8%
STRING : CHAR1S[?]
INTEGER : TCIMODE%,MS%,INTR%,INVECTOR%,INTENABLE%
INTEGER : INTMASK%,SUBLIB%,TIMEOUT%,INTSETUP%,POLLBYTE%,WRSTR%,WRFIL%,CB_FL
AG%
INTEGER : HCSR%,VCSR%,RDSTR%,RDFIL%
REAL : CSEG
INTEGER : SAVESTAT%,SYC%
STRING : DUMMYS[?]
INTEGER : NEOS%,TERM%,N1%,LAST_INT1%
INTEGER : INTSTAT%,CTR%
STRING : DATA_STRINGS[?]
INTEGER : CHAR%,STRLEN%,CB%,DAT%,LAST%,POLL_RESP%,STATUS%
INTEGER : SRQ%,SYS%,BIT%,SENSE%,X%,N2%,Y%,ADSTAT%
STRING : END_SEOS[?],LAST_CHARS[?],SEPS[?]
STRING ARRAY(?) : SARS
STRING : TEMPS[?]
INTEGER : M%,B%,N%,L%,Maxloop%,NCM%
INTEGER : NGM%,V%,T%,Cryostat%
INTEGER ARRAY(31) : Exp1%
INTEGER ARRAY(11) : Group%
STRING : FileS[5],Chks[3],NameS[60],FunctionS[?],MonikerS[13]
STRING : SensorS[4],Temp_wantedS[5],XtempS[16],HeaterS[4],Int_GainS[12],AS[3],Sensor1$[4]
REAL ARRAY(26,31) : Parameter!
REAL : Delta!,Time
REAL : Timerun!,Pulsevolt!,Bias!,Maxtemp!,Mintemp!,Temp!,O,Cdata!
REAL : Xdata!,Gdata!,Idata!,Delta_temp!,Temp_wanted!
INTEGER : Test_Passed%,Crostat%
REAL : Time!,L!,CURRENT!
REAL : VOL!,VOLT2!,Test_passed!,Offset!
STRING : XS[?]
REAL : TEMP_SET!,TEMP_PEAK!,Temp1!,Temp2!,TIFF!
REAL : Temp3!
INTEGER: Print_temp%,row%,p%
REAL: ACTUAL_TEMP,N

PROCEDURE: TIMEDELAY()
REAL: ARG: DELAY%/OPT=5!
END PROCEDURE

PROCEDURE: CLEARSCREEN()
INTEGER ARG: ROW%
END PROCEDURE

PROCEDURE: TIMEDELAY
REAL: Tyme!
STRING: OVERHEATS[%],Data_string$[?]
REAL: TEMP_PEAK%,PEAK_TEMP!
  22 REM * integer argvdelay%/opt = 5
  32 REM * real:tyme!
  100 Tyme! = 'TIMER
 110 IF TIMER < Tyme! + DELAY! THEN GOTO 110
 120 EXIT
END PROCEDURE

PROCEDURE: CLEARSCREEN
INTEGER: N%
  100 FOR N% = ROW% TO 24
  110 SET CURSOR N%,0
  120 PRINT SPC(80)
  130 NEXT N%
END PROCEDURE

'MAIN Program:

200 REM
201 REM **********************************************
202 REM *
203 REM * This section prints the entrance message to RUNIT. It *
204 REM * also initializes the devices on the IEEE bus. *
205 REM *
206 REM **********************************************
207 REM
210 CLEAR
220 ON ERROR GOSUB 42000
230 CLS: STATUSLINE OFF: SCREEN 0,0,0: COLOR 2,0: LOCATE 12,23
240 PRINT "Hit any key to begin Experiments"
250 COLOR 3,0:LOCATE 14,10
260 PRINT "All instruments must be on and the proper connections be made"
270 TIMEDELAY (2)
280 AS = INKEY$: IF AS = "" THEN GOTO 250
290 PARAMS = "INIT/1/&H310/?":GOSUB 50000
300 PARAMS = "SDR/5,7,8,12,13,16,17":GOSUB 50000
310 REM
311 REM *-------------------------------------------------------------------------------------*
312 REM *
313 REM * This section loads the data from the files stored by *
314 REM * the program PARAMETER. *
315 REM *
316 REM *-------------------------------------------------------------------------------------*
317 REM
320 CLS:COLOR 2,0,0:LOCATE 12,30
330 PRINT "Now LOADING data"
340 DRIVES = "C:"
350 DIRS = ""*
360 OPEN ":\DATA\EXPT" FOR INPUT AS #1
370 INPUT #1,FileS
380 FOR M% = 1 TO 30
390 INPUT #1,Expt%(M%)  
400 NEXT M%
410 CLOSE #1
420 OPEN ":\DATA\PARAM" FOR INPUT AS #1
430 FOR M% = 1 TO 6
440 INPUT #1,Parameter!(25,M%)
450 NEXT M%
460 RESTORE,60010
470 READ Maxloop%
480 FOR M% = 1 TO Maxloop%
490 READ Group%(M%)
500 NEXT M%
505 RESTORE,60410
507 READ Maxloop%
510 FOR N% = 1 TO Group%(Expt%(25))
520 FOR M% = 1 TO Maxloop%
530 IF Expt%(N%) = 0 THEN EXIT 1 LEVELS
540 INPUT #1,Parameter!(N%,M%)
550 NEXT M%
560 NEXT N%
570 CLOSE #1
575 ACTUAL TEMP=Parameter!(25,4)
600 REM
601 REM *-------------------------------------------------------------------------------------*
602 REM *
603 REM * This section finds which type of temperature run was *
604 REM * requested, then stores in in the string X TEMPS$ *
605 REM *
606 REM *-------------------------------------------------------------------------------------*
607 REM
610 RESTORE,60110
620 FOR M% = 1 TO 30
630 READ Xtemp$
640 IF M% = Expt%(26) THEN EXIT
650 NEXT M%
700 REM
704 P% = 0 : Print_temp% = 1
705 RESTORE,62000
710 DO
720 IF Expj%(25) <> 3 THEN EXIT TO,1000
730 CLS : COLOR 4,0,0 : SET CURSOR 0,35
740 PRINT "MOBILITY EXPERIMENT"
750 END DO
765 COLOR 3,0,0 : SET CURSOR 7,6 : PRINT "Connect the cables as follows"
766 COLOR 2,0,0
770 FOR N% = 1 TO 5
780 READ Name$
790 SET CURSOR 7 + N%*2,10
810 PRINT Name$
820 NEXT N%
830 COLOR 7,0,0 : SET CURSOR 22,6
840 PRINT "Hit any key to continue"
850 AS = INKEYS : IF AS = "" THEN GOTO 850
860 P% = P% + 1
870 GOSUB 15000
871 IF P% < 3 THEN GOTO 710
872 CLEARSCREEN(2)
873 COLOR 2,0,0 : SET CURSOR 13,6
874 PRINT "Hit any key to begin the experiment run"
875 AS = INKEYS : IF AS = "" THEN GOTO 875
880 Print_temp% = 2:GOTO 1000
1000 REM
1001 REM ******************************************************************************************************************
1002 REM *
1003 REM * This section print up the proper screen for each type of temperature run. Prints the time if a time run, *
1004 REM * and goes to the proper subroutine for the experiment chosen. It also exits to finish routines if either *
1005 REM * the final temperature or final time has been reached. *
1006 REM *
1007 REM ******************************************************************************************************************
1008 REM
1009 REM
1010 REM
1020 Mintemp! = Parameter!(25,1)
1030 Maxtemp! = Parameter!(25,2)
1040 COLOR 5,0:CLS
1050 IF Expj%(26) = 3 THEN PRINT SPC(60);"Room Temperature"
1060 COLOR 4,0:IF Expj%(26) = 3 THEN PRINT SPC(63);Parameter!(25,4);"K"
1070 COLOR 3,0:LOCATE 12,28:PRINT "The Correct Program is:"
1080 IF Expj%(26) = 3 THEN GOTO 1100
1090 GOSUB 9000
1100 IF Expj%(26) <> 1 THEN GOTO 1150
1110 Timerun! = 0,0
1120 TIMES$ = "00:00:00"
1130 LOCATE 1,8:COLOR 5,0:PRINT "Time"
1140 LOCATE 2,6:COLOR 4,0:PRINT TIMES$
1160 ON Expt%(25) GOSUB 2000,2500,3010,4010
1170 IF Expt%(26) = 1 THEN GOTO 1220
1180 Parameter!(25,4) = Parameter!(25,4) + Parameter!(25,3)
1190 IF Parameter!(25,4) > Maxtemp! THEN GOTO 40000
1210 GOTO 1040
1220 IF TIMER > Parameter!(25,5) THEN GOTO 40000
1230 Timerun! = Timerun! + Parameter!(25,6)
1240 TIMEDELAY (.5)
1250 LOCATE 14,35:PRINT "TIME SET"
1260 LOCATE 2,6:COLOR 4,6:PRINT TIMES
1270 IF TIMER <= Timerun! THEN GOTO 1250
1280 GOTO 1150
1999 STOP
2000 REM
2001 REM *********************************************
2002 REM *
2003 REM * This subroutine branches to the proper experiments *
2004 REM * chosen in group I. It also checks after each *
2005 REM * experiment to see if the COLD HEAD has overheated *
2006 REM * (except in the case of a Room Temperature run). *
2007 REM *
2008 REM *********************************************
2009 REM
2010 L% = 1
2020 ON Expt%(L%) GOSUB 10000,10000,11000,12000
2030 LOCATE 14,1:PRINT SPC(79)
2040 L% = L% + 1
2050 IF Expt%(26) <= 3 THEN GOSUB 9500
2060 IF L% <= Group%(Expt%(25)) THEN GOTO 2020
2070 RETURN
2499 STOP
2500 REM
2501 REM *********************************************
2502 REM *
2503 REM * This subroutine branches to the proper experiments *
2504 REM * chosen in group II. It also checks after each *
2505 REM * experiment to see if the COLD HEAD has overheated *
2506 REM * (except in the case of a Room Temperature run). *
2507 REM *
2508 REM *********************************************
2509 REM
2510 L% = 1
2520 ON Expt%(L%) GOSUB 13000,14000
2530 LOCATE 14,1:PRINT SPC(79)
2540 IF Expt%(26) <= 3 THEN GOSUB 9500
2550 L% = L% + 1
2560 IF L% <= Group%(Expt%(25)) THEN GOTO 2520
2570 RETURN
3000 REM
3001 REM *********************************************
3002 REM *
3003 REM * This subroutine branches to the proper experiments *
3004 REM * chosen in group III. It also checks after each *
3005 REM * experiment to see if the COLD HEAD has overheated *
3006 REM * (except in the case of a Room Temperature run). *
3007 REM *
3008 REM ***********************************************
3009 REM
3010 GOSUB 15000
3020 IF Expt%(26) <> 3 THEN GOSUB 9500
3030 RETURN
4000 REM
4001 REM ***********************************************
4002 REM *
4003 REM * This subroutine branches to the proper experiments *
4004 REM * chosen in group IV. It also checks after each *
4005 REM * experiment to see if the COLD HEAD has overheated *
4006 REM * (except in the case of a Room Temperature run). *
4007 REM *
4008 REM ***********************************************
4009 REM
4010 GOSUB 16000
4020 IF Expt%(26) <> 3 THEN GOSUB 9500
4030 RETURN
8999 STOP
9000 REM
9001 REM ***********************************************
9002 REM *
9003 REM * This sets the Cryostat to the correct temperature and *
9004 REM * then lets the temperature to stabilize to +/- 1 degree *
9005 REM *
9006 REM ***********************************************
9007 REM
9008 REM
9010 COLOR 2,0:LOCATE 14,24
9020 PRINT *Setting the Cryostat Temperature*
9030 LOCATE 1,30:PRINT SPC(49)
9040 COLOR 5,0:LOCATE 1,60:PRINT "Sample Setpoint"
9050 COLOR 4,0:LOCATE 2,63:PRINT Parameter!(25,4);"K"
9060 Test_Passed% = 0:Cryostat% = 0
9065 IF Parameter!(25,2) > 290 THEN Offset$ = 0 ELSE Offset$ = 0
9070 Temp_wanted! = Parameter!(25,4) + Offset!
9075 IF Temp_wanted! => 290 THEN LOCATE 22,10:COLOR 30,0:PRINT "Warning:"
9080 IF Temp_wanted! < 15 THEN Heater$ = "7"
9090 IF Temp_wanted! => 15 AND Temp_wanted! < 20 THEN Heater$ = "8" ELSE Heater$ = "9"
9100 IF Temp_wanted! > 290 THEN Int_Gain$="00100000400" ELSE Int_Gain$="00100000500"
9101 IF Temp_wanted! <= 100 THEN Int_Gain$="00150000600"
9102 IF Temp_wanted! < 50 THEN Int_Gain$="00200000700"
9110 Delta_temp! = 1
9120 Sensor$ = "33"
9130 DATAgetString$ = "P"
PARAMS = "WR.STR/7//EOS/":GOSUB 50000
Temp_wanted$ = MIDS(STR$(Temp_wanted$ + 1000),3)
DATA.Strings$ = Temp_wanted$ + Int_Gain$ + Sensor$ + Heater$ + ";505"
PARAMS = "WR.STR/7//EOS/":GOSUB 50000
PARAMS = "RD.STR//EOS/":GOSUB 50000
PARAMS = "RD.STR//EOS/":GOSUB 50000
TIMEDELAY(2)
DO 2 TIMES
PARAMS = "RD.STR//EOS/":GOSUB 50000
Temp! = VAL(MIDS(DATA.Strings$,2))
IF Temp! => Temp_wanted! - Delta_temp! AND Temp! <= Temp_wanted! + Delta_temp! THEN EXIT TO,9300
TIMEDELAY (1)
REPEAT
Cryostat% = 0
Test_Passed% = 0
IF Temp!< Temp_wanted! -2 OR Temp! > Temp_wanted! +2 THEN GOSUB 9500
GOTO 9130
STOP
Test_Passed% = Test_Passed% + 1
IF Test_Passed% > 1 THEN GOTO 9400
TIMEDELAY(35)
GOTO 9130
LOCATE 1,3;PRINT SPC(79)
COLOR 5,0;LOCATE 1,5;PRINT "Sample Temperature"
COLOR 4,0;LOCATE 2,6;PRINT Parameter!(25,4)
ACTUAL_TEMP=Temp!
COLOR 2,0
Cryostat% = 1
RETURN
STOP
REM
REM ***************************************************************
REM *
REM * This subroutine checks the Cold Head to make sure its "
REM * not overheating. *
REM *
REM ***************************************************************
REM
REM
IF Cryostat% = 0 THEN GOTO 9540
LOCATE 14,1;PRINT SPC(79)
COLOR 2,0;LOCATE 14,30;PRINT "Checking Cold Finger"
DATA.Strings$ = "P"
PARAMS = "WR.STR/7//EOS/":GOSUB 50000
Sensor$ = "31"
DATA.Strings$ = Temp_wanted$ + Int_Gain$ + Sensor$ + Heater$ + "505"
PARAMS = "WR.STR/7//EOS/":GOSUB 50000
DATA.Strings$ = "R"
PARAMS = "WR.STR/7//EOS/":GOSUB 50000
TIMEDELAY(3)
DO 5 TIMES
9610  PARAMS = "RD.STR7//EOS":GOSUB 50000
9620  Temp! = VAL(MIDS(DATA_STRING$,2))
9630  IF Temp! > 325 AND Temp! < 900 THEN GOTO 9700
9640  REPEAT
9650  IF Cryostat% = 0 THEN GOTO 9670
9660  LOCATE 14,1:PRINT SPC(79):LOCATE 14,1
9670  RETURN
9699  STOP
9700  REM
9701  REM  ......................................................................................
9702  REM *
9703  REM *   This subroutine is used if the Cold Finger overheats. *
9704  REM *   It writes death parameters to the Cryostat and gives *
9705  REM *   a warning message. *
9706  REM *
9707  REM  ......................................................................................
9708  REM
9710  DATA_STRING$ = "P"
9720  PARAMS = "WR.STR7//EOS":GOSUB 50000
9730  DATA_STRING$ = "0021000000001117501"
9740  PARAMS = "WR.STR7//EOS":GOSUB 50000
9750  DATA_STRING$ = "R"
9760  PARAMS = "WR.STR7//EOS":GOSUB 50000
9770  PARAMS = "RD.STR7//EOS":GOSUB 50000
9780  CLS:COLOR 2,0:LOCATE 11,21
9790  PRINT "The ";
9800  COLOR 19,0:PRINT "COLD FINGER";
9810  COLOR 2,0
9820  PRINT " has ";
9830  COLOR 20,0:PRINT "Overheated"
9840  LOCATE 14,16:COLOR 7,0
9850  PRINT "PLEASE CHECK FOR DAMAGE AND CONTACT ERIC COLE"
9860  LOCATE 16,12:COLOR 1,0:PRINT "ALSO TURN THE HEATER SWITCH ABOVE THE CONTROLLER OFF"
9870  CLEAR
9880  GOTO 9880
9999  STOP
10000  REM
10001  REM  ......................................................................................
10002  REM *
10003  REM *   This subroutine run the experiment [either C-t (1) or *
10004  REM *   G-t (2)] specified by 1%. It stores the data in the *
10005  REM *   file C:\DATA\#\#\?-t where ? is C or G respectively *
10006  REM *
10007  REM  ......................................................................................
10008  REM
10010  IF Expt%(26) = 1 OR Expt%(26) = 3 THEN GOTO 10030
10020  IF Parameter$(L%,4) <> Parameter$(25,4) GOTO 10500
10030  IF L% = 1 THEN GOTO 10080
10040  Function$ = "FN6"
10050  Name$ = "\G-t"
10060 Moniker$ = "CONDUCTANCE"
10070 GOTO 10110
10080 Function$ = "FN5"
10090 Name$ = "C-t"
10100 Moniker$ = "CAPACITANCE"
10110 OPEN "C:\DATA! + File$ + Name$ FOR APPEND AS #1
10120 LOCATE 14,20:PRINT "Taking "Moniker$;" vs. TIME Measurements"
10130 Pulsevolt! = Parameter!(L%,21)
10140 DATA_STRINGS$ = "M4CTU1T1W1HIL" + STR$(Pulsevolt!) + "VLOL" +
STR$(Parameter!(L%,24)) + "VD0"
10150 PARAMS = "W0.STR/13/EOS":GOSUB 50000
10160 PARAMS = "RD.STR/13/EOS":GOSUB 50000
10170 Bias! = Parameter!(L%,17)
10180 Time! = Parameter!(L%,14)
10190 IF Time! <= 0.01 THEN GOSUB 10580:GOTO 10220
10200 IF Time! < 10.1 * Parameter!(L%,13) AND Time! < .1 THEN GOSUB 10580:GOTO 10220
10210 GOSUB 10650
10220 N% = 1
10230 PRINT #1,Xtemp$:Parameter!(25,4);"ACTUALTEMP"
10240 PRINT #1,"TIME = ",Time!:" (;TIME;"")
10250 PRINT #1,"BIAS = ",Bias!
10260 PRINT #1,"HIGH PULSE = ",Pulsevolt!
10270 PRINT #1,"HOLD TYME = ",Time!
10280 PARAMS = "RD.STR/17/EOS":GOSUB 50000
10290 NCM% = INSTR(DATA_STRINGS$,"M") + 1
10300 Cdata! = VAL(MIDS(DATA_STRINGS$,NCM%))
10310 T% = INSTR(DATA_STRINGS$,"T") + 1
10320 Xdata! = VAL(MIDS(DATA_STRINGS$,T%))
10330 PRINT #1,Cdata!:";Xdata!
10340 N% = N% + 1
10350 IF N% <= Parameter!(L%,20) THEN GOTO 10280
10360 PRINT #1,"END"
10370 IF Parameter!(L%,15) = 0 OR Parameter!(L%,16) = 0 THEN GOTO 10400
10380 Time! = Time! + Parameter!(L%,16)
10390 IF Time! <= Parameter!(L%,15) THEN GOTO 10190
10400 IF Parameter!(L%,18) = 0 OR Parameter!(L%,19) = 0 THEN GOTO 10430
10410 Bias! = Bias! + Parameter!(L%,19)
10420 IF Bias! <= Parameter!(L%,18) THEN GOTO 10180
10430 IF Parameter!(L%,23) = 0 THEN GOTO 10460
10440 Pulsevolt! = Pulsevolt! + Parameter!(L%,23)
10450 IF Pulsevolt! <= Parameter!(L%,22) THEN GOTO 10140
10460 Parameter!(L%,4) = Parameter!(L%,4) + Parameter!(L%,3)
10470 CLOSE #1
10480 DATA_STRINGS$ = "BL0"
10490 PARAMS = "W0.STR/17/EOS":GOSUB 50000
10500 REM
10540 IF Parameter!(L%,4) < Parameter!(25,4) THEN Parameter!(L%,4) = Parameter!(L%,3) +
Parameter!(L%,4)
10550 IF Parameter!(L%,4) > Parameter!(L%,2) THEN Parameter!(L%,4) = 600
10560 PARAMS = "SDC/13,17":GOSUB 50000
157
10570 RETURN
10579 STOP
10580 REM
10581 REM *******************************************************************
10582 REM *
10583 REM * This subroutine is used if the step delay time and *
10584 REM * hold times do require the meter to be put into *
10585 REM * block mode.
10586 REM *
10587 REM *******************************************************************
10588 REM
10589 DATA STRING$ = Function$ + "CN13TR3LE2SA1PC" + STR$(Bias!) + ";PN" + STR$(Parameter!(L%.20)) + ";PH" + STR$(Time!) + ";PT" + STR$(Parameter!(L%.13)) + ";BL1;SW1"
10590 PARAMS = "WR.STR/17//EOS":GOSUB 50000
10591 TIMEDELAY(10)
10592 DATA STRING$ = "BD"
10593 PARAMS = "WR.STR/17//EOS":GOSUB 50000
10594 RETURN
10595 STOP
10596 REM
10597 REM *******************************************************************
10598 REM *
10599 REM * This subroutine is used if the step delay time and *
10600 REM * hold times do not require the meter to be put into *
10601 REM * block mode.
10602 REM *
10603 REM *
10604 REM *******************************************************************
10605 REM
10606 DATA STRING$ = Function$ + "CN13TR3LE2SA1PC" + STR$(Bias!) + ";PN" + STR$(Parameter!(L%.20)) + ";PH" + STR$(Time!) + ";PT" + STR$(Parameter!(L%.13)) + ";SW1"
10607 PARAMS = "WR.STR/17//EOS":GOSUB 50000
10608 RETURN
10609 STOP
10610 REM
10611 REM *******************************************************************
10612 REM *
10613 REM * This subroutine runs the C-G-V experiment and stores *
10614 REM * the data in the file "DATA\##\CGV".
10615 REM *
10616 REM *
10617 REM *******************************************************************
10618 REM
10619 REM
10620 IF ExpI%(26) = 1 OR ExpI%(26) = 3 THEN GOTO 11030
10621 IF Parameter!(3,4) <> Parameter!(25,4) GOTO 11310
10622 OPEN "C:\DATA" + File$ + "\CGV" FOR APPEND AS #1
10623 PRINT #1,Xtemp$;Parameter!(25,4);"ACTUAL\TEMPP"
10624 PRINT #1,"TIME = ";Time$;"(;TIMER;")"
10625 LOCATE 14,10;PRINT "taking CAPACITANCE AND CONDUCTANCE vs. VOLTAGE measurements"
10626 N% = 6
158
11080 M% = 6
11090 N% = N% + 1
11100 M% = M% + 2
11110 IF Parameter!(3,M%+1) = 0 THEN GOTO 11270
11112 DATA STRINGS$ = "FN1C1NO1B2LE2TR3PS" + STRS(Parameter!(3,N%)) + ";PP" + STRS(Parameter!(3,M%)) + ";PE" + STRS(Parameter!(3,N%+1)) + ";PL" + STRS(Parameter!(3,14)) + ";PD" + STRS(Parameter!(3,13)) + ";SW1"
11130 PARAM$ = "WR.STR/17//EOS":GOSUB 50000
11140 L! = Parameter!(3,N%)
11150 PARAM$ = "RD.STR/17//EOS":GOSUB 50000
11160 NCM% = 1INSTR(DATA STRINGS$,"M") + 1
11170 Cdata! = VAL(MID$(DATA STRINGS$,NCM%))
11180 NGM% = INSTR(DATA STRINGS$,"GM") + 2
11190 Gdata! = VAL(MID$(DATA STRINGS$,NGM%))
11200 V% = INSTR(DATA STRINGS$,"V") + 1
11210 Xdata! = VAL(MID$(DATA STRINGS$,V%))
11220 PRINT #1,Cdata!:"";Gdata!:"";Xdata!
11240 IF L! < Parameter!(3,M%) THEN L! = L! + Parameter!(3,M%+1): GOTO 11150
11250 DATA STRINGS$ = "SW0"
11260 PARAM$ = "WR.STR/17//EOS":GOSUB 50000
11270 IF N% = 7 THEN GOTO 11090
11280 Parameter!(3,4) = Parameter!(3,4) + Parameter!(3,3)
11290 PRINT #1,"END"
11300 CLOSE #1
11310 REM
11350 IF Parameter!(3,4) < Parameter!(25,4) THEN Parameter!(3,4) = Parameter!(3,4) + Parameter!(3,3)
11360 IF Parameter!(3,4) > Parameter!(3,2) THEN Parameter!(3,4) = 600
11370 PARAM$ = "SDC/17":GOSUB 50000
11380 RETURN
11999 STOP
12000 REM
12001 REM******************************************************************************
12002 REM*
12003 REM* This section takes the C-G readings and puts them in
12004 REM* file \DATA\###\CG*
12005 REM*
12006 REM******************************************************************************
12007 REM
12010 IF Expt%(26) = 1 OR Expt%(26) = 3 THEN GOTO 12030
12020 IF Parameter!(4,4) <> Parameter!(25,4) THEN GOTO 12250
12030 OPEN "C:\DATA\" + File$ + "CG" FOR APPEND AS #1
12040 Bias! = Parameter!(4,17)
12050 LOCATE 14,15:PRINT "taking CAPACITANCE AND CONDUCTANCE measurements"
12060 DATA STRINGS$ = "FN1C1NO1B1RA1MS3S2LE3TR3PV" + STRS(Bias!)
12070 PARAM$ = "WR.STR/17//EOS":GOSUB 50000
12072 DATA STRINGS$ = "VO1"
12074 PARAM$ = "WR.STR/17//EOS":GOSUB 50000
12080 PRINT #1,Xtemp$;Parameter!(25,4);"ACTUALTEMP"
12090 PRINT #1,"TIME = ";Timerun!:"(";TIMER," )"
12100 PRINT #1,"BIAS = ";Bias!
12105 TIMEDELAY(5)
12110 DATA.Strings$ = "EX"
12120 Param$ = "WR,STR/17//EOS":GOSUB 50000
12130 Param$ = "RD,STR/17//EOS":GOSUB 50000
12140 NCM% = INSTR(Data.Strings$,"M") + 1
12150 Cdata1 = VAL(MIDS(Data.Strings$,NCM%,))
12160 NGM% = INSTR(Data.Strings$,"GM") + 2
12170 Gdata1 = VAL(MIDS(Data.Strings$,NGM%,))
12180 PRINT #1,Cdata1,"Gdata1","ACTUALTEMP"
12185 Data.Strings$ = "V00"
12187 Param$ = "WR,STR/17//EOS":GOSUB 50000
12190 IF Parameter!(4,19) = 0 THEN GOTO 12220
12200 Bias! = Bias! + Parameter!(4,19)
12210 IF Bias! <= Parameter!(4,18) THEN GOTO 12060
12220 Parameter!(4,4) = Parameter!(4,4) + Parameter!(4,3)
12225 TIMEDELAY(10)
12230 CLOSE #1
12240 Param$ = "SDC/17":GOSUB 50000
12250 REM
12260 IF Parameter!(4,4) < Parameter!(25,4) THEN Parameter!(4,4) = Parameter!(4,4) + Parameter!(4,3)
12270 IF Parameter!(4,4) > Parameter!(4,2) THEN Parameter!(4,4) = 600
12280 Param$ = "SDC/17":GOSUB 50000
12290 RETURN
12300 STOP
13000 REM
13010 REM ************************************************************************************
13020 REM *                                                                                   *
13030 REM * This is the subroutine for actually running the I-V                                 *
13040 REM * experiment. It stores the data in the file                                        *
13050 REM * DDATA##IIV                                                                            *
13060 REM *                                                                                   *
13070 REM ************************************************************************************
13080 REM
13090 IF Exp1%26 = 1 OR Exp1%26 = 3 THEN GOTO 13030
13100 OPEN ";D:DATA! + File$ + "IIV" FOR APPEND AS #1
13110 LOCATE 14,20:PRINT "taking CURRENT vs. VOLTAGE measurements"
13120 PRINT #1,Xtemp$;Parameter!(25,4);"ACTUALTEMP"
13130 PRINT #1,"TIME = ";Timer!;"(";TIME;");"
13140 M% = 7
13150 N% = 8
13160 X$ = STR$(VAL(STR$(Parameter!(1,N% + 1))))
13170 DATA.Strings$ = "F212L3PS" + STR$(Parameter!(1,M%)) + ";PT" + STR$(Parameter!(1,N%)) + ";PE" + X$ + ";PH" + STR$(Parameter!(1,14)) + ";PD" + STR$(Parameter!(1,13)) + ";W1"
13180 TIMEDELAY(1)
13190 Param$ = "WR,STR/5//EOS":GOSUB 50000
13200 B% = 0
13210 Param$ = "RD,STR/5//EOS":GOSUB 50000
13220 NCM% = INSTR(Data.Strings$,"T") + 1
13150 ldata! = VAL(MIDS(DATA_STRING$\%NCM))
13160 V% = INSTR(DATA_STRING$,*A*) + 1
13170 Xdata! = VAL(MIDS(DATA_STRING$\%V%))
13180 PRINT #1,Idata!,*Xdata!
13190 B% = INSTR(DATA_STRING$,*L*)
13200 IF B% = 0 THEN GOTO 13130
13210 TIMEDELAY(2)
13215 PARAMS = *RD,STR/S//EOS*:GOSUB 50000
13220 IF Parameter!(1,11) = 0 THEN GOTO 13260
13230 M% = M% + 1
13240 N% = N% + 2
13250 IF M% = 8 THEN GOTO 13085
13260 Parameter!(1,4) = Parameter!(1,4) + Parameter!(1,3)
13270 PRINT #1,*END*
13280 CLOSE #1
13290 REM
13330 IF Parameter!(1,4) < Parameter!(25,4) THEN Parameter!(1,4) = Parameter!(1,4) + Parameter!(1,3)
13340 IF Parameter!(1,4) > Parameter!(1,2) THEN Parameter!(1,4) = 600
13350 PARAMS = *SDC/S*:GOSUB 50000
13360 RETURN
13999 STOP
14000 REM
14001 REM ********************************************
14002 REM *
14003 REM * This actually runs the Capacitance vs. Voltage *
14004 REM * experiment. *
14005 REM *
14006 REM ********************************************
14007 REM
14010 IF Exp\%\%26 = 1 OR Exp\%\%26 = 3 THEN GOTO 14030
14020 IF Parameter!(2,4) <> Parameter!(25,4) THEN GOTO 14210
14030 OPEN "C:\DATA\" + Files + "\CV" FOR APPEND AS #1
14040 LOCATE 14,18:PRINT "taking CAPACITANCE vs. VOLTAGE measurements"
14050 B% = 0
14060 PRINT #1,Temp\%,Parameter!(25,4);*ACTUALTEMP
14070 PRINT #1,TIME = ";Timerun!:; (";'TIMER:';)
14080 DATA_STRING$ = "F3I2L3PS" + STR$(Parameter!(2,7)-Parameter!(2,9)) + ";PT" + STR$(Parameter!(2,8)+Parameter!(2,9)) + ";PE" + STR$(Parameter!(2,9)) + ";PV" + STR$(Parameter!(2,12)) + ";PH" + STR$(Parameter!(2,14)) + ";W1"
14090 PARAMS = "WR,STR/S//EOS":GOSUB 50000
14100 PARAMS = *RD,STR/S//EOS*:GOSUB 50000
14110 NCM% = INSTR(DATA_STRING$,*C*) + 1
14120 Cdata! = VAL(MIDS(DATA_STRING$\%NCM))
14130 V% = INSTR(DATA_STRING$,*A*) + 1
14140 Xdata! = VAL(MIDS(DATA_STRING$\%V%))
14150 PRINT #1,Cdata!,*Xdata!
14160 B% = INSTR(DATA_STRING$,*L*)
14170 IF B% = 0 THEN GOTO 14100
14180 Parameter!(2,4) = Parameter!(2,4) + Parameter!(2,3)
14190 PRINT #1,*END*
14200 CLOSE #1
14210 REM
14250 IF Parameter!(2,4) < Parameter!(25,4) THEN Parameter!(2,4) = Parameter!(2,4) + Parameter!(2,3)
14260 IF Parameter!(2,4) > Parameter!(2,2) THEN Parameter!(2,4) = 600
14270 PARAMS = "SDC/5":GOSUB 50000
14280 RETURN
14999 STOP
15000 REM
15001 REM ************************************************************************************
15002 REM *
15003 REM * This subroutine runs the Van der Pauw/Mobility experiment and stores it in the file MOB. *
15004 REM *
15005 REM *
15006 REM ************************************************************************************
15007 REM
15009 IF Print_temp% = 1 THEN CLEARSCREEN(1)
15010 COLOR 2,0:LOCATE 14,20
15020 PRINT "taking Van der Pauw/Mobility Measurements"
15030 OPEN "DATA:" + File$ + "MOB" FOR APPEND AS #1
15032 DO
15035 IF Print_temp% <> 2 THEN EXIT 1 LEVELS
15040 PRINT #1,Xtemp$;Parameter!(25,4);"ACTUALTEMP"
15050 PRINT #1,"TIME = ";Time.run;" (;;TIME;"
15055 END DO
15060 DATA_STRINGS = "FN1C110I21B2TR3PS-10;PP10;PE.01;PL2;PD2"
15070 PARAMS = "WR.STR/17/EOS":GOSUB 50000
15078 DATA_STRINGS = "L3;"
15079 PARAMS = "WR.STR/5/EOS":GOSUB 50000
15089 DATA_STRINGS = "R0" + CHR$(13) + "X"
15096 PARAMS = "WR.STR/16/EOS":GOSUB 50000
15092 TIMEDELAY(15)
15095 DATA_STRINGS = "Z1" + CHR$(13) + "X"
15096 PARAMS = "WR.STR/16/EOS":GOSUB 50000
15100 FOR M% = 1 TO Print_temp%
15110 DATA_STRINGS = "F2RAI1210A3B2L3PS0;PT10;PE0.01;PH1;PD.1;W2"
15120 PARAMS = "WR.STR/5/EOS":GOSUB 50000
15130 DO 1000 TIMES
15140 DATA_STRINGS = "W6"
15150 PARAMS = "WR.STR/5/EOS":GOSUB 50000
15160 TIMEDELAY (2)
15170 PARAMS = "RD.STR/5/EOS":GOSUB 50000
15180 CURRENT! = VAL(MIDS(DATA_STRINGS$4))
15190 VOLT! = VAL(MIDS(DATA_STRINGS$,16))
15200 IF CURRENT! => Parameter!(1,17) THEN EXIT
15210 REPEAT
15211 DATA_STRINGS$ = "S1" + CHR$(13) + "X"
15212 PARAMS = "WR.STR/16/EOS":GOSUB 50000
15235 TIMEDELAY(10)
15237 PARAMS = "RD.STR/16/EOS":GOSUB 50000
15240 VOLT2! = VAL(MIDS(DATA_STRINGS$5))
15250  PRINT #1,M%;",";CURRENT%;",";VOLT1;",";VOLT2!
15260  IF M% = 1 AND Print_temp% = 2 THEN DATA_STRINGS = "SW1" ELSE
15270  DATA_STRINGS = "SW0"
15280  DATA_STRINGS = "W7"
15290  DATA_STRINGS = "WR.STR/17/EOS":GOSUB 50000
15300  IF M% = 1 AND Print_temp% = 2 THEN TIMEDELAY (20): DATA_STRINGS = "Z1"
15310  NEXT M%
15320  PRINT #1,"END"
15330  CLOSE #1
15340  PARAMS = "$SDC/5,16,17":GOSUB 50000
15350  RETURN
16000 REM
16001 REM *******************
16002 REM *
16003 REM * This section takes the RES readings and puts them in *
16004 REM * file \DATA\##\RES*
16005 REM *
16006 REM *
16007 REM
16030 OPEN "C:\DATA\" + File$ + "\RES" FOR APPEND AS #1
16040 Bias! = Parameter! (1,17)
16050 LOCATE 14,17PRINT "taking 4-POINT RESISTIVITY measurements"
16060 PRINT #1, Xtemp$:Parametrs(25,4),";ACTUALTEMP
16070 PRINT #1,"TIME = ";Timerun!;" (";TIMER;")
16100 PRINT #1,"BIAS = ";Bias!
16110 DATA.Strings = "R0" + CHR$(13) + "X"
16120 PARAMS = "WR.STR/16/EOS":GOSUB 50000
16130 TIMEDELAY(5)
16140 PARAMS = "RD.STR/16/EOS":GOSUB 50000
16150 VOLT2! = VAL(MIDS(DATA_STRINGS, 5))
16160 PRINT #1,VOLT2!;"VOLT2!":"ACTUALTEMP
16225 TIMEDELAY(1)
16230 CLOSE #1
16240 PARAMS = "$SDC/16":GOSUB 50000
16320 RETURN
16999 STOP
39999 STOP
40000 REM
40001 REM *******************
40002 REM *
40003 REM * This stops the program after a successful run.
40004 REM *
40005 REM *******************
40006 REM
40010 IF Expt%(26) <= 3 THEN GOSUB 41500
40020 IF Expt%(25) = 1 THEN RESTORE,60210
40030 IF Expt%(25) = 2 THEN RESTORE,60220
40040 IF Expt%(25) = 3 THEN RESTORE,60230
40050 IF Expt%(25) = 4 THEN RESTORE,60240
IF Exp$(25) = 5 THEN RESTORE,60250
淼 IF Exp$(25) = 6 THEN RESTORE,60260
淼 IF Exp$(25) = 7 THEN RESTORE,60270
淼 IF Exp$(25) = 8 THEN RESTORE,60280
淼 IF Exp$(25) = 9 THEN RESTORE,60290
淼 IF Exp$(25) = 10 THEN RESTORE,60300
淼 FOR M% = 1 TO Group%(Exp$(25))
淼 READ Name$
淼 IF Exp$(M%) = 0 THEN GOTO 40260
淼 OPEN "DATA" + File$ + "," + Name$ FOR APPEND AS #1
淼 PRINT #1,"END"
淼 CLOSE #1
淼 NEXT M%
淼
淼 REM
淼 1100 REM
淼 1101 REM ***************
淼 1102 REM *
淼 1103 REM * This section
淼 1104 REM * finds out whether the user wants to
淼 1105 REM * another experimental run.
淼 1106 REM *
淼 1107 REM ***************
淼 1108 REM
淼 1109 REM
淼 1110 PARAM$ = "ABORT";GOSUB 50000
淼 1111 CLS;COLOR 10,0,0;LOCATE 12,5
淼 1112 INPUT "Do you want to run another set of experiments (Defaults to No);Chk$
淼 1113 IF Chk$ = "Y" OR Chk$ = "Y" THEN GOTO 41060 ELSE GOTO 41100
淼 1114 OPEN "DATA\REDO" FOR OUTPUT AS #1
淼 1115 PRINT #1,File$
淼 1116 CLOSE #1
淼 1117 CLS
淼 1118 SYSTEM
淼 4149 STOP
淼 4150 REM
淼 4150 REM
淼 4150 REM ***************
淼 4150 REM *
淼 4150 REM * This subroutine gives the Cryostat default parameters *
淼 4150 REM * when the program RUNIT is finished.
淼 4150 REM *
淼 4150 REM ***************
淼 4150 REM
淼 4150 REM
淼 4150 REM DATA STRING$ = "P"
淼 4150 REM DATA STRING$ = "M";GOSUB 50000
淼 4150 REM DATA STRING$ = "M";GOSUB 50000
淼 4150 REM DATA STRING$ = "R"
淼 4150 REM DATA STRING$ = "R";GOSUB 50000
淼 4150 REM DATA STRING$ = "RD.STR/7/EOS";GOSUB 50000
淼 4150 RETURN
淼 4199 STOP
淼 4200 REM
淼 4200 REM
42003 REM *
42004 REM * This is the ERROR handling routines. It checks to see *
42005 REM * what error is and attempts to fix it.                  *
42006 REM *
42007 REM *******************************************************
42008 REM
42010 IF ERR = 1008 THEN RESUME
42020 PRINT"Sorry the main program is bombing. This is the error ";ERR
42030 PRINT"The line number is ";ERL
42040 STOP
60000 REM
60001 REM *******************************************************
60002 REM *
60003 REM * This data statement tells the program how many *
60004 REM * experiments exist in each group.                   *
60005 REM *
60006 REM *******************************************************
60007 REM
60009 DATA 4,4,2,1,2
60010 STOP
60100 REM
60101 REM *******************************************************
60102 REM *
60103 REM * This data statement stores what type of temperature *
60104 REM * it is in the string XTEMPS.                     *
60105 REM *
60106 REM *******************************************************
60107 REM
60108 DATA "TTEMP = ","LTEMP = ","RTEMP = 
60109 STOP
60200 REM
60201 REM *******************************************************
60202 REM *
60203 REM * These data statements are used to print the word "END" *
60204 REM * in each of the data files to show where the file quits. *
60205 REM *
60206 REM *******************************************************
60207 REM
60208 DATA "C-T","G-T","CGV","CG"
60209 DATA "TV","CV"
60210 DATA "MOB"
60211 DATA "RES"
60212 DATA 
60213 DATA 
60214 DATA 
60215 DATA 
60216 DATA 
60217 DATA 
60218 DATA 
60219 DATA 
60220 REM
60410 DATA 24
60400 REM
60401 DATA
62010 DATA "CABLE 1 to Va","CABLE 2 to I high","CABLE 3 to V- of the Keithley","CABLE 4 to V+ of the Keithley","I low to Ground"
62020 DATA "CABLE 1 to V- of the Keithley","CABLE 2 to I high","CABLE 3 to Va","CABLE 4 to V+ of the Keithley","I low to Ground"
62030 DATA "CABLE 1 to Va","CABLE 2 to V+ of the Keithley","CABLE 3 to I high","CABLE 4 to V- of the Keithley","I low to Ground"

