Electronic Conduction at Grain Boundaries in Silicon: Theory and Computer Aided Analysis

by

G.C. McGonigal

A thesis
presented to the University of Manitoba
in partial fulfillment of the
requirements for the degree of
Master of Science

Winnipeg, Manitoba, 1983

ELECTRONIC CONDUCTION AT GRAIN BOUNDARIES IN SILICON: THEORY AND COMPUTER AIDED ANALYSIS

BY

GORDON C. McGONIGAL

A thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

ABSTRACT

An experimental investigation of electronic conduction across isolated grain boundaries in large grain polycrystalline silicon is reported. A microcomputer-based automated data acquisition and processing (ADAP) system was designed and constructed to facilitate the measurements. An emission/diffusion conduction model has been developed to include the rates of carrier supply and collection at the grain boundary. Interface-state densities of approximately 1.0×10^{12} $eV^{-1} cm^{-2}$ have been measured. Activation energy measurements of the potential barrier suggest that a spacially-nonuniform distribution of charge exists over the plane of most grain boundaries. Documentation of the ADAP system, and of several algorithms employed in data reduction are included in the Appendices.

ACKNOWLEDGMENTS

I wish to extend my sincerest thanks to Professor H.C. Card for his guidance throughout the course of this work. Discussions with the other members of the Materials and Devices Research Group, especially Doug Thomson, Bob McLeod, Peter Hsieh, Lex De Groot, and Professor K.C. Kao are gratefully acknowledged. I would also like to thank Doug Thomson for his assistance with the experimental portions of this work.

LIST OF FIGURES

CONTENTS

Chapter I

INTRODUCTION

Recent applications of polycrystalline silicon in integrated circuit and solar cell devices [1,2] have sparked a renewed interest in the electronic characterization of these materials. With respect to solar cell applications, commercially available polycrystalline devices have shown energy conversion efficiencies comparable to those of single-crystal devices - but at a fraction of the cost.

Polycrystalline materials are constructed of grains of crystalline material that are randomly oriented with respect to one another. The interfaces between adjacent grains are called grain boundaries. Typical dimensions of the individual grains range from hundreds of angstroms to several millimeters depending upon the conditions of formation.

Usually polycrystalline semiconductors are modelled as volumes of crystalline material separated by grain boundaries. The properties of crystalline semiconductors are relatively well known and therefore the study of polycrystalline material should center upon a characterization of the grain boundary.

Most of the physical models of grain boundaries can be classified into one of two main subgroups: those that construct an amorphous (i.e. non-crystalline) intermediate layer between the grains [3]; and those that ignore such a layer and assume the crystal structure preserved to within one or two atomic spacings of the boundary [4]. The amorphous layer model might be justified for certain exceptional conditions such as heavy doping segregation to the boundary but TEM photographs of germanium grain boundaries by Krivanek [5] and silicon grain boundaries by Cunningham and Ast [6] coupled with intuitive feelings of the melt-zone crystal formation process lead us - and the majority of our co-workers - to adopt the latter model.

The grain boundary defines the interface between two unaligned crystal lattices. It is a discontinuity to the periodicity of each lattice and therefore the energy-band structure of each is not preserved locally. Bonds of varying length are formed at the interface. These "distorted" bonds manifest themselves as a continuum of energy levels lying within the forbidden gap. Such energy levels are refered to as interface states. Their interaction with electrical carriers is analogous to that of states introduced through bulk impurity doping [7] with the exception that interface states are spacially confined to the grain boundary.

Interface states can be classified as being either donor- or acceptor- like. Donor-like states are neutral when occupied by an electron and positively charged when empty. Conversly, acceptor-like states are negative when filled and neutral when empty. Under equilibrium conditions the states below the Fermi-level will be occupied, while the states above will be empty. If the number of empty donor-like states differs from that of filled acceptor-like states the grain boundary will hold a net charge. For polycrystalline silicon this charge is invariably positive for p-type doping and negative for n-type doping [4].

In order to maintain space-charge neutrality, a depletion region is formed on both sides of the grain boundary. The charge associated with the overall depletion region is equal in magnitude, but opposite in sign, to the net charge in the interface states.

We use the space-charge approximation [8] to see how this charge affects the energy bands in the vicinity of the grain boundary. Solving Poisson's equation we find the potential at the grain boundary V(x) to be

$$V(x) = \frac{qN_A}{2\varepsilon_s} x^2 \qquad (1.1)$$

where q is the electronic charge, N_A is the bulk impurity doping (assuming p-type material), and ε_s is the silicon permittivity. Figure 1.1 shows the corresponding energy-band diagram, with V_{do} the equilibrium diffusion potential. If we assume the acceptor- and donor-like states to be uniformly distributed in energy within the energy gap and of density N_{is} ($eV^{-1}cm^{-2}$), we can generate the relationship between the diffusion potential and the interface-state density as shown as a function of N_A and N_{is} in Fig. 1.2 .

When an external voltage V_a is applied across a grain boundary as shown in Fig. 1.1 a portion of V_a will reduce the barrier on what is termed the forward-biased side of the boundary, while the remaining voltage increases the barrier on the reverse-biased side.

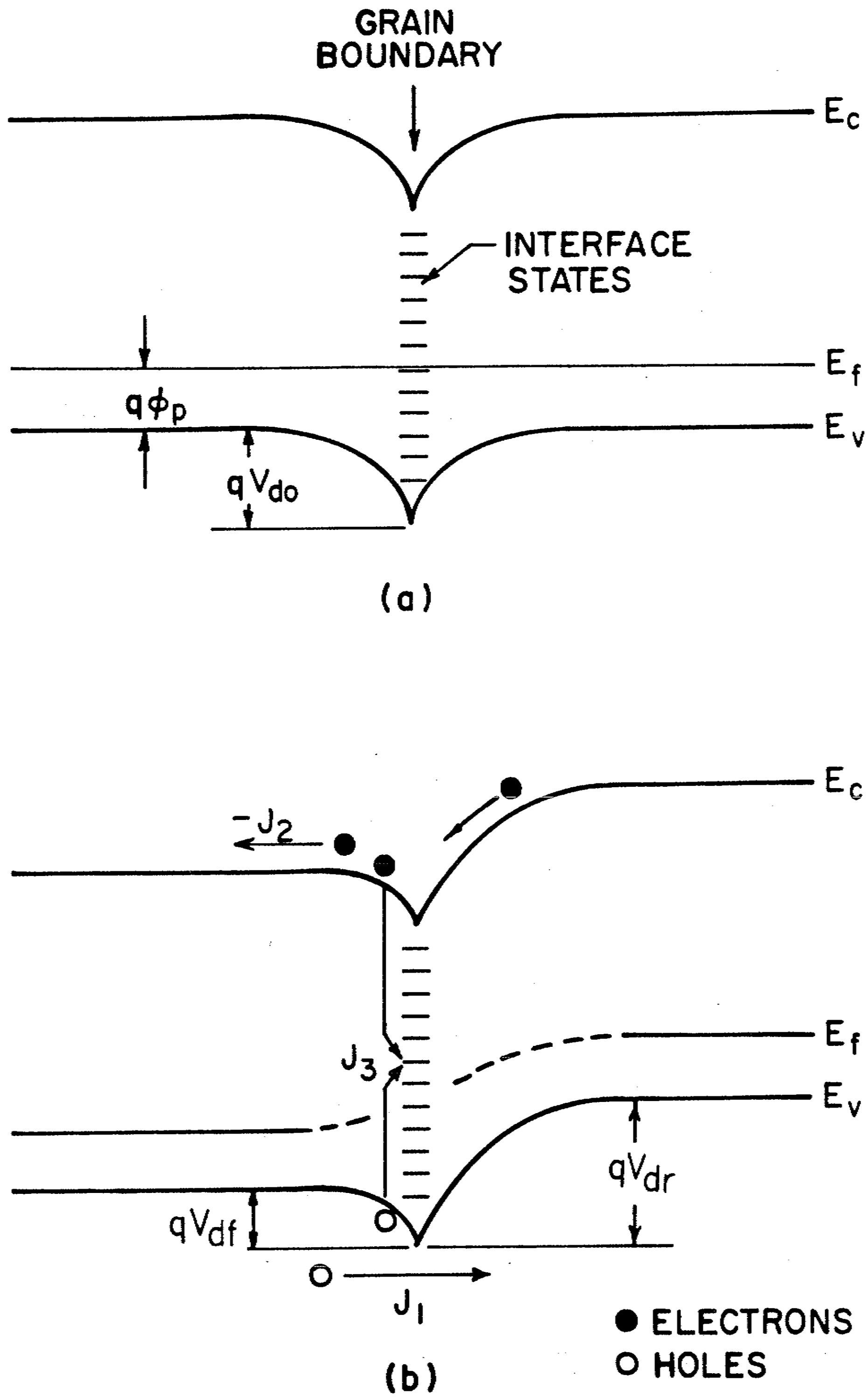


Figure 1.1: Electron energy-band diagrams for grain boundary with diffusion potential V_{do}. (a) V=0 ; (b) V>0.

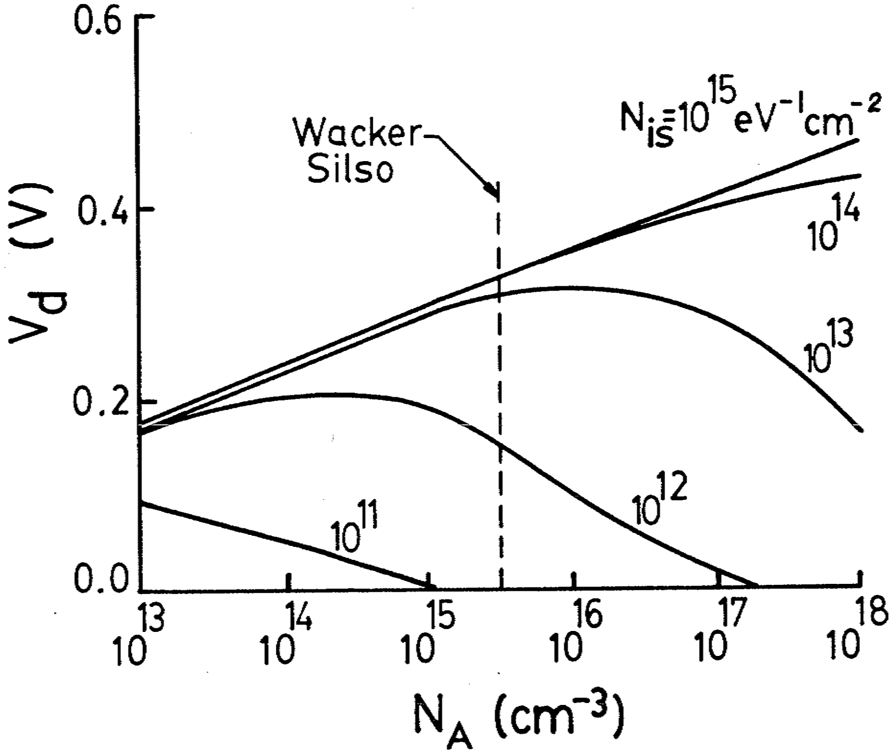


Figure 1.2: Diffusion potential V_{do} vs. bulk doping concentration N_A for several uniform interface-state densities N_{is}.

Chapter II

CARRIER TRANSPORT AT GRAIN BOUNDARIES

In this chapter we concentrate on developing an understanding of the mechanisms of carrier transport across (perpendicular to) a grain boundary. Here we isolate a grain boundary and ignore any interaction it may have with the other grain boundaries in the bulk polycrystalline material.

The steady-state current at a grain boundary consists of three distinct components. J_1 is the current due to transport of majority carriers over the potential barrier. J_2 is the component due to transport of minority carriers in the opposite direction. J_3 is the recombination current through the grain-boundary interface states. Figure 1.1 illustrates the three components. Under most conditions J_1 will dominate and the direct contribution of J_2 and J_3 to the total current can be ignored. These currents do however indirectly influence the magnitude of the majority-carrier current J_1 (as will be shown later).

2.1 THE EMISSION/DIFFUSION THEORY

The resemblence of the grain-boundary energy-band diagram to that of a pair of "back-to-back" Schottky-barrier contacts has lead many researchers [9-12] to adopt a current-voltage dependence modelled after the thermionic emission theory [13]. This is usually stated in the form

$$J = A^{*}T^{2} \exp\left(\frac{-\phi_b}{V_T}\right) \left[\exp\left(\frac{V_a}{V_T}\right) - 1\right] \tag{2.1}$$

where A^{*} is the modified Richardson constant $\simeq 30\ \mathrm{Acm^{-2}K^{-1}}$ for p-type silicon, ϕ_b is the barrier height defined as the potential difference between the Fermi potential and the valence band edge at the grain boundary under zero bias, V_a is the forward voltage, and V_T=kT/q is the thermal voltage.

The thermionic emission theory assumes there are a limited number of states capable of accepting a majority carrier on the metal side of the Schottky barrier. The reason is that there are very few available states in the metal with momentum parallel to the boundary to match that of the majority carriers from the semiconductor. This is because the energy at which the carriers are transfered is close to the valence band edge in the semiconductor but remote from the band edges in the metal.

Our physical model of the grain boundary is however devoid of any metal. Instead, silicon is both the emitting and accepting material. The energy-momentum transfer restrictions are greatly reduced. Moreover the reverse-biased side of the barrier acts as an efficient collector of majority carriers. After a majority carrier crosses the grain boundary it is swept into the bulk material by the large electric field of the depletion region. These objections to the applicability of the thermionic-emission theory have led us to a more general theory based on a combined emission/diffusion transport mechanism [14].

We observe that the magnitude of the current across a grain boundary depends upon the rate at which carriers can be removed from the boundary as well as the rate at which they are supplied. We will use the term emission/diffusion to describe a carrier transport model that incorporates both of these processes.

The majority-carrier current as given by the emission/diffusion theory is

$$J_1 = qN_A \frac{v_d v_r}{v_d + v_r} \exp\left(\frac{-V_{df}}{V_T}\right) \left[1 - \exp\left(\frac{-V_a}{V_T}\right)\right] \tag{2.2}$$

where N_A is the bulk acceptor impurity doping, v_r is the grain boundary majority-carrier collection velocity, v_d is the diffusion velocity for majority carriers on the forward-biased side of the grain boundary, and V_{df} is the diffusion potential on the forward-biased side. Equation (2.2) is an adaptation of the theory of carrier transport at grain boundaries proposed by Taylor, Odell, and Fan [15].

The diffusion velocity of majority carriers in the forward-biased depletion region v_d is

$$v_d = \mu E_{mf} \tag{2.3}$$

where μ is the majority-carrier mobility and E_{mf} is the maximum electric field. Similarly, the diffusion velocity away from the grain boundary on the reverse-biased side is given by

$$v_r = \mu E_{mr} \tag{2.4}$$

Note that v_d decreases as V_a increases. Thus $v_d < v_r$ which differs from the usual Schottky-barrier case. Using the depletion approximation

$$E_{mf} = \left(\frac{2qN_A}{\varepsilon_s} (V_{df} - V_T) \right)^{1/2} \tag{2.5}$$

and

$$E_{mr} = \left(\frac{2qN_A}{\varepsilon_s} (V_{dr} - V_T) \right)^{1/2} \tag{2.6}$$

where V_{dr} is the diffusion potential on the reverse-biased side and ε_s is the silicon permittivity.

2.2 THE EFFECT OF INTERFACE STATES

The amount of charge trapped in the grain-boundary interface states is not constant under all conditions. A change in the energy of the Fermi-level (due to a variation of temperature for example) will shift the distribution of interface states occupied by electrons. The net charge at the grain boundary is altered, resulting in a modification of the potential barrier V_{do}.

A similar effect occurs when the grain-boundary Fermi-level is disturbed by an applied voltage. In p-type material, the applied voltage will force the Fermi-level closer to the valence band edge, reducing the barrier height. This motion of the Fermi-level will however cause more interface states to empty, which tends to increase the barrier height. The "equilibrium" position of the Fermi-level will be determined by the interface-state density N_{is}. If N_{is} is small the barrier height can be reduced without adding significant charge to the grain boundary. If N_{is} is relatively large however, a small excursion of the Fermi-level will induce enough added charge to virtually maintain the barrier height. The Fermi-level is said to be "pinned" when this is the case. Equation (2.2) illustrates the role that the barrier height V_{df} plays in determining the current across the grain boundary. The interface-state distribution therefore indirectly contributes to the current-voltage characteristics.

The physics is somewhat complicated by "splitting" of the Fermi-level into separate electron and hole imrefs (quasi-Fermi-levels) under the nonequilibrium condition. Numerical calculations by Shaw (in [14]) have however revealed that the imrefs tend to intersect at the grain boundary. Knowing this we can still represent the carrier concentrations at the grain boundary with a single Fermi-level, although its exact energy under nonequilibrium conditions remains analytically uncertain.

2.3 NUMERICAL SIMULATION

A numerical simulation proved useful for visualizing certain aspects of the emission/diffusion carrier transport mechanism. The program, which used (2.2) as well as a consideration of the occupation of the interface states and the field dependence of the mobility, was implemented on an Apple II+ microcomputer (sec. 3.4). Parameter values were chosen to match the experimental samples measured later. For these calculations the impurity doping was $N_A=3x10^{15}$ cm^{-3} with an equilibrium diffusion potential of $V_{do}=0.28$ V. The interface-state density N_{is} was chosen to be independent of energy and of magnitude $1.0x10^{11}$ $eV^{-1}cm^{-2}$.

Figure 2.1 details how the applied voltage is shared between the forward- and reverse-biased sides of the grain boundary. We observe that at low voltages each side of the boundary consumes half of the applied voltage. As the voltage is increased, more than half of the applied voltage is developed across the reverse-biased depletion region at the expense of the forward-biased depletion region.

The maximum electric field on each side of the grain boundary is shown in Fig. 2.2 as a function of applied voltage. We note that although the magnitude of the electric field is always less than the value corresponding to a scattering-limited velocity ($\approx 10^5$ Vcm^{-1}), the carrier mobility cannot be considered to be independent of the electric field [8].

The preexponential term $v_d/(v_d+v_r)$ as a function of applied voltage is shown in Fig. 2.3 . This has a very weak voltage dependence, changing from 1/2 to approximately 1/3 as V_a increases to one volt.

Finally we note the voltage dependence of the Fermi-potential at the grain boundary as shown in Fig. 2.4 . With large N_{is} the Fermi-potential will be pinned at a fixed level but at the moderate density chosen here we see a shift towards the valence band with applied voltage. As previously noted, this will reduce the number of electrons trapped in the grain boundary interface states.

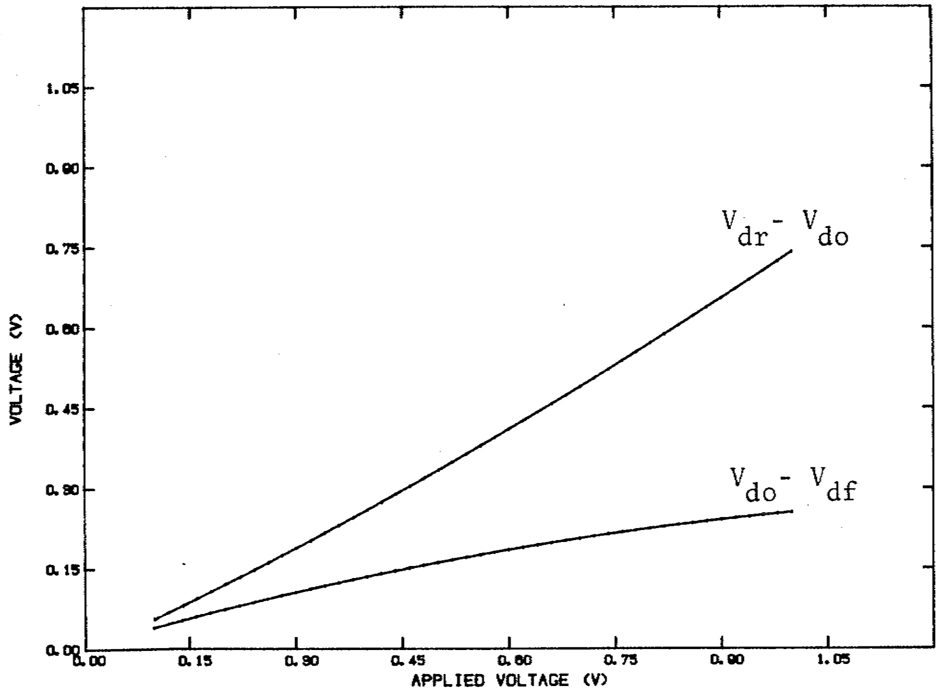


Figure 2.1: Voltage developed across the forward- and reverse-biased sides of a grain boundary as a function of applied voltage. $V_{dr} = V_a + V_{df}$.

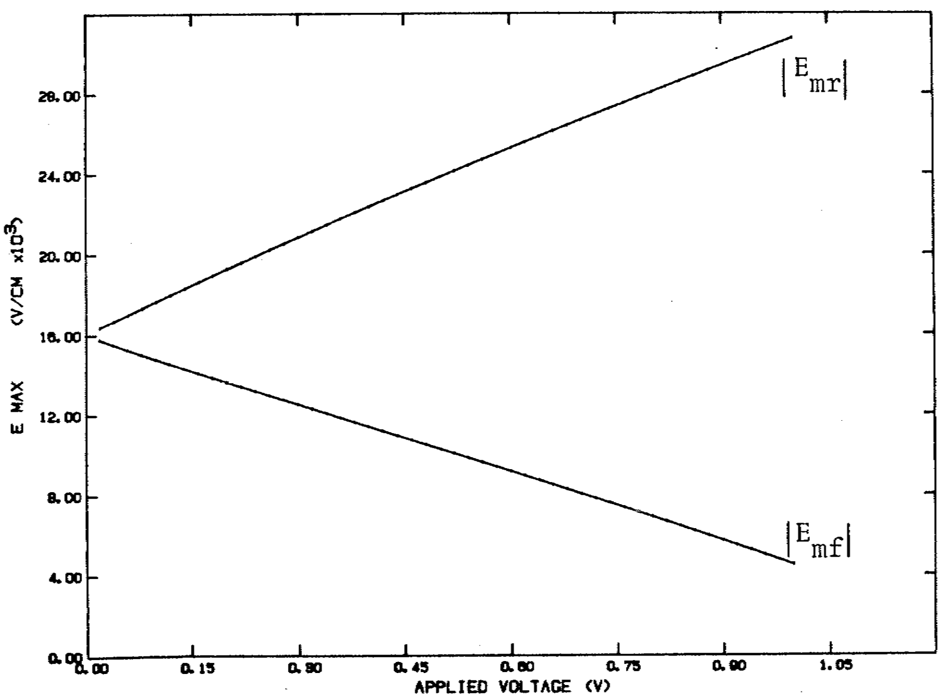


Figure 2.2: Maximum electric field at the forward- and reverse-biased sides, E_{mf} and E_{mr} respectively, as a function of applied voltage.

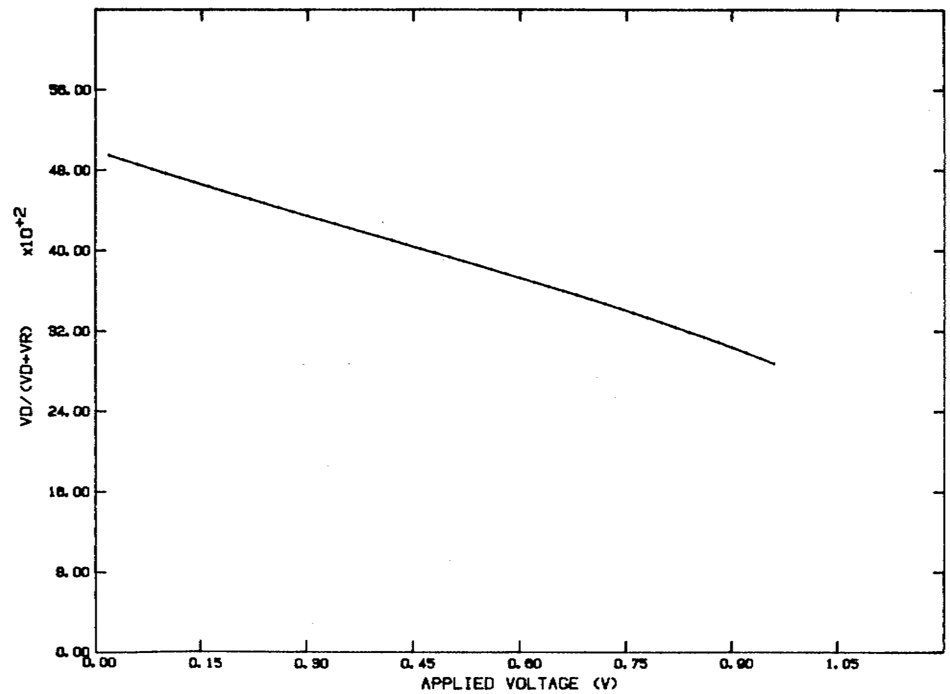


Figure 2.3: Preexponential factor $v_d/(v_d+v_r)$ as a function of applied voltage.