ENDFILE
GRAPHICS
SOURCE
PRECISION=7
AUTODEF=ON
OPTION BASE=1
ERL=ON
ERRORMODE=LOCAL
RESUME=LINE
FORMODE=BB
SCOPE=ON
PROC=49
STRING ARRAY(5,24)[32]: Graph_name$
STRING ARRAY(2,2)[5]: Directory$
STRING ARRAY(2)[20]: Tyme$,Temp$,Bias$,Hold$,Rate$,Slope$,Y_int$,X_int$
STRING ARRAY(2)[20]: Range$
STRING ARRAY(2)[20]: Correlation$,Pulse$
STRING ARRAY(2)[3]: Temp_type$
INTEGER ARRAY(5): Maxloop%
INTEGER ARRAY(2): Max_graph_points%
BYTE ARRAY(4003): Menu1,Menu2,Menu3,Menu4
BYTE ARRAY(10): Plus
REAL ARRAY(2,2,1200): Graph!
REAL ARRAY(2): X_int!,Y_int!,Slope!,Correlation!
REAL ARRAY(2): Min!,Max!
INTEGER: Row_number%,Col%,Menu_chosen%,Graph_chosen%,Error%,Minloop%,Print_it%
INTEGER:
Which_graph%,Graph_type%,M%,N%,Next_screen%,Choice%,Comp_graph%,Graph%
INTEGER:
File%,Graph_pick%,ID%,Print_out%,Info_pointer%,Max_graph_number%,Pointer%,Return%
INTEGER: Position%
REAL: MinX!,MaxX!,MinY!,Q!,K!,E_sub_S!,Area!,Slope!
REAL: Ac!,Vf!,I!,Min!,Max!,Y_int!,X_int!,Graph!
REAL: MaxX!,Temp_sought!,Time_sought!,Dummy!,DeltaX!,DeltaY!,Y_plot!,X_plot!
REAL: Mintyme!,Maxtyme!,Bias_sought!,Pulse_sought!,Hold_sought!,Temp!,Y!,Nc!
REAL: Nd!,X!,E_sub_O!,Elec_thick!,I!,V1!,V2!,R1!
REAL: R2!,G!,Eric!
STRING: File$[5],Xtitle$[77],Ytitle$[77],A$[3],Check$[3],Temp_found$[25]
STRING:
Time_found$[16],Bias_found$[12],Pulse_found$[17],Hold_found$[33],Name$[30],Save_file$[40],Area$[16],Nd$[16]
STRING: VS[?]
INTEGER: Counter%,Skip%
STRING ARRAY(12)[32]: Hand_info$
INTEGER: Run_type%
STRING ARRAY(2)[18]: GRange$
INTEGER: Minloop1%,Maxloop%
STRING: Change$[?],Fit$[?],Symbol$[?]
INTEGER: Error_LS%
STRING: Title$[?],Dummy$[?],IS[?]
INTEGER: Max_graph_points%
STRING ARRAY(?): Ra$
STRING: V1$[?],V2$[?]
REAL: Power!
INTEGER: Pen_selected%
STRING: Bias$,Line_type$,Line_type$,SYMBOLS$,?
STRING ARRAY(5)[16]: SYMBOLS
STRING ARRAY(5)[16]: LINETYPES
INTEGER ARRAY(8): PEN
REAL: X_plot!, Y_plot!
INTEGER: Graph1or2%, G1, G2, G
INTEGER: Onoff
STRING: SOURCES[16]
REAL: NT, Voltage, In, Volts, O
INTEGER ARRAY(?): Skippoint%
REAL ARRAY(?): Plot!
INTEGER: O%, P%
INTEGER ARRAY(?): Min%
INTEGER: Min%
REAL: Thick!, GW
STRING: HGS[
REAL: CF
INTEGER: MOD, GFLAG, Saveflag
STRING: SPEEDS[5]

PROCEDURE: TITLE()
  STRING ARG: Title$
END PROCEDURE

PROCEDURE: MENU()
  STRING ARG: Name$
  INTEGER ARG: Number%
  INTEGER ARG: Rownumber%/VAR
END PROCEDURE

PROCEDURE: FINISH()
  STRING ARG: Title$
  INTEGER ARG: Rownumber%/VAR, Col%/VAR
  INTEGER ARG: Number%
END PROCEDURE

PROCEDURE: BORDER()
  INTEGER ARG: Rownumber%/VAR
END PROCEDURE

PROCEDURE: CLEARSCREEN()
  INTEGER ARG: Row%
END PROCEDURE

PROCEDURE: DIRECTORY()
  STRING ARG: File$
  INTEGER ARG: Dir1or2%, Graph1or2%
  INTEGER ARG: Error%/VAR
END PROCEDURE
PROCEDURE: TIMEDELAY()
   REAL ARG: Delay!/OPT=5!
END PROCEDURE

PROCEDURE: XTITLE()
   STRING ARG: Title$
   REAL ARG: Min!,Max!
END PROCEDURE

PROCEDURE: YTITLE()
   STRING ARG: Title$
   REAL ARG: Min!,Max!
END PROCEDURE

PROCEDURE: LEASTSQUARES()
   INTEGER ARG: Which_graph%
   REAL ARG: Min!/VAR,Max!/VAR,Slope!/VAR,Correlation!/VAR,X_int!/VAR,Y_int!/VAR
   INTEGER ARG: Skip%
   !INTEGER ARG: Error_LS%/VAR
END PROCEDURE

PROCEDURE: RUNTYPE()
   !INTEGER ARG: Graph1or2%,Skip%
END PROCEDURE

PROCEDURE: DELTAMULTIPLY()
   INTEGER ARG: Graph1or2%,XorY%
   REAL ARG: Amount!
END PROCEDURE

PROCEDURE: DELTAIN()
   INTEGER ARG: Graph1or2%,XorY%
   REAL ARG: Amount!
END PROCEDURE

PROCEDURE: DATFORMIVAL()
   INTEGER ARG: Graph_chosen%,Graph1or2%,File%
   INTEGER ARG: Error%/VAR
   !INTEGER ARG: Run_type%
END PROCEDURE

PROCEDURE: DATAPRINT
END PROCEDURE

PROCEDURE: PARAMETERSET()
   INTEGER ARG: Graph_type%
END PROCEDURE

PROCEDURE: PLOTTING_POINTS()
   STRING ARG: Symbol$
   INTEGER ARG: Graph1or2%,Print_out%
END PROCEDURE

PROCEDURE: GETDATA()
   INTEGER ARG: Graph%
END PROCEDURE

PROCEDURE: PLOTTERAXES()
   STRING ARG: Title$
END PROCEDURE

PROCEDURE: SAVEDATA()
   INTEGER ARG: Graph%
END PROCEDURE

PROCEDURE: LOADDATA()
   INTEGER ARG: Graph%
END PROCEDURE

PROCEDURE: INVERSEPOWER()
   INTEGER ARG: Graph!or2%,Xory%
   REAL ARG: Amount!,Power!
END PROCEDURE

PROCEDURE: MAINKEY
END PROCEDURE

PROCEDURE: PLOTTER
END PROCEDURE

PROCEDURE: MODIFY
END PROCEDURE

PROCEDURE: COLORS
END PROCEDURE

PROCEDURE: SYMBOLS
END PROCEDURE

PROCEDURE: LINES
END PROCEDURE

PROCEDURE: Info
END PROCEDURE

PROCEDURE: CAPTION
END PROCEDURE

PROCEDURE: Fixpoints
   INTEGER ARG: Type%
   REAL ARG: Plotit!
   REAL ARG: Pluto/VAR,Plot1!/VAR
PROCEDURE: Title
   INTEGER: Col%
   EXTERNAL: G1,G2,GRAPH%,GFLAG
   100 CLS:COLOR 5,0
   104 SET CURSOR 0,70:COLOR 2,7:PRINT GFLAG:COLOR 5,0
   105 SET CURSOR 0,74:COLOR 8,7:PRINT G1;G2:COLOR 5,0
   110 Col% = CINT((60-LEN(Title$))/2)
   120 SET CURSOR 0,Col%
   130 PRINT Title$
END PROCEDURE

PROCEDURE: Menu
   INTEGER: M%,Placement%,Row%,Col%
   STRING: Number$[5]
90 COLOR 3,0
95 IF Number% = 1 THEN Row% = -1
100 FOR M% = 1 TO 4
110 IF Number% <= M% * 24 THEN EXIT
120 NEXT M%
130 Placement% = Number% - (M% - 1)*25
140 IF Number% > 24 THEN Placement% = Placement% + M% - 1
150 Number$ = STR$(Number%)
160 DELS(Number$,1,1)
170 IF Placement% < 13 THEN Col% = 5 ELSE Col% = 42
180 IF Placement% = 13 THEN Row% = 0 ELSE Row% = Row% + 1
190 SET CURSOR Row% + 4,Col%
200 PRINT "(*;Number$;*) ",Name$
210 IF Rownumber% < CSRLIN THEN Rownumber% = CSRLIN
220 COLOR 7,0
230 EXIT

END PROCEDURE

PROCEDURE: FINISH
STRING: Number$[5]
100 COLOR 3,0
110 Rownumber% = Rownumber%
120 Number$ = STR$(Number%)
130 DELS(Number$,1,1)
140 IF Col% = 0 THEN Col% = CINT((73 - LEN(Name$))/2)
150 SET CURSOR Rownumber%,Col%
160 PRINT "(*;Number$;*) ",Name$
170 Col% = 0
180 COLOR 7,0
190 EXIT

END PROCEDURE

PROCEDURE: BORDER
INTEGER: M%
100 COLOR 2,0
110 SET CURSOR 2,1
120 PRINT "##################################################################
130 PRINT "***************
140 Rownumber% = Rownumber% + 1
150 FOR M% = 3 TO Rownumber%
160 SET CURSOR M%,1
170 PRINT "**
180 SET CURSOR M%,79
190 PRINT "**
200 NEXT M%
210 Rownumber% = Rownumber% + 1
220 SET CURSOR Rownumber%,1
230 PRINT "##################################################################
240 PRINT "***************
250 Rownumber% = 0
260 COLOR 7,0
PROCEDURE: CLEARSCREEN
    INTEGER: M%
    100 FOR M% = Row% TO 23
    110 SET CURSOR M%,0
    120 PRINT SPC(80)
    130 NEXT M%
    140 EXIT
END PROCEDURE

PROCEDURE: DIRECTORY
    INTEGER: M%,Start%,Comp_graph%
    EXTERNAL: Directory$( )
    STRING: Number$[8]
    INTEGER: N%,Stop%
    REAL: M
    STRING: AS[16]
    EXTERNAL: SOURCES$?

PROCEDURE: TIMEDELAY
END PROCEDURE

PROCEDURE: TIMEDELAY
    REAL: Time!
    100 Time! = Timer + 4
    110 IF Timer < Time! THEN GOTO 110
    120 EXIT
    130 END
END PROCEDURE

90 CLOSE
100 ON ERROR GOTO 10000
110 Start% = 1
115 Stop% = 1
120 Error% = 0
131 SET CURSOR 4,6:COLOR 7,0
133 INPUT "IS THE DATA SOURCE C OR B DRIVE";SOURCES$
134 SOURCES=UPPERS(SOURCES$)
140 FOR M% = Start% TO Stop%
141 FOR N% = 5+2*M% TO 23
142 LOCATE N%,1:PRINT SPC(79)
143 NEXT N%
170 CLEAR (ERR)
180 COLOR 2,0;SET CURSOR 5 + 2*M%,6
190 PRINT "Enter the data source directory number (OR SAMPLE/RUN");
200 INPUT Directory$(Graph1or2%,M%)
210 IF Directory$(Graph1or2%,1) = "" AND Comp_graph% = 0 THEN GOTO 170
230 IF INSTR(SOURCES$,"B")=0 THEN OPEN "DATA" + Directory$(Graph1or2%,M%)
+ File$ FOR INPUT AS #M% ELSE OPEN "B:" + File$ + "+" +Directory$(Graph1or2%,M%)
FOR INPUT AS M%
240 NEXT M%
245 Directory$(Graph1or2%,2) = "NO"
250 Comp_graph% = 1
260 EXIT
270 STOP
10000 REM
10010 COLOR 7,0:SET CURSOR 20,5
10020 IF ERR = 1001 AND M% = 1 AND Comp_graph% = 0 THEN GOSUB 10500:RESCALE,260
10030 IF ERR = 1001 AND M% = 1 AND Comp_graph% = 1 THEN GOSUB 10600:RESCALE,110
10040 IF ERR = 1001 AND M% = 2 THEN GOSUB 10600:RESCALE,120
10050 IF ERR = 1007 THEN GOSUB 10700:RESCALE,120
10300 PRINT "THIS IS AN UNEXPECTED ERROR IN PROCEDURE DIRECTORY IN LINE ;;ERL;"AND ERROR NUMBER;"ERR"
10310 STOP
10500 REM
10510 PRINT "The graph you asked for does not exist in the directory chosen."
10520 SET CURSOR 21,5:PRINT "Please choose another graph."
10530 TIMEDELAY
10540 Error% = 1
10550 RETURN
10600 REM
10610 PRINT "The graph you asked for does not exist in the directory chosen."
10620 SET CURSOR 21,5:PRINT "Please choose another directory."
10630 TIMEDELAY
10640 Start% = M%
10650 RETURN
10700 REM
10710 PRINT "The directory you asked for does not exist."
10720 SET CURSOR 21,5:PRINT "Please choose another directory."
10730 TIMEDELAY
10740 Start% = M%
10750 RETURN
END PROCEDURE

PROCEDURE: TIMEDELAY
REAL: Waiting!, Waiting
100 Waiting! = TIMER + Delay!
110 IF TIMER < Waiting! THEN GOTO 110
120 EXIT
END PROCEDURE

PROCEDURE: XTITLE
INTEGER: Col%, M%
100 ON ERROR GOTO 10000
110 LOCATE 22,4:PRINT USING ";.####^~^~";Min!
120 LOCATE 22,72:PRINT USING ";.####^~^~";Max!
150 DRAW "BM105,163 D3 BR57 U3 BR57 D3 BR57 U3 BR57 D3 BR57 U3 BR57 D3 BR57 U3 BR57 D3 BR57 U3 BR57 D3 BR57 U3 BR57 D3 BR57 U3 BR57 U3 BR57 D3 BR57 U150 BD150 L570"
155 DO 1 TIMES
DO 1 TIMES
IF ABS(Min!) <> Max! THEN EXIT
LOCATE 22,42:PRINT "0"
EXIT 2 LEVELS
REPEAT
DO 1 TIMES
IF (Max! - Min!) MOD (10) <> 0 THEN EXIT
FOR M% = 1 TO 9
LOCATE 22.5 + M%*7.5
IF M% + Min! = 0 THEN EXIT
NEXT M%
IF M% < 10 THEN PRINT "0"
REPEAT
REPEAT
Col% = CINT((79 - LEN(Title$))/2)
LOCATE 23,Col%:PRINT Title$
EXIT
10000 REM
10010 VIEW:SCREEN 0,0,0:COLOR 7,0:SET CURSOR 12,5
10020 PRINT "Sorry the procedure XTITLE is bombing. This is error number ";ERR;" from line number ";ERL
10030 STOP
END PROCEDURE

PROCEDURE: YTITLE
INTEGER: Max%,Row%,M%
STRING: YS[3]
100 ON ERROR GOTO 10000
110 DRAW "BM48,15 R570 BL570 BD15 R3 BD15 L3 BD15 R3 BD15 L3 BD15 R3 BD15 L3 BD15 R3 BL15 U150"
120 Max% = LEN(Title$)
130 Row% = CINT((23 - Max%)2) - 2
135 IF Row% < 0 THEN Row% = 0
137 IF Max% > 23 THEN Max% =23
140 FOR M% = 1 TO Max%
150 YS = MIDS(Title$,M%,1)
160 SET CURSOR Row% + M%,0:PRINT YS
170 NEXT M%
180 SET CURSOR 1,0:PRINT USING ";#.##^^^^^";Max!
190 SET CURSOR 20,0:PRINT USING ";#.##^^^^^";Min!
200 EXIT
10000 CLS:VIEW:SCREEN 0,0,0:COLOR 7,0:SET CURSOR 11,5
10010 PRINT "The Xprocedure YTITLE is ^Ambig. This is error ";ERR;" from line ";ERL
10020 STOP
END PROCEDURE

PROCEDURE: LEASTSQUARES
EXTERNAL: Graph1(),Max_graph_points()%
INTEGER: Minloop%,Maxloop%,M%,Total_samples%
REAL: SumX2!,SumY2!,SumXY!,SumX!
REAL: SumY!,TotalX2!,TotalY2!,Xmean!,Ymean!,Sxx!,Syy!,Sxy!
INTEGER: Chich_graph%
REAL: Min1!,Max1!
INTEGER: N%
REAL: XTOTAL!,YTOTAL!
STRING: Min1$,[?],Max1$[?]
INTEGER: Minloop1%,Step%,Maxloop1%
STRING ARRAY(?) SLOPES,X_INTS,Y_INTS
STRING ARRAY(?) RANGES,CORRELATIONS,GRANGES
REAL#: Correlation!
   80 Error_LS% = 0
   90 ON ERROR GOTO 60000
   95 Minloop1% = 1: Maxloop1% = Max_graph_points%(Which_graph%):Step% = 1
  100 CLEAR
(SumX2!,SumY2!,SumX!,SumY!,SumXY!,Sxx!,Syy!,Sxy!,Xmean!,Ymean!,XTOTAL!,YTOTAL!)
  105 IF Skip% = 1 THEN GOTO 165
  107 GOSUB 20000
  110 COLOR 2,0:SET CURSOR 13,1
  120 INPUT "Over What X-axis range do you want your least square fit
(min,max)";Min1$,Max1$
  125 Min1! = VAL(Min1$) : Max1! = VAL(Max1$)
  130 GOSUB 20000
  135 DO
  140 SET CURSOR 13,32
  150 PRINT "Now LOADING data"
  160 END DO
  165 IF Graph!(Which_graph%,1,Max_graph_points%(Which_graph%)) <
Graph!(Which_graph%,1,1) THEN Minloop1% =
Max_graph_points%(Which_graph%):Maxloop1% = 1 :Step% = -1
   168 Min1! = Min1! : Max1! = Max1!
  170 FOR M% = Minloop1% TO Maxloop1% STEP Step%
  180 IF Graph!(Which_graph%,1,M%) => Min1! THEN EXIT
  190 NEXT M%
  200 Minloop% = M%
  210 FOR M% = Minloop1% TO Maxloop1% STEP Step%
  220 IF Graph!(Which_graph%,1,M%) => Max1! THEN EXIT
  230 NEXT M%
  240 Maxloop% = M%
  250 FOR M% = Minloop% TO Maxloop% STEP Step%
  260 SumX2! = SumX2! + Graph!(Which_graph%,1,M%) ^ 2.0
  270 SumY2! = SumY2! + Graph!(Which_graph%,2,M%) ^ 2.0
  280 SumXY! = SumXY! + Graph!(Which_graph%,1,M%) * Graph!(Which_graph%,2,M%)
  290 SumX! = SumX! + Graph!(Which_graph%,1,M%)
  300 SumY! = SumY! + Graph!(Which_graph%,2,M%)
  310 NEXT M%
  340 DO
  341 TotalX2! = SumX! * SumX!
  342 TotalY2! = SumY! * SumY!
  343 Total_samples% = Maxloop% - Minloop% + 1
  344 Xmean! = SumX!/Total_samples%
  345 Ymean! = SumY!/Total_samples%
  346 Sxx! = SumX2! - TotalX2!/Total_samples%
470  $yy! = \text{SumY}^2! - \text{TotalY}^2!/	ext{Total_samples}$%
480  $sx! = \text{SumXY}! - (\text{SumX}! \times \text{SumY}!)/\text{Total_samples}$%
490 END DO
600 DO
610  $\text{Slope}! = sx!/sx!$
620  $\text{Correlation}! = sx!/\text{SQR}(sx! \times sy!)$
630  $y! \_\text{int}! = c!y! \_\text{mean}! - (\text{Slope}! \times x! \_\text{mean}!)$
640  $x! \_\text{int}! = -1 \times (y! \_\text{int}! / \text{Slope}!)$
650 END DO
660  $\text{Min1}! = \text{Min1}! : \text{Max}! = \text{Max1}!
700 EXIT
10010 STOP
20000 DO
20010 FOR $n!% = 3 \text{ TO } 23$
20020  LOCATE $n!%:1 \colon \text{PRINT SPC}(79)$
20030  NEXT $n!$
20035 END DO
20040 RETURN
60000 IF ERR < 2 AND ERR > 3 THEN GOSUB 20000 ELSE GOTO 62000
60010 COLOR 7,0 : SET CURSOR 13,1
60999 COLOR 7,0 : SET CURSOR 13,1
61000 PRINT "The proc ^ Aure LEASTSQUARES is boxbing. ThXs is erro ^ A",ERR," from line ";ERL
61010 FOR $m% = -30000 \text{ TO } 30000 : \text{NEXT } m%
61020 Error_LS% = 1
61030 EXIT
62000 Slope! = 0:Correlation! = 0:y!_int! = 0:x!_int! = 0
62010 EXIT
END PROCEDURE

PROCEDURE: RUNTYPE
STRING: Check$[10$
INTEGER: $m$
EXTERNAL: Temp_sought!, Time_sought!, Temp_type$()
STRING: TS$[10$
100 DO
110 FOR $m% = 2 \text{ TO } 23$
120  LOCATE $m%:1 \colon \text{PRINT SPC}(79)$
130  NEXT $m$
140 COLOR 2,0,0:SET CURSOR 5,6
150 INPUT "Enter the Type of Temperature run (T,L,R):$;Temp_type$(Graph1or2$%)
160 Temp_type$(Graph1or2$%) = UPPERS$(Temp_type$(Graph1or2$%))
170 IF INSTR$(Temp_type$(Graph1or2$%),"L") < > 0 THEN EXIT
180 IF INSTR$(Temp_type$(Graph1or2$%),"R") < > 0 THEN EXIT TO,400
190 IF INSTR$(Temp_type$(Graph1or2$%),"T") < > 0 THEN EXIT TO,600
220 REPEAT
300 DO
305 IF Skip% = 1 THEN EXIT
310 SET CURSOR 7,6
320 INPUT "Enter the Temperature at which to read the data": Temp_sought!
330 END DO

178
400 DO
410   COLOR 7,0,0:SET CURSOR 22,6
420   INPUT "Do you want to change any of the above (Defaults to NO)";Check$ 
430   Check$ = UPPERS(Check$) 
440   IF INSTR(Check$,"Y") <> 0 THEN EXIT TO,100
450   END DO 
460   EXIT 
465   IF Skip% = 1 THEN EXIT 1 LEVELS 
470   SET CURSOR 7,6 
480   INPUT "Enter the time at which to read the data(min)";TS 
485   Temp_sought! = CNT(60*VAL(TS)) 
490   EXIT TO,400 
495   END DO 
500   EXIT TO,400 
505   END PROCEDURE 

PROCEDURE: DELTAMULTIPLY 
EXTERNAL: Graph!(),Max_graph_points%(),Miny!,Maxy!,Minx!,Maxx! 
INTEGER: M%,Xor% 
REAL: Min!,Max! 
EXTERNAL: G1,G2 
10 DO 
15   IF G1=1 AND Graph1or2% = 2 THEN EXIT 1 LEVELS 
20   DO 
25   IF XorY% = 2 THEN EXIT 1 LEVELS 
30     Minx! = 1E+29 : Maxx! = -1E+29 
35     EXIT 2 LEVELS 
37   END DO 
40   DO 
45     Miny! = 1E+29 : Maxy! = -1E+29 
50   END DO 
55   END DO 
60 DO 
120   Min! = 1E+26 : Max! = -1E+27 
130   END DO 
500 FOR M% = 1 TO Max_graph_points%(Graph1or2%) 
510   Graph!(Graph1or2%,XorY%,M%) = Graph!(Graph1or2%,XorY%,M%) * Amount! 
520   IF Min! > Graph!(Graph1or2%,XorY%,M%) THEN Min! = 
530   Graph!(Graph1or2%,XorY%,M%) 
540   IF Max! < Graph!(Graph1or2%,XorY%,M%) THEN Max! = 
545   Graph!(Graph1or2%,XorY%,M%) 
550   NEXT M% 
555   DO 
560   IF XorY% = 2 THEN EXIT 1 LEVELS 
570   IF Minx! > Min! THEN Minx! = Min! 
580   IF Maxx! < Max! THEN Maxx! = Max! 
590   EXIT 2 LEVELS 
600   END DO 
610   DO 
620   IF Miny! > Min! THEN Miny! = Min!
630 IF Maxy! < Max! THEN Maxy! = Max!
640 END DO
END PROCEDURE

PROCEDURE: DELTAIn
EXTERNAL: Graph(!), Max_graph_points%, Minx!, Maxx!, Miny!, Mxy!
REAL: Min!, Max!
INTEGER: M%
EXTERNAL: G1,G2
100 DO
110 DO
120 IF G1 = 1 AND Graph1or2% = 2 THEN EXIT
130 Min! = 1E+19
140 Max! = -1E+29
150 EXIT 2 LEVELS
160 END DO
170 DO
180 IF XorY% = 2 THEN EXIT
190 Min! = Minx!
200 Max! = Maxx!
210 EXIT 2 LEVELS
220 END DO
230 DO
240 Min! = Miny!
250 Max! = Maxy!
260 EXIT 2 LEVELS
270 END DO
280 END DO
300 FOR M% = 1 TO Max_graph_points%(Graph1or2%)
310 DO
320 DO
330 IF Graph!(Graph1or2%, XorY%, M%) <> 0 THEN EXIT 1 LEVELS
340 Graph!(Graph1or2%, XorY%, M%) = -1000
350 EXIT 2 LEVELS
360 END DO
370 Graph!(Graph1or2%, XorY%, M%) = LOG(ABS(Graph!(Graph1or2%, XorY%, M%)) * Amount!)
380 END DO
390 IF Min! > Graph!(Graph1or2%, XorY%, M%) THEN Min! = Graph!(Graph1or2%, XorY%, M%)
400 IF Max! < Graph!(Graph1or2%, XorY%, M%) THEN Max! = Graph!(Graph1or2%, XorY%, M%)
410 NEXT M%
500 DO
510 DO
520 IF XorY% = 2 THEN EXIT 1 LEVELS
530 Minx! = Min!
540 Maxx! = Max!
550 EXIT 2 LEVELS
560 END DO
570 DO
PROCEDURE: DATARETRIEVAL
EXTERNAL:
Graph1(),Max_graph_points%(),Minx!,Maxx!,Miny!,Maxy!,Temp_sought!,Bias_found$
EXTERNAL:
Hold_found$, Pulse_found$, Time_sought!, Bias_sought!, Pulse_sought!, Hold_sought!, Temp_type$(),
Temp_found$
EXTERNAL: Time_found$
INTEGER: Position%, Magnet_on%, Graph%, M%, N%
STRING: Data1$[32], Data2$[32], Data3$[32]
REAL: Min!, Max!
REAL: X!, Y1!, Y2!, X1!, Tyme!, R31!, R32!, R3!
REAL: Dummy!, Y1
EXTERNAL: Mintyme!, Maxyme!
STRING: Magnet_on$[?], Dummy$[?], IS$[?], V1$[?], Y1$[?]
INTEGER: Pointer%
REAL ARRAY(?): Graph
EXTERNAL: G!, Elec_thick!
INTEGER: Mistake%
EXTERNAL: Menu_chosen%
20 REM * GRAPH_CHOSEN%
RUN_TYPE%
30 REM *
40 REM * 1 = C-t, G-t  0 = Normal Run
41 REM * 2 = C-V, 1/C**2, 1/C**3  1 = Temp SCAN
42 REM * 3 = G-V  2 = Time SCAN
43 REM * 4 = C-T
44 REM * 5 = G-T
45 REM * 6 = I-V
46 REM * 7 = C-V, 1/C**2  (4140)
47 REM * 8 = DLTS
48 REM * 9 = MOB, Rho vs. Temp
49 REM * 10 = MOB, Mu vs. Temp
80 Error% = 0
90 ON ERROR GOTO 50000
100 DO
110 ON Graph_chosen% GOTO 150, 160, 170, 160, 170, 180, 190, 200, 210, 210
120 CLS: COLOR 7, 0: SET CURSOR 13, 6
130 PRINT *ERROR, the graph_chosen% variable is not defined in the procedure
dataretrieval"
140 STOP
150 Min! = 1E-15: Max! = 10E-03: EXIT
160 Min! = -1E01 : Max! = 2E01 : EXIT
170 Min! = 1E-08 : Max! = 10E-03 : EXIT
180 Min! = -020 : Max! = .020 : EXIT
190 Min! = 1E-15 : Max! = 10E-06: EXIT
200  Min! = 1E-15 : Max! = 2E-09 : N% = 0 : EXIT
210  Min! = -100 : Max! = 100 : EXIT
490 END DO
500 DO
510 DO
520   DO 1 TIMES
530      INPUT #File%,Temp_found$
540      IF INSTR(Temp_found$,”END”) <> 0 THEN Error% = 1:EXIT TO,50000
550      IF INSTR(Temp_type$,(Graph1or2%),”T”) <> 0 OR
560      INSTR(Temp_type$,(Graph1or2%),”R”) <> 0 OR Graph_chosen% = 4 OR Graph_chosen% = 5
570      IF VAL(Mid$(Temp_found$,Position%)) <> Temp_sought! THEN EXIT 2 LEVELS
580 REPEAT
590 DO 1 TIMES
600      INPUT #File%,Time_found$
610      IF INSTR(Temp_type$,(Graph1or2%),”T”) = 0 OR Graph_chosen% = 4 OR
620      Graph_chosen% = 5 OR Graph_chosen% = 8 THEN EXIT 1 LEVELS
630      Position% = INSTR(Time_found$,”=”)+1
640      IF VAL(Mid$(Time_found$,Position%)) <> Time_sought! THEN EXIT 2 LEVELS
650 REPEAT
660 DO 1 TIMES
670      IF Graph_chosen% <> 1 AND Graph_chosen% <> 4 AND Graph_chosen% <> 5
680      AND Graph_chosen% <> 8 THEN EXIT 3 LEVELS
690      INPUT #File%,Bias_found$
700      Position% = INSTR(Bias_found$,”=”)+1
710      IF VAL(Mid$(Bias_found$,Position%)) <> Bias_sought! THEN EXIT 2 LEVELS
720 REPEAT
730 DO 1 TIMES
740      IF Graph_chosen% <> 1 AND Graph_chosen% <> 8 THEN EXIT 3 LEVELS
750      INPUT #File%,Pulse_found$
760      Position% = INSTR(Pulse_found$,”=”)+1
770      IF VAL(Mid$(Pulse_found$,Position%)) <> Pulse_sought! THEN EXIT 2 LEVELS
780 REPEAT
790 REPEAT
800 DO
810      IF Graph_chosen% <> 4 AND Graph_chosen% <> 5 THEN EXIT 1 LEVELS
820 EXIT TO,100
830 REPEAT
840 DO
850      IF INSTR(Data1$,”END”) <> 0 THEN EXIT 1 LEVELS
860 EXIT TO,100
870 REPEAT
880 EXIT TO,100
890 REPEAT
900 EXIT TO,100
980 REPEAT
990 REPEAT
2000 FOR M% = Max_graph_points%(Graph1or2%) TO 1200
2005 IF Graph_chosen% = 9 OR Graph_chosen% = 10 THEN INPUT #File%,Magnet_on$:
2010 INPUT #File%,Data2$:
2020 IF Graph_chosen% > 1 AND Graph_chosen% < 6 THEN INPUT #File%,Data3$:
2025 IF Graph_chosen% = 9 OR Graph_chosen% = 10 THEN INPUT #File%,Data3$:
2030 INPUT #File%,Data1$:
2040 IF INSTR(Data1$,"END") <> 0 OR INSTR(Data2$,"END") <> 0 OR
2045 INSTR(Data3$,"END") <> 0 THEN EXIT 1 LEVELS
2050 X! = VAL(Data1$)
2060 Y1! = VAL(Data2$)
2070 Y2! = VAL(Data3$)
2071 DO
2072 IF Graph_chosen% > 2 AND Graph_chosen% < 6 OR Graph_chosen% = 8 THEN
2073 EXIT 1 LEVELS
2074 IF Y! < Min! OR Y1! > Max! THEN EXIT TO,2005
2075 END DO
2076 DO
2077 IF Graph_chosen% <> 5 THEN EXIT 1 LEVELS
2078 IF Y2! < Min! OR Y2! > Max! THEN EXIT TO,100
2079 END DO
2080 DO
2081 IF Graph_chosen% <> 4 THEN EXIT 1 LEVELS : PRINT Y!,2081
2082 IF Y! < Min! OR Y! > Max! THEN EXIT TO,100
2083 END DO
2084 DO
2085 IF Graph_chosen% <> 3 THEN EXIT 1 LEVELS
2086 IF Y2! < Min! OR Y2! > Max! THEN EXIT TO,2005
2087 END DO
2089 DO 1 TIMES
2090 ON Graph_chosen% GOTO 2100,2100,2200,2300,2300,2100,2100,2100,2600,2600,3000,3300
2100 DO
2110 Graph!(Graph1or2%,1,M%) = X!
2120 Graph!(Graph1or2%,2,M%) = Y1!
2130 EXIT 2 LEVELS
2140 REPEAT
2200 DO
2210 Graph!(Graph1or2%,1,M%) = X!
2220 Graph!(Graph1or2%,2,M%) = Y2!
2230 EXIT 2 LEVELS
2240 REPEAT
2300 REM
2310 DO
2400 Graph!(Graph1or2%,1,M%) = X!
2410 DO
2420 IF Graph_chosen% <> 4 THEN EXIT 1 LEVELS
2430 Graph!(Graph1or2%,2,M%) = Y1!
2440 END DO
2450 DO
IF Graph_chosen% <> 5 THEN EXIT 1 LEVELS
2470       Graph!(Graph1or2%,2,M%) = Y2!
2480       END DO
2500       Max_graph_points%(Graph1or2%) = Max_graph_points%(Graph1or2%) + 1
2520       EXIT 3 LEVELS
2530       REPEAT
2560       REM
2610       DO 1 TIMES
2620       IF X! <> Mintyme! THEN EXIT 1 LEVELS
2630       DO
2635       IF INSTR(Data2$,*END*) <> 0 AND INSTR(Data1$,*END*) <> 0 THEN EXIT
4 LEVELS
2636       IF INSTR(Data2$,*END*) <> 0 OR INSTR(Data1$,*END*) <> 0 THEN EXIT
TO,100
2640       IF Y! > Min! AND Y! < Max! THEN EXIT 1 LEVELS
2650       INPUT #File%,Data2$,Data1$:Y1! = VAL(Data2$):EXIT TO,2630
2670       REPEAT
2680       Graph!(Graph1or2%,2,M%) = Y1!
2690       N% = 2
2700       EXIT TO,2010
2710       REPEAT
2750       DO 1 TIMES
2760       IF X! <> Maxtyme! THEN EXIT 1 LEVELS
2770       DO
2780       IF Y! > Min! AND Y! < Max! THEN EXIT 1 LEVELS
2790       INPUT #File%,Data2$:INPUT #File%,Data1$:Y1! = VAL(Data2$):GOTO 2770
2810       REPEAT
2820       Graph!(Graph1or2%,2,M%) = Graph!(Graph1or2%,2,M%) - Y1!
2830       N% = 0
2840       Max_graph_points%(Graph1or2%) = M% + 1
2850       DO 1 TIMES
2860       Position% = INSTR(Temp_found$,*;*) + 1
2870       Graph!(Graph1or2%,1,M%) = VAL(MIDS(Temp_found$,Position%))
2880       REPEAT
2890       DO
2900       INPUT #File%,Data1$
2910       IF INSTR(Data1$,*END*) <> 0 THEN EXIT TO,4955
2920       REPEAT
2930       REPEAT
2935       EXIT TO,2010
3000       DO
3010       DO
3020       IF INSTR(Magnet_on$,*1*) <> 0 THEN EXIT 1 LEVELS
3030       INPUT #File%, Dummy$,Magnet_on$,Data2$,Data3$,Data1$
3040       REPEAT
3050       Y2! = VAL(Data3$)
3060       Y1! = VAL(Data2$)
3070       Graph!(Graph1or2%,2,M%) = Y2! / Y1! * (Elec_thick!/G!)
3080       DO
3090       IF INSTR(Temp_type$(Graph1or2%),"T") <> 0 THEN EXIT 1 LEVELS
3100       Position% = INSTR(Temp_found$,*;*) + 1
Graph1(Graph1or2%,1,M%) = VAL(MID$(Temp_foun$,Position$))
OPEN "BUCKET" FOR APPEND AS #3
PRINT #3,Graph1(Graph1or2%,2,M%)
CLOSE #3
EXIT TO,3700
REPEAT
DO
Position$ = INSTR(Time_foun$,"=") + 1
Graph1(Graph1or2%,1,M%) = VAL(MID$(Time_foun$,Position$))
OPEN "BUCKET" FOR APPEND AS #3
PRINT #3,Graph1(Graph1or2%,2,M%)
CLOSE #3
EXIT TO,3700
REPEAT
REPEAT
DO
IF INSTR(Magnet_on$,"2") <> 0 THEN Mistake% = 1 : EXIT TO,60000
R31! = X!/Y1!
END DO
DO
INPUT #File%,Magnet_on$,Data2$,Data3$,Data1$
IF INSTR(Magnet_on$,"1") <> 0 THEN Mistake% = 1 : EXIT TO,60000
X! = VAL(Data2$)
Y1! = VAL(Data1$)
R32! = Y1!/X!
END DO
DO
R3! = ABS(R32! - R31!)
Graph1(Graph1or2%,2,M%) = R3!/(Graph1(Graph1or2%,2,M%) * 3.4E-05)
END DO
DO
END DO
DO
END TO,60000
IF INSTR(Temp_type$(Graph1or2%),"T") <> 0 THEN EXIT 1 LEVELS
Position$ = INSTR(Temp_foun$,":" + 1
Dummy! = VAL(MID$(Temp_foun$,Position$))
EXIT TO,3700
REPEAT
DO
Position$ = INSTR(Time_foun$,":" + 1
Dummy! = VAL(MID$(Time_foun$,Position$))
END DO
IF Dummy! <> Graph1(Graph1or2%,1,M%) THEN Mistake% = 2 : EXIT TO,3700
REPEAT
REPEAT
DO
END DO
INPUT #File%,Data1$
IF INSTR(Data1$,"END") <> 0 THEN EXIT 1 LEVELS
REPEAT
DO
  INPUT #File%,Temp_found$
  IF INSTR(Temp_found$,"TEMP") <> 0 THEN EXIT 1 LEVELS
  IF INSTR(Temp_found$,"END") <> 0 THEN EXIT 4 LEVELS
  REPEAT
  INPUT #File%,Time_found$
  EXIT 1 LEVELS
REPEAT
REPEAT
DO
  IF Graph_chosen% = 1 THEN EXIT 1 LEVELS
  IF Graph_chosen% = 2 THEN EXIT 1 LEVELS
  IF Minx! > Graph!(Graph1or2%,1,M%) THEN Minx! = Graph!(Graph1or2%,1,M%)
  IF Maxx! < Graph!(Graph1or2%,1,M%) THEN Maxx! = Graph!(Graph1or2%,1,M%)
  IF Miny! > Graph!(Graph1or2%,2,M%) THEN Miny! = Graph!(Graph1or2%,2,M%)
  IF Maxy! < Graph!(Graph1or2%,2,M%) THEN Maxy! = Graph!(Graph1or2%,2,M%)
  IF Graph_chosen% = 8 THEN EXIT TO,500
END DO
NEXT M%
DO
  IF Graph_chosen% = 4 OR Graph_chosen% = 5 THEN EXIT TO,100
  IF Graph_chosen% < 9 OR Graph_chosen% > 10 THEN
  Max_graph_points%(Graph1or2%) = M% - 1 ELSE Max_graph_points%(Graph1or2%) = M%
  IF Run_type% = 0 THEN EXIT 1 LEVELS
  DO
    IF Run_type% <> 1 THEN EXIT 1 LEVELS
    DO
    DO
      IF Graph_chosen% = 6 OR Graph_chosen% > 8 THEN EXIT 1 LEVELS
    END DO
  END DO
  END DO
  IF INSTR(Data1$,"TEMP") = 0 THEN EXIT 1 LEVELS
  Position% = INSTR(Data1$,"=") + 1
  Temp_sought! = VAL(MID$(Data1$,Position%))
  EXIT 4 LEVELS
REPEAT
DO
  IF Graph_chosen% <> 2 THEN EXIT 1 LEVELS
  IF INSTR(Data3$,"TEMP") = 0 THEN EXIT 1 LEVELS
  Position% = INSTR(Data3$,"=") + 1
  Temp_sought! = VAL(MID$(Data3$,Position%))
  EXIT 4 LEVELS
REPEAT
DO
  IF Graph_chosen% <> 2 THEN EXIT 1 LEVELS
  IF Graph!(Graph1or2%,2,M%-1) = 0 THEN Error% = 1 : EXIT TO,50000
  Error% = 1 : EXIT 3 LEVELS
REPEAT
DO
  IF Graph_chosen% < 5 THEN EXIT 1 LEVELS
  INPUT #File%,Data1$
5340 END DO
5350 IF INSTR(Data1$,"TIME") = 0 THEN EXIT 1 LEVELS
5360 Position% = INSTR(Data1$,"=") + 1
5370 Time_sought! = VAL(MIDS(Data1$,Position%))
5380 EXIT 3 LEVELS
5390 REPEAT
5400 IF Graph!(Graph1or2%,2,M%-1) = 0 THEN Error% = 1 : EXIT TO,50000
5410 Error% = 1 : EXIT 2 LEVELS
5420 REPEAT
5430 EXIT
10006 EXIT
50000 REM
50010 COLOR 7,0,0;SET CURSOR 24,6
50500 PRINT "The procedure DATARETRIEVAL is bombing. This is error ";ERR;" from line"
"
50510 STOP
60000 REM
60001 FOR M% = 2 TO 24
60002 LOCATE M%,1 : PRINT SPC(79)
60003 NEXT M%
60010 IF Graph_chosen% = 4 AND Graph!(Graph1or2%,2,1) <> 0 THEN Error% =
60015 IF Max_graph_points%(Graph1or2%) = Max_graph_points%(Graph1or2%) - 1 : EXIT
60020 IF Graph_chosen% = 8 AND Graph!(Graph1or2%,2,2) <> 0 THEN Error% =
60025 IF Max_graph_points%(Graph1or2%) = Max_graph_points%(Graph1or2%) - 1 : EXIT
60030 IF Graph_chosen% > 8 AND Graph_chosen% < 11 AND Graph!(Graph1or2%,2,1) <>
60035 IF Error% = 0
60040 IF Mistake% = 1 THEN GOTO 60500
60050 IF Mistake% = 2 THEN GOTO 60600
60060 IF Graph_chosen% = 8 AND Graph!(Graph1or2%,2,1) <> 0 THEN Error% = 0 :
60070 IF Graph_chosen% = 8 AND Graph!(Graph1or2%,2,2) <> 0 THEN Error% = 0 :
60100 FOR N% = 2 TO 23
60110 SET CURSOR 2,0 : PRINT SPC(80)
60120 NEXT N%
60130 COLOR 7,0 : SET CURSOR 13,6
60140 PRINT "The parameters you have set are not in the directory chosen. Please choose new parameters"
60150 Tyne! = TIMER + 5
60160 IF Tyne! > TIMER THEN GOTO 60160
60200 EXIT
60500 DO
60510 CLS : COLOR 2,0,0 : SET CURSOR 13,6
60520 PRINT "The magnet data does not line up correctly. This is a major bomb. Sorry"
60530 END
60540 REPEAT
60600 DO
60610 CLS : COLOR 7,0,0 : SET CURSOR 13,6
60620 PRINT "The temperature does not line up from the Rho to Mu data. This is a major
bomb. Sorry