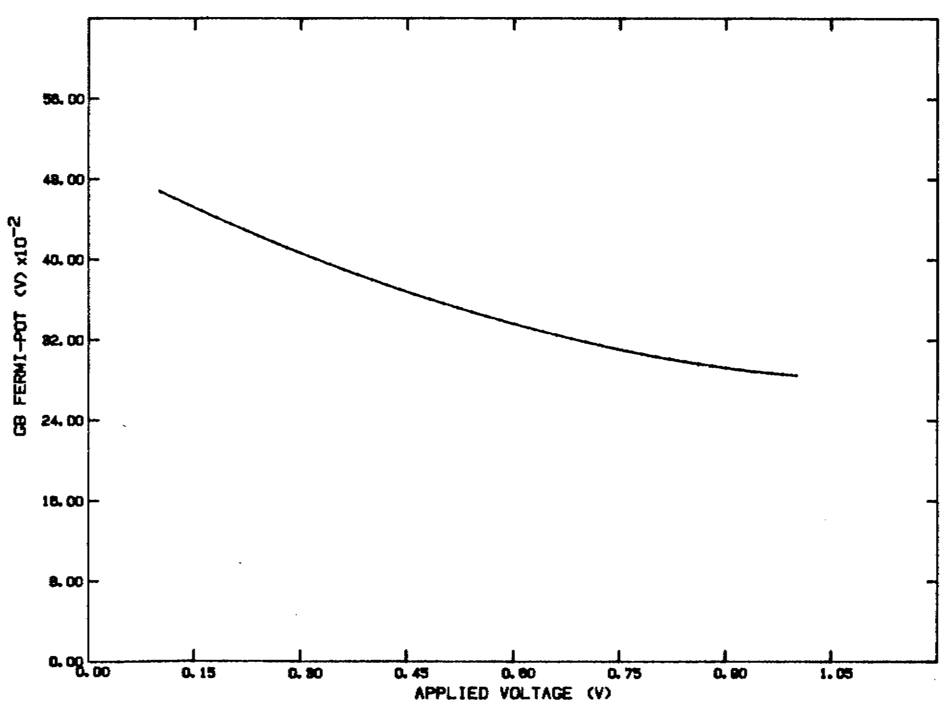


Figure 2.4: Fermi-potential at the grain boundary as a function of applied voltage. $N_{is}=1.0\times10^{11}$ $eV^{-1}cm^{-2}$.

2.4 ELECTROOPTICAL EFFECTS

The effects of minority carriers on the current transport at grain boundaries become important under conditions of optical illumination. We stress that minority carriers play an *indirect* role in determining the current by influencing the amount of charge trapped at the grain boundary. This determines the barrier height which in turn controls the majority-carrier current that dominates the total current.

The vicinity of the isolated grain boundary is illuminated such that there is a uniform volume photogeneration rate G_{ph}. If we assume that all minority carriers generated within a diffusion length L_n on either side of the grain boundary will be collected at the grain boundary due to the electric field of the depletion regions then for p-type material [16]

$$2L_nG_{ph} = \sigma_N n(0) v_{th} N_{is} \Delta E \qquad (2.7)$$

where σ_N is the capture cross-section for neutral traps, v_{th} is the thermal velocity of electrons, N_{is} is the interface-state density which will be assumed to be independent of energy for this analysis, and ΔE is the energy difference of the imrefs. Note that the left-hand side of (2.7) is the total flux of generated minority carriers to the grain boundary. The majority-carrier concentration at the grain boundary p(0) is

$$p(0) = N_V \exp\left(\frac{-\phi_p(0)}{V_T}\right) \qquad (2.8)$$

where N_V is the effective density of states in the valence band, V_T is the thermal voltage kT/q, and $\phi_p(0)$ is the hole imref-potential at the grain boundary (relative to E_V).

$$\phi_p(0) = \phi_p(\text{bulk}) + V_d \tag{2.9}$$

The minority-carrier concentration at the grain boundary n(0) is

$$n(0) = N_C \exp\left(\frac{-\phi_n(0)}{V_T}\right) \tag{2.10}$$

The electron imref-potential $\phi_n(0)$ can be written as

$$\phi_n(0) = \frac{E_g}{q} - \phi_p(0) - \frac{\Delta E}{q} \tag{2.11}$$

Figure 2.5 shows the energy-band structure near a uniformally-illuminated grain boundary in p-type silicon. Under illumination the energy states above the electron imref remain empty while those below the hole imref remain occupied. The occupation probability of states situated between the imrefs will depend on the relative electron and hole concentrations and their capture cross-sections at the grain boundary.

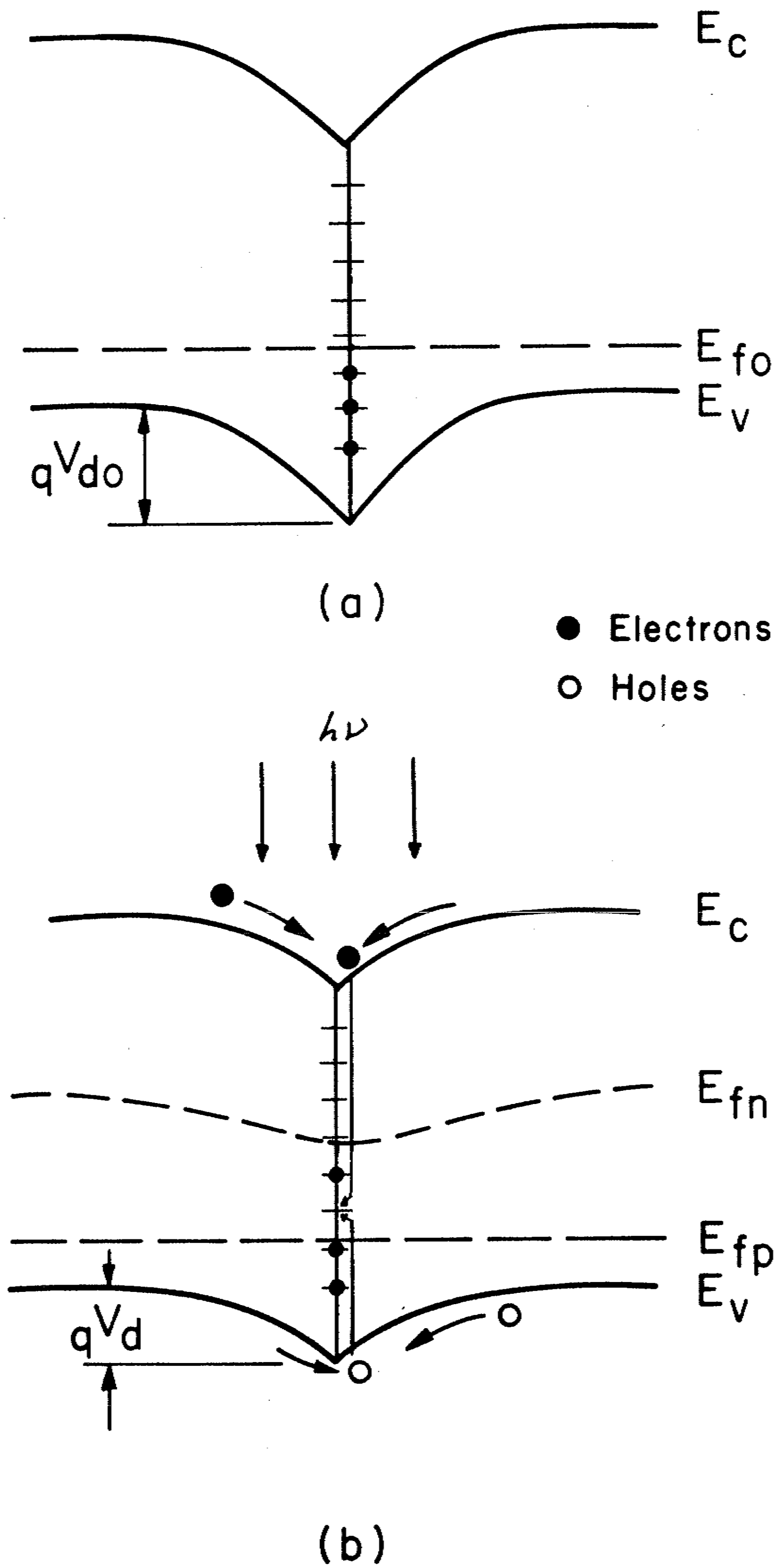


Figure 2.5: Energy-band diagram of an illuminated grain boundary. The states between the imrefs are partially filled.

The total charge at the grain boundary in the most general case is equated to the charge in the depletion regions as [16]

$$(8q\, N_A V_d)^{1/2} = -q\left[\int_{E_{fn}(0)}^{E_C(0)} N_{is}^{D}(E)\, dE + \int_{E_{fp}(0)}^{E_{fn}(0)} \left\{(1-f_D)\, N_{is}^{D}(E) - f_A\, N_{is}^{A}(E)\right\} dE - \int_{E_V(0)}^{E_{fp}(0)} N_{is}^{A}(E)\, dE\right] \tag{2.12}$$

where N_A is the doping concentration in the bulk, $N_{is}^{D}(E)$ and $N_{is}^{A}(E)$ are the donor- and acceptor-like interface-state densities respectively, $E_{fn} = q\phi_n$, $E_{fp} = q\phi_p$, and f_A and f_D are the trap occupation functions equal to [16]

$$f_A = \frac{\sigma_N\, n(0)}{\sigma_N\, n(0) + \sigma_C\, p(0)}$$

$$f_D = \frac{\sigma_C\, n(0)}{\sigma_C\, n(0) + \sigma_N\, p(0)} \tag{2.13}$$

where we have assumed the neutral (σ_N) and coulombic (σ_C) capture cross-sections to have the same value for the donor- and acceptor-like states.

Assuming N_{is}^D and N_{is}^A are independent of energy (2.12) can be rearranged to give the barrier height V_d

$$V_d = \frac{q}{8\varepsilon_s N_A}\left[-\phi_n N_{is}^D + \phi_p N_{is}^A - \Delta E\left\{(1-f_D)N_{is}^D - f_A N_{is}^A\right\}\right]^2 \tag{2.14}$$

For computation we rewrite (2.7) as

$$\Delta E = \frac{2G_{ph}L_n}{\sigma_N v_{th} N_{is} n(0)} \tag{2.15}$$

where

$$N_{is} = N_{is}^A + N_{is}^D \tag{2.16}$$

A computer program that solves this set of equations for V_d as a function of the photogeneration rate G_{ph} has been written and is included here as Appendix A. Sample results are shown graphically in Fig. 2.6 . We observe that the potential barrier V_d decreases with increasing separation of the imrefs ΔE . This implies that V_d decreases in proportion to the logarithm of the photogeneration rate of excess carriers. This effect may have serious implications for small-grain polycrystalline solar cell applications where the recombination lifetime of the excess carriers is important [17].

We note that optical illumination, by reducing the barrier height, effectively increases the conductivity of the barrier. This may eventually lead to the construction of polycrystalline optical sensors.

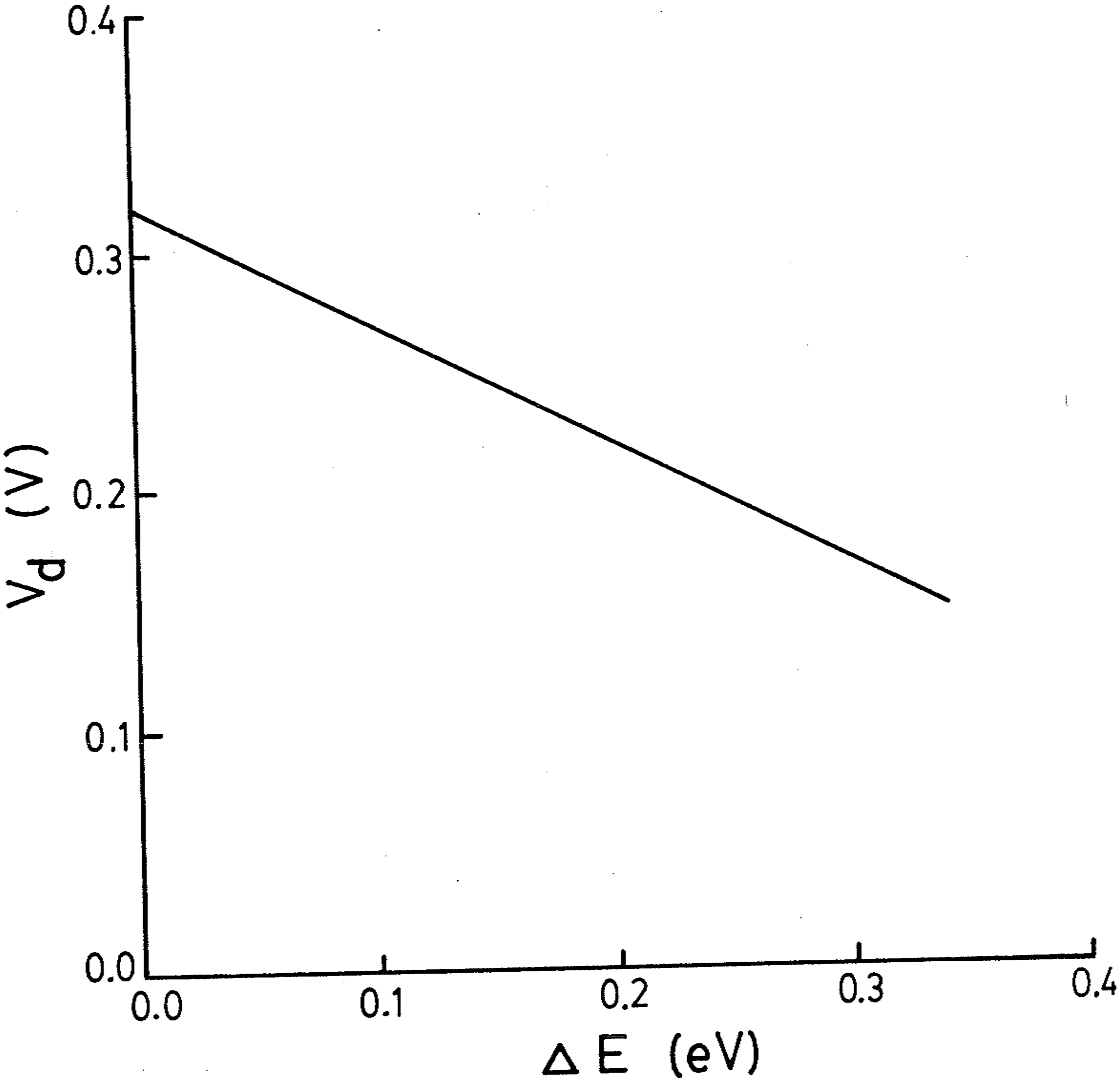


Figure 2.6: Reduction of the potential barrier due to optical illumination. In this example $L = 30\ \mu m$, $N_A = 10^{15}\ cm^{-3}$, $N_{is}^{A} = N_{is}^{D} = 10^{14}\ eV^{-1}\ cm^{-2}$, $\sigma_N = 10^{-16}\ cm^2$, $\sigma_C = 10^{-14}\ cm^2$, and $T = 300\ K$.

Chapter III

MEASUREMENTS ON ISOLATED SILICON GRAIN BOUNDARIES

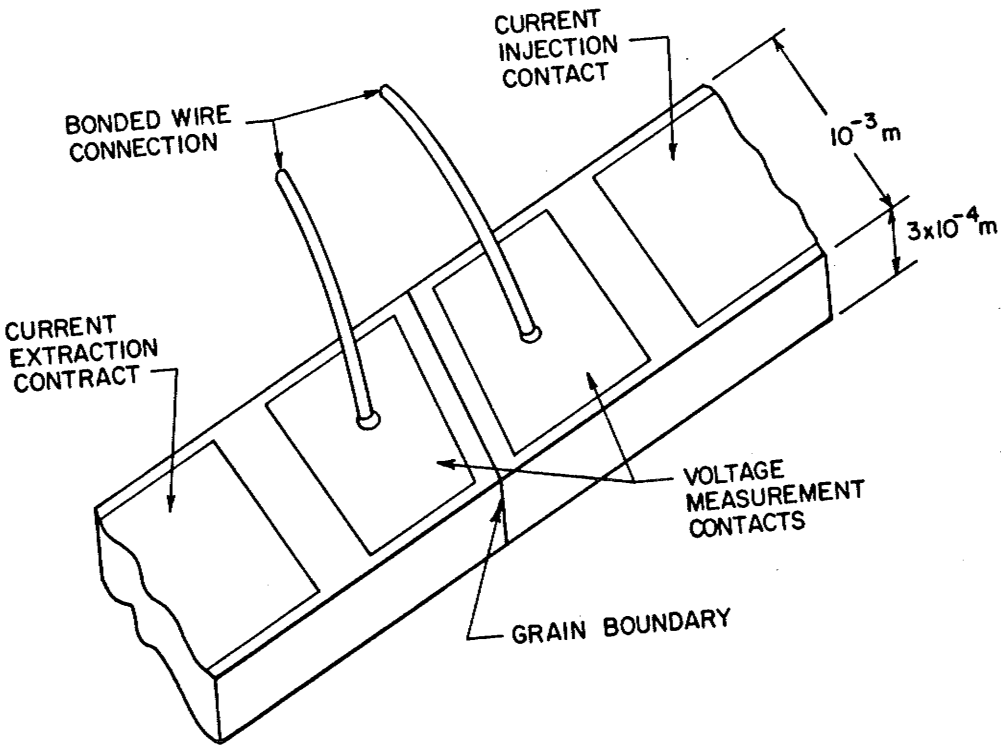
3.1 SAMPLE PREPARATION

An experimental investigation of the electrical properties of grain boundaries in polycrystalline silicon under d.c. conditions was performed. Samples for this purpose were cut from 'Silso' solar-grade polycrystalline silicon obtained from Wacker Chemitronic Ltd. This material had a typical grain size of 1 mm. The p-type (boron) doping concentration N_A was approximately $3x10^{15}$ cm^{-3}.

A diamond-edged wafering saw was used to cut the 100x100x0.4 mm wafers into strips approximately 20x1 mm. These strips were then etched in 3:1:1 HNO_3 (79%): HF (49%): glacial acetic acid for approximately three minutes under gentle agitation to ensure uniformity. This etching served to eliminate the saw damage [18] as well as to highlight the individual grain boundaries [19]. After etching the cross-sectional area was slightly reduced to $3x10^{-3}$ cm^2.

An optical microscope revealed which grain boundaries would be suitable for study. We selected boundaries that were planar throughout the cross-sectional extent of the sample strips and were somewhat remote from adjacent boundaries, i.e. those most similar to our theoretical model. One suitable grain boundary from each of the strips was masked by a thin wire prior to the vacuum deposition of aluminum contacts. A four-probe configuration was created by mechanically removing thin strips of the aluminum film from either side of the grain boundary. The sample geometry is shown in Fig. 3.1 . Ohmic behavior of the contacts was ensured by sintering at 600°C for 20 minutes in flowing nitrogen gas. The sample strips were mounted on test jigs made of printed-circuit-board by silver paste applied to the outer contacts. Aluminum wires were ultrasonically bonded to the inner contacts and brought out to contact pads on the jigs.

The four-probe configuration allows current to be injected and removed through the outer contacts while the voltage drop across the grain-boundary region can be accurately measured between the inner contacts with a high-impedance voltmeter.



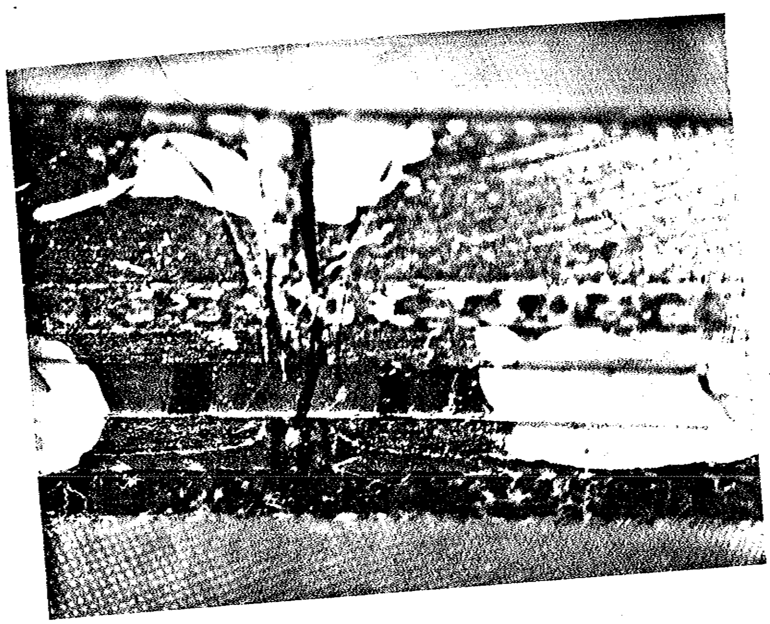


Figure 3.1: Sample geometry. Cross-sectional area of all samples=$3x10^{-3}$ cm^2.

3.2 EXPERIMENTAL RESULTS

A Keithly 610C electrometer and Tektronix DM 502 multimeter (for current and voltage functions respectively) were used to manually measure the current-voltage characteristics of the three sample grain boundaries shown in Figs. 3.2 - 3.4 . The differing current levels from sample to sample are attributed primarily to different magnitudes of the potential barrier V_{do} in the three samples.

The grain boundary samples were subjected to optical illumination from a Sylvania ELH projection lamp. An infrared filter eliminated all radiation with a wavelength of less than 1μm. This ensured a relatively uniform illumination throughout the sample. Figures 3.5 and 3.6 show the effects of the illumination on the current-voltage characteristics of a low and high barrier grain boundary respectively.

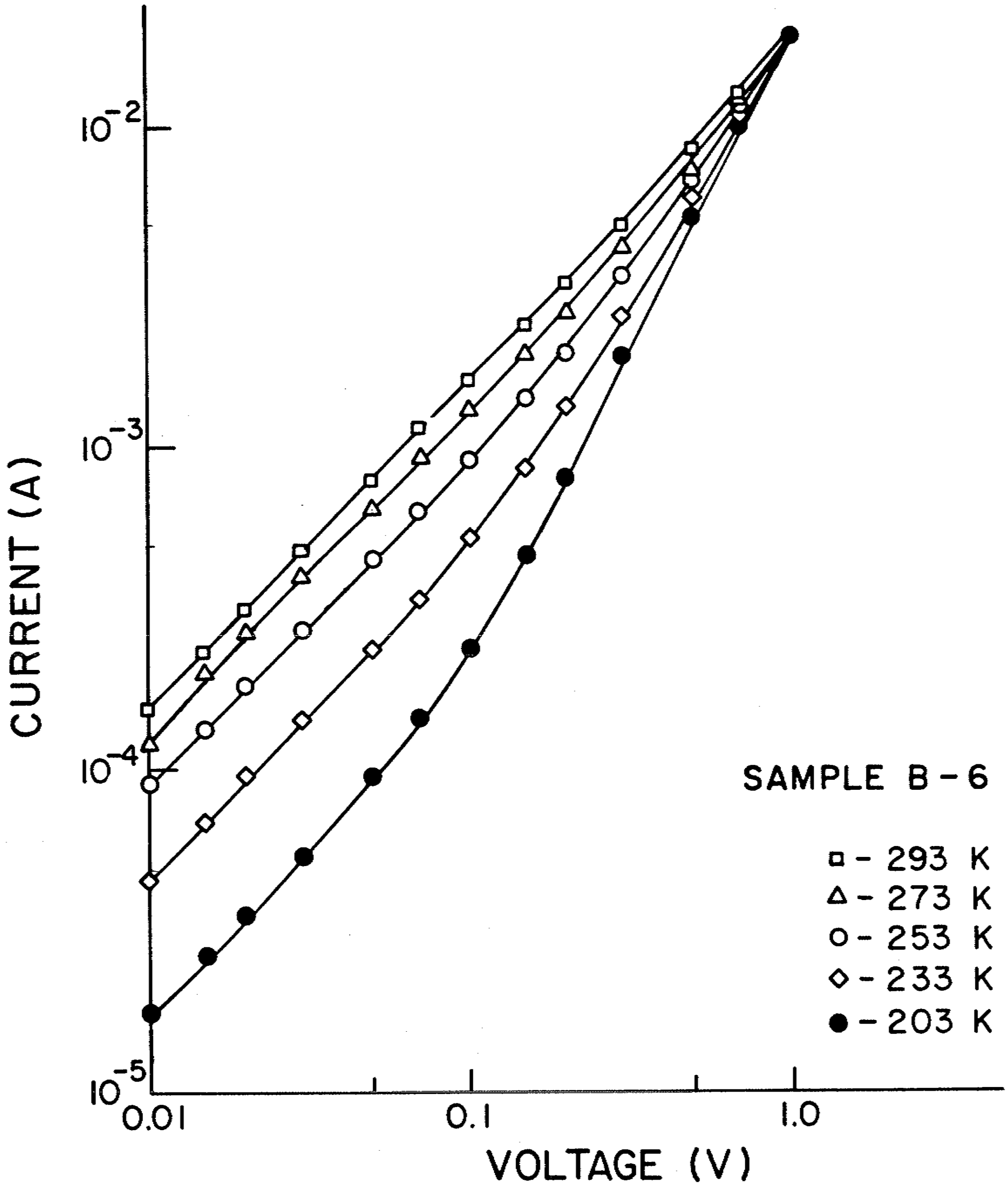


Figure 3.2: Dark characteristics of grain boundary sample B-6 at several measurement temperatures.