60630  END
60640  REPEAT
61200  EXIT
END PROCEDURE

PROCEDURE: DATAPRINT
INTEGER: N%
  50 FOR N% = 1 TO 23
  60   SET CURSOR N%,0 : PRINT SPC(80)
  70 NEXT N%
100 COLOR 2,0:SET CURSOR 13,23
110 PRINT "The ";
120 COLOR 3,0:PRINT "DATA";
130 COLOR 2,0:PRINT " is now being loaded"
140 EXIT
END PROCEDURE

PROCEDURE: PARAMETERSET
EXTERNAL: Bias_sought!, Hold_sought!, Pulse_sought!, Mintyme!, Maxtyme!
INTEGER: M%, Row%
REAL: L
STRING: Check$[10], Dummy$[?], Dummy2$[?]
REAL: Min!
REAL#: Max!
  38 REM * The numbers correspond to graph_type%
  39 REM *
  40 REM * 1 = C-t, G-t
  41 REM * 2 = C-G-V
  42 REM * 3 = C-G
  43 REM * 4 = DLTS
  44 REM *
100 Row% = 2:GOSUB 10000
110 COLOR 2,0,0
120 Row% = 5
200 DO 1 TIMES
210 IF Graph_type% <> 1 AND Graph_type% <> 3 AND Graph_type% <> 4 THEN EXIT L LEVELS
220   SET CURSOR Row%,6
230   INPUT "Enter the BIAS Voltage"; Dummy$
235   Bias_sought! = VAL(Dummy$)
240   Row% = Row% + 2
250   REPEAT
300 DO 1 TIMES
310 IF Graph_type% <> 1 AND Graph_type% <> 4 THEN EXIT 1 LEVELS
320   SET CURSOR Row%,6
330   INPUT "Enter the PULSE Voltage"; Dummy$
335   Pulse_sought! = VAL(Dummy$)
340   Row% = Row% + 2
350   REPEAT
400 DO 1 TIMES
410  IF Graph_type% <> 1 AND Graph_type% <> 4 THEN EXIT 1 LEVELS
420  SET CURSOR Row%,6
430  INPUT "Enter the HOLD Time in seconds";Dummy$
435  Hold_sought! = VAL(Dummy$)
440  Row% = Row% + 2
450  REPEAT
500  DO 1 TIMES
510  IF Graph_type% <> 4 THEN EXIT 1 LEVELS
520  SET CURSOR Row%,6
530  INPUT "Enter the box car times in seconds (min, max)";Dummy$;Dummy2$
535  Minyme! = VAL(Dummy$); Maxyme! = VAL(Dummy2$)
540  Row% = Row% + 2
550  REPEAT
8990  DO
9000  COLOR 7,0,0: SET CURSOR 22,6
9010  INPUT "Do you need to change any of the above (Defaults to NO)";Check$
9020  Check$ = UPPERS$(Check$)
9030  IF INSTR(Check$,"Y") <> 0 THEN EXIT TO,100
9040  END DO
9050  EXIT
10000 FOR M% = Row% TO 23
10010  LOCATE M%,1:PRINT SPC(79)
10020  NEXT M%
10030  RETURN
END PROCEDURE

PROCEDURE: PLOTTING_POINTS
EXTERNAL:
Graph!(),Minx!,Maxx!,Miny!,Mxy!,Max_graph_points%(),Info_pointer%,Directory$
EXTERNAL: Temp$(),Bias$(),Pulse$(),Hold$(),Tyme$(),Rate$(),Slope$(),X_int$
STRING: Line_type$[3],A$[3]
INTEGER: X%Y_pointer%
EXTERNAL: Y_int$(),Range$(),Correlation$,GRANGES()          REAL: X_factor!,Y_factor!,X!
REAL: Y!
STRING: Dummy$[?]
EXTERNAL: Pen_selected%
INTEGER: M%
STRING: FUCK$[?]
REAL: Ymin!,Xmin!
EXTERNAL: G,SYMBOLS(),LINETYPES(),PEN()
REAL ARRAY(?): LINETYPE
EXTERNAL: ONOFF,SPEED$

PROCEDURE: TIMEDELAY()
  REAL ARG: Delay!/OPT=1!
END PROCEDURE

PROCEDURE: CLEARSCREEN()
  INTEGER ARG: Line%
END PROCEDURE
PROCEDURE: EDITDATA()
  STRING ARG: DUMMY$VAR
END PROCEDURE

PROCEDURE: TIMEDELAY
  REAL: Tyme!
  100 Tyme! = TIMER + Delay!
  110 IF Tyme! > TIMER THEN GOTO 110
  120 EXIT
END PROCEDURE

PROCEDURE: CLEARSCREEN
  INTEGER: N%
  100 FOR N% = Line% TO 23
  110 LOCATE N%,1 : PRINT SPC(79)
  120 NEXT N%
  130 EXIT
END PROCEDURE

PROCEDURE: EDITDATA
  REAL: DUMMY!
  INTEGER: Position%,Sign%,N%
  STRING: Saver$[14]
  REAL: Saver!
  INTEGER: Position1%
  STRING: AS[3]
  REAL ARRAY(?): Sng
  50 OPEN "CONVERT" FOR OUTPUT AS #1
  60 CLOSE #1
  80 IF INSTR(DUMMY$,";") <> 0 THEN EXIT 1 LEVELS
  90 CLOSE #1
  100 OPEN "CONVERT" FOR OUTPUT AS #1
  110 DO
  120 IF INSTR(DUMMY$,";") = 0 THEN EXIT 1 LEVELS
  130 DUMMY! = VAL(DUMMY$)
  135 Sign% = SGN(DUMMY!)
  137 Position1% = INSTR(DUMMY$,";")-1
  145 FOR N% = 1 TO 2
  160  DO
  185  IF Sign% = -1 THEN Position% = INSTR(DUMMY$,";") ELSE Position% = 1
  200  Saver$ = MIDS(DUMMY$,Position%,Position1%)
  210  PRINT #1 Saver$;
  220  IF N% = 1 THEN PRINT #1," to ";
  230  Position% = Position1% + 2
  235  Position1% = 20
  240  END DO
  320  DUMMY$ = MIDS(DUMMY$,Position%)
  350  PRINT #1 Saver$;
  330  NEXT N%
  335  EXIT TO.5250
  350 REPEAT
400 DO
410 IF INSTR(DUMMY$,"X") = 0 THEN EXIT 1 LEVELS
420 PRINT "#1,"X"
425 EXIT TO,5250
430 END DO
460 DUMMY! = VAL(DUMMY$)
470 Sign% = SGN(DUMMY!)
480 IF Sign% = -1 THEN DUMMY$ = "*" + MIDS(DUMMY$,INSTR(DUMMY$,"-"),1)
490 IF INSTR(DUMMY$,"E") <> 0 THEN GOTO 5180
491 DO
492 IF DUMMY! <> 0 THEN EXIT 1 LEVELS
493 PRINT "#1",MIDS(DUMMY$,INSTR(DUMMY$,"0"),)
494 EXIT TO,5250
495 REPEAT
500 DO
510 IF DUMMY! > 10 OR DUMMY! < -10 THEN EXIT 1 LEVELS
520 IF Sign% = -1 THEN Saver$ = "*" + MIDS(DUMMY$2, ,4)
530 IF Sign% = 1 THEN Saver$ = MIDS(DUMMY$2, ,4)
540 PRINT "#1",Saver$
550 EXIT TO,5250
560 REPEAT
560 DO
560 IF DUMMY! > 1000 OR DUMMY! < -1000 THEN EXIT 1 LEVELS
560 IF Sign% = -1 THEN Saver$ = "*" + MIDS(DUMMY$2, ,5)
560 IF Sign% = 1 THEN Saver$ = MIDS(DUMMY$2, ,5)
560 PRINT "#1",Saver$
560 EXIT TO,5250
560 REPEAT
5000 DO
5010 DUMMY! = VAL(DUMMY$)
5020 Sign% = SGN(DUMMY!)
5030 IF INSTR(DUMMY$,"E") <> 0 THEN GOTO,5180
5040 DO
5050 IF DUMMY! > 10000 OR DUMMY! < -10000 THEN EXIT 1 LEVELS
5060 IF Sign% = -1 THEN Saver$ = "*" + MIDS(STR$(ABS(DUMMY! - .00001)),2)
5070 IF Sign% = 1 THEN Saver$ = MIDS(STR$(DUMMY! + .00001),2)
5080 PRINT "#1 USING \\ "Saver$
5090 EXIT 2 LEVELS
5100 REPEAT
5110 DO
5120 IF Sign% = -1 THEN Saver! = DUMMY! - 1
5130 IF Sign% = 1 THEN Saver! = DUMMY! + 1
5140 PRINT "#1 USING \\ "Saver!
5150 EXIT 2 LEVELS
5160 REPEAT
5170 REPEAT
5180 DO
5190 Position% = INSTR(DUMMY$,"E")
5192 Position1% = 4
5193 IF Position% < 5 THEN Position1% = Position%
5200 IF Sign% = 1 THEN Saver$ = MIDS(DUMMY$2, ,4) +
MIDS(DUMMYS,Position%)  
5205 IF Sign% = -1 THEN Saver$ = ":" + MIDS(DUMMYS,2,Position1%) +
MIDS(DUMMYS,Position%)  
5210 PRINT #1,Saver$
5220 CLOSE #1
5230 EXIT TO,5250
5240 REPEAT
5250 CLOSE #1
5260 OPEN "CONVERT" FOR INPUT AS #1
5265 CLEAR (DUMMYS)
5270 INPUT #1,DUMMYS
5280 CLOSE #1
5290 KILL "CONVERT"
5300 EXIT
END PROCEDURE
70 CLOSE #3
80 OPEN "COM1";9600,N,8,1,RS,CS65535,DS,CD* FOR OUTPUT AS #3
200 DO
210 X_factor! = (6634) / (Maxx! - Minx!)
220 Y_factor! = (5969) / (Miny! - Maxy!)
225 Ymin! = Miny!
226 Xmin! = Minx!
230 END DO
950 DO
955 CLEARSCREEN (2):SET CURSOR 0,70:COLOR 4,7:PRINT G
960 COLOR 2,0,0 : SET CURSOR 13,32
970 PRINT "PLOTTING POINTS"
980 END DO
1000 DO
1010 PRINT #3,"IN ; IP381,1016,6350,7650 ; ";
1020 PRINT #3,"IW381,1016,6350,7650 ; ";
1030 PRINT #3,"SP" PEN(G) ; SI2,2 ; SM"SYMBLS(G)" ; LT"LINETYPES(G)" ; DI 0,1 ; ";
1040 END DO
1100 FOR N% = 1 TO Max_graph_points%(Graph1or2%) STEP 1
1110 DO
1120 X! = Graph!(Graph1or2%,1,N%) * X_factor! + 1016 - Xmin! * X_factor!
1130 Y! = +Graph!(Graph1or2%,2,N%) * Y_factor! +6350 - Ymin! * Y_factor!
1132 IF X! > 32767 OR X! < -32768 THEN EXIT TO,1210
1133 IF Y! > 32767 OR Y! < -32768 THEN EXIT TO,1210
1140 END DO
1150 DO
1160 IF N% <> 1 THEN EXIT 1 LEVELS
1170 PRINT #3,"PA PU "Y!","X!" PD ;"
1180 END DO
1190 PRINT #3,"PA "Y!","X!" ; PD ;"
1200 IF VAL(SPEEDS)=0 THEN TIMEDELAY(1) ELSE TIMEDELAY(VALS(Speeds))
1210 NEXT N%
1220 IF ONOFF=0 THEN GOTO 4000
2000 DO
2010 Info_pointer% = Info_pointer% + 1
2020 IF info_pointer% > 5 THEN EXIT 1 LEVELS
2030  Y_pointer% = 1762 + 1179 * (Info_pointer% - 1)
2040  PRINT #3,"JW ; SM ; SP"PEN(G)" ; SI .16,2 ; DR0,1 ; "*
2050  DO
2060  PRINT #3,"PA PU 7754,"Y_pointer%" ; LB"SYMBS$(G) ; CHR$(3)
2065  TIMEDELAY(5)
2070  END DO
2080  DO
2090  PRINT #3,"PA PU 7954,"Y_pointer%" ; LB"Directory$(Graph1or2%,1) ; CHR$(3)
2095  TIMEDELAY(5)
2100  IF INSTR(Directory$(Graph1or2%,2),"NO") <> 0 THEN EXIT 1 LEVELS
2110  PRINT #3,"PA PU 7954,"Y_pointer%+480" ; LB"Directory$(Graph1or2%,2) ; CHR$(3)
2115  TIMEDELAY(5)
2120  END DO
2130  DO
2133  Dummy$ = "" + Temp$(Graph1or2%)
2135  EDITDATA(Dummy$)
2140  PRINT #3,"PA PU 8154,"Y_pointer%" ; LB"Dummy$ ; CHR$(3)
2145  TIMEDELAY(5)
2150  END DO
2152  EDITDATA(Tyme$(Graph1or2%))
2153  PRINT #3,"pa pu 8354,"Y_pointer%" ; LB"Tyme$(Graph1or2%) ; CHR$(3)
2154  TIMEDELAY(5)
2155  END DO
2159  DO
2165  EDITDATA(Bias$(Graph1or2%))
2170  PRINT #3,"PA PU 8554,"Y_pointer%" ; LB"Bias$(Graph1or2%) ; CHR$(3)
2175  TIMEDELAY(5)
2180  END DO
2185  DO
2195  EDITDATA(Pulse$(Graph1or2%))
2200  PRINT #3,"PA PU 8754,"Y_pointer%" ; LB"Pulse$(Graph1or2%) ; CHR$(3)
2205  TIMEDELAY(5)
2210  END DO
2215  DO
2225  EDITDATA(Hold$(Graph1or2%))
2230  PRINT #3,"PA PU 8954,"Y_pointer%" ; LB"Hold$(Graph1or2%) ; CHR$(3)
2235  TIMEDELAY(5)
2240  END DO
2245  DO
2255  EDITDATA(Rate$(Graph1or2%))
2260  PRINT #3,"PA PU 9154,"Y_pointer%" ; LB"Rate$(Graph1or2%) ; CHR$(3)
2265  TIMEDELAY(5)
2270  END DO
2275  DO
2285  EDITDATA(Slope$(Graph1or2%))
2290 PRINT #3,"PA PU 9554,"Y_pointer%" ; LB*Slope$(Graph1or2%) ; CHR$(3)
2295 TIMEDELAY(5)
2300 END DO
2310 DO
2315 EDITDATA(X_int$(Graph1or2%))
2320 PRINT #3,"PA PU 9554,"Y_pointer%" ; LB*X_int$(Graph1or2%) ; CHR$(3)
2325 TIMEDELAY(5)
2330 END DO
2340 DO
2345 EDITDATA(Y_int$(Graph1or2%))
2350 PRINT #3,"PA PU 9554,"Y_pointer%" ; LB*Y_int$(Graph1or2%) ; CHR$(3)
2355 TIMEDELAY(5)
2360 END DO
2370 DO
2375 EDITDATA(Range$(Graph1or2%))
2380 PRINT #3,"PA PU 10154,"Y_pointer%" ; LB*Range$(Graph1or2%) ; CHR$(3)
2385 TIMEDELAY(5)
2390 END DO
2400 DO
2405 EDITDATA(Correlation$(Graph1or2%))
2410 PRINT #3,"PA PU 10354,"Y_pointer%" ; LB*Correlation$(Graph1or2%) ; CHR$(3)
2415 TIMEDELAY(5)
2420 END DO
3000 END DO
4000 PRINT #3,"SP0 ; *
4010 CLOSE #3
4020 EXIT
30000 FOR N% = 2 TO 23
30010 LOCATE N%,1 : PRINT SPC(79)
30020 NEXT N%
30030 RETURN
END PROCEDURE

PROCEDURE: GETDATA
EXTERNAL: Graph1(),Minx!,Maxx!,Miny!,Maxy!,Max_graph_points%
INTEGER: M%,N%
REAL: Dummy,Dummy!
EXTERNAL: G1,G2
100 DO
110 IF G1=1 AND Graph% <> 1 THEN EXIT 1 LEVELS
120 Minx! = 1E+29
130 Maxx! = -1E+29
140 Miny! = 1E+29
150 Maxy! = -1E+29
160 END DO
500 OPEN "BUCKET" FOR INPUT AS #3
510 FOR M% = 1 TO Max_graph_points%(Graph%)
520 INPUT #3,Graph!(Graph%,1,M%) , Dummy!, Graph!(Graph%,2,M%)
530 IF Minx! > Graph!(Graph%,1,M%) THEN Minx! = Graph!(Graph%,1,M%)
540 IF Maxx! < Graph!(Graph%,1,M%) THEN Maxx! = Graph!(Graph%,1,M%)
550 IF Miny! > Graph!(Graph%,2,M%) THEN Miny! = Graph!(Graph%,2,M%)

560 IF Maxy! < Graph!(Graph%2,M%) THEN Maxy! = Graph!(Graph%2,M%)
570 NEXT N%
1000 KILL "BUCKET"
END PROCEDURE

PROCEDURE: PLOTTERAXES
EXTERNAL: Minx!,Miny!,Maxx!,Maxy!,Xtitle$,Ytitle$
STRING: Grid$[7],Name$[80],Dummy$[16]
REAL: Dummy!,Number!
INTEGER: M%,Maxloop%,Y_POINTER%,X_POINTER%,N%,Mini_pads%
EXTERNAL: ONOFF,lafo_pointer%

PROCEDURE: TIMEDELAY()
    REAL ARG: Delay!/OPT=10!
END PROCEDURE

PROCEDURE: CLEARSCREEN()
    INTEGER ARG: Line%
END PROCEDURE

PROCEDURE: EDITDATA()
    REAL ARG: DUMMY!
    STRING ARG: DUMMYS/VAR
END PROCEDURE

PROCEDURE: TIMEDELAY
    REAL: Tyme!
    100 Tyme! = TIMER + Delay!
    110 IF Tyme! > TIMER THEN GOTO 110
    120 EXIT
END PROCEDURE

PROCEDURE: CLEARSCREEN
    INTEGER: N%
    100 FOR N% = Line% TO 24
    110 LOCATE N%,1 : PRINT SPC(79)
    120 NEXT N%
    130 EXIT
END PROCEDURE

PROCEDURE: EDITDATA
    REAL: SAVER!
    INTEGER: Sign%
    STRING: Saver$[7]
    INTEGER: Position%,Position1%
    EXTERNAL: Mini_pads%
    REAL: DUMY!
    1 ON ERROR GOTO 1320!
    100 CLOSE #1
    110 OPEN "CONVERT" FOR OUTPUT AS #1
    111 DO
112 IF DUMMY! < .999 AND DUMMY! > -.999 THEN EXIT 1 LEVELS
113 IF DUMMY! > 10000 OR DUMMY! < -10000 THEN EXIT 1 LEVELS
117 IF ABS(CINT(DUMMY!)-DUMMY!)<.0009 THEN
118 DUMMY! = CINT(DUMMY!); DUMMY$ = STR$(DUMMY!); EXIT TO, 130
119 END DO
120 DUMMY$ = STR$(DUMMY!)
130 Sign% = SGN(DUMMY!)
135 DO
136 IF Sign% <> -1 THEN EXIT 1 LEVELS
137 DUMMY$ = MIDS(DUMMY$, INSTR(DUMMY$, "-"))
138 END DO
200 DO
205 IF Mini_pads% = 1 THEN EXIT 1 LEVELS
210 IF DUMMY! > .000999 OR DUMMY! < -.000999 THEN EXIT 1 LEVELS
220 PRINT #1 USING "#.##" ^ ^ ^* VAL(DUMMY$)
230 CLOSE #1
240 EXIT TO, 1250
250 END DO
500 DO
510 IF DUMMY! > 10 OR DUMMY! < -10 THEN EXIT 1 LEVELS
515 IF INSTR(DUMMY$, "0.000") <> 0 THEN DUMMY$ = "0"
520 Saver$ = MIDS(DUMMY$, 1, 6)
530 PRINT #1, Saver$
540 EXIT TO, 1250
550 REPEAT
560 DO
570 IF DUMMY! > 1000 OR DUMMY! < -1000 THEN EXIT 1 LEVELS
580 Saver$ = MIDS(DUMMY$, 1, 6)
590 PRINT #1, Saver$
600 EXIT TO, 1250
610 REPEAT
620 DO
630 IF DUMMY! > 9999.99 OR DUMMY! < -9999.99 THEN EXIT 1 LEVELS
640 Saver$ = MIDS(DUMMY$, 1, 6)
650 PRINT #1, Saver$
660 EXIT TO, 1250
670 REPEAT
1000 DO
1010 Position% = INSTR(DUMMY$, "E")
1020 IF Position% = 0 THEN EXIT 1 LEVELS
1025 DO
1030 IF Position% <= 5 THEN EXIT 1 LEVELS
1040 IF Sign% = -1 THEN Saver$ = MIDS(DUMMY$, INSTR(DUMMY$, "-"), 5) + MIDS(DUMMY$, Position%) ELSE Saver$ = MIDS(DUMMY$, 1, 5) + MIDS(DUMMY$, Position%)
1050 PRINT #1, Saver$
1060 EXIT TO, 1250
1070 REPEAT
1080 PRINT #1 USING "#.##" ^ ^ ^* DUMMY!
1090 EXIT TO, 1250
1100 REPEAT
1200 DO
1210  SAVER! = VAL(DUMMYS)
1220  PRINT #1 USING "#.### ^ ^ ^ ^ S AVER!"
1230  EXIT TO, 1250
1240  REPEAT
1250  CLOSE #1
1260  OPEN "CONVERT" FOR INPUT AS #1
1270  INPUT #1,DUMMYS
1280  CLOSE #1
1290  KILL "CONVERT"
1300  IF INSTR(DUMMYS,"E+00") <> 0 THEN DUMMYS = "0"
1310  EXIT
1320  PRINT "Error number "ERR" line "ERL"
1330  STOP

END PROCEDURE
90  Info_pointer% = 0
100  CLEARSCREEN(2)
110  COLOR 2,0,0 : SET CURSOR 13,6
120  INPUT "Do you want the grid plotted (Defaults to NO)"; GridS
130  GridS = UPPERS(GridS)
140  CLEARSCREEN(12)
150  COLOR 3,0,0 : SET CURSOR 13,36
160  PRINT "PLOTTING"
190  CLOSE #3
200  DO
210  OPEN "COM1 : 9600,N,8,1,RS,CS65535,DS,CD" FOR OUTPUT AS #3
220  PRINT #3,"IN ; IP 381,1016,6350,7650 : "
230  END DO
300  DO
310  PRINT #3," SP1 ; PA PU 381,1016 PD 381,7650,6350,7650,6350,1016,381,1016"
320  PRINT #3," ; PU ;"
330  TIMEDELAY(5)
340  END DO
400  DO
410  Y_POINTER% = CINT(1016 + (6634 - (LEN(Xtitle$) * 172)) / 2)
420  PRINT #3," SP2 ; SL3,3 ; DR 0,1 ;"
430  PRINT #3," PA PU 6858,"Y_POINTER%" ; LB*Xtitle$;CHR$(3)
440  TIMEDELAY(7)
450  END DO
500  DO
510  X_POINTER% = CINT(6340 - (5969 - (LEN(Ytitle$) * 172)) / 2)
520  PRINT #3," SP2 ; SL3,3 ; DR-1,0 ;"
530  PRINT #3," PA PU "X_POINTER%",170 ; LB*Ytitle$;CHR$(3)
540  TIMEDELAY()
550  END DO
600  DO
610  Y_POINTER% = CINT(1016 + (6634 - (LEN(Ytitle$) * 179)) / 2)
620  PRINT #3," SP1 ; SL3,3 ; DR0,1 ;"
630  PRINT #3," PA PU 254,"Y_POINTER%" ; LB*Ytitle$;CHR$(3)
640  TIMEDELAY()
650  END DO
700 DO
710 PRINT #3,"SP1 ; TL.9 ;"
720 FOR N% = 1 TO 9
730 X_POINTER% = CINT(381 + N% * 597)
740 PRINT #3, "PA PU "X_POINTER%;970 ; XT ;"
750 TIMEDELAY(1)
760 NEXT N%
770 FOR N% = 1 TO 9
780 Y_POINTER% = CINT(1016 + N% * 663)
790 PRINT #3, "PA PU 6350, "Y_POINTER%; YT;"
800 TIMEDELAY(1)
810 NEXT N%
820 END DO
830 DO
840 IF INSTR(Grid$,"Y") = 0 THEN EXIT 1 LEVELS
850 PRINT #3,"L:1,0.5 ; SP2 ;"
860 FOR N% = 1 TO 9
870 X_POINTER% = CINT(381 + N% * 597)
880 Y_POINTER% = CINT(1016 + N% * 663)
890 PRINT #3, "PA PU "X_POINTER%;1016 PD "X_POINTER%;7650 ;"
900 TIMEDELAY(25)
910 PRINT #3, "PA PU 381,"Y_POINTER%; PD 6350,"Y_POINTER%; ;"
920 TIMEDELAY(25)
930 NEXT N%
940 END DO
950 DO
960 PRINT #3,"SP2 ; SI.2,2 ; DI0,1 ;"
970 FOR N% = 0 TO 10 STEP 2
980 IF Minx! > .00099 OR Minx! < -.00099 THEN Mini_pads% = 1 ELSE Mini_pads% = 0
990 Number! = Minx! + (Maxx! - Minx!) * (N% / 10)
1000 EDITDATA(Number!, Dummy$)
1010 Y_POINTER% = CINT(980 + N% * 663 - LEN(Dummy$) * 36)
1020 IF N% = 10 THEN Y_POINTER% = CINT(7585 - LEN(Dummy$) * 72)
1030 PRINT #3,"PA PU 6530,"Y_POINTER%; ; LB"Dummy$;CHR$(3)
1040 TIMEDELAY(2)
1050 NEXT N%
1060 END DO
1070 DO
1080 PRINT #3, "SP2 ; SI.2,2 ; DI0,1 ;"
1090 FOR N% = 0 TO 10 STEP 2
1100 IF Miny! > .00099 OR Miny! < -.00099 THEN Mini_pads% = 1 ELSE Mini_pads% = 0
1110 Number! = Miny! + (Maxy! - Miny!) * (N%/10)
1120 EDITDATA(Number!, Dummy$)
1130 X_POINTER% = CINT(6355 - N% * 597)
1140 Y_POINTER% = CINT(584 - LEN(Dummy$) * 36)
1150 PRINT #3, "PA PU "X_POINTER%;"Y_POINTER%; LB"Dummy$;CHR$(3)
1160 TIMEDELAY(2)
1170 NEXT N%
1180 END DO
1190 NEXT N%
1200 END DO
1210 NEXT N%
1220 END
1510 IF ONOFF=0 THEN GOTO 1710
1600 DO
1610 PRINT #3,"PU ; SL.16.; DR0.1 ; SP2 ;"
1620 RESTORE,60100
1630 READ Maxloop%
1640 FOR N% = 1 TO Maxloop%
1650 READ Name$
1660 X_POINTER% = CINT(7554 + N% * 200)
1670 PRINT #3, "PA PU "X_POINTER%,508 ; LB"Name$ ; CHRS(3)
1680 TIMEDELAY(4)
1690 NEXT N%
1700 END DO
1710 PRINT #3,"SP0;"
1720 CLOSE #3
1730 EXIT
60000 STOP
60100 DATA 14
60110 DATA "SYMBOL","DIRECTORIES","TEMPERATURE","TIME","BIAS","PULSE HIGH"
60120 DATA "HOLD TIME","RATE WINDOW","GRAPH RANGE","SLOPE","X-INTERCEPT"
60130 DATA "Y-INTERCEPT","RANGE","CORRELATION"
END PROCEDURE

PROCEDURE: SAVEDATA
EXTERNAL:
Graph!(),Max_graph_points%(),Graph_name$(),Xtitle$,$title$,Pulse$(),Directory$(),Temp$()
EXTERNAL: Tyme$(),Bias$(),Hold$(),Rate$(),Grange$(),Slope$(),X_int$,Y_int$
EXTERNAL: Range$()
STRING: Save_file$[15]
INTEGER: Line%,Choice%,Position%,N%
EXTERNAL: Graph_chosen%,Menu_chosen%,Slope!(),Y_int!,X_int!
EXTERNAL: Correlation!(),Min!,Max!,Correlation$(),Y_in!,Min!,Max!
INTEGER: M%
REAL ARRAY(?): RAuge!
STRING: CHOICES$[2]
EXTERNAL: GFLAG,Saveflag
   1 ON ERROR GOTO 60000
100 DO
110 COLOR 6,0 : SET CURSOR 20,6
120 PRINT "Enter the driver and name you want to save the file under"
130 SET CURSOR 21,6 : PRINT "(Defaults to B:";Graph_name$(Menu_chosen%,Graph_chosen%)) ";
140 INPUT Save_file$
150 IF Save_file$ = "" THEN Save_file$ = "B:" +
Graph_name$(Menu_chosen%,Graph_chosen%)
160 END DO
170 Line% = 19 : GOSUB 20000
175 Choice% = Graph%
178 IF Saveflag = 1 THEN Saveflag = 0: GOTO 300
180 DO
200  SET CURSOR 20.6 : COLOR 2,0 : INPUT "Which do you want to save, (1) G1 or (2) G2 graph":CHOICES
202  Choice% = VAL(CHOICES):GFLAG = Choice%
203  SET CURSOR 0,70 : COLOR 2,7 : PRINT Choice% : COLOR 7,0
210  IF Choice% < 1 OR Choice% > 2 THEN EXIT TO,170
220  EXIT 1 LEVELS
230  REPEAT
300  Line% = 19 : GOSUB 20000
304  KILL Save_file$
305  DO
310  Position% = LEN(Save_file$)
320  SET CURSOR 17,10 : COLOR 5,0 : PRINT "SAVING FILE UNDER ":
330  SET CURSOR 17,28 : COLOR 5,0 : PRINT MIDS(Save_file$,1,Position%)
335  CLOSE
340  OPEN Save_file$ FOR OUTPUT AS #1
350  PRINT #1,Graph_name$(Menu_chosen%, Graph_chosen%)
360  PRINT #1,Xtitle$
370  PRINT #1,Ytitle$
380  PRINT #1,Directory$(Choice%, 1) :" Directory$(Choice%, 2)
390  PRINT #1,Temp$(Choice%)
400  PRINT #1,Tyme$(Choice%)
410  PRINT #1,Bias$(Choice%)
420  PRINT #1,Pulse$(Choice%)
430  PRINT #1,Hold$(Choice%)
440  PRINT #1,Rate$(Choice%)
450  PRINT #1,Grange$(Choice%)
460  PRINT #1,Calc!(Choice%)
470  PRINT #1,X_int!(Choice%)
480  PRINT #1,Y_int!(Choice%)
490  PRINT #1,Correlation!(Choice%)
495  PRINT #1,Min!(Choice%)
496  PRINT #1,Max!(Choice%)
500  PRINT #1,Max_graph_points%(Choice%)
510  FOR N% = 1 TO Max_graph_points%(Choice%)
520  PRINT #1,Graph!(Choice%, 1, N%) :": Graph!(Choice%, 2, N%)
530  NEXT N%
540  END DO
600  Line% = 17 : GOSUB 20000
610  CLOSE #1
1000 EXIT
20000 FOR N% = Line% TO 24
20010 LOCATE N%,.1 : PRINT SPC(79)
20020 NEXT N%
20030 RETURN
60000 IF ERR = 1001 AND ERL = 304 THEN RESUME NEXT
60010 IF ERR > 999 THEN GOSUB 63000
62000 CLS : COLOR 7,0 : SET CURSOR 13,1
62010 PRINT "Sorry the procedure SAVEDATA is bombing. This is error "ERR" from line "ERL"
62020 STOP
63000 Line% = 17 : GOSUB 20000
63010 COLOR 7,0 : SET CURSOR 20,1 : PRINT "There is a problem saving your data. Sorry"
63020 FOR M% = -30000 TO 30000 : NEXT N%
63030 RESUME,63040
63040 EXIT
END PROCEDURE

PROCEDURE: LOADDATA
EXTERNAL:
Graph!(), Max_graph_points%(), Graph_nameS(), XtitleS(), YtitleS(), PulseS(), DirectoryS(), TempS()
EXTERNAL: TymeS(), BiasS(), HoldS(), RateS(), GrangeS(), SlopeS(), X_intS(), Y_intS()
EXTERNAL: RangeS()
INTEGER: Line%, Choice%, Position%, N%
STRING: Load_fileS[15], XdataS[18], YdataS[18]
EXTERNAL: CorrelationS(), Min(), Max()
EXTERNAL:
Slope!(), X_int!(), Y_int!(), Correlation!(), CLEARGRAPH, G1, G2, COMP_GRAPH%
INTEGER: COICE%
1 ON ERROR GOSUB 60000
100 DO
110 COLOR 6,0 : SET CURSOR 20,6
120 PRINT "Enter the file name and source drive to be loaded"
130 SET CURSOR 21,6 : PRINT "(Defaults to B:";Graph_nameS(1,1)) "
140 INPUT Load_fileS
150 IF Load_fileS = "" THEN Load_fileS = "B:" + Graph_nameS(1,1)
160 END DO
200 Line% = 16 : GOSUB 20000
210 Choice% = Graph%
300 Line% = 16 : GOSUB 20000
310 DO
320 Position% = LEN(Load_fileS)
330 SET CURSOR 17,10 : COLOR 5,0 : PRINT "LOADING FILE"
340 SET CURSOR 17,23 : COLOR 5,0 : PRINT MID$(Load_fileS,1,Position%)
350 OPEN Load_fileS FOR INPUT AS #1
360 INPUT #1, Graph_name$(1,1)
370 INPUT #1, XtitleS
380 INPUT #1, YtitleS
390 INPUT #1, DirectoryS(Choice%,1), DirectoryS(Choice%,2)
400 INPUT #1, TempS(Choice%)
410 INPUT #1, TymeS(Choice%)
420 INPUT #1, BiasS(Choice%)
430 INPUT #1, PulseS(Choice%)
440 INPUT #1, HoldS(Choice%)
450 INPUT #1, RateS(Choice%)
455 DO
460 INPUT #1, XdataS
462 IF INSTR(XdataS,"X") <> 0 THEN EXIT 1 LEVELS
463 INPUT #1, YdataS
465 GrangeS(Choice%) = XdataS + ";" + YdataS
467 END DO
470 !INPUT #1, Slope!(Choice%)
480 INPUT #1, X_int!(Choice%)

201
490 INPUT #1,Y_int!(Choice%)
500 INPUT #1,Correlation!(Choice%)
507 INPUT #1,Min!(Choice%)
508 INPUT #1,Max!(Choice%)
510 INPUT #1,Max_graph_points%(Choice%)
520 FOR N% = 1 TO Max_graph_points%(Choice%)
530 INPUT #1,Xdata$,Ydata$
540 Graph!(Choice%,1,N%) = VAL(Xdata$)
550 Graph!(Choice%,2,N%) = VAL(Ydata$)
560 NEXT N%
570 END DO
600 Line% = 18 : GOSUB 20000
610 CLOSE #1
1000 EXIT
20000 FOR N% = Line% TO 24
20010 LOCATE N%,1: PRINT SPC(79)
20020 NEXT N%
20030 RETURN
59999 EXIT
60000 IF ERR > 999 THEN GOSUB 61000: RESUME, 59999
60005 Line% = 2 : GOSUB 20000
60010 SET CURSOR 13,6
60020 PRINT "The procedure LOADDATA is bombing. This is error "ERR" from line "ERL"
60030 STOP
61000 Line% = 19: GOSUB 20000
61005 SET CURSOR 19,6: PRINT "There is a file loading error. Check your files"
61010 FOR N% = -30000 TO 30000 : NEXT N%
61020 Line% = 13: GOSUB 20000
61030 RETURN
END PROCEDURE

PROCEDURE: INVERSEPOWER
EXTERNAL: MINX!,MAXX!,MINY!,MAXY!,GRAPH!,MAX_GRAPH_POINTS%
INTEGER: M%,Minloop%
10 ON ERROR GOTO 50000
20 Minloop% = 1
84 DO
85 IF Xory% = 2 THEN EXIT 1 LEVELS
90 MINX! = 1.0E+37: MAXX! = -1.0E+37
91 END DO
92 DO
93 IF Xory% = 1 THEN EXIT 1 LEVELS
98 MINY! = 1.0E+37: MAXY! = -1.0E+37
99 END DO
100 FOR M% = Minloop% TO MAX_GRAPH_POINTS%(Graph1or2%)
110 GRAPH!((Graph1or2%,Xory%,M%) = 1 / (GRAPH!((Graph1or2%,Xory%,M%) * Amount!) ^ Power!
120 DO
130 DO
140 IF Xory% <> 1 THEN EXIT 1 LEVELS
150 IF MINX! > GRAPH!((Graph1or2%,1,M%) THEN MINX! =
GRAPHS(Graph1or2%,1,M%)
    160   IF MAXX! < GRAPHS(Graph1or2%,1,M%) THEN MAXX! =
GRAPHS(Graph1or2%,1,M%)
    170   EXIT 2 LEVELS
    180   END DO
    190   DO
    200   IF MINY! > GRAPHS(Graph1or2%,2,M%) THEN MINY! =
GRAPHS(Graph1or2%,2,M%)
    210   IF MAXY! < GRAPHS(Graph1or2%,2,M%) THEN MAXY! =
GRAPHS(Graph1or2%,2,M%)
    220   END DO
    230   END DO
    240   NEXT M%
    250   EXIT
50000 DO
    50010   IF ERR = 2 AND ERL = 110 THEN Minloop% = M% + 1
    50020   EXIT TO,100
    51000   PRINT "The procedure INVERSEPOWER is bombing. This is error "ERR" from line "ERL"."
    51010   END
60000 END DO
END PROCEDURE

PROCEDURE: MAINKEY
REAL: I,L,N
STRING ARRAY(10)[16]: KEYFUNC$,FUNCS
REAL: O
EXTERNAL: G1,G2,Clearscreen,TITLE,GRAPH_NAMES()
EXTERNAL: MENU_CHOSEN%,GRAPH_CHOSEN%,GRAPH%,GFLAG
10 Clearscreen(2):STATUSLINE OFF
14 SET CURSOR 0,70:COLOR 2,7:PRINT GRAPH%:COLOR 7,0
15 IF MENU_CHOSEN%=0 OR GRAPH_CHOSEN%=0 THEN GOTO 20 ELSE
TITLE(Graph1%):MENU_CHOSEN%,GRAPH_CHOSEN%)):SET CURSOR
0,70:COLOR 2,7:PRINT GFLAG:COLOR 7,0
20 FOR l=1 TO 10
30 IF COS(I*3.14159)<0 THEN L=2*I:N=1 ELSE N=14
40 READ KEYFUNC$(I)
50 READ FUNCS$(I)
60 KEY I,FUNCS$(I)+CHRS(13)
70 SET CURSOR L,N;COLOR 2,0:PRINT "F":I;":COLOR 2,7:SET CURSOR
L,N+4:PRINT KEYFUNC$(I)
86 NEXT I
90 COLOR 2,0
95 STOP
END PROCEDURE

PROCEDURE: PLOTTER
REAL: I,L,N
STRING ARRAY(10)[16]: KEYFUNCS,Funcs
EXTERNAL: Onoff,Clearscreen,MENU_CHOSEN%,GRAPH_CHOSEN%,G1
EXTERNAL: G2,TITLE,GRAPH_NAMES(),G
10 Clearscreen(2)
15 IF MENU_CHOSEN%=0 OR GRAPH_CHOSEN%=0 THEN GOTO 20 ELSE
TITLE(GRAPH_NAMES(MENU_CHOSEN%,GRAPH_CHOSEN%)):SET CURSOR 0,7:COLOR 4,7:PRINT G:COLOR 7,0
20 FOR I=1 TO 10
30 IF COS(I*3.14159)<0 THEN L=2*I:N=1 ELSE N=14
40 READ KEYFUNCS(I)
50 READ Funcs(I)
60 SET CURSOR L,N:COLOR 4,0:PRINT "F";I:" *:COLOR 4,7:SET CURSOR L,N+4:PRINT KEYFUNCS(I)
70 KEY I,KEYFUNCS(I)+CHR$(13)
80 NEXT I
81 IF Onoff=1 THEN SET CURSOR 18,5:COLOR 0,7:PRINT KEYFUNCS(9)
90 COLOR 4,0
100 STOP
110 DATA "MAINKEY","MAINKEY","SPEED ","SPEED","LINES
","LINES","SYMBOLS","SYMBOLS","COLORS ","COLORS","PLOTAXS","EXIT TO 9570","PLOT G1","EXIT TO 9640","PLOT G2","EXIT TO 9220","ON/OFF ","INFO","CAPTION","CAPTION END PROCEDURE

PROCEDURE: MODIFY
REAL: I,L,N
STRING ARRAY(10)[16]: KEYFUNCS
EXTERNAL: Clearscreen,MENU_CHOSEN%,GRAPH_CHOSEN%,G1,G2,TITLE
EXTERNAL: GRAPH_NAMES()
REAL: MD
EXTERNAL: MOD
10 Clearscreen(2)
15 IF MENU_CHOSEN%=0 OR GRAPH_CHOSEN%=0 THEN GOTO 20 ELSE
TITLE(GRAPH_NAMES(MENU_CHOSEN%,GRAPH_CHOSEN%)):SET CURSOR 0,7:COLOR 1,7:PRINT MOD:COLOR 7,0
20 FOR I=1 TO 10
30 IF COS(I*3.14159)<0 THEN L=2*I:N=1 ELSE N=14
40 READ KEYFUNCS(I)
50 KEY I,KEYFUNCS(I)+CHR$(13)
60 SET CURSOR L,N:COLOR 1,0:PRINT "F";I:" *:COLOR 1,7:SET CURSOR L,N+4:PRINT KEYFUNCS(I)
70 NEXT I
80 COLOR 1,0
90 STOP
100 DATA "MAINKEY","ADDNUM ","MLTNUM ","POWER ","NATLOG ","EXPNT ","ABSLT ","DELPCT ","ADDPT ","CLGRPH" END PROCEDURE

PROCEDURE: COLORS
EXTERNAL: PLTTER
REAL: I,J

204
STRING: AS[16]
REAL: N
INTEGER ARRAY(5): EXIST
EXTERNAL: Pen_selected
REAL ARRAY(5): PN
EXTERNAL: PEN(), G1, G2, Clearsreen
STRING: NS[16], JS[16]
10 EXIST(1) = G1; EXIST(2) = G2
11 EXIST(3) = 0; EXIST(4) = 0; EXIST(5) = 0
20 Clearsreen(2): SET CURSOR 5,1: COLOR 7,0: PRINT "EXISTING GRAPHS:" 30 FOR I = 1 TO 4 40 IF EXIST(I) = 0 THEN GOTO 60 50 IF PEN(I) <= 0 THEN SET CURSOR 5,2*I+18: COLOR PEN(I)-2,7: PRINT EXIST(I)
ELSE SET CURSOR 5,2*I+18: COLOR 0,7: PRINT EXIST(I)
60 NEXT I
70 SET CURSOR 10,1: COLOR 7,0: PRINT "COLORS:" 80 FOR I = 1 TO 5 90 SET CURSOR 10,2*I+8: COLOR I-1,7: PRINT I+1
100 NEXT I
110 SET CURSOR 15,1: COLOR 7,0: INPUT "WHICH GRAPH" NS 111 N = VAL(NS)
120 SET CURSOR 17,1: INPUT "WHICH COLOR" JS 121 J = VAL(JS)
125 IF N < 1 OR N > 2 THEN GOTO 135
130 PEN(N) = J 131 FOR I = 1 TO 4 132 IF EXIST(I) = 0 THEN GOTO 134
133 IF PEN(I) <= 0 THEN SET CURSOR 5,2*I+18: COLOR PEN(I)-2,7: PRINT EXIST(I)
ELSE SET CURSOR 5,2*I+18: COLOR 0,7: PRINT EXIST(I)
134 NEXT I
135 COLOR 7,0
140 SET CURSOR 20,1: INPUT "FINISHED CHOOSING COLORS:"; AS
150 IF INSTR(UPPER$(AS), "Y") THEN GOTO 160 ELSE GOTO 20
160 PLOTTER
170 STOP
END PROCEDURE

PROCEDURE: SYMBOLS
INTEGER ARRAY(10): EXIST, LINES
EXTERNAL: PLOTTER
REAL: IJN
STRING: AS[16]
REAL ARRAY(10): SYMBOLS
EXTERNAL: Symbol$, SYMBOLS()
STRING ARRAY(5)[16]: SMBS$
EXTERNAL: G1, G2, CLEARSCREEN
STRING: NS[16], JS[16]
10 EXIST(1) = G1; EXIST(2) = G2
11 EXIST(3) = 0; EXIST(4) = 0; EXIST(5) = 0
20 CLEARSCREEN(2): SET CURSOR 5,1: COLOR 7,0: PRINT "EXISTING GRAPHS:" 30 FOR I = 1 TO 5

205
40 IF EXISTS(I)=0 THEN GOTO 60 ELSE SET CURSOR 5.2*1+18:COLOR 0.7:PRINT EXISTS(I)
50 IF SYMBOLS(I)<=>0 THEN SET CURSOR 5.2*1+18:COLOR SYMBOLS(I)+10.7:PRINT EXISTS(I)
60 NEXT I
70 SET CURSOR 10.1:COLOR 7.0:PRINT "SYMBOLS: "SET CURSOR 10.10:COLOR 11.0:PRINT "1)NONE":SET CURSOR 10.18:COLOR 12.0:PRINT "2) + "SET CURSOR 10.23:COLOR 13.0:PRINT "3) * "SET CURSOR 10.28:COLOR 14.0:PRINT "4) @ "SET CURSOR 15.1:COLOR 7.0:INPUT"WHICH GRAPH" NS
80 SET CURSOR 17.1:INPUT"WHICH SYMBOL" JS
81 N=VAL(NS)
85 SET CURSOR 17.1:INPUT"WHICH SYMBOL" JS
86 J=VAL(JS)
90 IF N<1 OR N>2 THEN GOTO 105
91 IF J=1 THEN SYMBOLS(N)=" *"
92 IF J=2 THEN SYMBOLS(N)=" +"
93 IF J=3 THEN SYMBOLS(N)=" **"
94 IF J=4 THEN SYMBOLS(N)=" @"
100 SYMBOLS(N)=J
101 FOR I=1 TO 5
102 IF EXISTS(I)=0 THEN GOTO 104 ELSE SET CURSOR 5.2*1+18:COLOR 0.7:PRINT EXISTS(I)
103 IF SYMBOLS(I)<=>0 THEN SET CURSOR 5.2*1+18:COLOR SYMBOLS(I)+10.7:PRINT EXISTS(I)
104 NEXT I
105 COLOR 7.0
110 SET CURSOR 20.1:INPUT"FINISHED CHOOSING SYMBOLS":AS
120 IF INSTR(UPPER$(AS),"Y") THEN GOTO 130 ELSE GOTO 20
130 PLOTTER
140 STOP
END PROCEDURE

PROCEDURE: LINES
INTEGER ARRAY(10): EXIST, LINES
EXTERNAL: PLOTTED
REAL: IJN
STRING: AS[16]
EXTERNAL: Line_type$ LINETYPE$, G1, G2, CLEARSCREEN
STRING: NS[?], JS[?]
10 IF EXISTS(1)=G1: EXISTS(2)=G2
11 EXISTS(3)=0: EXISTS(4)=0: EXISTS(5)=0
20 CLEARSCREEN(2): SET CURSOR 5.1:COLOR 7.0: PRINT "EXISTING GRAPHS:"
30 FOR I=1 TO 5
40 IF EXISTS(I)=0 THEN GOTO 60 ELSE SET CURSOR 5.2*1+18:COLOR 0.7:PRINT EXISTS(I)
50 IF LINES(I)<=>0 THEN SET CURSOR 5.2*1+18:COLOR LINES(I)+1.7:PRINT EXISTS(I)
60 NEXT I
70 SET CURSOR 10.1:COLOR 7.0:PRINT "LINES: "SET CURSOR 10.7:COLOR 2.0:PRINT "1)NONE":SET CURSOR 10.14:COLOR 3.0:PRINT "2) + "SET CURSOR 10.21:COLOR 4.0:PRINT "3) * "SET CURSOR 10.28:COLOR 5.0:PRINT "4) @ "SET CURSOR 15.1:COLOR 7.0:INPUT"WHICH GRAPH" NS
81 N=VAL(NS)
85 SET CURSOR 17,1;INPUT"WHICH LINE" J$  
86 J=VAL(J$)
90 IF N<1 OR N>2 THEN GOTO 105
91 IF J=1 THEN LINETYPES(N)="0"
92 IF J=2 THEN LINETYPES(N)=" "
93 IF J=3 THEN LINETYPES(N)="1"
94 IF J=4 THEN LINETYPES(N)="2"
100 LINES(N)=J
101 FOR I=1 TO 5
102 IF EXIST(I)=0 THEN GOTO 104 ELSE SET CURSOR 5,2*I+18;COLOR 0,7;PRINT EXIST(I)
103 IF LINES(I)<>=0 THEN SET CURSOR 5,2*I+18;COLOR LINES(I)+1,7;PRINT EXIST(I)
104 NEXT I
105 COLOR 7,0
110 SET CURSOR 20,1;INPUT"FINISHED CHOOSING LINES";AS
120 IF INSTR(UPPERS(AS),"Y") THEN GOTO 130 ELSE GOTO 20
130 PLOTTER
140 STOP
END PROCEDURE

PROCEDURE: Info
EXTERNAL: Onoff,Plotter
10 IF Onoff=0 THEN Onoff=1 ELSE Onoff=0
20 Plotter
30 STOP
END PROCEDURE

PROCEDURE: CAPTION
STRING: CPTIONS[80]
INTEGER: YPOINT
EXTERNAL: PLOTTER,TIMEDELAY
REAL: HELLO
5 WIDTH 80:CLS
10 SET CURSOR 15,5;COLOR 7,0;PRINT "ENTER THE PLOT CAPTION BELOW (80 CHARACTERS OR LESS)"
20 SET CURSOR 18,1;COLOR 0,7;INPUT">"; CPTIONS
30 CLOSE #3
40 OPEN *COM1: 9600,N,8,1,RS,CS65535,DS,CD* FOR OUTPUT AS #3
50 PRINT #3,"IN; "
70 PRINT #3, "SP2 ; SL15,.25 ; DR 0,1 ;"
80 PRINT #3, "PA PU 7200,900 ; LB*CTIIONS;CHR$(3)
85 TIMEDELAY(15)
86 PRINT #3, "SP0 ;"
90 CLOSE #3
95 COLOR 7,0
100 PLOTTER
110 STOP
END PROCEDURE
PROCEDURE: Fixpoints
INTEGER: O%
10 ON Type% GOTO 100,200,300,400,500
100 Plt! = 48
110 IF Plt! < 15 THEN Plt! = 15 : EXIT
120 IF Plt! > 166 THEN Plt! = 166 : EXIT
198 EXIT
199 STOP
200 Plt! = 618
210 IF Plt! < 15 THEN Plt! = 15 : EXIT
220 IF Plt! > 166 THEN Plt! = 166 : EXIT
298 EXIT
299 STOP
300 Plt! = 166
310 IF Plt! < 48 THEN Plt! = 48 : EXIT
320 IF Plt! > 618 THEN Plt! = 618 : EXIT
398 EXIT
399 STOP
400 Plt! = 15
410 IF Plt! < 48 THEN Plt! = 48 : EXIT
420 IF Plt! > 618 THEN Plt! = 618 : EXIT
500 EXIT
END PROCEDURE