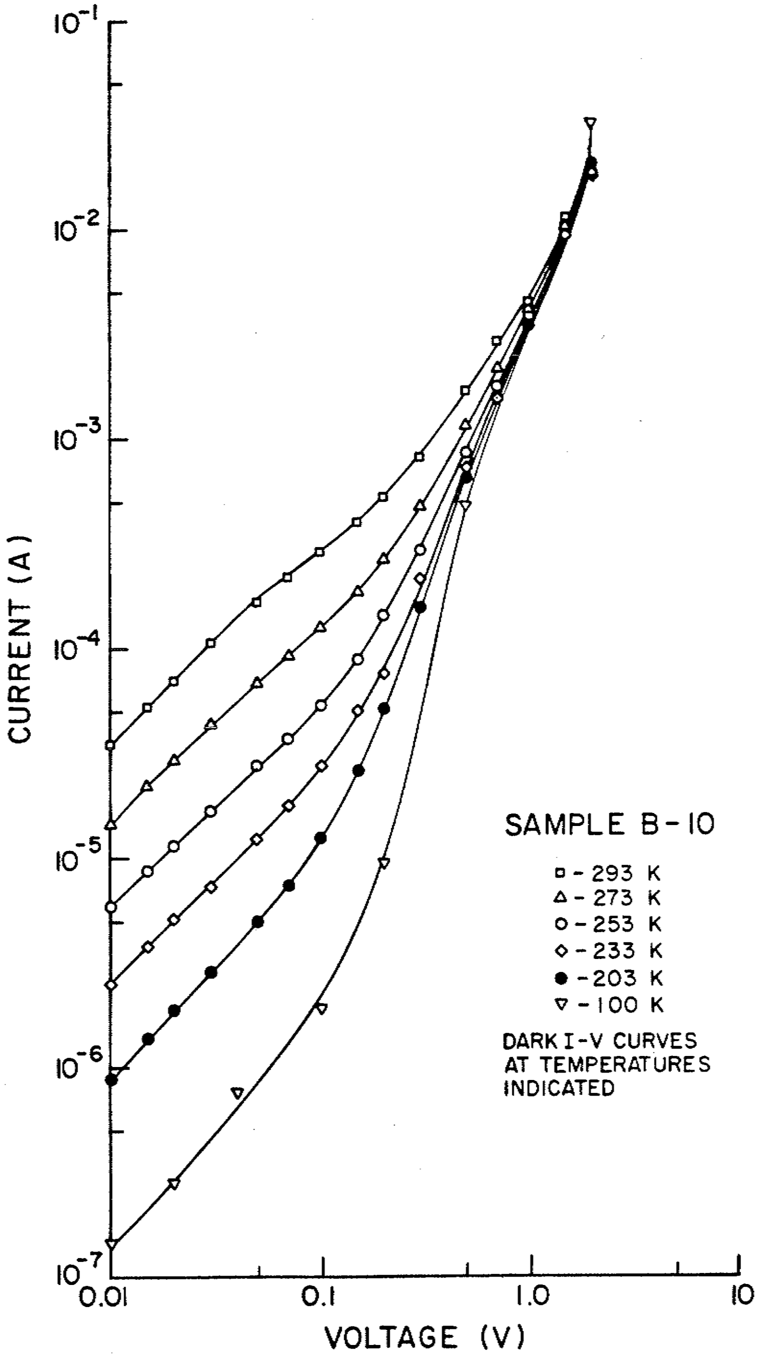


Figure 3.3: Dark characteristics of grain boundary sample B-10 at several measurement temperatures.

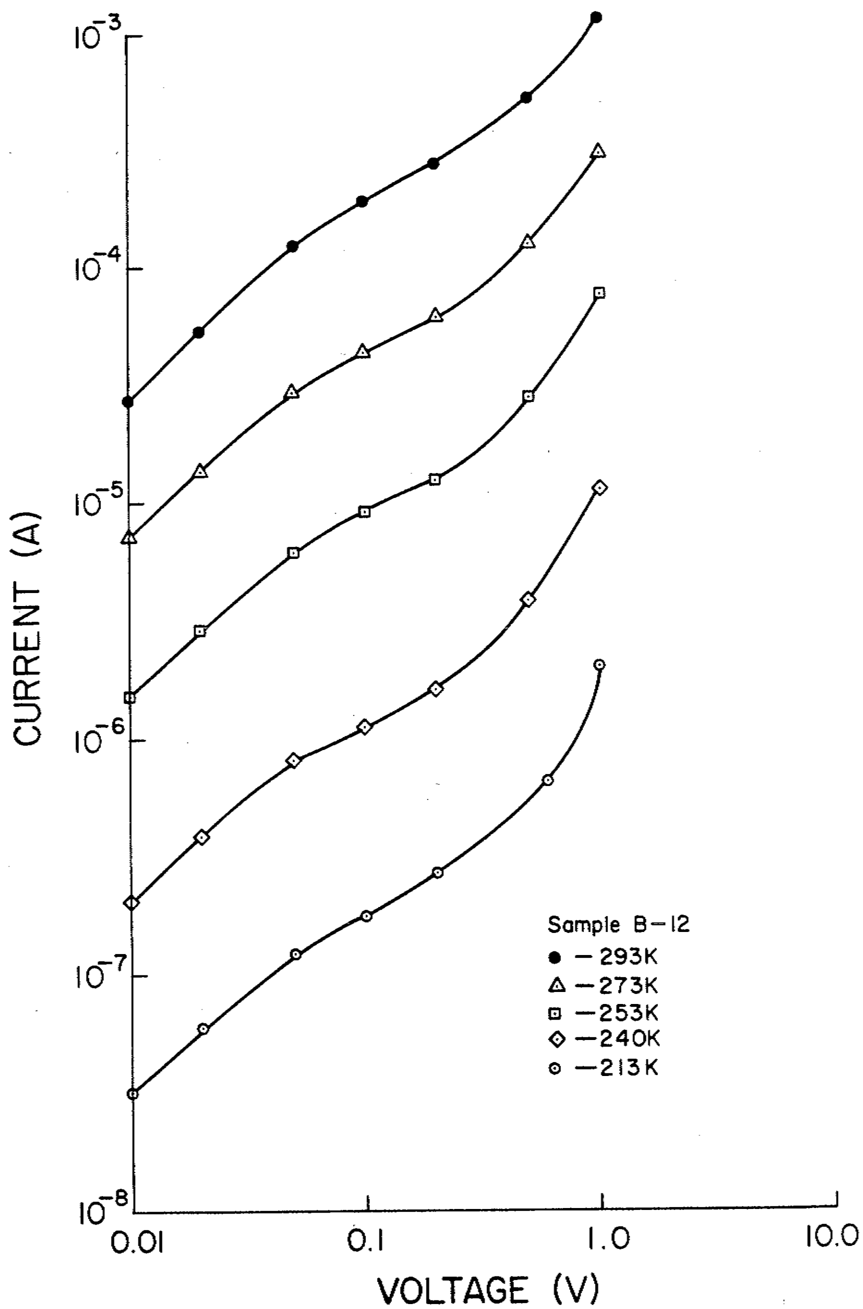


Figure 3.4: Dark characteristics of grain boundary sample B-12 at several measurement temperatures.

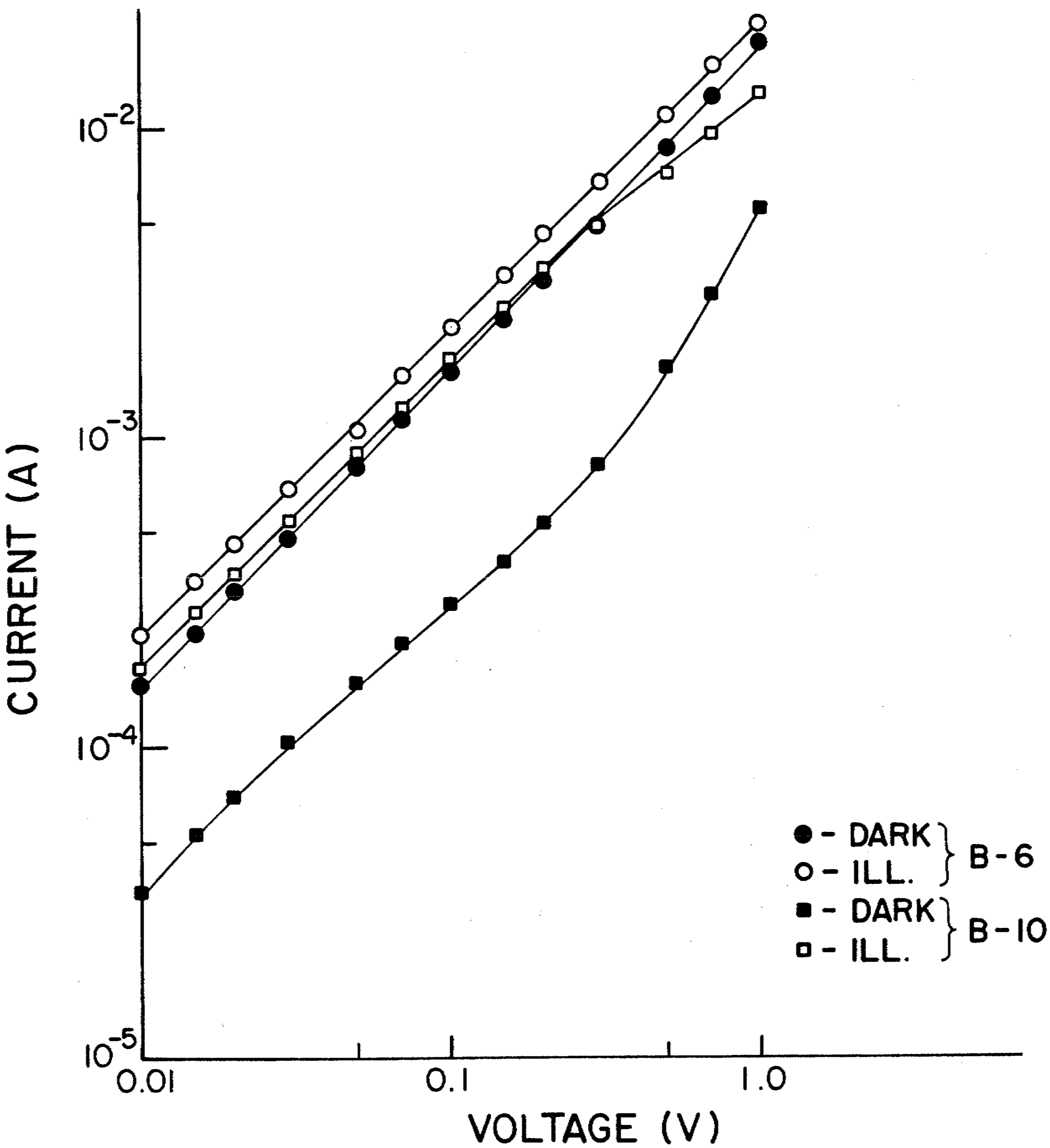


Figure 3.5: Characteristics of B-6 and B-10 under 5 mW cm^{-2} optical ilumination.

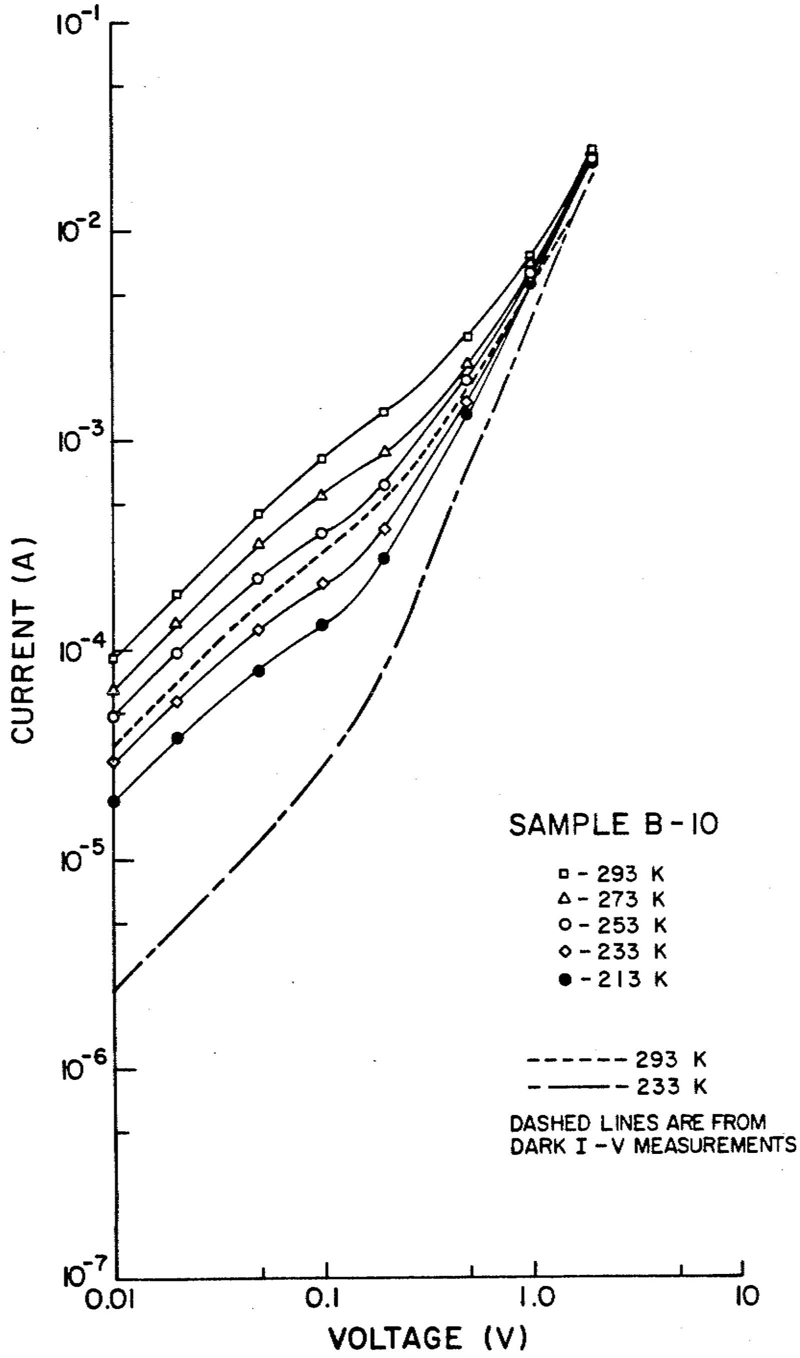


Figure 3.6: Characteristic of B-10 under 20 μW cm^{-2} optical illumination.

3.3 CURRENT OSCILLATIONS

Most grain boundary samples, when biased at a relatively high voltage, exhibited current oscillations such as those shown in Fig. 3.7 . The magnitude of these oscillations was typically 1% of the d.c. current. Voltages in excess of 10V, resulting in current densities in the range 40-100 Acm^{-2} were required to induce these oscillations.

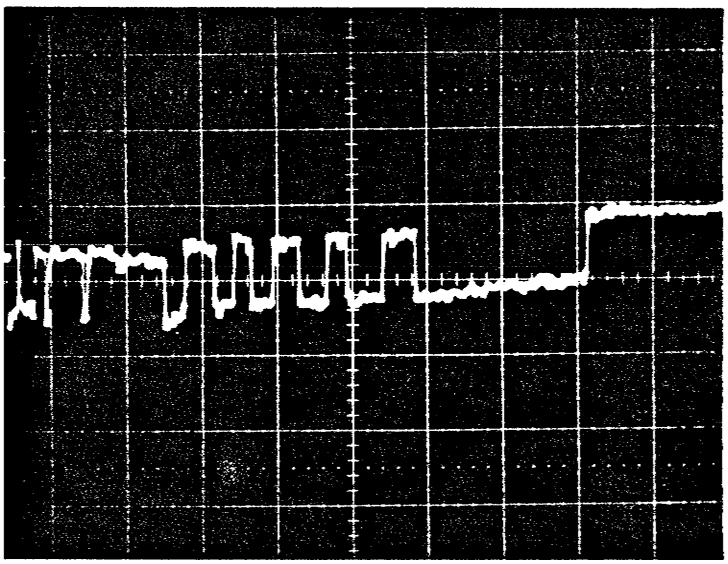


Figure 3.7: Current oscillations observed (as a voltage developed across a 10Ω resistor) at high d.c. current densities. Grain boundary sample B-11. V=15V, I=150mA. Vert. scale: 500 μA/div. Horz. scale: 1 ms/div.

Oscillations were not observed below this threshold level. Above the threshold, generally more and larger pulses were observed at higher current densities.

This experiment was originally intended to provide evidence of carrier multiplication processes at grain boundaries. Indeed, the observed oscillations are strikingly similar to those unambiguously arising from avalanche multiplication in p-n junctions [20]. It was however observed that the grain-boundary current oscillations could be thermally stimulated. That is, a sample, biased at a current below the threshold for oscillation, could be induced into oscillation by heating. The normal multiplication process is expected to be hindered by an increase in lattice temperature [8].

We now believe these oscillations to be due to electric-field-enhanced emission of carriers [21] from the localized interface states at the grain boundary. By this interpretation the large electric field on the reverse-biased side of the grain boundary lowers the coulombic potential well associated with the charged interface states, thus freeing a portion of the charge trapped at the grain boundary. The reduced grain boundary charge will result in a lower electric field and potential barrier. Field emission will cease temporarily until the charge is replenished, increasing the electric field. The current, being controlled by the potential barrier, exhibits rapid fluctuations in accordance with the modulation of the charge.

3.4 AUTOMATED MEASUREMENT OF GRAIN BOUNDARY SAMPLES

The direct current measurements of grain boundaries have been automated using an Automated Data Acquisition and Processing (ADAP) system. ADAP is an "in-house" data acquisition system specifically designed and constructed (as a part of this thesis project) for semiconductor material and device measurements. The system consists of an Apple II Plus microcomputer coupled to an Experiment Control Interface (ECI). ECI provides an 8-input channel, 12-bit analog-to-digital converter (A/D) and four 10-bit digital-to-analog (D/A) voltage outputs. Also incorporated is an ammeter in the form of a current-to-voltage converter with computer-controlled sensitivity. A detailed description of the ADAP and ECI systems can be found in the "ADAP User's Guide" included in this thesis as Appendix D.

The experimental setup is shown in Fig. 3.8 . The four-probe measurement configuration used previously is replaced here by the two-probe method which is better suited for automated measurements. The Z output provides a bias voltage of up to one volt to the sample. The resulting current is passed through the current-to-voltage converter whose output is connected to input channel 6 for "reading" by the A/D.

An FTS Multi-Cool refrigeration unit provided temperature control from +20°C to -60°C. The sample temperature was monitored with a Fluke 2100A thermometer. The analog output voltage of the thermometer was amplified by a Keithley 610C (gain set to 100) and connected to the ECI input channel 4.

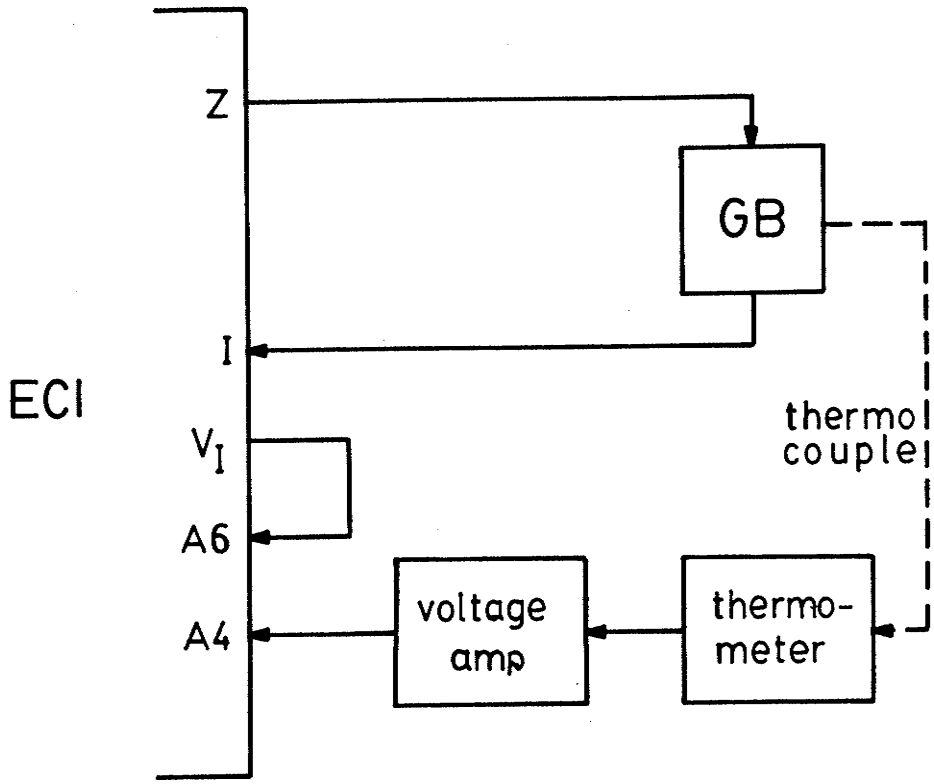


Figure 3.8: Connections between ECI and the grain boundary samples (GB) for automated measurements.

The current-voltage curves of a grain boundary were measured as a function of temperature by taping its jig to a thermal mass and placing it in the refrigeration unit at room temperature (+20°C). Cooling was then initiated and control of the experiment was handed to the ADAP system upon execution of the program "GB". This control program continuously monitored the sample temperature. At ten degree intervals (from +20° to -60°C) the bias voltage was swept through a series of levels from 0.020 to 0.500 volts and the respective currents recorded. These current values were entered into a matrix (i.e. with temperature and voltage as coefficients) and stored on a floppy-disk for reduction by other programs. Appendix B is a listing of "GB".

Figure 3.9 shows a set of current-voltage characteristics that were collected and plotted on a log-log scale using "GB".

The computer facilitates these measurements in several ways. Most importantly, the experiment can run unattended for its two to three hour duration. Also, measurements at each temperature are completed in less than five seconds, thus ensuring isothermality. Clerical errors are eliminated by storing the data directly on floppy disk in a form suitable for subsequent processing. Graphical output in the convenient form of a log-current vs. log-voltage plot is immediately available.

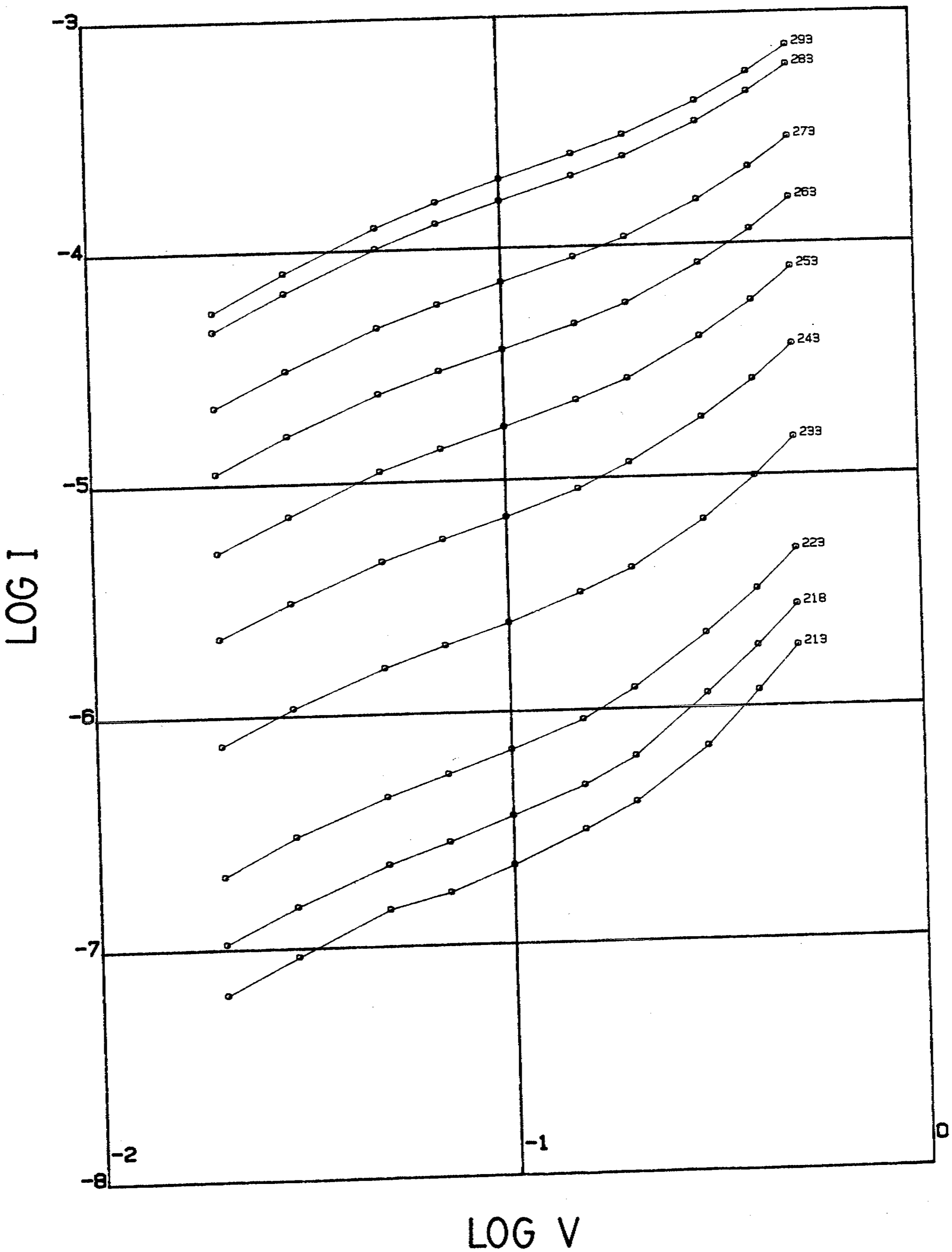


Figure 3.9: Log-current vs. log-voltage plot of a grain boundary characteristic as measured and plotted using the "GB" control program. Sample B-99.

3.5 DETERMINATION OF THE INTERFACE STATE DENSITY BY DECONVOLUTION

In sec. 2.2 we discussed the important role that the interface states play in determining the electrical conduction at grain boundaries. It is thus desireable to experimentally characterize the interface states of the samples previously described. A complete characterization would include the densities and capture cross-sections as a function of energy for both the acceptor- and donor-like states. The capture cross-sections are more relevant to transient and photo-response measurements and will not be dealt with in our present steady-state analysis. An iterative technique has been developed to provide an indication of the interface-state densities at grain boundaries from their current-voltage characteristics. This type of calculation has been dubbed "deconvolution" of the current-voltage characteristics [18].

We begin our formulation by assuming that the diffusion and recombination velocities at the grain boundary are approximately equal for the small applied voltages (V_a<1V) we are considering. This assumption was justified by the results of the computer simulation (sec. 2.3) which revealed that $1/2 < v_d/(v_d+v_r) < 1/3$. Equation (2.2) is then

$$J \simeq \frac{qN_A \mu E_{mf}}{2} \exp\left(\frac{-V_{df}}{V_T}\right)\left[1-\exp\left(\frac{-V_a}{V_T}\right)\right] \tag{3.1}$$

We have used the electric field on the forward-biased side of the grain boundary E_{mf} because it has a slightly greater influence over the current than E_{mr}.