PROCEDURE: Cleargraph
EXTERNAL: Pulse$(,Hold$(,Rate$(,Slope$(,X_int$(,Y_int$(,Range$(,Correlation$(
EXTERNAL: Tyme$(,Temp$(,Grange$(,G1,G2,Comp_graph%
REAL: IN
EXTERNAL: Bias$(,Max_graph_points%
EXTERNAL: Graph$(
INTEGER: J
10 Bias$(G1orG2) = "X"
20 Pulse$(G1orG2) = "X"
30 Hold$(G1orG2) = "X"
40 Rate$(G1orG2) = "X"
50 Slope$(G1orG2) = "X"
60 X_int$(G1orG2) = "X"
70 Y_int$(G1orG2) = "X"
80 Range$(G1orG2) = "X"
90 Correlation$(G1orG2) = "X"
100 Tyme$(G1orG2) = "X"
110 Temp$(G1orG2) = "X"
120 Grange$(G1orG2) = "X"
130 IF Max_graph_points%(G1orG2)>300 THEN J=Max_graph_points%(G1orG2) ELSE J=300
140 FOR I=1 TO 2
150 FOR N=1 TO J
160 Graph!(G1orG2,I,N)=0
170 NEXT N
180 NEXT I
185 Max_graph_points%(G1orG2)=1
190 IF G1orG2=1 THEN G1=0
200 IF G1orG2=2 THEN G2=0:Comp_graph%=0
210 EXIT
220 STOP
END PROCEDURE

PROCEDURE: CLRGRPH
EXTERNAL: G2, COMP_GRAPH%, MODIFY, Cleargraph, G1
REAL ARRAY(10): Exist
EXTERNAL: Clearscreen
REAL: I
STRING: NS[?]
REAL: N
STRING: AS[?]
EXTERNAL: MOD
10 Exist(1)=G1:Exist(2)=G2
11 Exist(3)=0:Exist(4)=0:Exist(5)=0
20 Clearscreen(2):SET CURSOR 5,1:COLOR 7,0:PRINT "EXISTING GRAPHS;"
30 FOR I=1 TO 4
40 IF Exist(I)=0 THEN GOTO 60
50 SET CURSOR 5,2*I+18:COLOR 0,7:PRINT Exist(I)
60 NEXT I
70 SET CURSOR 15,1:COLOR 7,0:INPUT "CLEAR WHICH GRAPH" NS
80 N=VAL(NS):MOD=N
81 IF N<1 OR N>2 THEN GOTO 100
90 Cleargraph N
100 Clearscreen(2):SET CURSOR 5,1:COLOR 7,0:PRINT "EXISTING GRAPHS;"
105 Exist(1)=G1:Exist(2)=G2
110 FOR I=1 TO 4
120 IF Exist(I)=0 THEN GOTO 140
130 SET CURSOR 5,2*I+18:COLOR 0,7:PRINT Exist(I)
140 NEXT I
150 SET CURSOR 18,1:COLOR 7,0:INPUT "FINISHED CLEARING GRAPHS" AS
160 AS=UPPER$(AS)
170 IF INSTR(AS,"Y")=0 THEN GOTO 10
180 MODIFY
190 STOP
END PROCEDURE

PROCEDURE: ADDNUM
EXTERNAL: GRAPH!(,G1,G2, MAX_GRAPH_POINTS%)
INTEGER ARRAY(8): EXIST
STRING: WS[2], XS[2], NUMS[10], AS[2]
EXTERNAL: Clearscreen
INTEGER: I, W
EXTERNAL: MODIFY
INTEGER: M
REAL: NUM
INTEGER: J
EXTERNAL: MOD
10 EXIST(1)=G1:EXIST(2)=G2
11 EXIST(3)=0:EXIST(4)=0:EXIST(5)=0
20 CLEARSCREEN(2):SET CURSOR 5,1:COLOR 7,0:PRINT "EXISTING GRAPHS:" 
30 FOR I=1 TO 4
40 IF EXIST(I)=0 THEN GOTO 60
50 SET CURSOR 5,2*I+18:COLOR 0,7:PRINT EXIST(I)
60 NEXT I
70 SET CURSOR 15,1:COLOR 7,0:INPUT"ADD TO WHICH GRAPH" WS$ 
80 W=VAL(W$):MOD=W 
81 IF W<1 OR W>2 THEN GOTO 100
82 SET CURSOR 17,1:COLOR 7,0:INPUT"ADD TO X OR Y" XS$ 
83 XS=UPPERS(XS$) 
84 M=1 
85 IF INSTR(XS,"X")=0 THEN M=2
87 SET CURSOR 19,1:COLOR 7,0:INPUT"ADD WHAT NUMBER" NUMS$ 
88 NUM=VAL(NUMS$) 
90 FOR J=1 TO MAX_GRAPH_POINTS%(W)
92 GRAPH%(W,M,J)=GRAPH%(W,M,J)+NUM 
94 NEXT J
100 REM
150 SET CURSOR 21,1:COLOR 7,0:INPUT"FINISHED ADDING" AS$ 
160 AS=UPPERS(AS$) 
170 IF INSTR(AS,"Y")=0 THEN GOTO 10 
180 MODIFY 
190 STOP
END PROCEDURE

PROCEDURE: MltNum
EXTERNAL: GRAPH%(,),G1,G2,MAX_GRAPH_POINTS%( )
INTEGER ARRAY(8): EXIST 
STRING: WS[2],XS[2],NUMS[10],AS[2] 
EXTERNAL: CLEARSCREEN 
INTEGER: I,W 
EXTERNAL: MODIFY 
INTEGER: M 
REAL: NUM 
INTEGER: J 
EXTERNAL: MOD 
10 EXIST(1)=G1:EXIST(2)=G2 
11 EXIST(3)=0:EXIST(4)=0:EXIST(5)=0
20 CLEARSCREEN(2):SET CURSOR 5,1:COLOR 7,0:PRINT "EXISTING GRAPHS:" 
30 FOR I=1 TO 4
40 IF EXIST(I)=0 THEN GOTO 60
50 SET CURSOR 5,2*I+18:COLOR 0,7:PRINT EXIST(I)
60 NEXT I
70 SET CURSOR 15,1:COLOR 7,0:INPUT"MLTNUM WHICH GRAPH" WS$ 
80 W=VAL(W$):MOD=W 
81 IF W<1 OR W>2 THEN GOTO 100
82 SET CURSOR 17,1:COLOR 7,0:INPUT"MLT TO X OR Y" XS$ 
83 XS=UPPERS(XS$) 
84 M=1 
85 IF INSTR(XS,"X")=0 THEN M=2
87 SET CURSOR 19,1:COLOR 7,0:INPUT"MLT WHAT NUMBER" NUMS
88 NUM=VAL(NUMS)
90 FOR J=1 TO MAX_GRAPH_POINTS%(W)
92 GRAPH!(W,M,J)=GRAPH!(W,M,J)\*NUM
94 NEXT J
100 REM
150 SET CURSOR 21,1:COLOR 7,0:INPUT"FINISHED MULTIPLYING" AS$~
160 AS$=UPPERS(AS$)
170 IF INSTR(AS$,"Y")=0 THEN GOTO 10
180 MODIFY
190 STOP
END PROCEDURE

PROCEDURE: POWER
EXTERNAL: GRAPH!((),G1,G2,MAX_GRAPH_POINTS%()~
INTEGER ARRAY(8): EXIST~
STRING: WS[2],XS[2],NUMS[10],AS[2]
EXTERNAL: CLEARSCREEN
INTEGER: I,W
EXTERNAL: MODIFY
INTEGER: M
REAL: NUM
INTEGER: J
EXTERNAL: MOD
10 EXIST(1)=G1:EXIST(2)=G2
11 EXIST(3)=0:EXIST(4)=0:EXIST(5)=0
20 CLEARSCREEN(2):SET CURSOR 5,1:COLOR 7,0:PRINT "EXISTING GRAPHS:"
30 FOR I=1 TO 4
40 IF EXIST(I)=0 THEN GOTO 60
50 SET CURSOR 5,2*I+18:COLOR 0,7:PRINT EXIST(I)
60 NEXT I
70 SET CURSOR 15,1:COLOR 7,0:INPUT"POWER WHICH GRAPH" WS$~
80 W=VAL(WS$):MOD=W
81 IF W<1 OR W>2 THEN GOTO 100
82 SET CURSOR 17,1:COLOR 7,0:INPUT"POWER X OR Y" XS$
83 XS$=UPPERS(XS$)
84 M=1
85 IF INSTR(XS$,"X")=0 THEN M=2
87 SET CURSOR 19,1:COLOR 7,0:INPUT"WHAT POWER" NUMS
88 NUM=VAL(NUMS)
90 FOR J=1 TO MAX_GRAPH_POINTS%(W)
92 GRAPH!(W,M,J)=GRAPH!(W,M,J)\^NUM
94 NEXT J
100 REM
150 SET CURSOR 21,1:COLOR 7,0:INPUT"FINISHED POWERING" AS$~
160 AS$=UPPERS(AS$)
170 IF INSTR(AS$,"Y")=0 THEN GOTO 10
180 MODIFY
190 STOP
END PROCEDURE
PROCEDURE: NATLOG
EXTERNAL: GRAPH(),G1,G2,MAX_GRAPH_POINTS%
INTEGER ARRAY(8): EXIST
EXTERNAL: CLEARSCREEN
INTEGER: I,W
EXTERNAL: MOD
INTEGER: M
REAL: NUM
INTEGER: J
EXTERNAL: MOD
10 EXIST(1)=G1:EXIST(2)=G2
11 EXIST(3)=0:EXIST(4)=0:EXIST(5)=0
20 CLEARSCREEN(2):SET CURSOR 5,1:COLOR 7,0:PRINT "EXISTING GRAPHS:"
30 FOR I=1 TO 4
40 IF EXIST(I)=0 THEN GOTO 60
50 SET CURSOR 5,2*I+18:COLOR 0,7:PRINT EXIST(I)
60 NEXT I
70 SET CURSOR 15,1:COLOR 7,0:INPUT "NATLOG WHICH GRAPH" WS
80 W=VAL(WS):MOD=W
81 IF W<1 OR W>2 THEN GOTO 100
82 SET CURSOR 17,1:COLOR 7,0:INPUT "NATLOG X OR Y" XS
83 XS=UPPER$(XS)
84 M=1
85 IF INSTR(XS,"X")=0 THEN M=2
90 FOR J=1 TO MAX_GRAPH_POINTS%(W)
92 GRAPH!(W,M,J)=LOG(GRAPH!(W,M,J))
94 NEXT J
100 REM
150 SET CURSOR 21,1:COLOR 7,0:INPUT "FINISHED NATLOG" AS
160 AS=UPPER$(AS)
170 IF INSTR(AS,"Y")=0 THEN GOTO 10
180 MODIFY
190 STOP
END PROCEDURE

PROCEDURE: EXPNT
EXTERNAL: GRAPH(),G1,G2,MAX_GRAPH_POINTS%
INTEGER ARRAY(8): EXIST
EXTERNAL: CLEARSCREEN
INTEGER: I,W
EXTERNAL: MOD
INTEGER: M
REAL: NUM
INTEGER: J
EXTERNAL: MOD
10 EXIST(1)=G1:EXIST(2)=G2
11 EXIST(3)=0:EXIST(4)=0:EXIST(5)=0
20 CLEARSCREEN(2):SET CURSOR 5,1:COLOR 7,0:PRINT "EXISTING GRAPHS:"
30 FOR I=1 TO 4

212
40 IF EXIST(I)=0 THEN GOTO 60
50 SET CURSOR 5,2*1+18:COLOR 0,7:PRINT EXIST(I)
60 NEXT I
70 SET CURSOR 15,1:COLOR 7,0:INPUT"EXPNT WHICH GRAPH" W$
80 W=VAL(W$):MOD=W
81 IF W<1 OR W>2 THEN GOTO 100
82 SET CURSOR 17,1:COLOR 7,0:INPUT"EXPNT X OR Y" XS
83 XS=UPPERS(XS)
84 M=1
85 IF INSTR(XS,"X")=0 THEN M=2
90 FOR J=1 TO MAX_GRAPH_POINTS%(W)
92 GRAPH!(W,M,J)=EXP(GRA0H!(W,M,J))
94 NEXT J
100 REM
150 SET CURSOR 21,1:COLOR 7,0:INPUT"FINISHED EXPNT" AS
160 AS=UPPERS(AS)
170 IF INSTR(AS,"Y")=0 THEN GOTO 10
180 MODIFY
190 STOP

END PROCEDURE

PROCEDURE: ABSLT
EXTERNAL: GRAPH!,G1,G2,MAX_GRAPH_POINTS%
INTEGER ARRAY(8): EXIST
STRING: WS[2],XS[2],NUMS[10],AS[2]
EXTERNAL: CLEARSCREEN
INTEGER: I,W
EXTERNAL: MODIFY
INTEGER: M
REAL: NUM
INTEGER: J
EXTERNAL: MOD
10 EXIST(I)=G1:EXIST(2)=G2
11 EXIST(3)=0:EXIST(4)=0:EXIST(5)=0
20 CLEARSCREEN(2):SET CURSOR 5,1:COLOR 7,0:PRINT "EXISTING GRAPHS:"
30 FOR I=1 TO 4
40 IF EXIST(I)=0 THEN GOTO 60
50 SET CURSOR 5,2*1+18:COLOR 0,7:PRINT EXIST(I)
60 NEXT I
70 SET CURSOR 15,1:COLOR 7,0:INPUT"ABSLT WHICH GRAPH" W$
80 W=VAL(W$):MOD=W
81 IF W<1 OR W>2 THEN GOTO 100
82 SET CURSOR 17,1:COLOR 7,0:INPUT"ABSLT X OR Y" XS
83 XS=UPPERS(XS)
84 M=1
85 IF INSTR(XS,"X")=0 THEN M=2
90 FOR J=1 TO MAX_GRAPH_POINTS%(W)
92 GRAPH!(W,M,J)=ABS(GRA0H!(W,M,J))
94 NEXT J
100 REM
150 SET CURSOR 21,1:COLOR 7,0:INPUT"FINISHED ABSLT" AS
160 AS=UPPER$(AS)
170 IF INSTR(AS,"Y")=0 THEN GOTO 10
180 MODIFY
190 STOP
END PROCEDURE

PROCEDURE: DELPT
EXTERNAL: GRAPH!,G1,G2,MAX_GRAPH_POINTS%
INTEGER ARRAY(8): EXIST
STRING: WS[2],XS[2],NUMS[10],AS[2]
EXTERNAL: CLEARSCREEN
INTEGER: I,W
EXTERNAL: MODIFY
INTEGER: M
REAL: NUM
INTEGER: J,X
STRING: DS[4]
INTEGER: D,N,FLAG
STRING: DPS[9]
INTEGER: DP
REAL: MAIN
STRING: NULLS[9]
EXTERNAL: MOD
10 EXIST(1)=G1,EXIST(2)=G2
11 EXIST(3)=0:EXIST(4)=0:EXIST(5)=0
20 CLEARSCREEN(2):SET CURSOR 5,1:COLOR 7,0:PRINT "EXISTING GRAPHS:"
30 FOR I=1 TO 4
40 IF EXIST(I)=0 THEN GOTO 60
50 SET CURSOR 5,2*I+18:COLOR 0,7:PRINT EXIST(I)
60 NEXT I
70 SET CURSOR 15,1:COLOR 7,0:INPUT"DELPT FROM WHICH GRAPH* W$"
80 W=VAL(W$):MOD=W
81 IF W<1 OR W>2 THEN GOTO 240
90 N=2
100 M=1
101 FOR J=3 TO 20
102 SET CURSOR J,40:COLOR 2,0:PRINT ""  
104 NEXT J
110 FOR J=M TO MAX_GRAPH_POINTS%(W)
120 SET CURSOR J-N,40:COLOR 2,0:PRINT J;";GRAPH!(W,1,J);";GRAPH!(W,2,J)
130 IF J-N=15 THEN EXIT TO,150
135 FLAG=J
140 NEXT J
150 N=N-3:M=J
160 SET CURSOR 17,1:COLOR 7,0:INPUT"DELETE WHICH POINT*;DS"
170 D=VAL(D$)
180 IF D=0 AND FLAG<>MAX_GRAPH_POINTS%(W) THEN GOTO 101
190 IF D=0 THEN GOTO 240
210 FOR J=D TO MAX_GRAPH_POINTS%(W)
220 GRAPH!(W,1,J)=GRAPH!(W,1,J+1)
221 GRAPH!(W,2,J)=GRAPH!(W,2,J+1)
230 NEXT J
231 MAX_GRAPH_POINTS%(W)=MAX_GRAPH_POINTS%(W)-1
240 REM
250 SET CURSOR 21,1;COLOR 7,0;INPUT"FINISHED DELPT" AS$
260 AS$=UPPER$(AS$)
270 IF INSTR(AS$,"Y")=0 THEN GOTO 10
280 MODIFY
290 STOP
END PROCEDURE

PROCEDURE: ADDPT
EXTERNAL: GRAPH!,G1,G2,MAX_GRAPH_POINTS()
INTEGER ARRAY(8): EXIST
STRING: WS[2],XS[2],NMS[10],AS[2]
EXTERNAL: CLEARSCREEN
INTEGER: I,W
EXTERNAL: MODIFY
INTEGER: M
REAL: NUM
INTEGER: J,X
STRING: APS[4]
INTEGER: AP,N,FLAG
STRING: XADS[10],YADS[10]
REAL: XAD,YAD
EXTERNAL: MOD
10 EXIST(1)=G1:EXIST(2)=G2
11 EXIST(3)=0:EXIST(4)=0:EXIST(5)=0
20 CLEARSCREEN(2):SET CURSOR 5,1;COLOR 7,0;PRINT "EXISTING GRAPHS:"
30 FOR I=1 TO 4
40 IF EXIST(I)=0 THEN GOTO 60
50 SET CURSOR 5,2*I+18:COLOR 0,7:PRINT EXIST(I)
60 NEXT I
70 SET CURSOR 15,1;COLOR 7,0;INPUT"ADDPT TO WHICH GRAPH" W$
80 W=VAL(W$):MOD=W
81 IF W<1 OR W>2 THEN GOTO 240
90 N=-2
100 M=1
101 FOR J=3 TO 20
102 SET CURSOR J,40;COLOR 2,0;PRINT "
104 NEXT J
110 FOR J=M TO MAX_GRAPH_POINTS%(W)
120 SET CURSOR J,N,40;COLOR 2,0;PRINT J:".;GRAPH!(W,1,J);".;GRAPH!(W,2,J)
130 IF J-N=15 THEN EXIT TO,150
135 FLAG=J
140 NEXT J
150 N=J-3;M=J
160 SET CURSOR 17,1;COLOR 7,0;INPUT"ADD BEFORE WHICH POINT" APS$
170 AP=VAL(AP$)
180 IF AP=0 AND FLAG<MAX_GRAPH_POINTS%(W) THEN GOTO 101
190 IF AP=0 THEN GOTO 240
191 SET CURSOR 19,1;COLOR 7,0;INPUT"ENTER NEW X,Y";XADS,YADS
210 FOR J=MAX_GRAPH_POINTS%(W) TO AP STEP -1
220   GRAPH!(W,1,J+1)=GRAPH!(W,1,J)
221   GRAPH!(W,2,J+1)=GRAPH!(W,2,J)
230 NEXT J
231 MAX_GRAPH_POINTS%(W)=MAX_GRAPH_POINTS%(W)+1
232 GRAPH!(W,1,AP)=VAL(XADS$)
233 GRAPH!(W,2,AP)=VAL(YADS$)
240 REM
250 SET CURSOR 21,1:COLOR 7,0.INPUT *FINISHED ADDPT* AS
260 AS=UPPERS(AS$)
270 IF INSTR(AS$,"Y")=0 THEN GOTO 10
280 MODIFY
290 STOP
END PROCEDURE

PROCEDURE: SPEED
EXTERNAL: SPEEDS,PLOTTER
   10 CLS:COLOR 9,1:SET CURSOR 5,1
   20 INPUT *ENTER PLOTTER PEN SPEED .1(fast) to 2(slow)*;SPEEDS$  
   30 COLOR 7,0:PLOTTER  
   40 STOP
END PROCEDURE

'MAIN Program:

100 DO
110 SCREEN 0,0,0 : VIEW : CLS : STATUSLINE OFF : TROFF
120 COLOR 2,1:CLS:SET CURSOR 13,30:PRINT "Now Entering Graphics"
130 TIMEDELAY(3)
140 COLOR 2,0,0
145 CLS
150 END DO
155 SET CURSOR 0,74:COLOR 8,7:PRINT G1;G2:COLOR 7,0
160 MAINKEY
200 CLEAR:WIDTH 80
210 ON ERROR GOTO 55000
230 CHDIR "."
240 MinX! = 1E+19
250 MaxX! = -1E+19
260 MinY! = 1E+19
270 MaxY! = -1E+19
275 FOR N% = 1 TO 2
280 Cleargraph N%
350 NEXT N%
390 Q! = 1.60218E-19
400 K! = 1.38066E-23
410 E_sub_O! = 8.85418E-14
490 Minloop% = 1
500 REM
510 RESTORE,60200
520 READ Max_graph_number%
530 FOR N% = 1 TO 5
540 FOR M% = 1 TO 24
550 READ Graph_name$(N%,M%)
560 IF (N%-1)*24 + M% = > Max_graph_number% THEN EXIT 2 LEVELS
570 NEXT M%
580 Maxloop%(N%) = 24
590 NEXT N%
600 Maxloop%(N%) = M% - 1
605 IF M% = 24 THEN Maxloop%(N%) = 24
610 IF N% = 6 THEN Next_screen% = 5 ELSE Next_screen% = N%
700 REM
710 FOR N% = 1 TO Next_screen%
720 TITLE ("Graphics Menu")
730 FOR M% = 1 TO Maxloop%(N%)
740 MENU (Graph_name$(N%,M%),M%,Rownumber%)
750 NEXT M%
760 DO
770 IF N% = Next_screen% THEN EXIT
780 IF N% > 1 THEN Col% = 5
790 FINISH ("Go to Next Graphics Menu",Rownumber%,Col%,25):EXIT
800 REPEAT
810 DO
820 IF N% < 2 THEN EXIT
830 IF N% < Next_screen% THEN Col% = 42
835 IF N% = Next_screen% THEN Col% = 5
840 FINISH ("Go to Previous Graphics Menu",Rownumber%,Col%,26):EXIT
850 REPEAT
860 DO
870 IF N% < Next_screen% THEN EXIT
875 Col% = 42
880 FINISH ("Exit to DOS",Rownumber%,Col%,27):EXIT
890 REPEAT
895 BORDER (Rownumber%)
900 REM
910 DO
920 ON N% GOTO 940,950,960,970
930 CLS:PRINT "ERROR SAVE SCREEN ROUTINE":STOP
940 SAVE SCREEN 0,0,24,79,MENU1:EXIT
950 SAVE SCREEN 0,0,24,79,MENU2:EXIT
960 SAVE SCREEN 0,0,24,79,MENU3:EXIT
970 SAVE SCREEN 0,0,24,79,MENU4:EXIT
980 REPEAT
1000 REM
1010 DO
1020 CLEARSCREEN (21)
1030 COLOR 6,0,SET CURSOR 21,5
1040 INPUT "Enter the number to be executed";Choice%
1050 IF Choice% > 0 AND Choice% <= Maxloop%(N%) THEN EXIT 2 LEVELS
1060 IF Choice% = 26 AND N% > 1 THEN EXIT TO,1100
1070 IF Choice% = 25 AND N% < Next_screen% THEN EXIT TO,1300
1080 IF Choice% = 27 AND N% = Next_screen% THEN CLS:SYSTEM
1090 REPEAT
1100 REM
1110 DO
1120 N% = N% - 1
1130 ON N% GOTO 1150,1160,1170,1180
1140 CLS:PRINT *ERROR IN RESTORE ROUTINE*:STOP
1150 RESTORE SCREEN 0,0,24,79,Menu1:EXIT
1160 RESTORE SCREEN 0,0,24,79,Menu2:EXIT
1170 RESTORE SCREEN 0,0,24,79,Menu3:EXIT
1180 RESTORE SCREEN 0,0,24,79,Menu4:EXIT
1200 REPEAT
1210 GOTO 1000
1300 REM
1310 NEXT N%
1320 Col% = CINT((80 - LEN(Graph_name$(N%,Choice%))/2)
1330 CLS:COLOR 3,0:SET CURSOR 0,Col%
1340 PRINT Graph_name$(N%,Choice%)
1350 Max_graph_points%(1) = 1
1360 Max_graph_points%(2) = 1
1400 REM
1410 Menu_chosen% = N%
1420 Graph_chosen% = Choice%
1425 GOTO 160
1429 Cleargraph 1:G1=1:Graph%=1
1430 WIDTH
80:GFLAG=Graph%.TITLE(Graph_name$(Menu_chosen%,Graph_chosen%)):ON
Menu_chosen% GOTO 1500,1600,1700,1800,1900
1440 CLS:COLOR 7,0:PRINT *ERROR IN MENU_CHOSEN ROUTINE (1420)*:STOP
1500 REM
1510 ON Graph_chosen% GOSUB 2000,2000,10000,10000,25000,23000,23000,17000,17000,23000,23000,23000,10000,11000,11000,11000,11000,11000,11000,11000,11000,11000,00,11000,11000,11000,11000,15000,15000,15000,15000
1520 CLS:PRINT *ERROR IN FIRST MENU ROUTINE (1500)*:STOP
1600 REM
1620 CLS:PRINT *ERROR IN SECOND MENU ROUTINE (1600)*:STOP
1700 REM
1720 CLS:PRINT *ERROR IN THIRD MENU ROUTINE (1700)*:STOP
1800 REM
1820 CLS:PRINT *ERROR IN FOURTH MENU ROUTINE (1800)*:STOP
1900 REM
1920 CLS:PRINT *ERROR IN FIFTH MENU ROUTINE (1900)*:STOP
2000 CLS:PRINT *THESE GRAPHS ARE NOT INSTALLED.*
2010 TIMEDELAY ()
2020 GOTO 200
2425 GRange$(Graph%) = STR$(Min!) + ";" + STR$(Max!)
2430 GOTO 160
3000 WIDTH 80: Minloop% = 1
3002 IF G1 = 1 THEN DELTAMULTIPLY(1,1,1,0): DELTAMULTIPLY(1,2,1,0)
3003 IF G2 = 2 THEN DELTAMULTIPLY(2,1,1,0): DELTAMULTIPLY(2,2,1,0)
3007 CLOSE
3010 DO
3020 TITLE (*Extrema for X and Y axes*)
3030 COLOR 4.0: SET CURSOR 2,5
3040 PRINT *Minimum X*; SPC(10); *Maximum X*; SPC(10); *Minimum Y*; SPC(10); *Maximum Y*
3050 COLOR 3,0
3060 SET CURSOR 3,7: PRINT MinX!
3070 SET CURSOR 3,26: PRINT MaxX!
3080 SET CURSOR 3,45: PRINT MinY!
3090 SET CURSOR 3,64: PRINT MaxY!
3100 COLOR 7,0: SET CURSOR 19,5
3110 INPUT "Do you want to change any of the extrema (Defaults to NO)"; Check$.
3120 Check$ = UPPER$(Check$)
3130 IF INSTR(Check$, "Y") = 0 THEN EXIT 1 LEVELS
3140 CLEARSCREEN (19)
3150 DO 1 TIMES
3160 COLOR 6,0: SET CURSOR 9,5
3170 PRINT "Enter the Minimum X value (Defaults to ""; MinX!; ");"
3180 INPUT Change$
3190 IF Change$ <> "" THEN MinX! = VAL(Change$)
3200 COLOR 6,0: SET CURSOR 11,5
3210 PRINT "Enter the Maximum X value (Defaults to ""; MaxX!; ");"
3220 INPUT Change$
3230 IF Change$ <> "" THEN MaxX! = VAL(Change$)
3240 COLOR 6,0: SET CURSOR 13,5
3250 PRINT "Enter the Minimum Y value (Defaults to ""; MinY!; ");"
3260 INPUT Change$
3270 IF Change$ <> "" THEN MinY! = VAL(Change$)
3280 COLOR 6,0: SET CURSOR 15,5
3290 PRINT "Enter the Maximum Y value (Defaults to ""; MaxY!; ");"
3300 INPUT Change$
3310 IF Change$ <> "" THEN MaxY! = VAL(Change$)
3320 REPEAT
3330 REPEAT
3340 GOTO 160
4000 REM
4010 DO
4020 CLS: COLOR 2,0: SET CURSOR 11,5
4030 INPUT "Do you want a dot (1) or a line (2) graph"; Choice$
4040 IF Choice$ > 0 AND Choice$ < 3 THEN EXIT
4050 REPEAT
4060 Graph_type% = Choice$
4000 REM
4100 DO 1 TIMES
4120 COLOR 2,0
4130 SCREEN 2
4540  DRAW "BM200,100 U2 D4 U2 L2 R4"
4550  GET (198,102) - (202,98),PLUS
4560  CLS
4570  REPEAT
5000  REM
5010  Col% = CINT((79 - LEN(Graph_name$)(Menu_chosen%,Graph_chosen%)))/2)
5020  LOCATE 1,Col%;PRINT Graph_name$(Menu_chosen%,Graph_chosen%)
5050  XTITLE (Xtitle$,MinX!,MaxX!)
5060  YTITLE (Ytitle$,MinY!,MaxY!)
5070  REM
5510  DeltaX! = (570)/(MaxX! - MinX!)
5520  DeltaY! = (150)/(MaxY! - MinY!)
6000  REM
6004  ERASE Plot!,Skippoint%
6005  DIM Skippoint%(2,600)
6010  FOR M% = 1 TO Max_graph_points%(1)
6015    DO
6017    IF MinX! > Graph!(1,1,M%) THEN Skippoint%(1,M%) = 1 : EXIT 1 LEVELS
6019    IF MaxX! < Graph!(1,1,M%) THEN Skippoint%(1,M%) = 2 : EXIT 1 LEVELS
6021    IF MinY! > Graph!(1,2,M%) THEN Skippoint%(1,M%) = 3 : EXIT 1 LEVELS
6023    IF MaxY! < Graph!(1,2,M%) THEN Skippoint%(1,M%) = 4 : EXIT 1 LEVELS
6025    X_plot! = DeltaX!*(Graph!(1,1,M%) - MinX!) + 46
6026    Y_plot! = DeltaY!*(Graph!(1,2,M%) - MinY!) + 165
6030    PSET (X_plot!,Y_plot!)
6031    Skippoint%(1,M%) = 5
6035    END DO
6040  NEXT M%
6045  DO 1 TIMES
6050  IF Comp_graph% = 0 THEN EXIT
6120  FOR M% = Minloop% TO Max_graph_points%(2)
6130    DO
6140    IF MinX! > Graph!(2,1,M%) THEN Skippoint%(2,M%) = 1 : EXIT 1 LEVELS
6145    IF MaxX! < Graph!(2,1,M%) THEN Skippoint%(2,M%) = 2 : EXIT 1 LEVELS
6150    IF MinY! > Graph!(2,2,M%) THEN Skippoint%(2,M%) = 3 : EXIT 1 LEVELS
6155    IF MaxY! < Graph!(2,2,M%) THEN Skippoint%(2,M%) = 4 : EXIT 1 LEVELS
6160    X_plot! = DeltaX!*(Graph!(2,1,M%) - MinX!) + 46
6170    Y_plot! = DeltaY!*(Graph!(2,2,M%) - MinY!) + 165
6180    PUT (X_plot!,Y_plot!),PLUS,OR
6190    Skippoint%(2,M%) = 5
6200    END DO
6208  NEXT M%
6209  REPEAT
6210  DIM Plot%(2,2)
6212  DO
6213  IF Graph_type% <> 2 THEN EXIT 1 LEVELS
6214  FOR N% = 1 TO 2
6215  IF N% = 2 AND Comp_graph% <> 1 THEN EXIT 2 LEVELS
6216  FOR M% = 1 TO Max_graph_points%(N%) - 1
6217    DO
6218      FOR O% = 1 TO 2
6219      Plot!(O%,1) = DeltaX!*(Graph!(N%,1,M%+O%-1) - MinX!) + 46
Plot(O%,2) = DeltaY*(Graph(N%,2,M%+O%-1) - MinY) + 165
Fixpoints Skippoint%(N%,M%+O%-1),0,Plot(O%,1),Plot(O%,2)
NEXT O%
IF N% = 2 AND (Skippoint%(1,M%) = 5 OR Skippoint%(2,M%) = 5) THEN
Plot(1,1) = Plot(1,1) + 2:Plot(1,2)=Plot(1,2)+2:Plot(2,1)=
Plot(2,1)+2:Plot(2,2)=Plot(2,2)+2
LINE (Plot(1,1),Plot(1,2)) - (Plot(2,1),Plot(2,2))
END DO
NEXT M%
NEXT N%
END DO
DO
TIMEDELAY(2)
AS = INKEY$: IF INKEY$ = "" THEN GOTO 6250
SCREEN 0,0,0: VIEW: CLS
ERASE Skippoint%,Plot!
END DO
GOTO 160
WIDTH 80: TITLE (Graph_name$(Menu_chosen%,Graph_chosen$))
Fit$=UPPERS$"Y"
Error_LS%=0
FOR M% = 1 TO 2
DO 1 TIMES
DO
IF INSTR(Fit$,"Y")=0 THEN EXIT 1 LEVELS
CLEAR (Slope!(M%),Correlation!(M%),X_int!(M%),Y_int!(M%))
LEASTSQUARES
(M%,Min!(M%),Max!(M%),Slope!(M%),Correlation!(M%),X_int!(M%),Y_int!(M%),M%,-1,Error_LS%)%
IF Error_LS% <> 0 THEN Fit$ = "END"
END DO
REPEAT
DO
IF Error_LS% = 1 THEN EXIT TO 160
IF INSTR(Fit$,"Y") = 0 THEN EXIT
DO 1 TIMES
IF M% = 2 THEN EXIT
CLS: TITLE ("Least Squares Fit")
COLOR 2,0,0
SET CURSOR 8,0:PRINT "X - Intercept"
SET CURSOR 10,0:PRINT "Y - Intercept"
SET CURSOR 12,0:PRINT "Slope"
SET CURSOR 14,0:PRINT "Correlation"
SET CURSOR 16,0:PRINT "Range"
REPEAT
DO
IF M% = 1 THEN Col%= 20
IF M% = 2 THEN Col%= 50
SET CURSOR 6,Col%
IF M% = 1 THEN PRINT ". CURVE 1" ELSE PRINT "+ CURVE 2"
FOR N% = 1 TO 6
SET CURSOR N%*2 + 6,Col%
DO
ON N% GOTO 7870,7880,7890,7900,7910,7920
Dummy! = X_int!(M%):EXIT 1 LEVELS
Dummy! = Y_int!(M%):EXIT 1 LEVELS
Dummy! = Slope!(M%):EXIT 1 LEVELS
Dummy! = Correlation!(M%):EXIT 1 LEVELS
Dummy! = Min!(M%):EXIT 1 LEVELS
SET CURSOR 16,Col% + 8:PRINT ":
SET CURSOR 16,Col% + 10:Dummy! = Max!(M%):EXIT 1 LEVELS
REPEAT
DO 1 TIMES
IF Dummy! => .01 AND Dummy! <= 1000 THEN PRINT USING ".;Dummy!:EXIT
IF Dummy! <= -.01 AND Dummy! => -1000 THEN PRINT USING ".;Dummy!:EXIT
IF Dummy! = 0 THEN PRINT "0" ELSE PRINT USING ".##.##;Dummy!:EXIT
REPEAT
NEXT N%
END DO
END DO
NEXT M%
DO 1 TIMES
IF INSTR(Fit$,"Y") = 0 THEN EXIT
TIMEDELAY (4)
SET CURSOR 21,5:COLOR 7,0:PRINT "HIT ANY KEY TO CONTINUE"
AS = INKEY$:IF AS = "" THEN GOTO 8040
REPEAT
IF X_int!(1) <> 0 OR X_int!(2) <> 0 THEN Fit$ = "Y"
GOTO 160
DO
CLEARSCREEN(2): SET CURSOR 13,6
SAVEDATA(Comp_graph%+1)
END DO
GOTO 160
Cleargraph 2;G2=2;Graph%=2;Comp_graph%=1
CLS:COLOR 3,0:SET CURSOR 0,CINT((80 - LEN(Graph_name$(Menu_chosen%,Graph_chosen%)))/2)
PRINT Graph_name$(Menu_chosen%,Graph_chosen%)
EXIT TO,1430
REM
TITLE (Graph_name$(Menu_chosen%,Graph_chosen%))
DO 1 TIMES
IF Info_pointer% < 5 THEN EXIT 1 LEVELS
COLOR 7,0: SET CURSOR 13,6
PRINT "WARNING! If you plan on plotting the another graph, the information block"
SET CURSOR 14,6: PRINT "will not be printed!"
TIMEDELAY (5)
REPEAT
PLOTTER
9220 DO
9230 IF INSTR(Fit$,"Y") = 0 THEN EXIT 1 LEVELS
9240 Slope$2 = STR$(Slope!(2))
9250 X_int$2 = STR$(X_int!(2))
9260 Y_int$2 = STR$(Y_int!(2))
9265 Range$2 = STR$(Min!(2)) + "," + STR$(Max!(2))
9266 Correlation$2 = STR$(Correlation!(2))
9267 END DO
9350 G=G2:Print_out%=1
9360 PLOTTING_POINTS(Symbol$,2,1)
9370 EXIT TO,9999
9560 GOTO 9000
9570 DO
9580 Title$ = UPPERS(Graph_name$(Menu_chosen%,Graph_chosen%))
9590 PLOTTERAXES (UPPERS(Graph_name$(Menu_chosen%,Graph_chosen%)))
9600 END DO
9601 GOTO 9000
9640 DO
9650 IF INSTR(Fit$,"Y") = 0 THEN EXIT 1 LEVELS
9660 Slope$(1) = STR$(Slope!(1))
9670 X_int$(1) = STR$(X_int!(1))
9680 Y_int$(1) = STR$(Y_int!(1))
9690 Range$(1) = STR$(Min!(1)) + "," + STR$(Max!(1))
9695 Correlation$(1) = STR$(Correlation!(1))
9696 END DO
9700 DO
9701 Temp$(1) = "" + Temp$(1)
9702 Tyme$(1) = "" + Tyme$(1)
9703 Bias$(1) = "" + Bias$(1)
9704 Pulse$(1) = "" + Pulse$(1)
9705 Hold$(1) = "" + Hold$(1)
9706 Rate$(1) = "" + Rate$(1)
9707 GRange$(1) = "" + GRange$(1)
9708 G=G1
9710 PLOTTING_POINTS(Symbol$,1,0)
9715 Print_out% = 1
9720 END DO
9999 GOTO 9000
10000 REM
10007 Error% = 0
10010 DO
10020 IF Graph_chosen% = 11 THEN EXIT 1 LEVELS
10030 Directory$(Graph%,2) = "NO"
10040 END DO
10040 DO
10410 IF Graph_chosen% = 3 THEN File$ = "G-T" ELSE File$ = "C-T"
10415 IF Graph_chosen% = 11 THEN ID% = 0 ELSE ID% = 1
10420 DIRECTORY (File$,ID%,Graph%,Error%)
10430 IF Error% <> 0 THEN EXIT TO,160
10434 DO 1 TIMES
10435 IF Graph_chosen% = 11 THEN ID% = 1 ELSE ID% = 0
RUNTYPE(Graph%,ID%)

DO

IF Graph_chosen% <> 11 THEN EXIT 1 LEVELS

IF INSTR(Temp_type$(Graph%),"T") = 0 AND INSTR(Temp_type$(Graph%),"R") = 0 THEN EXIT 1 LEVELS

CLEARSCREEN(2)

SET CURSOR 13,6; COLOR 7,0

PRINT "The DLTS Spectrum cannot be done with either a TIME or ROOM TEMPERATURE run."

TIMEDELAY()

EXIT TO,10400

REPEAT

REPEAT

IF Graph_chosen% <> 11 THEN ID% = 1 ELSE ID% = 4

PARAMETERSET (ID%)

IF Graph_chosen% = 11 THEN ID% = 8 ELSE ID% = 1

DATAPRINT

FOR N% = 1 TO 2

DO

IF INSTR(Directory$(Graph%,N%),"NO") <> 0 THEN EXIT 1 LEVELS

DATARETRIEVAL (ID%,Graph%,N%,Error%,0)

IF Error% <> 0 THEN EXIT TO,10400

END DO

NEXT N%

END DO

DO

DO

IF Graph_chosen% <> 2 THEN EXIT 1 LEVELS

DELTMULTIPLY (Graph%,2,1.0E+09)

Ytitle$ = "Capacitance in nF"

DELTMULTIPLY (Graph%,1,1000.0)

Xtitle$ = "Time in milliSeconds"

IF INSTR(Temp_type$(Graph%),"T") = 0 THEN Temp$(Graph%) = STRS(Temp_sought!)

IF INSTR(Temp_type$(Graph%),"T") <> 0 THEN Time$(Graph%) = STRS(Time_sought!)

Bias$(Graph%) = MIDS(Bias_found$,INSTR(Bias_found$,"=")+2)

Pulse$(Graph%) = MIDS(Pulse_found$,INSTR(Pulse_found$,"=")+2)

Hold$(Graph%) = MIDS(Hold_found$.INSTR(Hold_found$,"=")+2)

EXIT 2 LEVELS

END DO

DO

DO

IF Graph_chosen% <> 3 THEN EXIT 1 LEVELS

DELTMULTIPLY (Graph%,2,1000.0)

Ytitle$ = "Conductance in mMhos"

DELTMULTIPLY (Graph%,1,1000.0)

Xtitle$ = "Time in milliSeconds"

IF INSTR(Temp_type$(Graph%),"T") = 0 THEN Temp$(Graph%) = STRS(Temp_sought!)