The difference between the diffusion potentials on the reverse- and forward-biased sides of the grain boundary V_{dr} and V_{df} respectively is equal to the applied voltage V_a and can be related to the electric field as

$$V_a = V_{dr} - V_{df} = \frac{\varepsilon_s}{2qN_A}\left[E_{mr}^2 - E_{mf}^2\right] \tag{3.2}$$

The total charge trapped at the grain boundary Q_{GB} is

$$Q_{GB} = \varepsilon_s (E_{mf} - E_{mr}) \tag{3.3}$$

From (3.2) and (3.3) we solve for the electric fields

$$E_{mf} = \frac{-V_a qN_A}{Q_{GB}} + \frac{Q_{GB}}{2\varepsilon_s} = \left(\frac{2qN_A}{\varepsilon_s} V_{df}\right)^{1/2}$$

$$E_{mr} = \frac{-V_a qN_A}{Q_{GB}} - \frac{Q_{GB}}{2\varepsilon_s} = \left(\frac{2qN_A}{\varepsilon_s} V_{dr}\right)^{1/2} \tag{3.4}$$

Solving for the diffusion potentials

$$V_{df} = \frac{\varepsilon_s}{2qN_A}\left(\frac{-V_a qN_A}{Q_{GB}} + \frac{Q_{GB}}{2\varepsilon_s}\right)^2$$

$$V_{dr} = \frac{\varepsilon_s}{2qN_A}\left(\frac{-V_a qN_A}{Q_{GB}} - \frac{Q_{GB}}{2\varepsilon_s}\right)^2 \tag{3.5}$$

We begin the iterative solution for the charge at the grain boundary (for a particular current-voltage pair) by chosing an initial value for the charge Q_{GB}. This value is used in (3.5) to establish a first estimate of the diffusion potentials at the grain boundary for the given applied voltage V_a. The resulting forward-biased diffusion potential V_{df} is used in (3.4) to calculate the corresponding electric field E_{mf}. The current equation (3.1) can now be solved for another value of V_{df}. An improved calculation of Q_{GB} follows from (3.4) and (3.3). The next iteration uses this Q_{GB} to calculate new diffusion potentials and so on. The iterations continue until Q_{GB} converges to a constant value.

When the applied voltage is increased, the Fermi-level at the grain boundary will shift closer to the valence-band edge in p-type material due to the reduction of the forward-biased diffusion potential. The concentration of holes (majority carriers) at the grain boundary p(o) is expressed as [15]

$$p(0) = \frac{N_A}{E_{mr} - E_{mf}} \left\{ E_{mr} \exp\left(\frac{-V_{dr}}{V_T}\right) - E_{mf} \exp\left(\frac{-V_{df}}{V_T}\right) \right\} \qquad (3.6)$$

which has a corresponding Fermi-level

$$E_f(0) = -qV_T \, \ell n\left[\frac{p(0)}{N_V}\right] \qquad (3.7)$$

where N_V is the effective density of states in the valence band. The values of E_{mf}, E_{mr}, V_{df}, and V_{dr} from the iterative solution are used to calculate $E_f(0)$ at each applied voltage.

Successive steps in applied voltage produce an incremental shift of the Fermi-level ΔE_f (as noted in sec. 2.3) and a corresponding alteration of the grain boundary charge ΔQ_{GB}. Provided that the interface states are in equilibrium with the majority-carrier Fermi-level we may write the interface-state density $N_{is}(E)$ as

$$N_{is}(E_f) = \frac{1}{q}\frac{\Delta Q_{GB}}{\Delta E_f} \tag{3.8}$$

The deconvolution of current-voltage characteristics measured at different temperatures are combined in Fig. 3.10 to show the experimentally determined relationship between the interface-state density and the electronic energy of these states. The upper energy limit that may be investigated is the sum of the bulk and zero-bias diffusion potentials. The lowest measureable energy is determined by several factors including pinning of the Fermi-level and the inevitable breakdown of our approximations under the large deviations from equilibrium required for probing close to the valence band.

It is interesting to note that the numerical simulation of sec. 2.3 has shown that at high voltages the emission approximation made by Seager [18] can lead to an underestimation of the interface-state density by a factor of two or three as compared to the emission/diffusion model employed here.

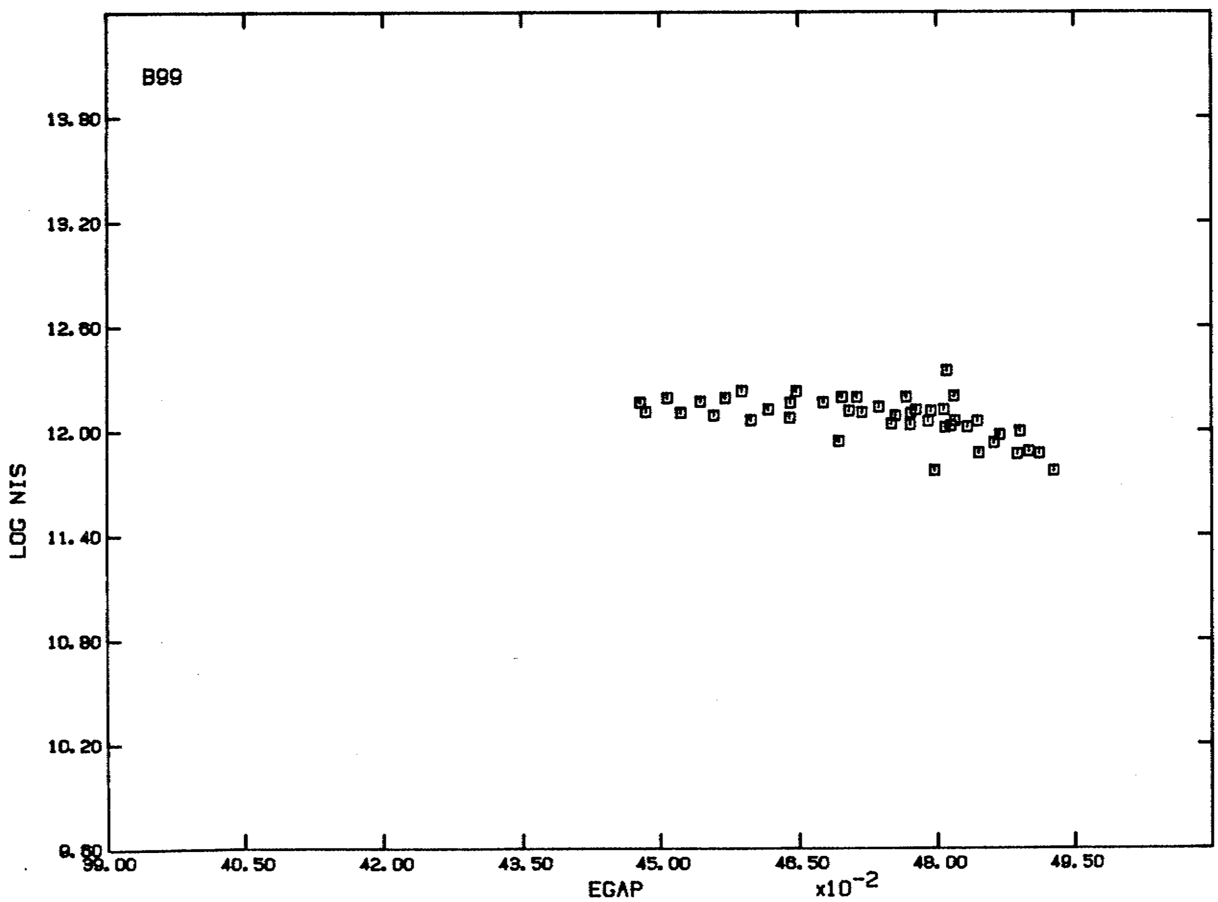


Figure 3.10: Interface-state densities within the forbidden gap of silicon at a grain boundary obtained from the deconvolution of current-voltage characteristics. The plot was made by the program AUTODECON from data collected by GB. Sample B-99.

3.6 GRAIN BOUNDARY ACTIVATION ENERGY

For $V_a >> V_T$ the current equation (2.2) simplifies to

$$J \simeq q \frac{v_d}{2} N_A \exp\left[\frac{-qV_{df}}{kT}\right] \tag{3.9}$$

and it follows that

$$\ell n \left[\frac{J}{v_d}\right] = \ell n\left[\frac{qN_A}{2}\right] - \frac{q}{k} V_{df} \left(\frac{1}{T}\right) \tag{3.10}$$

Taking the derivative of both sides of (3.10) with respect to 1/T we define an activation energy E_a as

$$E_a = - \frac{d\ \ell n(J/v_d)}{d(1/T)} = \frac{d[\frac{q}{k} V_{df}(1/T)]}{d(1/T)}$$

$$= \frac{q}{k}\left(V_{df}(T) + \frac{1}{T}\frac{dV_{df}(T)}{d(1/T)}\right) = \frac{q}{k}\left(V_{df}(T) - T\frac{dV_{df}(T)}{dT}\right) \tag{3.11}$$

Similarly for $V_a << V_T$

$$J \simeq q^2 \frac{v_d N_A}{2} \frac{V_a}{kT} \exp\left(\frac{-qV_{do}}{kT}\right) \tag{3.12}$$

which results in the activation energy

$$E_a = \frac{d\ \ell n(J\cdot T)}{d(1/T)} = \frac{q}{k}\left(V_{do}(T) - T\frac{dV_{do}(T)}{dT}\right) \tag{3.13}$$

The difference of activation energy measured at any two temperatures T_1 and T_2 will be

$$E_a(T_2) - E_a(T_1) = \frac{q}{k}\left(V_{df}(T_2) - T_2\frac{dV_{df}(T_2)}{dT} - V_{df}(T_1) + T_1\frac{dV_{df}(T_1)}{dT}\right)$$
$$= \frac{q}{k}\left(V_{df}(T_2) - V_{df}(T_1) - \frac{dV_{df}}{dT}(T_2 - T_1)\right) = 0 \qquad (3.14)$$

so that E_a is independent of temperature for grain boundaries that conform to our one-dimensional model, provided that $dV_{df}(T_1)/dT=dV_{df}(T_2)/dT$.

The diffusion potential is temperature dependent due to the temperature dependence of the bulk Fermi-potential given as

$$\phi_p = \frac{kT}{q}\,\ell n(N_V/N_A) \qquad (3.15)$$

As the temperature is decreased the Fermi-potential shifts closer to the valence band edge (p-type), increasing the charge trapped in the interface states. Thus $dV_{do}/dT<0$. Quantitatively,

$$\frac{dV_{do}}{dT} = -\gamma\frac{d\phi_p}{dT} = -\gamma\frac{k}{q}\,\ell n(N_V/N_A) \qquad (3.16)$$

where we have neglected the weak temperature dependence of $\ln N_V$.

The parameter γ accounts for the increased charge at the grain boundary. If no interface states existed in the energy range in question ($N_{is}=0$) then $dV_{do}/dT=0$ which leads to $\gamma=0$. Conversely if N_{is} is relatively large $dV_{do}/dT=-d\phi_p/dT$, hence $\gamma=1$.

To derive γ we recognize that at equilibrium

$$qN_{is}\,\frac{d(\phi_p + V_{do})}{dT} = \frac{dQ_{GB}}{dV_{do}} \cdot \frac{dV_{do}}{dT} \tag{3.17}$$

where $\phi_p + V_{do}$ is the Fermi-potential at the grain boundary. We then have

$$\frac{dV_{do}}{dT} = \frac{1}{\dfrac{1}{qN_{is}} \cdot \dfrac{dQ_{GB}}{dV_{do}} - 1} \cdot \frac{d\phi_p}{dT} \tag{3.18}$$

Comparing (3.18) and (3.16) we find γ to be

$$\gamma = \frac{1}{1 - \dfrac{1}{qN_{is}}\dfrac{dQ_{GB}}{dV_{do}}} \tag{3.19}$$

where

$$\frac{dQ_{GB}}{dV_{do}} = \left(\frac{2q\varepsilon_s N_A}{V_{do}}\right)^{1/2} \tag{3.20}$$

provided that $V_{do} >> V_T$. In general γ is a function of temperature; however, it can be considered constant provided N_{is} is relatively independent of energy (for the range of Fermi-level excursions to be considered).

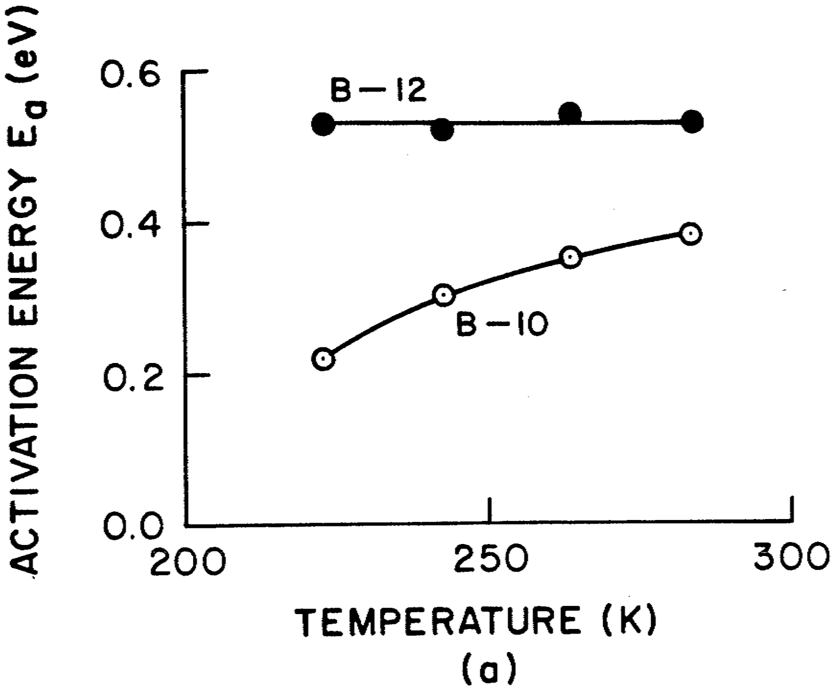
Chapter IV

DISCUSSION

4.1 EXPERIMENTAL DETERMINATION OF THE POTENTIAL BARRIER

The potential barrier at a grain boundary V_{do} has been measured by several techniques. The first uses (3.12) and currents measured at low voltage to obtain V_{do} at near-equilibrium conditions. The second employs (3.13) to obtain V_{do} from the measured activation energies. This procedure requires an independent determination of the interface-state density which is acquired by deconvolution of the current-voltage characteristics (sec. 3.5). The results of these calculations for V_{do} as functions of measurement temperature are shown in Fig. 4.1 for several samples.

From Fig 4.1 it is evident that sample B-12 conforms, at least qualitatively, to the results predicted by (3.14) with $\gamma=1$. The marked decrease of E_a and V_{do} with decreasing temperature for sample B-10 is however incompatible with the present theory. Several possible causes of this effect will be discussed in the following section.



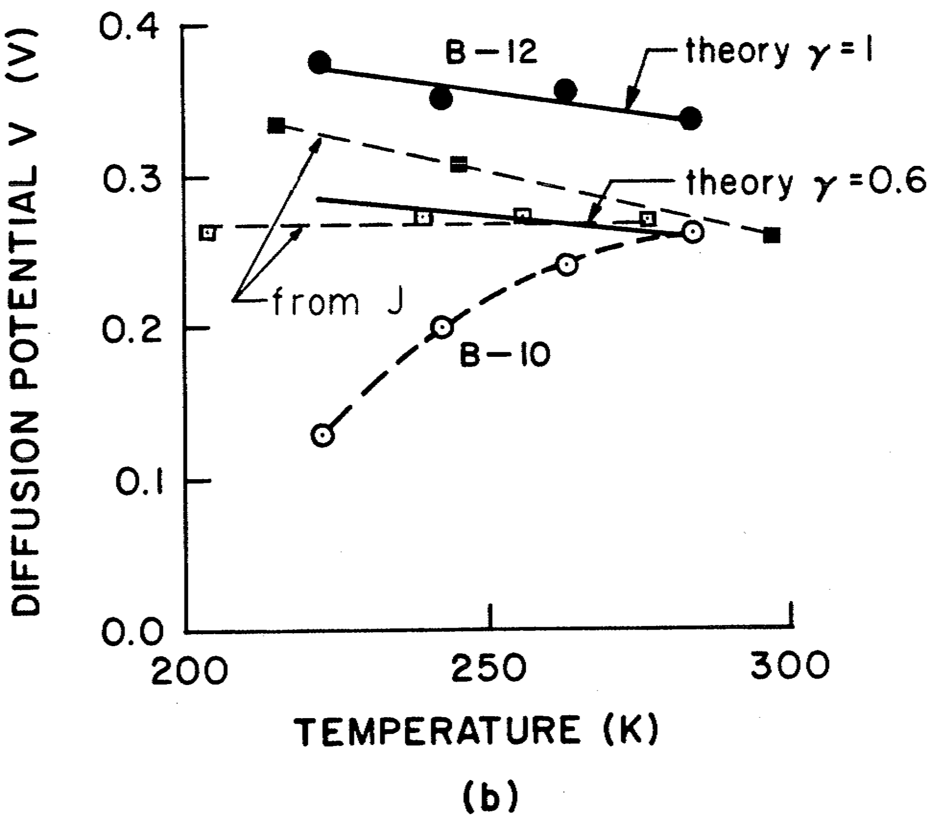


Figure 4.1: The dependence of (a) the activation energy and (b) the diffusion potential on temperature for B-10 and B-12. Dark conditions.

4.2 NONUNIFORMITY OF THE GRAIN BOUNDARY DIFFUSION POTENTIAL

So far we have assumed the grain boundary potential barrier to arise from a macroscopically uniform distribution of charge trapped in the interface states. There are however at least two reasons to suspect that most of our samples will have nonuniform potential barriers.

The material used for our experiments, Wacker "Silso" polycrystalline silicon, is cast in blocks with the boron doping incorporated in the melt. Upon cooling the boron atoms may cluster and be forced to the grain boundaries during recrystallization [22]. The impurity doping concentration could be considerably enhanced in the vicinity of the grain boundary resulting in a local reduction of the potential barrier.

The second possible source of the potential barrier nonuniformity is a consequence of the origin of the interface states. We have assumed that these states arise from the lattice mismatch between adjacent crystals. Figure 4.2 is a Thomson penny model (TPM) of the crystal lattices at the grain boundary. It is apparent that different mismatch angles will result in different bond energies across the boundary. A variation of the mismatch angle could alter the energies and densities of the interface states. This would lead to a localized variation of the grain boundary charge which would produce a nonuniform potential barrier.

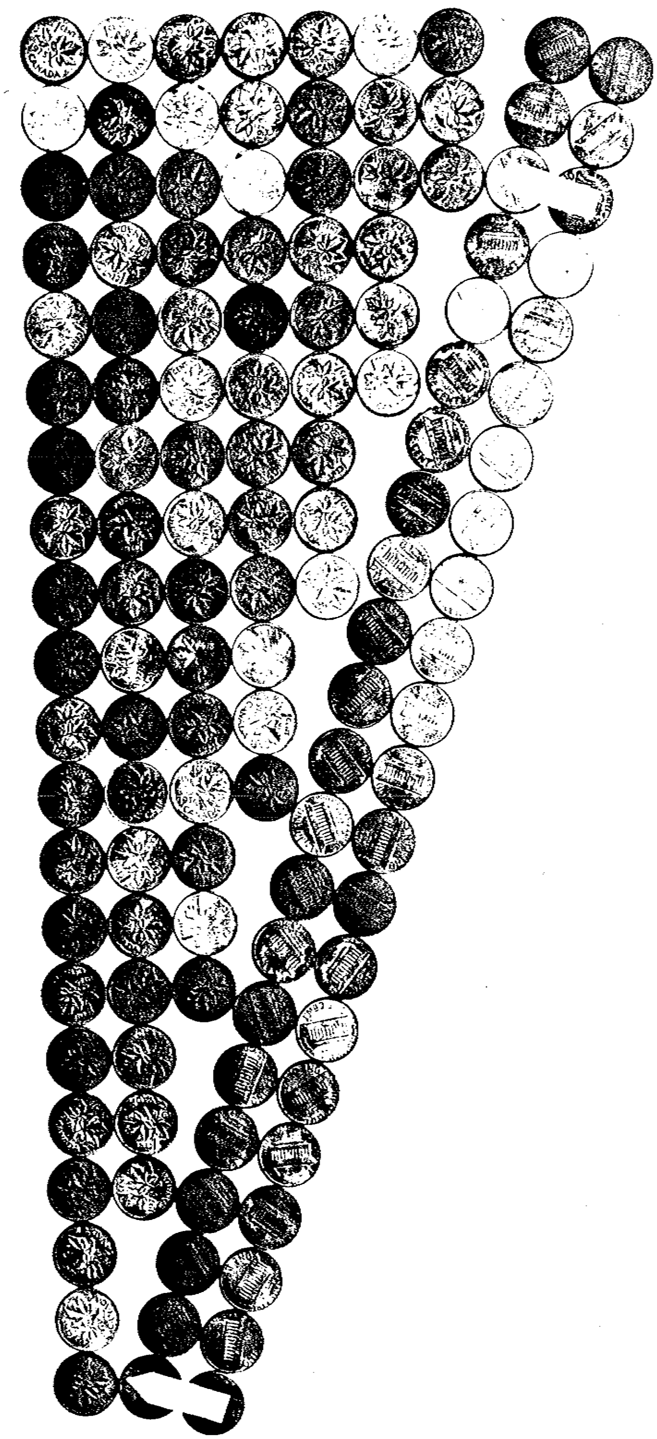


Figure 4.2: Thomson penny model of grain boundary for an orientational mismatch of approx. 20° showing the periodic nature of a particular defect structure. A variety of bonding disorder is suggested over one period.

We further suggest that the bonding defects at the grain boundary could be periodic in nature. Generally, large mismatch angles would produce bonding defects with periods of a few atomic spacings while low mismatch angles will generate longer periods. It is interesting to note that, by virtue of the periodic nature of the defect structure, a two-dimensional band conduction should be possible in the grain boundary plane.

Thomson [23] has hypothesized that the irregularities of the B-10 activation energies at low temperatures are a result of nonuniformities of the diffusion potential over the plane of the grain boundary. After assuming a truncated gaussian distribution of boundary potentials this approach produced qualitative agreement with the observed behavior of B-10. Quantitative agreement was not good, possibly because a gaussian distribution is unphysical for this problem. Also, many simplifications such as the neglect of spreading resistance (which could be significant, especially at low temperatures) were in effect. A Monte Carlo simulation of the grain boundary potentials may lead to a more favorable result. Experimental determination of the grain boundary potential distribution, possibly by scanning light spot techniques [24], would greatly enhance the usefulness of these numerical simulations.

The reduction of the potential barrier under illuminated conditions as described previously in sec. 2.3 has been experimentally verified (by Thomson in [14]). Low voltage activation energy measurements over a range of illumination intensities were used to construct the relation between the potential barrier and the intensity shown in Fig. 4.3 .

We wish to point out that sample B-12, which is thought to have a uniform potential barrier, conforms well to the predicted barrier reduction of 0.07 eV per decade of illumination. On the other hand, B-10 shows a significantly greater reaction to illumination. This again may be attributed to the nonuniform nature of its potential barrier.

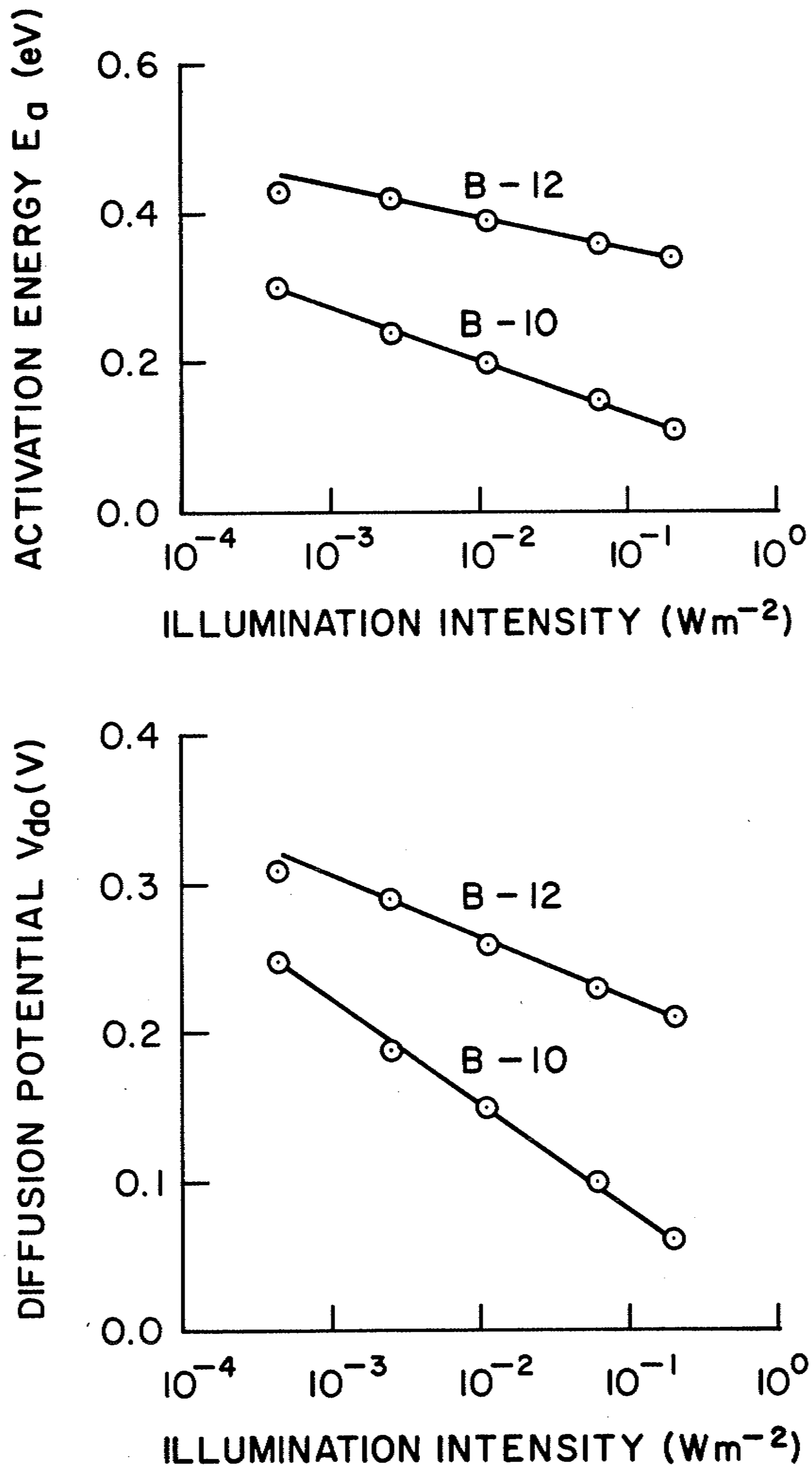


Figure 4.3: The dependence of (a) the activation energy and (b) the diffusion potential on illumination intensity for B-10 and B-12. T=300 K.

Chapter V

CONCLUSIONS

The majority-carrier current across silicon grain boundary potential barriers is controlled by an emission/diffusion transport mechanism. The diffusion velocity of majority carriers to the grain boundary is the current limiting factor.

The interface states at the grain boundary are found to play an important role in determining the electronic properties of polycrystalline silicon. Indeed, it is this fact that allows us to deconvolve the density of these states from current-voltage measurements of isolated grain boundaries. Interface-state densities of approximately 10^{12} $eV^{-1}cm^{-2}$ were measured. Under optical illumination, minority carriers interact with the interface states in such a way as to decrease the potential barrier to majority carriers.

Grain boundaries with a variety of diffusion potentials exist in Wacker 'Silso' polycrystalline silicon. Activation energy measurements have shown irregularities that were attributed to the nonuniformity of the diffusion potential over the plane of the grain boundary. The special case of a spacially uniform boundary has also been observed.

A microcomputer has proven useful for automating the current-voltage measurements and deconvolving the interface-state densities of grain boundary samples.

The many applications of polycrystalline semiconductors necessitates a complete understanding of their electrical and electrooptical properties. The large number of processes involved will make this a challenging research endeavor for years to come.

REFERENCES

1. H.Yamanaka, T. Wada, O. Kudoh, and M. Sakamoto, "Polysilicon interconnection technology for IC devices," J. Electrochem. Soc., 126, pp. 1415-1418, (1979).

2. C.M. Wu, E.S. Yang, W. Hwang, and H.C. Card, "Grain boundary effects on the electrical behavior of Al-poly-Si Schottky barrier solar cells," IEEE Trans. Electron Dev., ED-27, pp.687-692, (1980).

3. M.M. Mandurah, K.C. Saraswat, and T.I. Kamins, "A Model for Conduction in Polycrystalline Silicon - Part I:Theory," IEEE Trans. Electron Dev., ED-28, pp. 1163-1171, (1981).

4. L.L. Kazmerski, Polycrystalline and Amorphous Thin Films and Devices, New York: Academic Press, 1980. chap. 3.

5. O.L. Krivanek, S. Isoda, and K. Kobayashi, "Lattice imaging of a grain boundary in crystalline germanium," Phil. Mag., 36, pp. 931-940, (1977).

6. B. Cunningham and D.G. Ast, "High Resolution Electron Microscopy," in Grain Boundaries in Semiconductors, ed. C. Seager, New York: North-Holland, pp. 21-26, (1982).

7. W. Shockley and W.T. Read, "Statistics of the Recombination of Holes and Electrons," Phys. Rev., 87, pp. 835-842, (1952).

8. S.M. Sze, Physics of Semiconductor Devices, 2nd ed., New York: Wiley-Interscience, 1981.

9. J.Y.W. Seto, "The electrical properties of polycrystalline silicon films", J. Appl. Phys., 46, pp. 5247-5254, (1975).

10. J. Martinez, and J. Piqueras, "On the Mobility of Polycrystalline Semiconductors", Solid State Electron., 23, pp. 297-303, (1980).

11. C.H. Seager and T.G. Castner, "Zero-bias resistance of grain boundaries in neutron-transmutation-doped polycrystalline silicon", J. Appl. Phys., 49, pp. 3879-3889, (1978).

12. C.H. Seager and G.E. Pike, "Grain boundary states and varistor behavior in silicon bicrystals", Appl. Phys. Lett., 35, pp. 709-711, (1979).

13. E.H. Rhoderick, Metal-Semiconductor Contacts, Oxford: Oxford Press, 1978.

14. G.C. McGonigal, D.J. Thomson, J.G. Shaw, and H.C. Card, "Electronic Transport at Grain Boundaries in Silicon," Phys. Rev. B, to be published.

15. W.E. Taylor, N.H. Odell, and H.Y. Fan, "Grain Boundary Barriers in Germanium," Phys. Rev., 88, pp. 867-875, (1952).

16. P. Panayotatos and H.C. Card, "Recombination Velocity at Grain Boundaries in Polycrystalline Si under Optical Illumination," IEEE Electron Dev. Lett., EDL-1, pp. 263-266, (1980).

17. L.L. Kazmerski, "The Effects of Grain Boundary and Interface Recombination on the Performance of Thin-film Solar Cells," Solid St. Electron., 21, pp. 1545-1550, (1978).

18. C.H. Seager, "The Electronic Properties of Semiconductor Grain Boundaries," in ref. 6, pp. 85-98.

19. D.J. Thomson, S.R. Mejia, and H.C. Card, "The Influence of Surface Preparation on Rectification in Aluminum-Polycrystalline Silicon Solar Cells," J. Power Sources, 7, pp. 191-194, (1982).

20. K.G. McKay, "Avalanche Breakdown in Silicon," Phys. Rev., 94, pp. 877-884, (1954).

21. J. Frenkel, "On pre-breakdown phenomena in insulator and electronic semiconductors," Phys. Rev., 54, pp. 647-648, (1938).

22. L.L. Kazmerski, P.J. Ireland, and T.F. Ciszek, "Evidence for the segregation of impurities to grain boundaries in multigrained silicon using Auger electron spectroscopy and secondary ion mass spectroscopy," Appl. Phys. Lett., 36, pp. 323-325, (1980).

23. D.J. Thomson and H.C. Card, "Effects of Interface-Potential Nonuniformities on Carrier Transport across Silicon Grain Boundaries," J. Appl. Phys., in press.

24. J. Martinez, A. Criado, and J. Piqueras, "Grain boundary potential determination in polycrystalline silicon by the scanning light spot technique," J. Appl. Phys., 52, pp. 1301-1305, (1981).

Appendix A

PHOTO GB

```
//PHOTOGB JOB ',,,,,,T=10,L=2,C=0',MCGON
// EXEC PASCCG,PARM.PASC='NOXREF,NOPXREF,NOWARNING'
//SYSIN DD *
PROGRAM PHOTO_GB(INPUT,OUTPUT); (* IBM PASCAL/VS REL. 2 *)
(************************* PHOTO_GB ******************************)
(*                                                               *)
(*                 WRITTEN BY G.C. MCGONIGAL                     *)
(*                                                               *)
(*              MATERIALS & DEVICES RESEARCH LAB                 *)
(*              DEPT.  OF ELECTRICAL ENGINEERING                 *)
(*                   UNIVERSITY OF MANITOBA                      *)
(*                           1981.                               *)
(*                                                               *)
(*****************************************************************)
(*                                                               *)
(*PHOTO_GB CALCULATES THE POTENTIAL BARRIER AT A ONE-DIMENSIONAL*)
(*GRAIN BOUNDARY AS A FUNCTION OF A UNIFORM PHOTOGENERATION RATE*)
(*THE CORRESP. IMREF SEP. AND  CAPACITANCE IS ALSO GIVEN.       *)
(*                                                               *)
(*ALL UNITS ARE CGS                                              *)
(*                                                               *)
(*INPUT PARAMETERS ARE:                                          *)
(*       TEMPERATURE (K)                                         *)
(*       BULK IMPURITY DOPING CONCENTRATION                      *)
(*       MINORITY CARRIER DIFFUSION LENGTH (BULK)                *)
(*       ACCEPTOR INTERFACE ST. DENSITY,  DONOR INT. ST. DENS.   *)
(*       NEUTRAL CAPTURE X-SECTION,  COULOMBIC CAP. X-SECTION    *)
(*                                                               *)
(*****************************************************************)
VAR DELTA_E,DELTA_E_STAR,V_T,V_D,V_D_STAR,V_TH,N_V,N_C,N_D,G_PH,
    PHI_N,PHI_P,PHI_N_GB,PHI_P_GB,F_A,F_D,T,P_O,N_O,N_IS,L_P,Q,M_E,
    CAPACITANCE,
    N_IS_A,N_IS_D,SIG_N,SIG_C,EPSILON_S,P_O_STAR,G_PRINT,INC:REAL;
(*)

(*)
BEGIN
READLN(T);              WRITELN( ' TEMPERATURE = ',T:6:1,' K');
READLN(N_D);            WRITELN( ' ND = ',N_D:10,' PER CM CUBED');
READLN(L_P);            WRITELN( ' MIN. CAR. DIFF. L. = ',L_P:10);
READLN(N_IS_A,N_IS_D);
READLN(SIG_N,SIG_C);
WRITE( ' N_IS_A = ',N_IS_A:10);
WRITELN( '   N_IS_D = ',N_IS_D:10,'  PER CM SQUARED');
```

```
WRITE( ' NEUTRAL CAP. X-SECTION = ',SIG_N:10);
WRITELN( ' COULOMB CAP. X-SECTION = ',SIG_C:10,' CM SQUARED');
WRITELN;WRITELN;
WRITELN( ' G_PH,V_D,DELTA_E,P_O,N_O,F_A,F_D,PHI_P,PHI_N,CAP');
WRITELN;
(*
  .... CALCULATE CONSTANTS ....
*)
Q := 1.6E-19;                                    (* ELECTRONIC CHARGE *)
M_E := 9.1E-31;                                  (* ELECTRONIC  MASS  *)
V_T := 8.67E-5 * T;                              (* K*T/Q *)
V_TH := SQRT(3.0*V_T*Q/M_E)*100.0;               (* THERMAL VELOC. *)
N_V := 2.0E15 * SQRT(T * T * T);                 (* # OF VAL. STS. *)
N_C := 5.389E15 * SQRT( T*T*T );                 (* # OF COND. STS.*)
PHI_N := LN(N_C/N_D) * V_T;                      (* BULK MAJ C.FER LVL *)
N_IS := N_IS_A + N_IS_D;                         (* TOTAL INTERF. STS. *)
EPSILON_S := 11.8*8.854E-14;                     (* DIEL. CONSTANT  *)
(*
  .... DEFINE INITIAL CONDITIONS FOR ITERATIONS ....
*)
DELTA_E := 0.00;                                 (* SEP. OF FERMI LVLS *)
V_D :=   0.30;                                   (* DIFFUSION POTENTIAL*)
V_D_STAR := V_D;
G_PH := 0.0;                                     (* PHOTO-GENERATION *)
(*
  .... REPEAT CALC. FOR VALUES OF G_PH ....
*)
WHILE G_PH < 1.0E20 DO
  BEGIN
  (*
    .... BEGIN MAJOR ITERATION FOR V_D ....
  *)
  REPEAT
    V_D := V_D+(V_D_STAR-V_D)*1.0E-2;
    PHI_N_GB := PHI_N + V_D;                   (* GB MAJ. CAR. FER. LVL *)
    N_O := N_C*EXP(-PHI_N_GB/V_T);             (* GB MAJ. CAR. CONCENTR.*)
    (*
      .... BEGIN MINOR ITERATION FOR P_O & DELTA_E ....
    *)
    REPEAT
      PHI_P_GB := 1.12-PHI_N_GB-DELTA_E;(*GB MIN. CAR. F. LVL.*)
      P_O := N_V*EXP(-PHI_P_GB/V_T); (* GB MIN. CAR. CONCENTR.*)
      DELTA_E_STAR := 2.0*G_PH*L_P/SIG_N/V_TH/N_IS/P_O;
      DELTA_E := DELTA_E-(DELTA_E-DELTA_E_STAR)*0.001;
    UNTIL ABS(DELTA_E-DELTA_E_STAR) <= 0.0001;
    F_A := SIG_N*N_O/(SIG_N*N_O+SIG_C*P_O);(* ACCEPTOR OCCUP.*)
    F_D := SIG_C*N_O/(SIG_C*N_O+SIG_N*P_O);(* DONOR OCCUPATION*)
    V_D_STAR := SQR(Q*(-PHI_N_GB*N_IS_D+PHI_P_GB*N_IS_A
                -DELTA_E*((1.0-F_D)*N_IS_D-F_A
                *N_IS_A)))/(8.0*Q*N_D*EPSILON_S);
  UNTIL ABS(V_D_STAR-V_D) <= 0.0100;
  CAPACITANCE := SQRT(Q*EPSILON_S*N_D/8.0/V_D);
  WRITELN(G_PH:10,V_D:11:4,DELTA_E:11:4,P_O:11,N_O:11,F_A:11:4,
        F_D:11:4,PHI_P_GB:11:4,PHI_N_GB:11:4,CAPACITANCE:11);
```

```
  (* ARBITRAR. INCREMENT G_PH *)
  G_PH := G_PH*26.0/(LN(G_PH+1.0)/2.3+5.0)+1.0E13;
  END;(*WHILE*)
END.
//* SAMPLE INPUT PARMS....
//GO.INPUT DD *
300.0                 TEMP
1.0E15                DOPING
30.0E-4               DIFF LENGTH
7.0E12 7.0E12         N_IS_A N_IS_D
1.0E-16 1.0E-14       SIGMA_N SIGMA_C
/*
```

Appendix B

GB

```
(*$S+*)
PROGRAM GB;

(*
   ANALOG INPUTS:
      0-
      1-
      2-
      3-
      4- TEMP:FROM KEITHLEY(G=100) FROM FLUKE TEMPERATURE METER.
      5-
      6- CURRENT:VOLTAGE FROM I/V
      7-

   ANALOG OUTPUTS:
      X-
      Y-
      W-
      Z- SAMPLE BIAS
*)
USES TRANSCEND,ECIDAS,APPLESTUFF,TURTLEGRAPHICS;

TYPE MATRIX=ARRAY[1..10] OF ARRAY[1..10] OF REAL;

VAR ITV:MATRIX;
    MODECHAR,INCHAR:CHAR;
    TEMPCO,J,I:INTEGER;
    V,VTEMP,TEMP:ARRAY[1..10] OF REAL;
    TOTAL:REAL;
    PLOT:BOOLEAN;
    TO7470,IVDATA,T:TEXT;


PROCEDURE LOGGRID(XDECNUM,XMINMAG,YDECNUM,YMINMAG:INTEGER);
(* GENERATES  A LOG-LOG GRID *)
(* DECNUM=NO. OF DECADES
   MINMAG=SMALLEST DECADE TO BE DISP *)
VAR I,XMAXMAG,YMAXMAG:INTEGER;
    ISTR:STRING[3];
BEGIN
XMAXMAG:=XMINMAG+XDECNUM;
YMAXMAG:=YMINMAG+YDECNUM;
REWRITE(TO7470,'#8:');
WRITE(TO7470,'IN;SC ',XMAXMAG,',',XMINMAG,',',YMINMAG,',',YMAXMAG,';');
WRITE(TO7470,'SP1;'); (* LEFT PEN *)
WRITE(TO7470,'DI0.0,1.0;');
FOR I:=XMINMAG TO XMAXMAG DO
```

```
  BEGIN
  WRITE(TO7470,'PU',I,',',YMAXMAG,';');
  WRITE(TO7470,'PD',I,',',YMINMAG,';');
  STR(I,ISTR);
  WRITE(TO7470,'PU;DT',CHR(3),';LB',CHR(8),CHR(8),CHR(3),'PU;LB',ISTR,CHR(3));
  END;
FOR I:=YMINMAG TO YMAXMAG DO
  BEGIN
  WRITE(TO7470,'PU',XMAXMAG,',',I,';');
  WRITE(TO7470,'PD',XMINMAG,',',I,';');
  STR(I,ISTR);
  WRITE(TO7470,'PU;DI0.2,1.0;LB',CHR(11),CHR(3),'DI0.0,1.0;LB',ISTR,CHR(3));
  END;
WRITE(TO7470,'SP0;');  (* STORE PEN *)
WRITE(TO7470,'DI;');
END;


PROCEDURE LOGPLOT(X,Y:REAL;XDECNUM,XMINMAG,YDECNUM,YMINMAG:INTEGER;
                  CONPTS:BOOLEAN;CHARMARKER:CHAR);
(* PLOTS POINTS ON THE LOG PLOT *)
(* DECNUM,MINMAG SAME AS FOR LOGGRID
   X,Y=CO-ORDS. OF POINT TO BE PLOTTED
   CONPTS=TRUE DRAWS A LINE TO NEXT PT.
   CHARMARKER=CHAR TO BE PRINTED *)
BEGIN
WRITE(TO7470,'SP2;SM',CHARMARKER,';');
IF CONPTS=TRUE THEN
  WRITE(TO7470,'PD;') (*PEN DOWN*)
ELSE
  WRITE(TO7470,'PU;');  (*PEN UP*)
WRITE(TO7470,'PA',LOG(X):5:3,',',LOG(Y):5:3,';SM;');
END;


PROCEDURE LAYOUT;
(* DRAWS ELECTRICAL CONNECTIONS *)
VAR C:CHAR;
BEGIN
INITTURTLE;
PENCOLOR(NONE);
MOVETO(15,180);
WSTRING('ECI      INSTRUMENT CONNECTIONS FOR GB');
MOVETO(40,0);
PENCOLOR(WHITE);
MOVETO(40,200);
MOVETO(40,160);
MOVETO(190,160);
MOVETO(190,100);
MOVETO(40,100);
MOVETO(40,75);
MOVETO(60,75);
MOVETO(60,55);
MOVETO(40,55);
MOVETO(40,30);
MOVETO(220,30);

VIEWPORT(160,220,110,150);
FILLSCREEN(WHITE);

VIEWPORT(80,140,10,50);
FILLSCREEN(WHITE);

VIEWPORT(180,240,10,50);
FILLSCREEN(WHITE);
```

```
VIEWPORT(0,300,0,200);
PENCOLOR(NONE);
MOVETO(183,125);
CHARTYPE(7);
WSTRING('GB');
MOVETO(82,32);
WSTRING('KEITHLEY');
MOVETO(92,22);
WSTRING('G=100');
MOVETO(190,32);
WSTRING('FLUKE');
MOVETO(190,22);
WSTRING('2100A');
CHARTYPE(10);
MOVETO(32,157);
WSTRING('Z');
MOVETO(32,97);
WSTRING('I');
MOVETO(25,72);
WSTRING('VI');
MOVETO(25,52);
WSTRING('A6');
MOVETO(25,27);
WSTRING('A4');
READLN(C);
TEXTMODE;
END;

PROCEDURE COLLECT;
(* SWEEPS THE OUTPUT VOLTAGE AND RECORDS
   THE RESULTING CURRENTS IN ITV *)
VAR SVOLTS,TOTAL,IVALUE,IVOLTS:REAL;
    RANGE:INTEGER;

BEGIN

INITECIDAS; (*<<<<<<<<<<<<<<<<<<<<<<<<*)

WRITELN(CHR(12));
WRITELN('I-V DATA FOR      TEMP>>>>>>>>>>>>>',TEMP[TEMPCO]:5:1);
WRITELN('                  V-TEMP..........',VTEMP[TEMPCO]:5:3);
WRITELN;
WRITE('           ');
WRITELN('    V         I');WRITELN;
FOR J:=1 TO 10 DO (* 10 VOLTAGES *)
  BEGIN
  ANALOGOUT(V[J]*10.0,Z);
  RANGE:=2;
  I2V(RANGE);
  SAMPLE(6,IVOLTS);
  (* FIND SUITABLE CURRENT SENSITIVITY *)
  WHILE (-IVOLTS < 1.0) AND (RANGE <> 5) DO
    BEGIN
    RANGE:=RANGE+1;
    I2V(RANGE);
    SAMPLE(6,IVOLTS);
    END;
  TOTAL:=0.0;
  FOR I:=1 TO 10 DO (* AVE. OF 10 SAMPLES *)
    BEGIN
    SAMPLE(6,IVOLTS);
    TOTAL:=TOTAL+IVOLTS;
    END;
  IVOLTS:=TOTAL/10.0;
  IVALUE:=-IVOLTS/PWROFTEN(RANGE+1);(* REAL-VALUE CURRENT *)
  IF IVALUE < 1.0E-6 THEN (*FOR SMALL I MAKE 100 SAMPLES*)
```

```
      BEGIN
      TOTAL:=0.0;
      FOR I:=1 TO 100 DO
        BEGIN
        SAMPLE(6,IVOLTS);
        TOTAL:=TOTAL+IVOLTS;
        END;
      IVOLTS:=TOTAL/100.0;
      IVALUE:=-IVOLTS/PWROFTEN(RANGE+1);
      END;
   ITV[TEMPCO,J]:=IVALUE;(*SAVE I*)
   WRITE('            ');
   WRITELN(V[J]:5:3,'   ',IVALUE);
   END;
ANALOGOUT(0.0,Z);(*ZERO OUTPUT*)
END;


PROCEDURE CASEO;
(* USES THE TEMPERATURES PREVIOUSLY
   ENTERED IN THE TEMP.TEXT FILE
   AS MEASUREMENT TEMPS. *)
VAR TVOLTS:REAL;
BEGIN
RESET(T,'TEMP.TEXT');
FOR I:=1 TO 10 DO READLN(T,TEMP[I],VTEMP[I]);
WRITELN;
FOR I:=1 TO 10 DO WRITELN('          T(',I:2,')=',TEMP[I]:3:1,'K');
WRITELN('HIT <RET> TO INITIATE EXPERIMENT');
REPEAT UNTIL KEYPRESS;
READ(KEYBOARD,INCHAR);
WRITELN('NEXT T=',TEMP[1]:5:1);
TEMPCO:=1;
WHILE  TEMPCO < 11 DO (*FOR 10 TEMPS*)
   BEGIN
   REPEAT (*WAIT UNTIL SPECIFIED TEMP*)
      TOTAL:=0.0;
      FOR I:=1 TO 10 DO (*AVE. 10 SAMPLES*)
        BEGIN
        SAMPLE(4,TVOLTS);
        TOTAL:=TOTAL+TVOLTS;
        END;
      TVOLTS:=TOTAL/10.0;
   UNTIL TVOLTS > VTEMP[TEMPCO];
   COLLECT;(*SWEEP THROUGH VOLTAGE STEPS*)
   TEMPCO:=TEMPCO+1;(*NEXT TEMP*)
   WRITELN;
   WRITELN('******** EXPERIMENT IN PROGRESS ********');
   FOR I:=1 TO 5 DO WRITELN;
   FOR I:=1 TO 14 DO WRITE(' ');
   IF TEMPCO<11 THEN WRITE('NEXT T=',TEMP[TEMPCO]:5:1);
   FOR I:=1 TO 27 DO WRITE(' ');
   END;
END;

PROCEDURE CASEN;
(* USED FOR CALIBRATING A NEW SET OF TEMPERATURE POINTS.
   TEN TEMPS MUST BE ENTERED.
   MAIN FCN IS TO ASSOCIATE A VOLTAGE FROM
   THE TEMP-METER WITH THE MEASURED TEMP..
   THE VOLTAGE-TEMP PAIRS (10) ARE ENTERED IN TEMP.TEXT *)
VAR TVOLTS:REAL;
BEGIN
WRITELN(CHR(12));
WRITELN('MANUAL TEMP INPUT MODE...');
WRITELN;
```

```
WRITELN('ENTER TEMP (K) TO INITIATE SAMPLING');
WRITELN('ENTER 0 TO EXIT.  # OF T-PTS IS 10.');
WRITELN;
TEMPCO:=1;
WHILE TEMPCO<11 DO
  BEGIN
  WRITE('T(',TEMPCO:2,')=');
  READLN(TEMP[TEMPCO]);
  TOTAL:=0.0;
  FOR I:=1 TO 10 DO (* AVE SIGNAL FROM TEMP-METER*)
    BEGIN
    SAMPLE(4,TVOLTS);
    TOTAL:=TOTAL+TVOLTS;
    END;
  VTEMP[TEMPCO]:=TOTAL/10.0;
  COLLECT;(*SWEEP VOLTAGE STEPS*)
  IF TEMP[TEMPCO]=0.0 THEN TEMPCO:=11;
  WRITELN;
  WRITELN('******** EXPERIMENT IN PROGRESS ********');
  TEMPCO:=TEMPCO+1;(* NEXT TEMP *)
  END;
IF TEMPCO=11 THEN WRITELN('LAST TEMP PT.');
REWRITE(T,'TEMP.TEXT');
FOR I:=1 TO 10 DO WRITELN(T,TEMP[I]:5:1,VTEMP[I]:6);
CLOSE(T,LOCK);(*SAVE TEMPS*)
END;

PROCEDURE WRITEDATA;
(* GENERATES A TABLE OF CURRENTS FROM THE ITV MATRIX*)
VAR YMINMAG,YDECNUM:INTEGER;
    CONPTS:BOOLEAN;
BEGIN
IF (MODECHAR='O') OR (MODECHAR='N') THEN
  BEGIN
  REWRITE(IVDATA,'ITV.TEXT');
  FOR TEMPCO:=1 TO 10 DO
    FOR J:=1 TO 10 DO
      WRITELN(IVDATA,ITV[TEMPCO,J]);
  CLOSE(IVDATA,LOCK);
  END;
IF MODECHAR='P' THEN
  BEGIN
  RESET(T,'TEMP.TEXT');
  FOR I:=1 TO 10 DO READLN(T,TEMP[I],VTEMP[I]);
  END;
RESET(IVDATA,'ITV.TEXT');
FOR TEMPCO:=1 TO 10 DO
  FOR J:=1 TO 10 DO
    READLN(IVDATA,ITV[TEMPCO,J]);
WRITELN;
WRITE('            ');
FOR TEMPCO:=1 TO 5 DO
  BEGIN
  WRITE(TEMP[TEMPCO]:5:1);
  WRITE('        ');
  END;WRITELN;
FOR J:=1 TO 10 DO
  BEGIN
  WRITE(V[J]:5:2,'   ');
  FOR TEMPCO:=1 TO 5 DO WRITE(ITV[TEMPCO,J]:13);
  WRITE('   ',V[J]:5:2);
  WRITELN;
  END;
WRITE('            ');
FOR TEMPCO:=6 TO 10 DO
  BEGIN
```

```
  WRITE(TEMP[TEMPCO]:5:1);
  WRITE('        ');
  END;WRITELN;
FOR J:=1 TO 10 DO
  BEGIN
  WRITE(V[J]:5:2,'   ');
  FOR TEMPCO:=6 TO 10 DO WRITE(ITV[TEMPCO,J]:13);
  WRITE('   ',V[J]:5:2);
  WRITELN;
  END;
IF PLOT THEN(* GEN LOGI-LOGV PLOT ON 7470 FROM ITV*)
  BEGIN
  YMINMAG:=TRUNC(LOG(ITV[10,1]))-1;
  YDECNUM:=TRUNC(LOG(ITV[1,10]))-YMINMAG;
  LOGGRID(YDECNUM,YMINMAG,2,-2);
  WRITE(TO7470,'DIO.0,1.0;SIO.11,0.17;');
  FOR TEMPCO:=1 TO 10 DO
    BEGIN
    CONPTS:=FALSE;
    FOR J:=1 TO 10 DO
      BEGIN
      LOGPLOT(ITV[TEMPCO,J],V[J],YDECNUM,YMINMAG,2,-2,CONPTS,CHR(111));
      CONPTS:=TRUE;
      END;
    WRITE(TO7470,'PU;LB',ROUND(TEMP[TEMPCO]):4,CHR(3));
    END;
  WRITE(TO7470,'SPO;');
  END;
END;

BEGIN (*MAIN*)
V[1]:=0.02;(* SPECIFY VOLTAGE STEPS *)
V[2]:=0.03;
V[3]:=0.05;
V[4]:=0.07;
V[5]:=0.10;
V[6]:=0.15;
V[7]:=0.20;
V[8]:=0.30;
V[9]:=0.40;
V[10]:=0.50;
WRITE('N)EW OR O)LD TEMP PTS OR P)RESENT DATA? * ');
(* N=CALIB OF NEW TEMP PTS
   O=PREVIOUSLY CALIB TEMPS FOUND IN TEMP.TEXT
   P=WRITE DATA PRESENTLY IN ITV.TEXT *)
REPEAT UNTIL KEYPRESS;
READ(MODECHAR);WRITELN;
WRITE('PLOT? (Y/N) * ');(*LOGI-LOGV ON 7470A*)
REPEAT UNTIL KEYPRESS;
READ(INCHAR);
IF INCHAR='Y' THEN PLOT:=TRUE ELSE PLOT:=FALSE;
IF (MODECHAR='N') OR (MODECHAR='O') THEN LAYOUT;
CASE MODECHAR OF
  'O':CASEO;
  'N':CASEN;
  END;
WRITEDATA;
END.
```