IF INSTR(Temp_type$(Graph%),"T") <> 0 THEN Time$(Graph%) = STRS(Time_sought!)
Bias$(Graph%) = MID$(Bias_found$,INSTR(Bias_found$,"=") + 1)
Pulse$(Graph%) = MID$(Pulse_found$,INSTR(Pulse_found$,"=") + 1)
Hold$(Graph%) = MID$(Hold_found$,INSTR(Hold_found$,"=") + 1)
EXIT 2 LEVELS
END DO
DO
IF Graph_chosen% <> 11 THEN EXIT 1 LEVELS
DELTAMULTIPLY (Graph%,2,1E+12)
Ytitle$ = "DLTS in pF"
Xtitle$ = "Temperature in Kelvin"
Bias$(Graph%) = MID$(Bias_found$,INSTR(Bias_found$,"=") + 1)
Pulse$(Graph%) = MID$(Pulse_found$,INSTR(Pulse_found$,"=") + 1)
Hold$(Graph%) = MID$(Hold_found$,INSTR(Hold_found$,"=") + 1)
Rate$(Graph%) = STR$(LOG(Mintyme! / Maxtyme!) / (Mintyme! - Maxtyme!))
EXIT 2 LEVELS
END DO
END DO
CLOSE
RETURN,160
REM
Counter% = 0
File% = 1
DO
IF Graph_chosen% = 13 OR Graph_chosen% = 14 THEN File$ = "CV" ELSE File$ = "UV"
IF Graph_chosen% > 16 AND Graph_chosen% < 25 THEN ID% = 0 ELSE ID% = 1
IF Graph_chosen% = 3 THEN ID% = 0
DIRECTORY(File$,ID%,Graph%,Error%)
IF Error% <> 0 THEN EXIT TO,160
END DO
DO
IF Graph_chosen% > 16 OR Graph_chosen% = 3 THEN ID% = 1 ELSE ID% = 0
RUNTYPE(Graph%,ID%)
DO
IF INSTR(Temp_type$(Graph%),"R") = 0 OR Graph_chosen% <> 17 THEN EXIT 1 LEVELS
CLEARSCREEN(2)
COLOR 7,0,0 : SET CURSOR 13,6
PRINT "You cannot do the chosen graph with a room temperature run."
TIMEDELAY(5)
EXIT TO,11500
REPEAT
END DO
DO
IF Graph_chosen% <> 20 THEN EXIT 1 LEVELS
CLEARSCREEN(2)
SET CURSOR 13,6 : COLOR 2,0
INPUT "Enter the AE*A ",Dummy$
AeA! = VAL(Dummy$)
IF AeA! <= 0 THEN EXIT TO,11590
END DO
DO
IF Graph_chosen% <> 17 THEN EXIT 1 LEVELS
CLEARSCREEN(2)
SET CURSOR 13,6 : COLOR 2,0,0
INPUT *Enter the forward bias. *Dummy$;
Vf! = VAL(Dummy$)
END DO
DATAPRINT
DO
IF Graph_chosen% < 17 AND Graph_chosen% <> 3 THEN EXIT 1 LEVELS
IF INSTR(Temp_type$,("T") <> 0 THEN Run_type% = 2 ELSE Run_type% = 1
IF INSTR(Temp_type$,("T") = 0 THEN Temp_type$(Graph%) = "R"
EXIT TO,11900
REPEAT
Run_type% = 0
DO
IF Graph_chosen% = 13 OR Graph_chosen% = 14 THEN ID% = 7 ELSE ID% = 6
DATARETRIEVAL (ID%,Graph%,File%,Error%,Run_type%)
IF Error% = 1 AND Graph!(Graph%,2,2) <> 0 AND Graph_chosen% = 16 THEN EXIT 1 LEVELS
IF Error% = 1 AND Graph!(Graph%,2,2) <> 0 AND Graph_chosen% = 12 THEN EXIT 1 LEVELS
IF Error% = 1 AND Graph!(Graph%,2,2) <> 0 AND Graph_chosen% = 15 THEN EXIT 1 LEVELS
IF Error% <> 0 AND Graph_chosen% < 17 AND Graph_chosen% <> 3 THEN EXIT TO,11500
END DO
DO
IF Graph_chosen% <> 12 THEN EXIT 1 LEVELS
DELTAMULTIPLY(Graph%,2,1000.0)
Xtitle$ = "Voltage in Volts";
Ytitle$ = "Current in mA"
EXIT 2 LEVELS
REPEAT
DO
IF Graph_chosen% <> 13 THEN EXIT 1 LEVELS
DELTAMULTIPLY(Graph%,2,1E+09)
Ytitle$ = "Cap in nF"
EXIT 2 LEVELS
REPEAT
DO
IF Graph_chosen% <> 14 THEN EXIT 3 LEVELS
INVERSEPOWER(Graph%,2,1E+09,2,000)
Ytitle$ = "1/C**2 C in nF"
END DO
END DO
Xtitle$ = "Voltage in Volts"
12220 EXIT 2 LEVELS
12230 REPEAT
12240 DO
12250 IF Graph_chosen% > 16 OR Graph_chosen% = 3 THEN EXIT 1 LEVELS
12260 DELTAIn(Graph%,2,1.0)
12270 Ytitle$ = "Ln(I) 1 in A"
12280 DO
12290 IF Graph_chosen% <= 15 THEN EXIT 1 LEVELS
12300 Xtitle$ = "Voltage in Volts"
12305 DELTAMULTIPLY(Graph%,2,1/LOG(10))
12310 EXIT 3 LEVELS
12320 REPEAT
12330 DO
12340 Position% = INSTR(Temp_found$,,") + 1
12350 Temp! = VAL(MIDS(Temp_found$,Position%))
12360 DELTAMULTIPLY(Graph%,1,((Q!/(K! * Temp!)))
12370 Xtitle$ = "QV / kT"
12375 IF Graph_chosen% = 17 THEN EXIT 1 LEVELS
12380 EXIT 3 LEVELS
12390 REPEAT
12400 END DO
12410 DO
12420 IF Graph_chosen% <= 17 THEN EXIT 1 LEVELS
12430 DO
12440 Position% = INSTR(Temp_found$,,") + 1
12450 Temp! = VAL(MIDS(Temp_found$,Position%))
12460 END DO
12470 Counter% = Counter% + 1
12480 DO
12490 FOR M% = 1 TO Max_graph_points%(Graph%)
12500 IF Graph!(Graph%,1,M%) = Vf! THEN EXIT 2 LEVELS
12510 NEXT M%
12515 IF Counter% = 1 THEN EXIT TO,12630
12520 CLEARSCREEN(2)
12530 COLOR 7,0 : SET CURSOR 13,6
12540 PRINT "The Forward bias you specified was not found"
12550 TIMEDELAY (5)
12560 EXIT TO,11670
12570 REPEAT
12580 IF! = Graph!(Graph%,2,M%)
12590 X! = (Q!) / (K! * Temp!)
12600 Y! = LOG (ABS(IF!) / Temp! ^ 2.0)
12610 OPEN "$BUCKET" FOR APPEND AS #3
12620 PRINT #3,X!;"",Y!
12630 CLOSE #File% : CLOSE #3
12632 IF INSTR(SOURCES",B")=0 THEN OPEN "$DATA\" + Directory$(Graph%,File%) + File$ FOR INPUT AS #File% ELSE OPEN "B:\" + File$ + ":\" + Directory$(Graph%,File%) FOR INPUT AS #File%
12635 Max_graph_points%(Graph%) = 1
12637 IF INSTR(Temp_type$(Graph%),"T") = 0 THEN Temp_type$(Graph%) = "L"
12640 IF Error% = 0 THEN EXIT TO,11900
12642 DO
12643 CLOSE #File%
12644 File% = File% + 1
12645 IF File% = 3 OR INSTR(Directory$(Graph%,2),"NO") <> 0 THEN EXIT 1
12646 LEVELS
12647 IF INSTR(Temp_type$(Graph%,"T") = 0 THEN Temp_type$(Graph%) = "R"
12648 EXIT TO,11900
12649 REPEAT
12650 DO
12651 Max_graph_points%(Graph%) = Counter%
12652 GETDATA(Graph%)
12653 Ytitle$ = "Ln (If / T**2) If in A"
12654 Xtitle$ = "q / KT"
12655 Graph_name$(1,17) = "Activation Energy"
12656 Bias$(Graph%) = STR$(VF!)
12657 EXIT 3 LEVELS
12658 REPEAT
12659 REPEAT
12660 DO
12661 IF Graph_chosen% = 3 THEN EXIT 1 LEVELS
12662 IF Graph_chosen% < 18 OR Graph_chosen% > 20 THEN EXIT 2 LEVELS
12663 END DO
12664 Counter% = Counter% + 1
12665 DO
12666 Position% = INSTR(Temp_found$,,") + 1
12667 Temp! = VAL(MIDS(Temp_found$,,Position%))
12668 DELTAMultiPLY(Graph%,1,(Q!/(K! * Temp!)))
12669 IF Counter% = 1 THEN ID% = 0 ELSE ID% = 1
12670 LEASTSQUARES
12671 (Graph%,Min!,Max!,Slope!,Dummy!,Dummy!,Y_int!,ID%,Error_LS%)
12672 IF Error_LS% = 1 THEN EXIT TO,160
12673 GRange$(Graph%) = STR$(Min!) + "," + STR$(Max!)
12674 END DO
12675 DO
12676 IF Graph_chosen% <> 18 AND Graph_chosen% <> 3 THEN EXIT 1 LEVELS
12677 DO
12678 IF Graph_chosen% = 3 THEN EXIT 1 LEVELS
12679 IF Y_int! < -39 OR Y_int! > 1 THEN Counter% = Counter% - 1 : EXIT
12680 TO,13313
12681 Y! = EXP(Y_int!)
12682 Ytitle$ = "Isat in Amps"
12683 Graph_name$(1,18) = "I Saturation"
12684 END DO
12685 DO
12686 IF Graph_chosen% <> 3 THEN EXIT 1 LEVELS
12687 Y! = Y_int!
12688 Ytitle$ = "ln(Isat) in Amps"
12689 Graph_name$(2,3) = "ln(Isat)"
END DO
EXIT 2 LEVELS
REPEAT
DO
IF Graph_chosen% <> 19 THEN EXIT 1 LEVELS
Y! = 1/Slope!
Ytitle$ = "Ideality Factor"
Graph_name$(1,19) = "Ideality Factor n"
EXIT 2 LEVELS
REPEAT
DO
IF Graph_chosen% <> 20 THEN EXIT 1 LEVELS
IF Y_int! < -39 OR Y_int! > 1 THEN Counter% = Counter% - 1 : EXIT TO,13313
Y! = (K! * Temp!/ Q!) * LOG (AeA! * Temp! ^ 2.0/EXP(Y_int!))
Ytitle$ = "Barrier in Volts"
Graph_name$(1,20) = "Barrier Height (Current)"
EXIT 2 LEVELS
REPEAT
END DO
DO
DO
IF INSTR(Temp_type$(Graph%), "T") <> 0 THEN EXIT 1 LEVELS
X! = Temp!
Xtitle$ = "Temperature in Kelvin"
EXIT 2 LEVELS
REPEAT
DO
Position% = INSTR(Time_found$,"=") + 1
X! = VAL (MIDS(Time_found$,Position%))
Xtitle$ = "Time in Seconds"
END DO
END DO
DO
OPEN "BUCKET" FOR APPEND AS #3
PRINT #3, X! ** Y!
CLOSE #File% : CLOSE #3
END DO
DO
Max_graph_points%(Graph%) = 1
CLOSE #File%
IF INSTR(SOURCES,$B")=0 THEN OPEN "DATA"+ Directory$(Graph%,File%) + File$ FOR INPUT AS #File% ELSE OPEN "B:" + File$ + ":"+Directory$(Graph%,File%) FOR INPUT AS #File%
IF INSTR(Temp_type$(Graph%),"T") = 0 THEN Temp_type$(Graph%) = "L"
IF Error% = 0 THEN EXIT TO,11900
END DO
DO
CLOSE #File%
File% = File% + 1
IF File% = 3 OR INSTR(Directory$(Graph%,2),"NO") <> 0 THEN EXIT 1 LEVELS
13336   IF INSTR(Temp_type$(Graph%),"T") = 0 THEN Temp_type$(Graph%) = "R"
13337   EXIT TO,11900
13338   REPEAT
13340   DO
13350   Max_graph_points%(Graph%) = Counter%
13360   GETDATA(Graph%)
13370   END DO
13380   EXIT 2 LEVELS
13385   END DO
13390   END DO
13400   DO
13410   IF Graph_chosen% > 17 AND Graph_chosen% < 21 OR Graph_chosen% = 3 THEN
13420     EXIT 1 LEVELS
13430   DO
13440   IF INSTR(Temp_type$(Graph%), "T") <> 0 THEN EXIT 1 LEVELS
13450   Position% = INSTR(Temp_found$, ";", 1) + 1
13460   Temp$(Graph%) = MIDS(Temp_found$, Position%, )
13470   EXIT 2 LEVELS
13480   REPEAT
13490   DO
13500   Position% = INSTR(Time_found$, ";", 1) + 1
13510   Tyme$(Graph%) = MIDS(Time_found$, Position%, 14)
13520   END DO
13530   END DO
13540   RETURN, 160
13550   REM
13560   DO
13570   IF Graph_chosen% > 20 AND Graph_chosen% < 25 THEN EXIT 1 LEVELS
13580   CLEARSCREEN(2)
13590   COLOR 7,0,0; SET CURSOR 13,6
13600   PRINT "The graph for the mobility experiment chosen is not installed (oops)"
13610   END
13620   REPEAT
13630   File% = 1
13640   File$ = "MOB"
13650   DO
13660   DIRECTORY(File$, 0, Graph%, Error%)
13670   END DO
13680   RUNTYPE(Graph%, 1)
13690   DO
13700   CLEARSCREEN(2)
13710   COLOR 2,0,0; SET CURSOR 10,6
13720   INPUT "Enter the samples electrical thickness. ", Dummy$
13730   Elec_thick! = VAL(Dummy$)
13740   IF Elec_thick! <= 0 THEN EXIT TO, 15200
13750   END DO
13760   DO
13770   IF Graph_chosen% <> 24 THEN EXIT 1 LEVELS
13780   CLEARSCREEN(16)
13790   SET CURSOR 16,6
13800   INPUT "Enter the average dopant density. ", Dummy$
15350   Nd! = VAL(Dummy$)
15360   IF Nd! <= 0 THEN EXIT TO, 15300
15370   END DO
15380   DATAPRINT
15390   DO
15400   DO
15410   IF INSTR(Temp_type$(Graph%),"T") = 0 THEN Temp_type$(Graph%) = "R"
15420   END DO
15430   IF INSTR(Directory$(Graph%, File%),"NO") <> 0 THEN EXIT 2 LEVELS
15440   INPUT #File%, Dummy$, IS, V1$, V2$, Dummy$
15450   R1! = VAL(V2$) / VAL(IS)
15460   INPUT #File%, Dummy$, IS, V1$, V2$, Dummy$
15470   R2! = VAL(V2$) / VAL(IS)
15480   INPUT #File%, Dummy$, IS, V1$, V2$, Dummy$
15490   Dummy! = MIN(R2!, R1!)
15500   Eric! = MAX(R2!, R1!)
15510   G! = 1.0 / ((3.141592654 * VAL(IS) / (2 * VAL(V1$) * .693147181) * (R1!+R2!)*(1-.138974234 * LOG(ABS(Eric! / Dummy!))))
15520   DATARETRIEVAL(9, Graph%, File%, Error%, 0)
15530   IF Error% <> 0 THEN EXIT TO, 15200
15540   END DO
15550   DO
15560   DO
15570   IF Graph_chosen% <> 21 THEN EXIT 1 LEVELS
15580   Graph_name$(1, 21) = "Resistivity"
15590   Ytitle$ = "Resistivity in ohms*cm"
15600   DO
15610   CLOSE #File%
15620   File% = File% + 1
15630   IF File% = 3 OR INSTR(Directory$(Graph%, 2), "NO") <> 0 THEN EXIT 1 LEVELS
15640   Max_graph_points%(Graph%) = Max_graph_points%(Graph%) + 1
15650   EXIT TO, 15500
15660   REPEAT
15670   EXIT 2 LEVELS
15680   END DO
15690   DO
15700   IF Graph_chosen% < 22 OR Graph_chosen% > 24 THEN EXIT 1 LEVELS
15710   CLOSE #File%
15720   DO
15730   IF INSTR(SOURCES$, "B") = 0 THEN OPEN "DATA\" + Directory$(Graph%, File%) + File$ FOR INPUT AS #File% ELSE OPEN "B:\" + File$ + "\" + Directory$(Graph%, File%) FOR INPUT AS #File%
15740   DO 15 TIMES
15750   INPUT #File%, Dummy$
15760   REPEAT
15770   END DO
15780   IF File% = 1 THEN Counter% = Max_graph_points%(Graph%) + 1
15790   IF File% = 1 THEN Max_graph_points%(Graph%) = 1
15800   IF File% = 2 THEN Max_graph_points%(Graph%) = Counter%
15870 IF INSTR(Directory$(Graph%,File%),"NO") <> 0 THEN EXIT 1 LEVELS
15880 DATA RETRIEVAL(10,Graph%,File%,Error%,0)
15890 IF Error% <> 0 THEN EXIT TO,15200
15900 END DO
15910 DO
15913 CLOSE #File%
15914 File% = File% + 1
15915 IF File% = 3 OR INSTR(Directory$(Graph%,2),"NO") <> 0 THEN EXIT 1 LEVELS
15916 Max_graph_points%(Graph%) = Max_graph_points%(Graph%) + 1
15917 EXIT TO,15500
15918 REPEAT
15919 DELTAMULTIPLY(Graph%,2,Elec_thick!)
15920 DO
15930 IF Graph_chosen% <> 22 THEN EXIT 1 LEVELS
15940 Ytitle$ = "Mobility in cm**2/V-s"
15945 Graph_name$(1,22) = "Mobility"
15950 EXIT 3 LEVELS
15960 END DO
15970 DO
15980 IF Graph_chosen% <> 23 AND Graph_chosen% <> 24 THEN EXIT 1 LEVELS
15990 OPEN "BUCKET" FOR INPUT AS #3
16000 FOR N% = 1 TO Max_graph_points%(Graph%)
16010 INPUT #3,Y!
16020 Graph!(Graph%,2,N%) = 1/(Q!/Y! * Graph!(Graph%,2,N%))
16030 NEXT N%
16040 DO
16050 IF Graph_chosen% <> 23 THEN EXIT 1 LEVELS
16060 Ytitle$ = "Carrier Conc in 1/cm**3"
16065 Graph_name$(1,23) = "Carrier Concentration n"
16065 EXIT 3 LEVELS
16070 EXIT 4 LEVELS
16080 END DO
16090 DO
16100 DELTAMULTIPLY(Graph%,2,100.0/Nd!)
16110 Ytitle$ = "% Activation"
16115 Graph_name$(1,24) = "Dopant Activation %"
16120 EXIT 3 LEVELS
16130 END DO
16140 REPEAT
16150 REPEAT
16160 END DO
16165 DO
16170 DO
16180 IF INSTR(Temp_type$(Graph%),"T") <> 0 THEN EXIT 1 LEVELS
16190 Xtitle$ = "Temperature in Kelvin"
16200 EXIT 2 LEVELS
16210 END DO
16220 DO
16230 Xtitle$ = "Time in Seconds"
16240 END DO
16245 END DO
16250 KILL "BUCKET"
16260 RETURN, 160
17000 REM
17010 DO
17020 IF Graph_chosen% = 6 OR Graph_chosen% = 7 THEN EXIT 1 LEVELS
17030 CLEARSCREEN(2)
17040 SET CURSOR 13, 1: COLOR 7, 0
17050 PRINT "The graph chosen has not been implemented (17000)."
17060 STOP
17070 REPEAT
17100 DO
17120 DIRECTORY("CG", 0, Graph%, Error%)
17130 IF Error% <> 0 THEN EXIT TO, 160
17135 RUNTYPE(Graph%, 1)
17240 END DO
17250 DO
17260 CLEARSCREEN(2)
17270 SET CURSOR 13, 6: COLOR 2, 0, 0
17280 INPUT "Enter the Bias Voltage. ", Dummy$
17285 Bias_sought! = VAL(Dummy$)
17290 END DO
17500 FOR N% = 1 TO 2
17510 IF INSTR(Directory$(Graph%, N%), "NO") <> 0 THEN EXIT 1 LEVELS
17512 DATAPRINT
17515 IF N% = 2 THEN Max_graph_points%(Graph%) = Max_graph_points%(Graph%) + 1
17520 IF Graph_chosen% = 6 THEN ID% = 4 ELSE ID% = 5
17530 DATARETRIEVAL(ID%, Graph%, N%, Error%, 0)
17540 IF Error% = 1 THEN EXIT TO, 17210
17550 NEXT N%
17700 DO
17710 IF Graph_chosen% <> 6 THEN EXIT 1 LEVELS
17720 DELTAMULTIPLY(Graph%, 2, 1E+09)
17730 Ytitle$ = "Capacitance in nF"
17735 Graph_name$(1, 6) = "Capacitance vs."
17740 END DO
17740 DO
17750 IF Graph_chosen% <> 7 THEN EXIT 1 LEVELS
17760 DELTAMULTIPLY(Graph%, 2, 1000.0)
17775 Graph_name$(1, 7) = "Conductance vs."
17780 Ytitle$ = "Conductance in mHos"
17790 END DO
17800 DO
17810 Position% = INSTR(Bias_found$, "," ) + 1
17820 Bias$(Graph%) = MIDS(Bias_found$, Position%)
17830 DO
17840 IF INSTR(Temp_type$(Graph%), "T") <> 0 THEN EXIT 1 LEVELS
17850 Xtitle$ = "Temperature in Kelvin"
17860 Graph_name$(1, Graph_chosen%) = Graph_name$(1, Graph_chosen%) + "Temperature"
17870 EXIT 2 LEVELS
17880 REPEAT
17890  DO
17900   Xtitle$ = "Time in Seconds"
17910   Graph_name$(1,Graph_chosen%) = Graph_name$(1,Graph_chosen%) + "Time"
17920  END DO
17930  END DO
17940  RETURN,160
20000  REM
20010  DO
20020  CLEARSCREEN(2)
20030  RESTORE,62000
20040  FOR N% = 4 TO 11
20050    READ Hand_info$(N%)
20060    COLOR 11,0 ; LOCATE N%,1
20070    PRINT Hand_info$(N%)
20080    NEXT N%
20090  END DO
20100  LOCATE 4,1 ; COLOR 0,7 ; PRINT Hand_info$(4)
20110  LOCATE 5,0 ; LOCATE 13,40 ; PRINT "MAKE SELECTION"
20120  LOCATE 4,1
20125  AS = INKEYS$ ; IF AS = "" THEN GOTO 20220
20130  IF ASC(RIGHT$(AS,1)) = 13 THEN GOTO 20300
20140  IF ASC(RIGHT$(AS,1)) = 72 THEN GOSUB 20400
20150  IF ASC(RIGHT$(AS,1)) = 80 THEN GOSUB 20500
20160  GOTO 20220
20170  ON CSRLIN .3 GOSUB 20600,20700,20800,20900,21100,22000,22600,22400
20180  PRINT CSRLIN : STOP
20190  Position% = CSRLIN
20200  COLOR 11,0 ; LOCATE Position%,1 ; PRINT Hand_info$(Position%)
20210  IF Position% < 5 THEN Position% = 12
20220  COLOR 0,7 ; LOCATE Position%-1,1 ; PRINT Hand_info$(Position%-1);
20230  RETURN,20220
20240  Position% = CSRLIN
20250  COLOR 11,0 ; LOCATE Position%,1 ; PRINT Hand_info$(Position%)
20260  IF Position% > 10 THEN Position% = 3
20270  COLOR 0,7 ; LOCATE Position%+1,1 ; PRINT Hand_info$(Position%+1);
20280  RETURN,20220
20290  GOSUB 22500
20310  DO
20320    LINE INPUT "Enter the TITLE of the graph (up to 60 letters) ";Graph_name$(1,1)
20325    Position% = INSTR(Graph_name$(1,1)," ") - 1
20330    Graph_name$(1,1) = MID$(Graph_name$(1,1),1,Position%) & Position%
20340    CLEARSCREEN (20)
20350  END DO
20360  END DO
20370  RETURN,20200
20370  GOSUB 22500
20370  DO
20370    LINE INPUT "Enter the X AXIS title (up to 60 letters) ";Xtitle$
20375    Position% = INSTR(Xtitle$," ") - 1
20377    Xtitle$ = MID$(Xtitle$,1,Position%)
20370    CLEARSCREEN(20)
20370  END DO
20750 RETURN, 20200
20800 GOSUB 22500
20810 DO
20820 LINE INPUT "Enter the Y AXIS title (up to 20 letters) ", Ytitle$
20825 Position% = INSTR(Ytitle$, " ") - 1
20827 Ytitle$ = MIDS(Ytitle$, 1, Position%) CLEARSCREEN(20)
20840 END DO
20850 RETURN, 20200
20900 GOSUB 22500
20920 PRINT "Enter the X, Y data point <RETURN> ".
20930 COLOR 7, 0
20940 FOR N% = Minloop% TO 1200
20947 SET CURSOR 21, 43: PRINT SPC(34)
20950 SET CURSOR 21, 43
20955 DO
20960 LINE INPUT VS
20961 IF VS = " " THEN EXIT TO 21005
20962 Pointer% = INSTR(VS, ",")
20963 IF Pointer% = 0 THEN EXIT TO 21005
20964 END DO
20965 Graph!(Graph%, 1, N%) = VAL(MIDS(VS, 1, Pointer%))
20966 Graph!(Graph%, 2, N%) = VAL(MIDS(VS, Pointer% + 1))
20985 SET CURSOR 21, 43: PRINT SPC(34)
20990 CLEARSCREEN(22)
21000 NEXT N%
21005 Minloop% = N%
21007 Max_graph_points%(Graph%) = N% - 1
21010 CLEARSCREEN (21)
21020 RETURN, 20200
21050 RETURN
21100 GOSUB 22500
21110 SET CURSOR 3, 40: COLOR 5, 0: PRINT "Changing X-Y data"
21120 Minloop% = 1
21130 FOR N% = Minloop1% TO Max_graph_points%(Graph%)
21140 COLOR 2, 0: SET CURSOR N% + 5 - Minloop1%, 45
21150 PRINT Graph!(Graph%, 1, N%) ", Graph!(Graph%, 2, N%)
21155 IF N% + 5 - Minloop1% > 17 THEN EXIT 1 LEVELS
21160 NEXT N%
21170 Maxloop% = N% - 1
21200 FOR N% = Minloop1% TO Maxloop%  
21210 COLOR 0, 2: SET CURSOR N% + 5 - Minloop1%, 45
21220 PRINT Graph!(Graph%, 1, N%) ", Graph!(Graph%, 2, N%)
21225 COLOR 0, 0: CLEARSCREEN(20)
21230 SET CURSOR N% + 5 - Minloop1%, 45
21240 COLOR 0, 2: LINE INPUT VS
21245 COLOR 0, 0: CLEARSCREEN(20)
21250 COLOR 0, 0: Position% = INSTR(VS, ",") + 1
21260 IF Position% = 1 THEN EXIT TO 21400
21270 Graph!(Graph%, 1, N%) = VAL(MIDS(VS, 1, Position%))
21280 Graph!(Graph%, 2, N%) = VAL(MIDS(VS, Position%))
COLOR 2,0 : SET CURSOR N%+%5-Minloop1%,45
PRINT SPC(35)
COLOR 2,0 : SET CURSOR N%+%5-Minloop1%,45
PRINT Graph!(Graph%,1,N%) "," Graph!(Graph%,2,N%)
CLEARSCREEN(20)
NEXT N%
DO
IF Maxloop% => Max_graph_points%(Graph%) THEN EXIT TO,21500
FOR N% = 4 TO 23
COLOR 7,0 : SET CURSOR N%,30
PRINT SPC(50)
NEXT N%
Minloop1% = Maxloop%
EXIT TO,21130
REPEAT
FOR N% = 2 TO 23
COLOR 7,0 : SET CURSOR N%,30
PRINT SPC(50)
NEXT N%
RETURN,20200
GOSUB 22500
DO
Choice% = 1:Saveflag=1
SAVEDATA(Graph%)
END DO
SET CURSOR 13,40 : PRINT SPC(40)
CLEARSCREEN(16)
CLOSE #1
RETURN,20200
COLOR 7,0
CLEARSCREEN(2)
RETURN,160
REM
COLOR 11,0 : LOCATE CSRLIN,1 : PRINT Hand_info$(CSRLIN)
DO
COLOR 0,0 : CLEARSCREEN(12)
SET CURSOR 13,40:PRINT SPC(39)
SET CURSOR 21,6 : COLOR 2,0
END DO
RETURN
STOP
GOSUB 22500
LOADDATA(Graph%)
RETURN,20200
STOP
REM
DO
IF Graph_chosen% <> 10 OR Graph_chosen% <> 2 THEN Directory$(Graph%,2) = "NO"
END DO
DO
IF Graph_chosen% = 10 OR Graph_chosen% = 2 THEN ID% = 0 ELSE ID% = 1

DIRECTORY("CGV",ID%,Graph%,Error%)

IF Error% <> 0 THEN EXIT TO.160

END DO

DO

IF Graph_chosen% = 10 OR (Graph_chosen% <> 5 AND Graph_chosen% <> 4 AND Graph_chosen% <> 2 AND Graph_chosen% <> 8 AND Graph_chosen% <> 9) THEN ID% = 1 ELSE ID% = 0

RUNTYPE(Graph%,ID%)

END DO

DO

IF Graph_chosen% > 2 AND Graph_chosen% < 6 THEN EXIT 1 LEVELS

CLEARSCREEN(2)

SET CURSOR 13,6 : COLOR 2,0,0

INPUT "Enter the area of the device in cm squared. ".Area$

Area! = VAL(Area$)

IF Area! = 0 THEN EXIT TO.23650

END DO

DO

IF Graph_chosen% <> 10 AND Graph_chosen% <> 9 AND Graph_chosen% <> 2 THEN EXIT 1 LEVELS

CLEARSCREEN(15)

SET CURSOR 15,6 : INPUT "Enter the type of material you have tested (GaAs, Si)":Name$

Name$ = UPPERS(Name$)

IF INSTR(Name$,"GAAS") <> 0 AND INSTR(Name$,"Si") = 0 THEN EXIT TO.23671

IF INSTR(Name$,"G") <> 0 THEN RESTORE,61000 ELSE RESTORE,61500

READ Nc!,Dummy!

E_sub_S! = E_sub_O! * Dummy!

IF Graph_chosen% = 9 THEN EXIT 1 LEVELS

CLEARSCREEN (17)

SET CURSOR 17,6 : INPUT "Enter the donor density (Nd) ":Nd$

Nd! = VAL(Nd$)

IF Nd! = 0 THEN GOTO 23682

END DO

DATAPRINT

DO

DO

IF Graph_chosen% < 10 AND Graph_chosen% <> 2 THEN EXIT 1 LEVELS

IF INSTR(Temp_type$(Graph%),"T") = 0 THEN Temp_type$(Graph%) = "R"

IF INSTR(Temp_type$(Graph%),"T") = 0 THEN Run_type% = 1 ELSE Run_type% = 2

EXIT 2 LEVELS

REPEAT

Run_type% = 0

END DO

File% = 1

DO

IF Graph_chosen% = 5 THEN ID% = 3 ELSE ID% = 2

DATARETRIEVAL(ID%,Graph%,File%,Error%,Run_type%)

IF Error% <> 0 AND Graph_chosen% = 10 THEN EXIT 1 LEVELS
IF Error% <> 0 AND Graph_chosen% = 2 THEN EXIT 1 LEVELS
 IF Error% <> 0 THEN EXIT TO 23540
 END DO
 DO
 IF Graph_chosen% <> 4 THEN EXIT 1 LEVELS
 DELTAMULTIPLY(Graph%2,1E+09)
 Ytitle$ = "Cap in nF"
 EXIT 2 LEVELS
 REPEAT
 DO
 IF Graph_chosen% <> 5 THEN EXIT 1 LEVELS
 DELTAMULTIPLY(Graph%2,1E+03)
 Ytitle$ = "Cond in mMho"
 EXIT 2 LEVELS
 REPEAT
 DO
 IF Graph_chosen% <> 8 AND Graph_chosen% <> 1 THEN EXIT 1 LEVELS
 IF Graph_chosen% <> 8 THEN Power! = 3.000 ELSE Power! = 2.0000
 INVERSEPPOWER(Graph%,2,10E+09/Area!,Power!)
 IF Graph_chosen% = 8 THEN Ytitle$ = "1/C**2 C in nF" ELSE Ytitle$ = "1/C**3 in nF"
 EXIT 2 LEVELS
 REPEAT
 DO
 IF Graph_chosen% <> 9 THEN EXIT 1 LEVELS
 INVERSEPPOWER(Graph%,2,1.0/Area!,2.000)
 MinX! = 1E+37 : MaxX! = -1E+37
 MinY! = 1E+37 : MaxY! = -1E+37
 FOR N% = 1 TO Max_graph_points%(Graph%)-1
 Dummy! = E_sub_S! * SQR(Graph!(Graph%,2,N%))
 Slope! = (Graph!(Graph%,1,N%+1) - Graph!(Graph%,1,N%)) / (Graph!(Graph%,1,N%+1) - Graph!(Graph%,1,N%))
 Graph!(Graph%,1,N%) = Dummy!
 IF Slope! = 0 THEN GOTO 24120
 Graph!(Graph%,2,N%) = (-2/(Q! * E_sub_S!)) * 1/Slope!
 NEXT N%
 Max_graph_points%(Graph%) = Max_graph_points%(Graph%) - 1
 Ytitle$ = "Dopant Density in 1/cm**3 X 1e+10"
 Xtitle$ = "Depth in microns"
 DELTAMULTIPLY(Graph%2,1E-10)
 EXIT 2 LEVELS
 REPEAT
 DO
 IF Graph_chosen% <> 10 AND Graph_chosen% <> 2 THEN EXIT 1 LEVELS
 Counter% = Counter% + 1
 IF Graph_chosen% <> 10 THEN Power! = 3.0000 ELSE Power! = 2.0000
 INVERSEPPOWER(Graph%,2,1/Area!,Power!)
 DELTAMULTIPLY(Graph%2,1E-15)
 IF Counter% = 1 THEN ID% = 0 ELSE ID% = 1

LEASTSQUARES(Graph%, Min!, Max!, Dummy!, Slope!, X_int!, Y_int!, ID%, Error_LS%) 
IF Counter% = 1 THEN Range$(Graph%) = STR$(Min!) + "," + STR$(Max!) 
IF Error_LS% = 1 THEN RUN,200 
Position% = INSTR(Temp_found$, ",") + 1 
Temp! = VAL(MID$(Temp_found$, Position%)) 
Y! = X_int! + ((K! * Temp!) / Q!) * LOG(Nc! / Nd!) 
DO 
  IF INSTR(Temp_type$(Graph%), ",") <> 0 THEN EXIT 1 LEVELS 
  X! = Temp! 
  EXIT 2 LEVELS 
  REPEAT 
  DO 
    Position% = INSTR(Time_found$, ",") + 1 
    X! = VAL(MID$(Time_found$, Position%)) 
  END DO 
  END DO 
OPEN "BUCKET" FOR APPEND AS #3 
PRINT #3,X!,",Y!
CLOSE #3 : CLOSE #File% 
PRINT File%, Graph1or2% 
IF INSTR(Directory$(Graph%, File%), ",NO") <> 0 THEN EXIT 1 LEVELS 
OPEN "C:\DATA!" + Directory$(Graph%, File%) + ",CGV" FOR INPUT AS #File% 
IF INSTR(Temp_type$(Graph%), ",T") = 0 THEN Temp_type$(Graph%) = ",LH" 
Max_graph_points% (Graph%) = 1 
IF Error% = 0 THEN EXIT TO,23705 
DO 
  File% = File% + 1 
  IF File% = 3 OR INSTR(Directory$(Graph%, 2), ",NO") <> 0 THEN EXIT 1 LEVELS 
  CLOSE #1 : Error% = 0 
  IF INSTR(Temp_type$(Graph%), ",T") = 0 THEN Temp_type$(Graph%) = ",R" 
  EXIT TO,23705 
  REPEAT 
  DO 
    Max_graph_points% (Graph%) = Counter% 
    GETDATA(Graph%) 
    Graph_name$(1,10) = "Barrier Height (Capacitive)" 
    Ytitle$ = "Barrier in Volts" 
    IF INSTR(Temp_type$(Graph%), ",T") THEN Xtitle$ = "Time in Seconds" ELSE Xtitle$ = "Temperature in Kelvin" 
    EXIT TO,3000 
  END DO 
  END DO 
  END DO 
IF Graph_chosen% = 9 THEN EXIT TO,24560 
Xtitle$ = "Voltage in Volts" 
DO 
  IF Temp_type$(Graph%) = ",T" THEN EXIT 1 LEVELS 
  Position% = INSTR(Temp_found$, ",") + 1
TempS(Graph%) = MID$(Temp_found$, Position%) 
EXIT 2 LEVELS 
REPEAT 
DO 
Position% = INSTR(Time_found$, '*') + 1 
TymeS(Graph%) = MID$(Time_found$, Position%) 
END DO 
EXIT TO, 160 
END DO 
RETURN, 160 
REM 
DIRECTORY("RES", 0, Graph%, Error%) 
IF Error% <> 0 THEN EXIT TO, 160 
RUNTYPE(Graph%, 1) 
END DO 
DO 
CLEARSCREEN(2) 
SET CURSOR 13, 6 : COLOR 2, 0, 0 
INPUT "Enter the Current Bias (mA)", Dummy$ 
Bias_sought! = VAL(Dummy$) 
SET CURSOR 15, 6 : COLOR 2, 0, 0 
INPUT "Enter the 4-pt. Correction Factor(CF: pg. 31 of Sze)", Dummy$ 
CF = VAL(Dummy$) 
SET CURSOR 17, 6 : COLOR 2, 0, 0 
INPUT "Enter the Sample thickness", Dummy$ 
Thick! = VAL(Dummy$) 
END DO 
FOR N% = 1 TO 2 
IF INSTR(Directory$, Graph%, N%, "NO") <> 0 THEN EXIT 1 LEVELS 
DATAPRINT 
IF N% = 2 THEN Max_graph_points%(Graph%) = Max_graph_points%(Graph%) + 1 
ID% = 4 
DATARETRIEVAL(ID%, Graph%, N%, Error%, 0) 
IF Error% = 1 THEN EXIT TO, 25210 
NEXT N% 
DO 
DELTA乘多PLY(Graph%, 2, CF*Thick!/(0.01*Bias_sought!)) 
Ytitle$ = "Resistivity in ohm*cms" 
Graph_name$(1, 6) = "Resistivity vs." 
END DO 
DO 
Position% = INSTR(Bias_found$, '*') + 1 
Bias$(Graph%) = MID$(Bias_found$, Position%) 
DO 
IF INSTR(Temp_type$(Graph%), 'T') <> 0 THEN EXIT 1 LEVELS 
Xtitle$ = "Temperature in Kelvin" 
Graph_name$(1, Graph_chosen%) = Graph_name$(1, Graph_chosen%) + "Temperature" 
EXIT 2 LEVELS 
REPEAT 
END DO
Vita

Phillip Johnson is a prince of a man. A true giant in Lilliput. There never has been or will be another man like him (barring people in towers with rifles). But I regress, how can you judge a man without knowing his background? Phil was born and did lots of boring stuff then he got a drivers license and then things really got interesting. Yes a true AMERICAN with a ’69 Camaro Super Sport with a 396 cid engine and Mag wheels. Life began at this point and probably almost ended. Stories are probably still being told around the mighty metropolis of Arlington. We were thinking of entering some of his exploits in "stupid people tricks". Then his life changed for the better he entered VPI&SU. Life just doesn’t get any better than this. He decided to devote his life to more philosophical pursuits, bike riding, beer drinking, and aluminum can avoiding. This was the height of his life. After this he did more boring things like got meaningful employment, got married, had lots of children and named them after Doonesbury characters. Bye!!  - Chris Turman

p.s. I had nothing to do with this vita.