Appendix C

AUTODECON

```
        (*$S+*)
PROGRAM AUTODECON;

(**************** APPLE AUTODECON I.0 **************)

(* BY G.C.MCGONIGAL/1982
   MATERIALS & DEVICES RESEARCH LAB
   DEPT.  OF ELECTRICAL ENGINEERING
   UNIVERSITY OF MANITOBA.

   FUNCTION:
   DECONVOLVES THE INTERFACE-STATE DENSITY FROM
   I-V CHARACTERISTICS OF GB SAMPLES MEASURED
   BY THE PROGRAM 'GB'.

   DATA:
   1) FILE 'TEMP.TEXT' MUST BE PRESENT ON #4.
   2) EITHER 'ITV.TEXT' SHOULD BE PRESENT ON #4
        OR
      A TEXT FILE ON #5 MAY BE USED AS DATA.

   OUTPUT: ROUTED TO THE 7470A PLOTTER.
*)

USES TRANSCEND,APPLESTUFF,TURTLEGRAPHICS,PLOTAIDS,PLOTTERGRAPHICS,PENPLOTAIDS;

CONST EPSILONO=8.854E-14;
      EPSILONR=11.7;
      Q=1.6E-19;
      ND=3.0E15;
      AREA=3.0E-3;

VAR T,NC,QGBA,QGB1,MU,VT,EPSILONS,
    VDF,VDR,QGB,I,DELTAQ,DELTAEF,EF,EF1,NIS,
    XBOT,YBOT,XDIV,YDIV,
    PHIN,V,J,JOE:REAL;
    FILENAME,TLABEL,SAMPLENAME:STRING[20];
    TDECON:ARRAY[1..10] OF BOOLEAN;
    CONPTS:BOOLEAN;
    TVAL,VAPP:ARRAY[1..10] OF REAL;
    INCHAR:CHAR;
    STEPMAX,VCO,TEMPCO,II:INTEGER;
    TVALUES,IVDATA:TEXT;
    ITV:ARRAY[1..10] OF ARRAY[1..10] OF REAL;
    SYM:POINTMARKER;
```

```
PROCEDURE VOLTAGEDIV;
(* FORW & REV. DIFFUSION POTS *)
BEGIN
VDF:=EPSILONS/2.0/Q/ND*SQR(-V*Q*ND/QGB+QGB/2.0/EPSILONS);
VDR:=EPSILONS/2.0/Q/ND*SQR(QGB/2.0/EPSILONS+V*Q*ND/QGB);
END;

PROCEDURE SATCUR;
(* SATURATION CURRENT *)
BEGIN
JOE:=J/(1.0-EXP(-V/VT));
END;

PROCEDURE BARRIER;
(* BARRIER HEIGHT *)
VAR E1:REAL;
BEGIN
E1:=SQRT(2.0*Q*ND*VDF/EPSILONS);
VDF:=-VT*LN(2.0*JOE/(Q*ND*MU*E1));
END;

PROCEDURE CHARGE;
(* GRAIN BOUNDARY CHARGE *)
VAR E1,E2:REAL;
BEGIN
E1:=SQRT(2.0*Q*ND*VDF/EPSILONS);
E2:=-SQRT(2.0*Q*ND*VDR/EPSILONS);
QGB:=EPSILONS*(E1-E2);
END;


PROCEDURE WRITEHEADER;
BEGIN
VAPP[1]:=0.02;
VAPP[2]:=0.03;
VAPP[3]:=0.05;
VAPP[4]:=0.07;
VAPP[5]:=0.10;
VAPP[6]:=0.15;
VAPP[7]:=0.20;
VAPP[8]:=0.30;
VAPP[9]:=0.40;
VAPP[10]:=0.50;

WRITELN(CHR(12));
WRITELN;
WRITELN('        MDRL AUTODECON I.0  OCT/82');
WRITELN;
WRITELN('DATAFILE NAME ( ''*'' FOR PRESENT)');
READLN(FILENAME);
IF POS('*',FILENAME)=1 THEN
   (* FIRST CHAR='*' USE PRESENT ITV *)
   BEGIN
   FILENAME:='ITV.TEXT';
   WRITE('SAMPLE ');
   READLN(SAMPLENAME);
   END
ELSE
   (* USE FILE FROM #5 FOR ITV *)
   BEGIN
   SAMPLENAME:=FILENAME;
   FILENAME:=CONCAT('#5:',FILENAME,'.TEXT');
   END;
WRITELN;
```

```
FOR II:=1 TO 10 DO
  BEGIN
  WRITE('          DECON T(',II:2,')=',TVAL[II]:4:1,'? (Y/N) ');
  REPEAT UNTIL KEYPRESS;
  READ(INCHAR);
  IF INCHAR='Y' THEN
    TDECON[II]:=TRUE
  ELSE
    TDECON[II]:=FALSE;
  WRITELN;
  END;
WRITELN('MAX VSTEPS=? ');(* # OF VOLTAGE STEPS TO BE DECONED*)
READLN(STEPMAX);
WRITELN(CHR(12));
WRITELN('DECON FOR: ');
FOR II:=1 TO 10 DO
  IF TDECON[II]=TRUE THEN WRITELN('                    ',TVAL[II]:4:1);
HDRAWGRID(0.5,0.40,14.5,10.0,XBOT,YBOT,XDIV,YDIV,FALSE);
HAXISLABEL('     EGAP','LOG NIS');
HMOVETO(30.0,178.0);
HWSTRING(SAMPLENAME);
END;

PROCEDURE WRITEDATA;
(* GEN CURRENT TABLE FROM ITV *)
BEGIN
WRITELN;
WRITE('               ');
FOR TEMPCO:=1 TO 5 DO
  BEGIN
  WRITE(TVAL[TEMPCO]:5:1);
  WRITE('        ');
  END;WRITELN;
FOR II:=1 TO 10 DO
  BEGIN
  WRITE(VAPP[II]:5:2,'   ');
  FOR TEMPCO:=1 TO 5 DO WRITE(ITV[TEMPCO,II]:13);
  WRITE('   ',VAPP[II]:5:2);
  WRITELN;
  END;
WRITE('               ');
FOR TEMPCO:=6 TO 10 DO
  BEGIN
  WRITE(TVAL[TEMPCO]:5:1);
  WRITE('        ');
  END;WRITELN;
FOR II:=1 TO 10 DO
  BEGIN
  WRITE(VAPP[II]:5:2,'   ');
  FOR TEMPCO:=6 TO 10 DO WRITE(ITV[TEMPCO,II]:13);
  WRITE('   ',VAPP[II]:5:2);
  WRITELN;
  END;
END;


FUNCTION MUP(T:REAL):REAL;
(* HOLE MOBILITY(TEMP) -LOW FIELD *)
(* C.F. ARORA ET. AL. *)
VAR MU1,MU2,TN:REAL;
BEGIN
TN:=T/300.0;
MU1:=1.36E8*EXP(-2.23*LN(T));
MU2:=ND/2.35E17/EXP(2.4*LN(TN))*0.88*EXP(-0.146*LN(TN));
MUP:=54.3*EXP(-0.57*LN(TN))+MU1/(1.0+MU2);
END;
```

```
PROCEDURE GBFERMI;
(* GRAIN BOUNDARY FERMI-LEVEL *)
VAR NO,E1,E2:REAL;
BEGIN
E2:=-V*Q*ND/QGB-QGB/2.0/EPSILONS;
E1:=-V*Q*ND/QGB+QGB/2.0/EPSILONS;
NO:=ND/(E2-E1)*(E2*EXP(-VDR/VT)-E1*EXP(-VDF/VT));
EF:=-VT*LN(NO/NC);
END;

(* END SUBROUTINES *)

(* MAIN BEGINS HERE *)

BEGIN

RESET(TVALUES,'TEMP.TEXT');

FOR TEMPCO:=1 TO 10 DO
  READLN(TVALUES,TVAL[TEMPCO]);

WRITEHEADER;

RESET(IVDATA,FILENAME);

FOR TEMPCO:=1 TO 10 DO
  FOR VCO:=1 TO 10 DO
    READ(IVDATA,ITV[TEMPCO,VCO]);

WRITEDATA;

EPSILONS:=EPSILONR*EPSILONO;
TEMPCO:=1;
WHILE TEMPCO <= 10 DO
   BEGIN
   WRITE('TEMPCO=');
   IF TDECON[TEMPCO] THEN
     BEGIN
     WRITELN(TEMPCO);
     DELTAQ:=0.0;
     QGB1:=0.0;
     QGB:=0.0;
     CONPTS:=FALSE;

     I:=ITV[TEMPCO,1];
     V:=VAPP[1];
     T:=TVAL[TEMPCO];

     MU:=MUP(T);
     NC:=1.925E15*SQRT(T*T*T);
     VT:=8.67E-5*T;
     PHIN:=LN(NC/ND)*VT;
     J:=I/AREA;
     VDF:=0.1;
     VDR:=0.1;
     SATCUR;
     BARRIER;
     CHARGE;
     FOR VCO:=1 TO STEPMAX DO (* EACH VOLTAGE PT *)
       BEGIN
       V:=VAPP[VCO];
       I:=ITV[TEMPCO,VCO];
       J:=I/AREA;
```

```
      REPEAT (* FOR BARRIER HEIGHT *)
        QGBA:=QGB;
        VOLTAGEDIV;
        SATCUR;
        BARRIER;
        CHARGE;
      UNTIL ABS(QGBA-QGB) = 0.0;
      GBFERMI;(* ENERGY OF FERMI-LEVEL AT BOUNDARY *)
      IF QGB1 <> 0.0 THEN
        (* FIND SHIFT OF FERMI FROM PREVIOUS VALUE *)
        BEGIN
        DELTAEF:=EF-EF1;
        DELTAQ:=QGB-QGB1;
        NIS:=-DELTAQ/DELTAEF/Q;(*INTERFACE-STATE DENSITY*)
        WRITE('                    >');
        WRITELN(DELTAEF/2.0+EF1:3:3,NIS);
        IF NIS >= 0.0 THEN SYM:=SQUARE ELSE SYM:=CROSS;
        HPLOTPOINT(DELTAEF/2.0+EF1,LOG(ABS(NIS)),XBOT,YBOT,XDIV,YDIV,CONPTS,SYM);
        END;
      EF1:=EF;(* SAVE THESE CONDITIONS *)
      QGB1:=QGB;
      WRITELN(V:2:2,I,QGB);
      END;
    END;
  TEMPCO:=TEMPCO+1;(* REPEAT FOR NEXT TEMP*)
  END;
END.
```

Appendix D

ADAP USERS' GUIDE

ADAP

Automated Data Aquisition & Processing System

Users' Guide

by

G.C. McGonigal

Materials & Devices Research Laboratory
Department of Electrical Engineering
University of Manitoba

OVERVIEW

ADAP is a computer-controlled data acquisition and processing system specifically designed for materials and devices testing and experimentation. An Apple II+ microcomputer provides the computing power required for data collection, reduction, and display. Experiments are interfaced to the computer via an Experiment Control Interface (ECI). ECI was designed by the author to provide:

1. 12-bit A/D converter

2. 8 buffered voltage inputs

3. 4 10-bit D/A converters

4. counter/timer

5. current-to-voltage converter

6. several digital inputs and outputs.

ACKNOWLEDGMENTS

The author wishes to acknowledge John Shaw, Robert Allen and Bruce Klimpke for their helpful suggestions concerning this project.

CONTENTS

LIST OF FIGURES

1

SYSTEM COMPONENTS

This section is intended to introduce the user to the ADAP system. Figure 1 is a schematic diagram of this system.

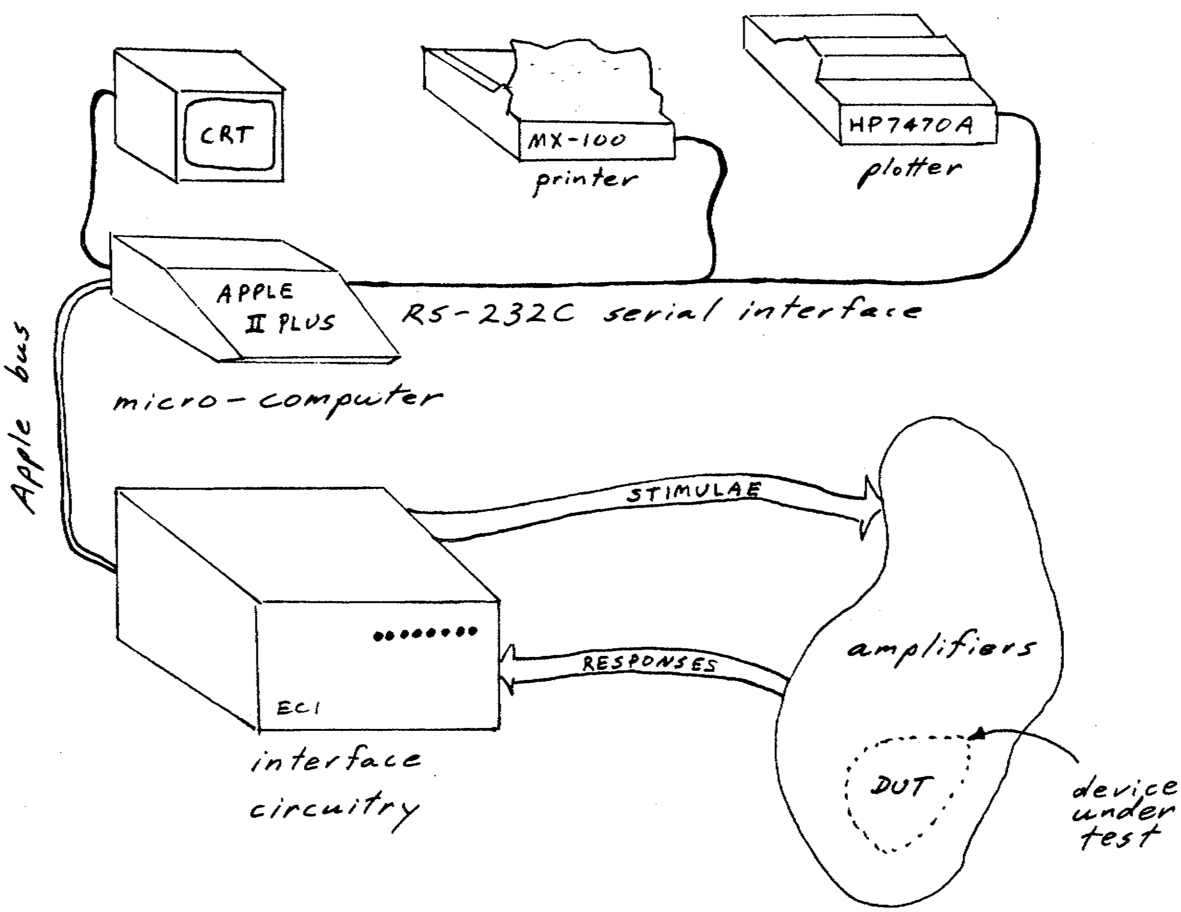


Figure 1: Schematic diagram of the ADAP system.

1.1 APPLE II+ MICROCOMPUTER

The heart of ADAP is an Apple II+ microcomputer outfitted with 64k RAM, two Apple 5-1/4 in. disk-drives and an NEC green-phosphor television monitor.

1.2 PROGRAMMING

A working knowledge of the (UCSD) Pascal computer language is required to program ADAP. This manual assumes that the user is familiar with Pascal. The user is referred to the Apple II Pascal manuals [1,2] for a detailed description of this language and operating system.

The ADAP software resides on four floppy-disks labeled 'ADAP1', 'ADAP2','ADAP3', and 'ADAP4'. ADAP1 and ADAP2 contain the Apple II Pascal 1.1 compiler, editor, and operating system. ADAP3 contains the source and code files that have been added to the system library for use by the ADAP system. ADAP4 contains many example programs that have been written for the ADAP system. ADAP1 includes the system library which contains the subroutines written specifically for ADAP. These subroutines are written in Pascal and are intended to simplify the implementation of ADAP programs. Included in the system library are programs for controlling ECI and plotting results on the Apple TV screen or HP7470A pen-plotter.

The ECI subroutines are especially important for they perform the machine-level tasks required to control the individual devices in the ECI interface. The use of these subroutines allows the user to access the various devices by simple names and pass parameters in convenient units (e.g. volts instead of their binary equivalent).

A complete description of the ADAP software is given in chapter 2.

1.3 SSC SERIAL INTERFACE

An Apple SSC RS-232C interface card provides serial communication to various peripherals. The SSC resides in card-slot 2 on the Apple bus. The Communications Mode of operation is used with all peripherals including plotters,printers, and links to other computers. The SSC appears as volume #8 to the Pascal operating system and is also referred to as REMIN and REMOUT. It is suggested that the data rate be set to 1200 baud. Detailed information on the SSC may be obtained from the "Super Serial Card, Installation and Operating Manual" [3].

1.4 HP7470A PEN-PLOTTER

Hard-copy of graphical results is obtained from the Hewlett-Packard 7470A pen-plotter. This plotter features two pens and an addressable resolution of 0.025mm (0.001 in.). Communication with the plotter is made through the SSC. The plotter is connected to the SSC on the Apple via an RS-232 cable. The dip-switch on the plotter beside the RS-232 connector should be configured as 00011101.

Commands are sent to the plotter in HP Graphics Language (HPGL) format. These instructions are of a high-level nature and include commands for automatic scaling, straight-line pen movements from A to B, and the printing of ASCII character strings. Details of the HPGL may be obtained from the "HP7470A Interfacing and Programming Manual" [4].

A Pascal UNIT of subroutines called PENPLOTAIDS is included in the ADAP system library. PENPLOTAIDS enables the user to obtain graphical hard-copy without a knowledge of the HPGL. Included are programs capable of drawing and appropriately scaling graphs as well as plotting points at the desired values. Also in the system library is a UNIT called PLOTTERGRAPHICS which is similar to TURTLEGRAPHICS. PLOTTERGRAPHICS provides many of the subroutines for drawing with the pen-plotter that TURTLEGRAPHICS provides for drawing on the Apple screen. A detailed description of these UNITS is found in chapter 2.

With PENPLOTAIDS and PLOTTERGRAPHICS the user is able to generate plots of usable quality. If further embelishments to the plots are required the user is referred to the 7470A manual mentioned above to take full advantage of its features.

1.5 PRINTER

Programs and numerical results may be transferred to a line printer such as the Epson MX-100 via the serial interface. Programs may be printed as they are compiled by adding the compiler directive (*$L #8:*) to the top of the program. Disk files may be printed using the filer's T)RANSFER command with the destination specified as #8:. During program execution numerical results may be written to a TEXT file associated with #8: (see Pascal manual for syntax).

Note that the printer and plotter cannot be on-line simultaneously as they both require the SSC port.

1.6 EXPERIMENT CONTROL INTERFACE

The Experiment Control Interface (ECI) provides the interface between the user's experiment and the Apple computer.

ECI was designed to monitor the analog outputs of instruments as well as provide voltage stimulae to their analog inputs. Direct connection to a device under test should be undertaken only with extreme caution. ECI is a peripheral board housed in its own cabinet external to the Apple and is connected to card-slot 3 of the Apple bus via ribbon cable. ECI provides:

1. 12-bit A/D converter
2. 8 buffered voltage inputs
3. 4 10-bit D/A converters
4. counter/timer
5. current-to-voltage converter
6. several digital inputs and outputs.

Full details of the ECI hardware are given in chapter 3.

2

ADAP SOFTWARE

This chapter endeavours to present detailed documentation of the existing ADAP software as well as provide information regarding the more subtle aspects of the Apple Pascal system that the user may find useful.

2.1 APPLE II PASCAL 1.1

Apple II Pascal 1.1 was chosen as the programming language for ADAP because of its logical structure, flexibility, and readability. Apple Pascal closely follows the syntax of UCSD Pascal. This manual will assume the user has a working knowledge of this language. The particulars of Apple II Pascal may be obtained from the manuals [1,2].

The Apple Pascal provides the following facilities:

1. Pascal Compiler. Generates code for execution from a Pascal source file.

2. Editor. This is a full-screen editor used to create the Pascal source files.

3. Filer. This facility handles the bookkeeping for the files on the floppy-disks.

4. Assembler. Used to create assembly language subroutines that may be linked into a Pascal host file for execution. The assembler is required only when using the system at the machine-code level and is therefore not normally needed for ADAP programs.

5. Linker. The linker is used for adding assembly language subroutines to host Pascal programs and adding Pascal UNITs to the system library. The linker is not usually invoked by the ADAP user.

2.2 SYSTEM LIBRARY

The system library resides in the file SYSTEM.LIBRARY. This file contains the subroutines written for ADAP as well as many subroutines used by the Pascal system which are also available to the user. These subroutines take the form of Pascal PROCEDUREs and FUNCTIONs. Groups of related subroutines are grouped into packages called UNITs. UNITs are accessed from the system library by the USES declaration which is required before the subroutines may be called.

Apple Pascal provides several useful units. The user may wish to refer to the Pascal manuals regarding all of these units but two are worth mentioning here. The user should be aware of the TRANSCEND unit which contains the basic transcendental functions such as LOG and SIN. The other important unit is TURTLEGRAPHICS. This unit comprises of the subroutines required to implement the UCSD Turtle Graphics language of graphics commands. These commands are useful for constructing output on the Apple's screen.

Several units written specifically for ADAP have been added to the system library. Their names and functions are as follows:

1. ECI performs the machine-level tasks required to control the individual devices in the ECI interface. These subroutines allow the user to access the various devices by simple names and pass parameters in convenient units (e.g. volts instead of their binary equivalent). A complete description of the ECI unit follows.

2. PLOTAIDS provides subroutines for drawing graphs on the Apple screen. Included is a subroutine to draw and scale a grid given its coordinate extrema. Another subroutine places markers at the desired values on the graph. A complete description of the PLOTAIDS unit follows.

3. PLOTTERGRAPHICS implements a version of UCSD Turtle Graphics that will drive the pen-plotter rather than the TV screen. PLOTTERGRAPHICS may be used to embelish plots made by PENPLOTAIDS. A complete description of the PLOTTERGRAPHICS unit follows.

4. PENPLOTAIDS is a slightly modified version of PLOTAIDS that is used to draw graphical output on the pen-plotter. A complete description of the PENPLOTAIDS unit follows.

ECI Unit

The ECI unit is a group of subroutines used to access the various devices resident on the ECI board. These subroutines perform the required machine-level tasks, thus the user is not required to learn the addresses or protocols of the various integrated circuits employed.

Apple Pascal allows assembly language subroutines to be referenced with the same syntax as a Pascal subroutine. The assembled subroutines may either be made available directly to the user (e.g. I2V) or called by another Pascal subroutine which in turn is made available (e.g. ANALOGOUT is the available Pascal subroutine that calls the assembled subroutine DIG2AN, which is not available). In the latter case the Pascal subroutines present a more intelligible parameter list to the user (e.g. Z rather than a device address).

Programs that call ECI subroutines must reference ECIDAS in the USES declaration, e.g.

```
USES ECI;
```

A listing of the compiled ECI unit and its assembled subroutines is included in Appendix A.

A complete description of the subroutines available through ECI is now given. The user may wish to consult chapter 3 regarding the hardware aspects of the devices employed.

PROCEDURE I2V(SCALE:INTEGER)

The current-to-voltage converter sensitivity is controlled by I2V. The integer parameter SCALE specifies the current that is required to produce a full-scale voltage output. The desired SCALE is calculated from the magnitude of the desired value of current (in powers of ten) multiplyed by -1. The following table lists the possible values of SCALE and their corresponding full-scale currents.

SCALE	Full-scale current (A)
2	10mA
3	1mA
4	0.1mA
5	0.01mA

FUNCTION ASSAMPLE(CHANNEL:INTEGER):INTEGER

ASSAMPLE is an assembly language subroutine that controls the analog-to-digital converter and the analog multiplexer. The operations performed by ASSAMPLE are as follows:

1. Set the analog multiplexer to the desired channel specified by the input parameter CHANNEL. This is an integer value that corresponds to the number of the input to be sampled, e.g. if CHANNEL=3 then the A3 input voltage will be routed to the analog-to-digital converter.

2. A "start of conversion" request is sent to the A/D.

3. Wait until "conversion complete" is received from the A/D.

4. Read the resulting conversion value. This value will be an integer in the range 0 to 4095. A value of 0 corresponds to an input voltage of -10.240 volts while 4095 represents 10.235 volts. Zero volts returns a value of 2047.

5. Return value and exit.

PROCEDURE SAMPLE(CHANNEL:INTEGER;VAR RRESULT:REAL)

This subroutine performs the same functions as ASSAMPLE except that the conversion result is returned as a real-numbered voltage in the range -10.240 to 10.235 volts. Note that SAMPLE is a procedure wheras ASSAMPLE is a function.

PROCEDURE DOUT(HIORLO,LINENUM:INTEGER)

The digital outputs labeled D0,D1,D2, and D3 are controled with DOUT. The parameter HIORLO specifies the state that the output LINENUM will assume after execution,e.g. the call DOUT(1,2) will make D2 go high while DOUT(0,3) would put D3 low.

FUNCTION SINPUT(LINE:INTEGER):INTEGER

SINPUT returns the status of either the S1 or S2 input.

If a low-to-high transition has occurred on the S1 input then SINPUT(1)=1, else SINPUT(1)=0. If a high-to-low transition has occurred on the S2 input then SINPUT(2)=1, else SINPUT(2)=0.

PROCEDURE ANALOGOUT(VOUT:REAL;D2A:DEVICE)

The voltage on any one of the four analog-to-digital converters is set by a call to ANALOGOUT. This procedure assumes that the converters are wired for a voltage range of -10.240 to +10.220. The Z output is, however, wired for 1 V operation so that the actual output voltage will be 1/10 of that specified by VOUT.

A new variable type called DEVICE has been created for this procedure. The members of this type are {X,Y,W,Z} which refer to the voltage outputs on the ECI module. The parameter D2A specifies which device the voltage control signals will be sent to.

The voltage to be converted is specified by the VOUT parameter. VOUT is the real-valued voltage in volts that is to appear at the output specified by D2A. Any value of type REAL between -10.24 and +10.22 can be specified.

The 10-bit converters provide a conversion resolution of 0.020 volts. Values of VOUT are thus rounded to the nearest 0.020 volts before they are sent to the converter.

As an example we may wish to set output Y to +3.20 volts. The procedure call ANALOGOUT(3.20,Y) will accomplish this.

ANALOGOUT calls the assembled procedure DIG2AN to perform the machine-level operations.

PROCEDURE INITECIDAS

This parameter-less procedure <u>must be called</u> before most of the ECIDAS devices can be accessed. One call at the start of a program is usually sufficient.

The primary function of INITECIDAS is to configure the PIA to read the A/D and S inputs, and write to the I/V converter. It also configures the three stages of the Programmable Timer Module for counting in microseconds, milliseconds and seconds. S1 is set to become active on a low-to-high input transition while S2 is active after a high-to-low input transition (see chap. 3). The assembled subroutine INITPIA is called to perform the necessary byte manipulations.

INITECIDAS also sets the voltage outputs X,Y,W,Z to 0.0 volts and the digital outputs D0..D3 to logic 0.

FUNCTION TIME(SECTION:INTEGER):INTEGER

Timing functions are provided by a 6840 Programmable Timer Module (PTM). This device contains three 16-bit counters which may be programmed to perform a variety of duties. An invocation of INITECI will configure the three counters T1, T2, and T3 to count microseconds, milliseconds, and seconds respectively. The input parameter to this assembly language function is either 1, 2, or 3 in reference to the three timer sections.

PROCEDURE ZEROT

This parameter-less procedure resets the timing counters by activating the PTM's master reset.

PLOTAIDS Unit

The unit PLOTAIDS provides facilities for constructing 2-D graphical output on the Apple's monitor. Included in this unit is a self-scaling axis generator and a subroutine for placing markers at the desired coordinates.

PLOTAIDS employs the UCSD graphics language Turtle Graphics for controlling the monitor display. The TURTLEGRAPHICS unit in the system library must be referenced before PLOTAIDS in the Pascal USES declaration, e.g. USES TURTLEGRAPHICS,PLOTAIDS. Documentation of the Turtle Graphics language can be found in the Apple Pascal "Language Reference Manual" [1].

The available resolution is 192 vertical by 282 horizontal pixels.

A listing of the compiled PLOTAIDS unit is included in Appendix A.

We now present a detailed explanation of the PLOTAIDS subroutines.

```
DRAWGRID(XMAX,XMIN,YMAX,YMIN:REAL;
         VAR XBOTTOMLINE,YBOTTOMLINE,
         XDIVSIZE,YDIVSIZE:REAL;
         DOTTEDLINES:BOOLEAN)
```

This procedure draws and labels a two-dimensional coordinate system on the monitor.

The first four parameters, XMAX, XMIN, YMAX, and YMIN, specify the value extrema that are to be plotted. This is a four-quadrant axis generator so any values may be specified. If their range is known, these values may be set at compile time. Alternatively, during execution, the data to be plotted may be scanned to determine the range of values present.

The four parameters that DRAWGRID returns, XBOTTOMLINE, YBOTTOMLINE, XDIVSIZE, and YDIVSIZE, contain information concerning the axis scaling that has been selected. These values are required by the PLOTPOINT procedure and should not be altered.

If DOTTEDLINES=TRUE, a grid of dotted reference lines will be added to the plot.

Scaling values are shown along the bottom and left-hand side of the plot. An exponential factor may be indicated by the character "E". For example if "E-3" is displayed, all values should be multiplied by 0.001.

```
PLOTPOINT(XNUM,YNUM,XBOTTOMLINE,YBOTTOMLINE,
          XDIVSIZE,YDIVSIZE:REAL;
          CONNECTPOINTS:BOOLEAN;POINTSYMBOL:POINTMARKER)
```

This procedure places markers at the desired coordinates on a plot previously constructed by DRAWGRID.

DRAWGRID must be called before PLOTPOINT.

The parameters XNUM and YNUM specify the x and y coordinate values of the point to be plotted. These values should be within the range previously established by XMAX, XMIN, YMAX, YMIN of the DRAWGRID procedure.

The input parameters XBOTTOMLINE, YBOTTOMLINE, XDIVSIZE, and YDIVSIZE should be obtained from the DRAWGRID procedure's output parameters of the same name. In this way PLOTPOINT is able to obtain information about the axis scaling that DRAWGRID has established.

If CONNECTPOINTS=TRUE a line will be drawn from the preceding point to the point currently being plotted. If CONNECTPOINTS=FALSE no line will be drawn and only the marker will appear at the specified point. This has an analogy with a mechanical plotter in that CONNECTPOINTS=TRUE puts the pen down before moving to the specified point whereas CONNECTPOINTS=FALSE will lift the pen before moving.

POINTSYMBOL specifies which marker is to be drawn at the data points. This parameter is of the special type POINTMARKER and may have the value POINT, CROSS, or SQUARE.

PLOTTERGRAPHICS Unit

PLOTTERGRAPHICS allows the user to control the Hewlett-Packard 7470A penplotter in much the same manner that TURTLEGRAPHICS controls the Apple monitor.

PLOTTERGRAPHICS supports eight of the TURTLEGRAPHICS subroutines. Their functions are analogous to those of TURTLEGRAPHICS. The similarities between the two graphics units requires that the user need only learn one graphics language in order to manipulate both the screen and the plotter. Their compatibility also allows graphics programs to be easily transferred from one medium to the other.

There are two main differences between PLOTTERGRAPHICS and TURTLEGRAPHICS:

1. All length and angular parameters in PLOTTERGRAPHICS are of type REAL (whereas they are type INTEGER in TURTLEGRAPHICS). This enables full utilization of the plotter's 25 micron addressable resolution.

2. The SCREENCOLOR type has been modified in accordance with the requirements of a two-pen plotter such as the HP 7470A. See the description of the HPENCOLOR procedure for details.

A listing of the compiled PLOTTERGRAPHICS unit is included in Appendix A.

We now present a detailed explanation of the PLOTTERGRAPHICS subroutines.

HINITTURTLE

Initialization routine for the PLOTTERGRAPHICS mode. Its main function is to create a file called PLOTTER and associate it with the serial-port file REMOUT. The label PLOTTER may not be redefined.

The 7470A plotter is initialized so that the x- and y-coordinate scaling resembles the Apple monitor, i.e. user units are 282 x 192. All pens are stored and the holder moves to position 0,0.

PROCEDURE HTURNTO(ANGLE:REAL)

Explicitly sets the angular orientation of the turtle. The parameter ANGLE is specified in degrees. Angle=0 points the turtle to the right.

PROCEDURE HTURN(ANGLE:REAL)

Adds the ANGLE value (counter-clockwise) to the current angle.

PROCEDURE HMOVETO(X,Y:REAL)

This procedure moves the pen to the absolute position (X,Y) in user units. The parameters are of type REAL to take full advantage of the plotter's resolution.

PROCEDURE HMOVE(DIST:REAL)

HMOVE moves the pen relative to its current position. This motion will be in the direction specified by the current angular variable (set by HTURN and HTURNTO) for a distance DIST specified in user units.

PROCEDURE HPENCOLOR(PENMODE:HSCREENCOLOR)

HPENCOLOR directs the pen configuration.

For this procedure a new TYPE has been created called HSCREENCOLOR.

1. HPENCOLOR(NO) Lifts the pen. If the pen is already up, no action results.

2. HPENCOLOR(YES) Puts the pen down.

3. HPENCOLOR(ONE) Loads the pen stored in the lefthand well of the 7470A. If the pen is already in use no action results.

4. HPENCOLOR(TWO) - Loads the pen stored in the right-hand pen well of the 7470A.

5. HPENCOLOR(AWAY) - Stores any pen that is currenty in use.

Note that to change pens a HPENCOLOR(AWAY) call is not required.

HVIEWPORT(LEFT,RIGHT,BOTTOM,TOP:REAL)

The plotting "window" is set by the parameters of the HVIEWPORT procedure. Values outside of the window will not be plotted. A "pen-up" command is executed by the plotter when this boundary is reached.

PROCEDURE HWSTRING(S:STRING)

The character string S is printed horizontally, at the current pen position, starting with the lower lefthand corner of the first character. The string S may be of any length. The pen should be raised prior to a HWSTRING call or else it will be returned to the paper at the starting position of the next (nonexistent) character.

PENPLOTAIDS Unit

PENPLOTAIDS is useful for obtaining graphical output on the 7470A plotter. PENPOTAIDS is analogous to PLOTAIDS in the same manner that PLOTTERGRAPHICS is analogous to TURTLEGRAPHICS. In fact, portions of PLOTTERGRAPHICS were created by changing the TURTLEGRAPHICS routines in PLOTAIDS to the PLOTTERGRAPHICS equivalent.

Included in this unit is a self-scaling axis generator and a subroutine for placing markers at the desired coordinates.

A listing of the compiled PENPLOTAIDS unit is included in Appendix A.

We now present a detailed explanation of the PENPLOTAIDS subroutines.

```
HDRAWGRID(XMAX,XMIN,YMAX,YMIN:REAL;
         VAR XBOTTOMLINE,YBOTTOMLINE,
         XDIVSIZE,YDIVSIZE:REAL;
         DOTTEDLINES:BOOLEAN)
```

This procedure draws and labels a two-dimensional coordinate system on the plotter.

The first four parameters, XMAX, XMIN, YMAX, and YMIN, specify the value extrema that are to be plotted. This is a four-quadrant axis generator so any values may be specified. If their range is known, these values may be set at compile time. Alternatively, during execution, the data to be plotted may be scanned to determine the range of values present.

The four parameters returned by HDRAWGRID, XBOTTOMLINE, YBOTTOMLINE, XDIVSIZE, and YDIVSIZE, contain information concerning the axis scaling that has been selected. These values are required by the HPLOTPOINT procedure and should not be altered.

If DOTTEDLINES=TRUE, a grid of reference lines will be added to the plot.

Scaling values are shown along the bottom and left-hand side of the plot.

```
HPLOTPOINT(XNUM,YNUM,XBOTTOMLINE,YBOTTOMLINE,
          XDIVSIZE,YDIVSIZE:REAL;
          CONNECTPOINTS:BOOLEAN;POINTSYMBOL:HPOINTMARKER)
```

This procedure places markers at the desired coordinates on a plot previously constructed by HDRAWGRID.

HDRAWGRID must be called before HPLOTPOINT.

The parameters XNUM and YNUM specify the x and y coordinate values of the point to be plotted. These values should be within the range previously established by XMAX, XMIN, YMAX, YMIN of the HDRAWGRID procedure.

The input parameters XBOTTOMLINE, YBOTTOMLINE, XDIVSIZE, and YDIVSIZE should be obtained from the HDRAWGRID procedure's output parameters of the same name. In this way HPLOTPOINT is able to obtain information about the axis scaling that HDRAWGRID has established.

If CONNECTPOINTS=TRUE a line will be drawn from the preceding point to the point currently being plotted. If CONNECTPOINTS=FALSE no line will be drawn and only the marker will appear at the specified point. This has an analogy with a mechanical plotter in that CONNECTPOINTS=TRUE puts the pen down before moving to the specified point whereas CONNECTPOINTS=FALSE will lift the pen before moving.

POINTSYMBOL specifies which marker is to be drawn at the data points. This parameter is of the special type HPOINTMARKER and may have the value HPOINT, HDOT, HCROSS, or HSQUARE. Note that the HDOT marker is not supported in PLOTAIDS.

2.3 COMPILING AND RUNNING LARGE PROGRAMS

Special considerations are required for running large programs. Unpredictable results will occur if the program is too large for the Apple's memory.

The Apple Pascal has a segmentation system which allows sections of the code to be loaded into the main memory from the floppy-disk only when they are needed. UNITs and PROCEDURE subroutines are eligible for this segmentation.

The user is referred to the discussion of the segment swapping ($S+) and $RESIDENT compiler directives in [1]. Also, [5] contains a discussion of other methods for handling large programs.

2.4 UNIT SUMMARY

ECI

DEFINED TYPES:

DEVICE=(X,Y,W,Z)

RESERVED VARIABLES:

TEM, COPY, RETAD, LINE

AVAILABLE SUBROUTINES:

```
PROCEDURE I2V(SCALE:INTEGER)

FUNCTION  TIME(SECTION:INTEGER):INTEGER

FUNCTION  SINPUT(LINE:INTEGER):INTEGER

PROCEDURE DOUT(HIORLO,LINENUM:INTEGER)

FUNCTION  ASSAMPLE(CHANNEL:INTEGER):INTEGER

PROCEDURE ZEROT

PROCEDURE SAMPLE(CHANNEL:INTEGER; VAR RRESULT:REAL)

PROCEDURE ANALOGOUT(VOUT:REAL; D2A:DEVICE)

PROCEDURE INITECI
```

PLOTAIDS

DEFINED TYPES:

```
POINTMARKER=(POINT,CROSS,SQUARE)

VECTOR = ARRAY[0..7] OF REAL
```

AUXILIARY UNITS REQUIRED:

TRANSCEND, TURTLEGRAPHICS

AVAILABLE SUBROUTINES:

```
PROCEDURE PLOTPOINT(XNUM,YNUM,XBOTTOMLINE,YBOTTOMLINE,
                    XDIVSIZE,YDIVSIZE:REAL;
                    CONNECTPOINTS:BOOLEAN;
                    POINTSYMBOL:POINTMARKER)

PROCEDURE DRAWGRID(XMAX,XMIN,YMAX,YMIN:REAL;
                   VAR XBOTTOMLINE,YBOTTOMLINE,
                   XDIVSIZE,YDIVSIZE:REAL;
                   DOTTEDLINES:BOOLEAN)
```

PLOTTERGRAPHICS

DEFINED TYPES:

HSCREENCOLOR=(NO,YES,ONE,TWO,AWAY)

RESERVED VARIABLES:

PLOTTER

AUXILIARY UNITS REQUIRED:

TRANSCEND

AVAILABLE SUBROUTINES:

PROCEDURE HINITTURTLE

PROCEDURE HTURN(ANGLE:REAL)

PROCEDURE HTURNTO(ANGLE:REAL)

PROCEDURE HMOVE(DIST:REAL)

PROCEDURE HMOVETO(X,Y:REAL)

PROCEDURE HPENCOLOR(PENMODE:HSCREENCOLOR)

PROCEDURE HVIEWPORT(LEFT,RIGHT,BOTTOM,TOP:REAL)

PROCEDURE HWSTRING(S:STRING)

PENPLOTAIDS

DEFINED TYPES:

```
HPOINTMARKER=(HPOINT,HDOT,HCROSS,HSQUARE)
```

AUXILIARY UNITS REQUIRED:

TRANSCEND, PLOTTERGRAPHICS

AVAILABLE SUBROUTINES:

```
PROCEDURE HPLOTPOINT(XNUM,YNUM,XBOTTOMLINE,YBOTTOMLINE,
                     XDIVSIZE,YDIVSIZE:REAL;
                     CONNECTPOINTS:BOOLEAN;
                     POINTSYMBOL:POINTMARKER)

PROCEDURE HDRAWGRID(XMAX,XMIN,YMAX,YMIN:REAL;
                    VAR XBOTTOMLINE,YBOTTOMLINE,
                    XDIVSIZE,YDIVSIZE:REAL;
                    DOTTEDLINES:BOOLEAN)

PROCEDURE HAXISLABEL(XLABEL,YLABEL:STRING)
```

3

ECI HARDWARE

3.1 APPLE BUS BUFFERING AND ADDRESS DECODING

Communication between the Apple computer and the ECI peripheral occurs via the Apple bus. A 24-wire ribbon cable connects a card in the Apple bus card-slot 3 to the ECI board where three 74LS245's, one for each of the data, address, and control busses provide the necessary buffering. The IRQ line is not buffered.

The Apple bus provides an 'I/O SELECT' line which is active when the memory associated with the card slot is being addressed. This provides decoding to the card-slot level so that only the eight least-significant address lines are required by ECI.

From the A3, A4, and A5 address lines, eight lines are further decoded to select the eight devices on the ECI board. These are as follows:

Chip Select Line	Device
CS0	Z D/A
CS1	W D/A
CS2	Y D/A
CS3	X D/A
CS4	PIA
CS5	amux latch
CS6	PTM
CS7	D-out latch

Each device is allocated eight consecutive memory locations. The buffering and decoding details are shown in fig. 2 .

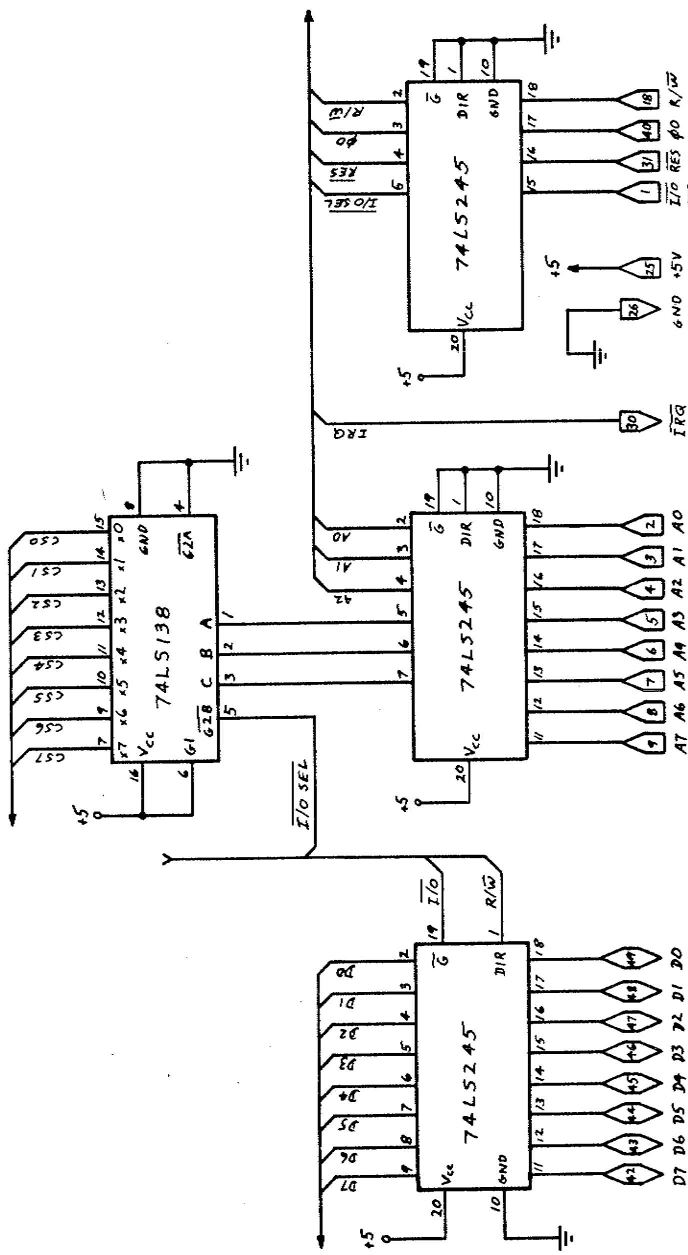


Figure 2: Bus buffering and address decoding

3.2 MEMORY MAP

The following is a compilation of the memory locations used by ECI. The ECI card resides in card-slot 3. The memory allocated for that slot is from C300 to C3FF.

LOCATION (hex)	REGISTER	DEVICE
C300	LSB	Z D/A
C301	MSB	
C308	LSB	W D/A
C309	MSB	
C310	LSB	Y D/A
C311	MSB	
C318	LSB	X D/A
C319	MSB	
C320	PRA	PIA
C321	CRA	
C322	PRB	
C323	CRB	
C328	latch	AMUX
C330	WCR1/3	PTM
C331	WCR2	
C332	read T1	
C333	write T1	
C334	read T2	
C335	write T2	
C336	read T3	
C337	write T3	
C338	latch	D-outputs

3.3 I/O LINES

The following is a list of the available ECI inputs and outputs, their functions, and electrical characteristics.

A0 - A3

These four input-pairs are connected to differential-input instrumentation amplifiers. Figure 3 is the amplifier circuit. The outputs are connected to the A/D converter's analog multiplexer for reading.

These devices are the National LH0037C. They have an input impedance of 300 MΩ. Common-mode rejection is greater than 100 dB.

Offset adjustment is via the 100K pot.

The gain of these amplifiers is normally unity, however a variable gain of up to 100 is available. This feature may be exercised by connecting a jumper between pins 7 and 8 on the amplifier's socket. Gain adjustment can now be made via the 10-turn pot which is accessible through the case.

A4 - A7

These four inputs are connected to single-ended amplifiers. Figure 4 is the amplifier circuit. The outputs are connected to the A/D converter's analog multiplexer for reading. These are non-inverting amplifiers constructed with the National LF356 op amp. This device has an input impedance of 1 GΩ.

Offset adjustment is via the 25K pot.

The gain of these amplifiers is normally unity, however a variable gain of up to 100 is available. Gain adjustment is via the 10-turn pot which is accessible through the case.

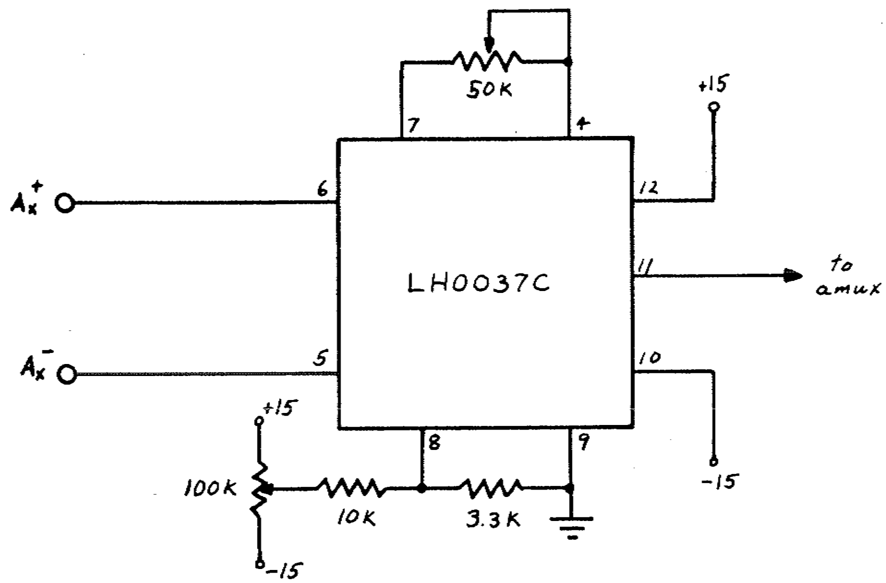


Figure 3: A0-A3 amplifier wiring connections.

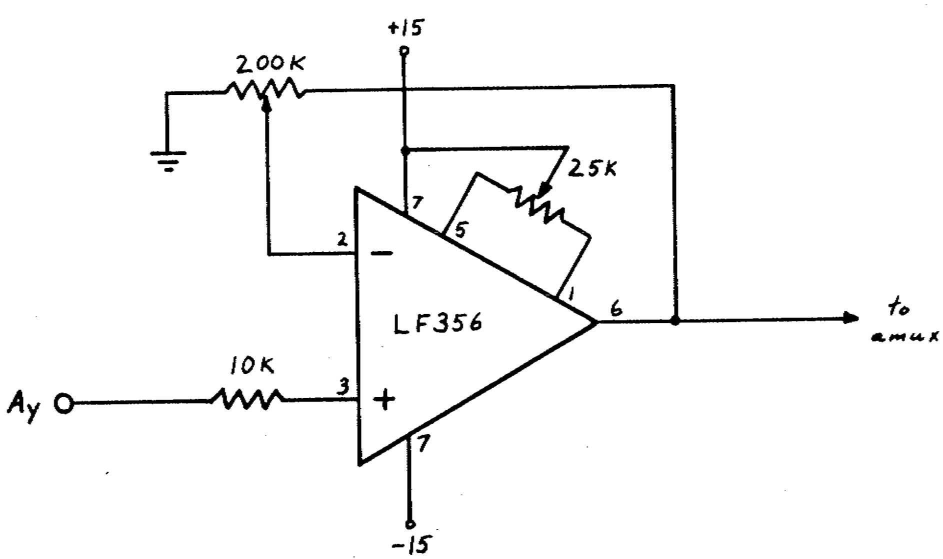


Figure 4: A4-A7 amplifier wiring connections.

D0 - D3

These are general-purpose 0-5 volt digital outputs. Their states are controlled independently with the DOUT Pascal procedure.

A 7417 provides open-collector buffering. These outputs may be wire-ORed. Figure 5 shows the wiring connections.

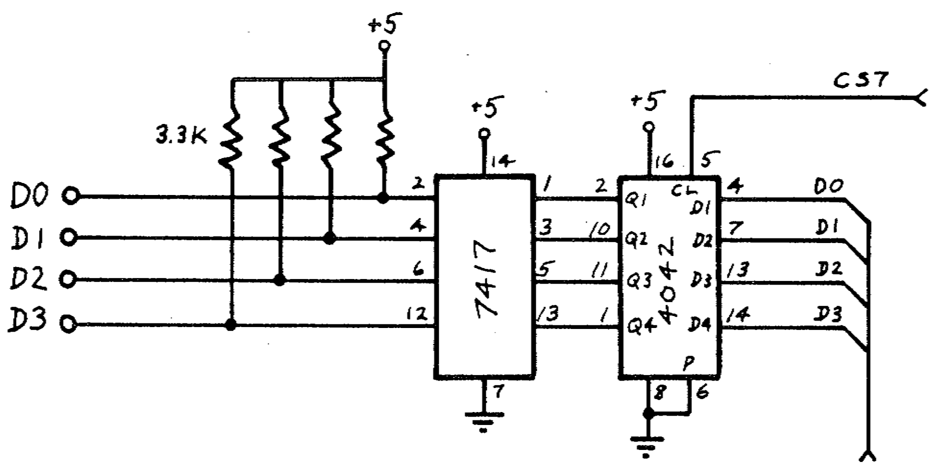


Figure 5: D-output wiring diagram.

I

This is the input for the selectable-sensitivity current-to-voltage converter.

The basic converter circuit consists of an op amp and a feedback resistor. The output voltage (see the V_I output) is the negative of the input current multiplied by the resistance. In this particular circuit, a wide range of current magnitudes may be accurrately measured because an analog multiplexer is used to select an appropriate resistance. Figure 6 shows the connection of the op amp, analog multiplexer, and measurement resistors.

The I input is at virtual ground.

The analog multiplexer channel is controlled via PB4, PB5, and PB6 on the 6821 PIA. These lines can be set with the I2V Pascal subroutine.

Four resistance values are currently supported (1K, 10K, 100K, 1M). If the 1K resistance is selected, a 10 mA current will produce a -10 V output. At the other extreme the 1 M resistance will respond to a one micro-amp input with a -1 V output. The minimum measureable current is determined by noise. If statistical averaging is employed, currents as low as 0.1 micro-amp are measureable. Four channels of the 8-input analog multiplexer are available for future use.

Output-offset adjust is provided by a 25K pot. Each of the measurement resistances have an associated pot for calibration to a known input.

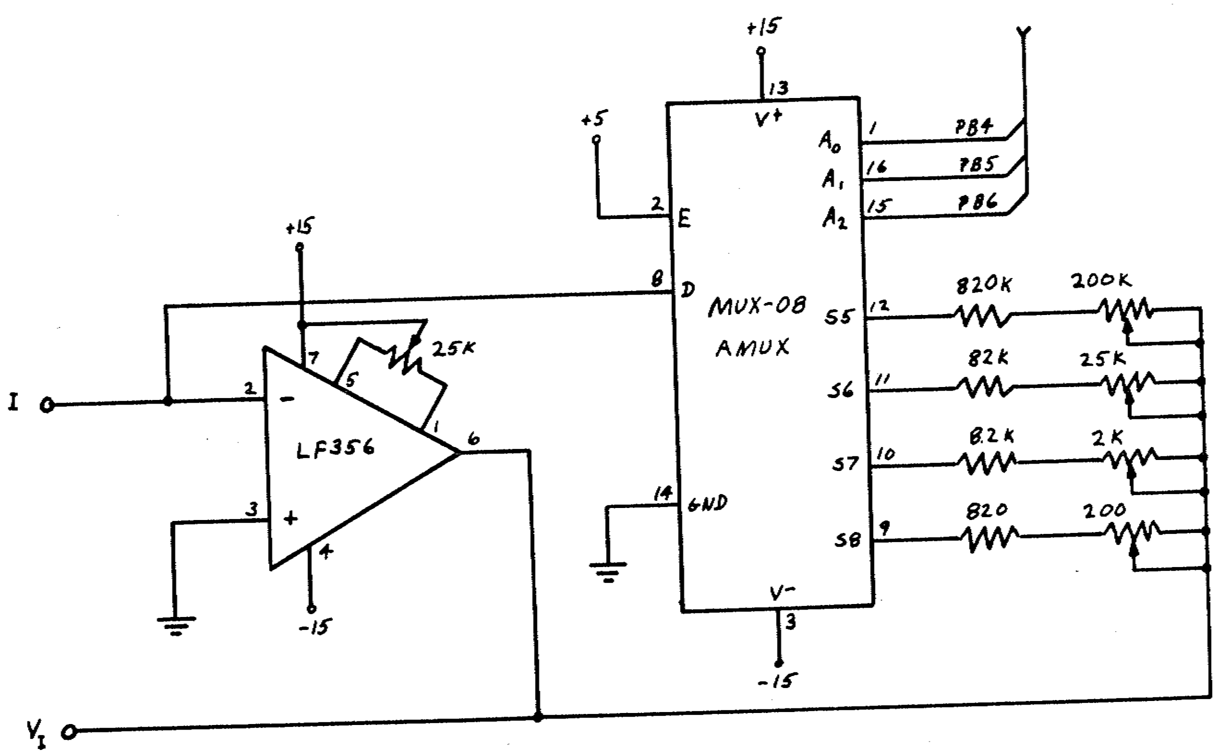


Figure 6: Current-to-voltage converter wiring details.

S1, S2

The S-inputs are general-purpose, edge-triggered digital inputs.

S1,and S2 are buffered by a 7414 Schmidt-trigger and connected to CB1 and CB2 on the 6821 PIA respectively. The status of either input may be examined with the Pascal subroutine SINPUT.

ECI is usually configured (with the INITECI procedure) so that S1 is active on a low-to-high (0-5 volt) transition and S2 is active after a high-to-low (5-0 volt) transition.

V_I

This is the output voltage of the current-to-voltage converter (see I).

Maximum output voltage is 10 volts. This output is usually connected to one of the A-inputs for reading by the A/D. Output-offset adjust is provided by a 25K pot. Each of the measurement resistances have an associated pot for calibration to a known input.

X, Y, W, Z

These four voltage outputs are each controlled by a 10-bit digital-to-analog converter (D/A).

The converter employed here is the microprocessor-compatible National DAC1006. The DAC1006 has 10-bit resolution and 10-bit accurracy.

Figure 7 is the complete wiring diagram for one of the outputs.

The output-voltage swing is determined by the feedback resistance of the final amplifier stage. A 10K resistance will produce a voltage swing from -10 to +10 volts. A 1K resistance will result in -1 to +1 volt.

The National LH0071 provides a reference voltage of 10.24 V. In this way each of the 1024 bits is approximately 20 mV over the 20.48 V output voltage range.

These outputs are controlled by the assembly-language program DIG2AN which is called by the Pascal procedure ANALOGOUT.

The Z output is equipped with pass transistors which allow this output to supply up to the power supply limit of 500 mA. All other outputs are restricted to less than 10 mA. The Z output, in standard configuration, has an output voltage swing of 1.024 volts. All other outputs have a 10.24 voltage range.

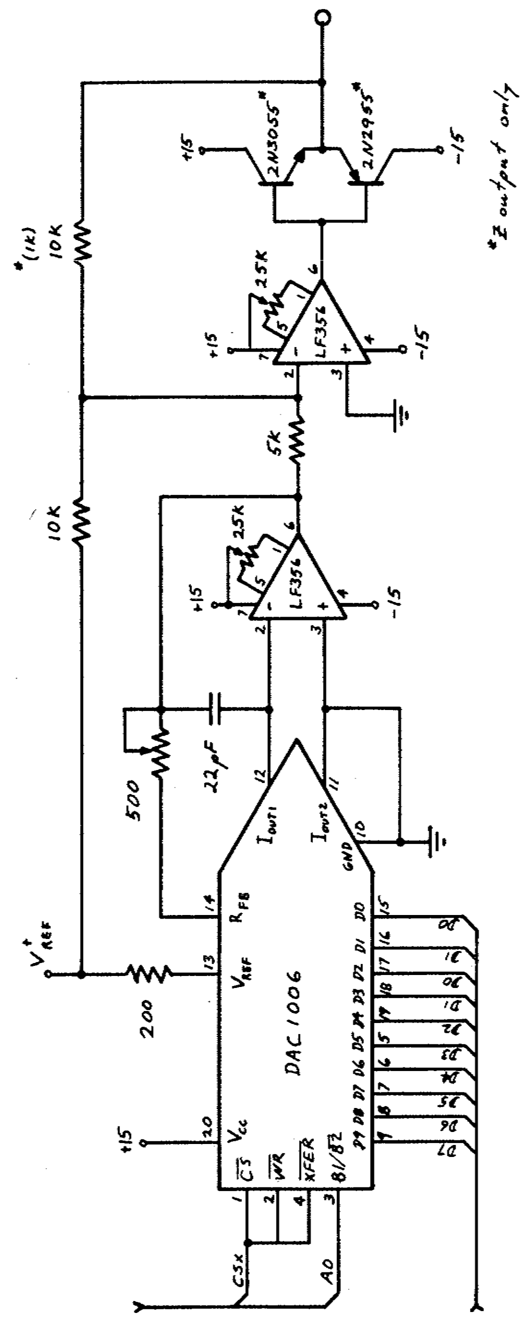


Figure 7: D/A converter wiring

3.4 ANALOG-TO-DIGITAL CONVERSION

The main function of the ECI system is to convert analog signals into a representative digital form. This is accomplished by the National ADC1211 12-bit, successive-approximation, analog-to-digital converter (A/D). An 8-channel analog multiplexer (PMI MUX-8) selects which of the A-inputs (0-7) will be routed to the input of the A/D prior to conversion initiation. The selected analog signal is preconditioned by a 1:2 voltage divider and connected through a buffer amp to the converter input.

Figure 8 is a wiring diagram of the A/D section.

A 10.24 V reference voltage is provided by a National LH0071. The converter is wired to accept bipolar inputs of one-half the reference voltage. Due to the voltage divider, the resulting input voltage range is -10.240 to +10.235 volts and each of the 4096 resolvable bits corresponds to 5 mV at the A-input (assuming unity gain).

A 140 kHz clock signal is generated by a 555 timer IC to drive the successive-approximation sequence.

The conversion time is approximately 100 microseconds.

The outputs from the 1211 are not tri-stated and therefore a Motorola 6821 Periferal Interface Adapter (PIA) is required to provide the interface with the Apple bus. The A side of the PIA is connected to the eight least-significant bits of the A/D result while PB0-PB3 monitor the four most-significant bits. The PIA also serves as the source of the 'start conversion' pulse and monitors the 'end of conversion' line. Because the 1211 is a CMOS device, level-converters (MM74C901,MM74C906) are required to interface the digital signals.

The analog multiplexer is controlled by three outputs from an independent latch (4042). For diagnostic and cosmetic reasons, a string of LED's have been added to the front panel of the ECI case. These LED's show which input channel is currently routed to the A/D. This function is accomplished by driving the LED's from another analog multiplexer (4051) that is also receiving the control signals from the latch.

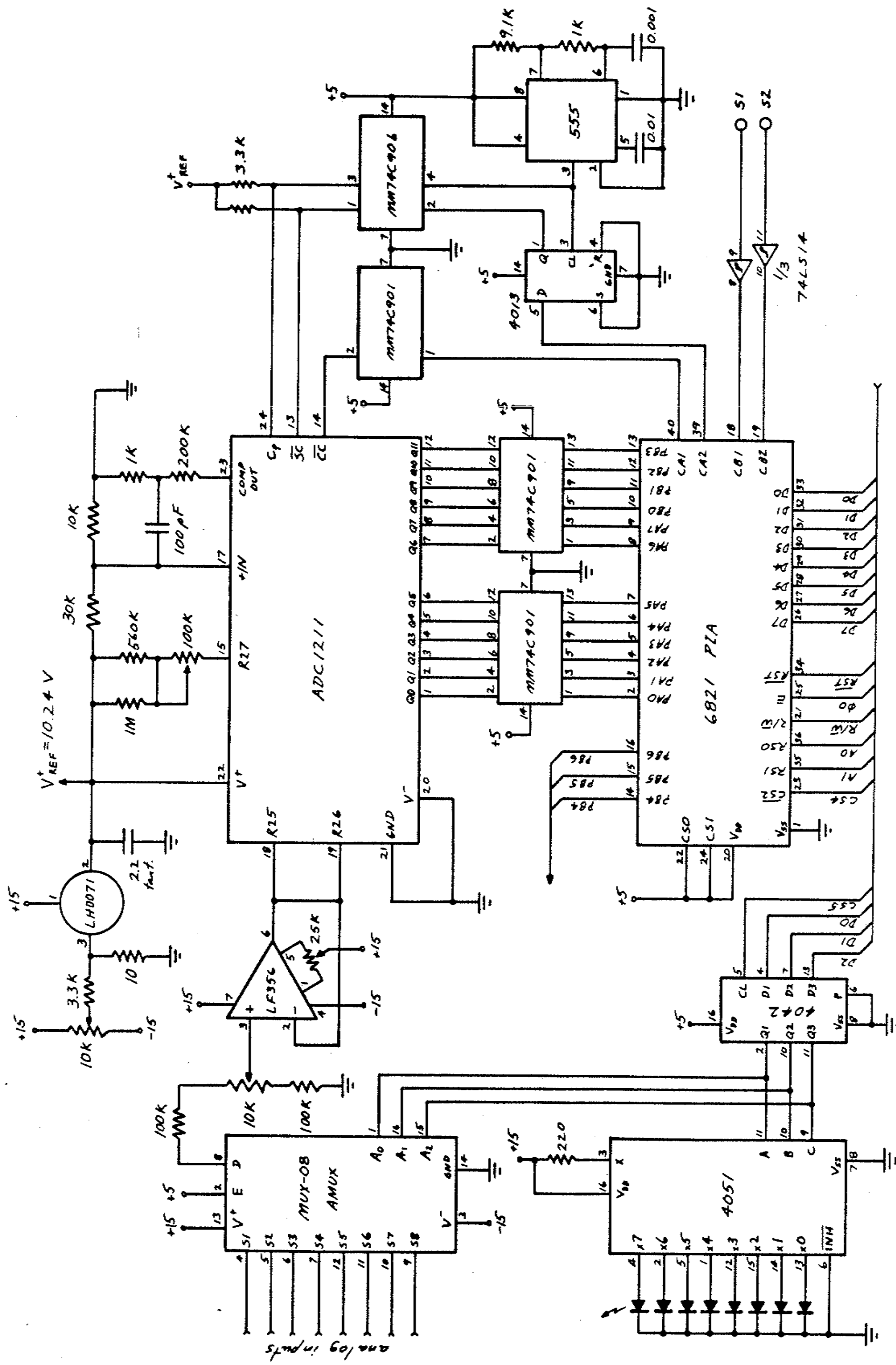


Figure 8: A/D section wiring.

3.5 PROGRAMMABLE TIMER MODULE (PTM)

The Motorola 6840 Programmable Timer Module (PTM) is capable of timing, event counting, frequency comparison, and astable- and monostable-pulse generation.

The PTM comprises three independent 16-bit counting registers. In the standard ECI configuration these registers are cascaded as shown in fig. 9 . The output of register 1 is connected to the input clock of register 2. Similarly the output of register 2 is connected to the clock of register 3. The first register, and consequently all three, are clocked from the 1 MHz system clock. This allows the software (INITECI) to establish a system time-base for counting microseconds, milliseconds, and seconds.

The user may wish to rewire the timer and add his own assembly-language programs to utilize the timer in another mode. For example, a pulse generator may be required for a DLTS system [6]. In this case, one of the PTM's output lines could be brought out to one of the unused terminals on the ECI case. Another example might be an event counter for a shaft-encoder. In this case the external input would be connected to one of the timer's clock inputs.

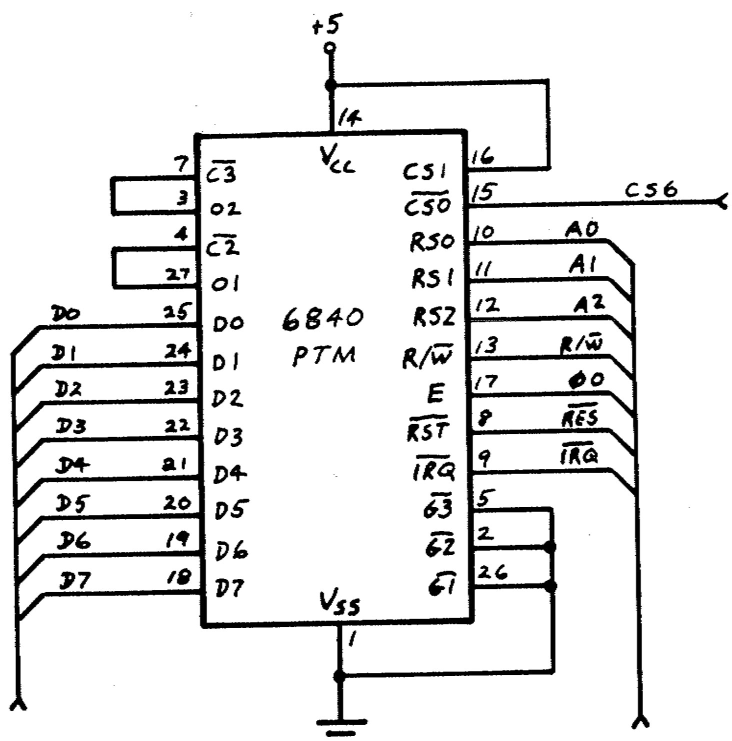


Figure 9: PTM wiring connections.

3.6 POWER SUPPLY

Power for ECI's analog devices is derived from a local ±15V power supply. A Hammond 166 J28 tranformer converts the line voltage to 28 V at 1 amp which is then rectified and regulated by a complementary pair of 15 V regulator ICs (LM78M15CP,LM79M15CP). Power for the 5V digital circuits is derived from the Apple bus. The power supply wiring is shown in fig. 10 .

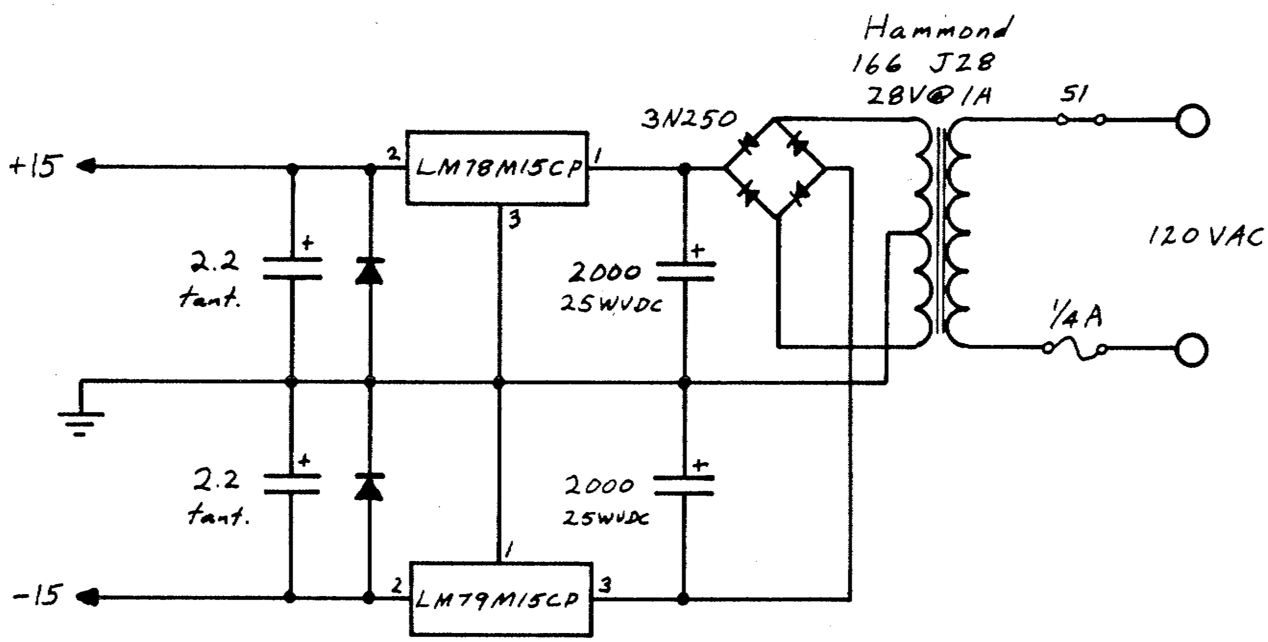


Figure 10: ±15V power supply wiring details.

REFERENCES

1. "Apple Pascal Language Reference Manual," Apple Computer Inc., Cupertino, Calif., 1980.

2. "Apple Pascal Operating System Reference Manual," Apple Computer Inc., Cupertino, Calif., 1980.

3. "Super Serial Card, Installation and Operating Manual," Apple Computer Inc., Cupertino, Calif., 1981.

4. "HP 7470A Interfacing and Programming Manual," Hewlett-Packard, San Diego, Calif., 1982.

5. C. B. Grant and J. Butah, "Introduction to the UCSD p-System," Berkeley:Sybex, 1982.

6. D.V. Lang, "Deep-level transient spectroscopy: A new method to characterize traps in semiconductors," J. Appl. Phys., 45, pp. 3023-3032, (1974).

Appendix

ADAP SYSTEM LIBRARY UNITS

```
 1    1:D   1 (*$L #8:*)
 1    1:D   1 (*$S+*)
16    1:D   1 UNIT ECI;INTRINSIC CODE 16 DATA 31;
16    1:D   1
16    1:D   1    (***************************** ECI ******************************)
16    1:D   1    (*                                                                *)
16    1:D   1    (*                       WRITTEN BY G.C. MCGONIGAL                *)
16    1:D   1    (*                                                                *)
16    1:D   1    (*               MATERIALS & DEVICES RESEARCH LABORATORY          *)
16    1:D   1    (*                DEPTARTMENT OF ELECTRICAL ENGINEERING           *)
16    1:D   1    (*                        UNIVERSITY OF MANITOBA                  *)
16    1:D   1    (*                                                                *)
16    1:D   1    (*                           VER. I.0/1982                        *)
16    1:D   1    (*                                                                *)
16    1:D   1    (****************************************************************)
16    1:D   1    (*                                                                *)
16    1:D   1    (*   ECI IS A UNIT OF PASCAL PROCEDURES DESIGNED TO SIMPLIFY      *)
16    1:D   1    (* THE IMPLEMENTATION OF CONTROL PROGRAMS FOR THE "EXPERIMENT     *)
16    1:D   1    (* CONTROL INTERFACE" [ECI].                                      *)
16    1:D   1    (*                                                                *)
16    1:D   1    (*   RECOGNIZABLE FUNCTION NAMES AND "HIGH-LEVEL" PARAMETERS      *)
16    1:D   1    (* ARE EMPLOYED TO "BUFFER" THE PROGRAMMER FROM THE MACHINE-LEVEL *)
16    1:D   1    (* PARTICULARS.                                                   *)
16    1:D   1    (*                                                                *)
16    1:D   1    (* SUMMARY OF AVAILABLE PROCEDURES:                               *)
16    1:D   1    (* NAME:       FUNCTION:                                          *)
16    1:D   1    (*                                                                *)
16    1:D   1    (* I2V         SETS SENSITIVITY OF THE I TO V CONVERTER           *)
16    1:D   1    (* TIME        READS A COUNTER                                    *)
16    1:D   1    (* SINPUT      POLLS THE S1 OR S2 INPUT                           *)
16    1:D   1    (* DOUT        CONTROLS THE D OUTPUTS                             *)
16    1:D   1    (* ASSAMPLE    ASSEMBLED SAMPLING ROUTINE                         *)
16    1:D   1    (* ZEROT       RESETS THE COUNTERS                                *)
16    1:D   1    (* SAMPLE      RETURNS THE SAMPLED VOLTAGE                        *)
16    1:D   1    (* ANALOGOUT   WRITES VOLTAGES TO THE DACS                        *)
16    1:D   1    (* INITECI     SYSTEM INITIALIZATION ROUTINE                      *)
16    1:D   1    (*                                                                *)
16    1:D   1    (****************************************************************)
16    1:D   1
16    1:D   1
16    1:D   1    INTERFACE
16    1:D   1
16    1:D   1      TYPE DEVICE=(X,Y,W,Z);
16    1:D   1
16    1:D   1      VAR TEM,COPY,RETAD,LINE:INTEGER;
16    1:D   5
16    2:D   1    PROCEDURE I2V(SCALE:INTEGER);
16    2:D   2
16    3:D   3    FUNCTION  TIME(SECTION:INTEGER):INTEGER;
16    3:D   4
16    4:D   3    FUNCTION  SINPUT(LINE:INTEGER):INTEGER;
16    4:D   4
16    5:D   1    PROCEDURE DOUT(HIORLO,LINENUM:INTEGER);
16    5:D   3
16    6:D   3    FUNCTION  ASSAMPLE(CHANNEL:INTEGER):INTEGER;
16    6:D   4
16    7:D   1    PROCEDURE ZEROT;
16    7:D   1
16    8:D   1    PROCEDURE SAMPLE(CHANNEL:INTEGER;VAR RRESULT:REAL);
16    8:D   3
16    9:D   1    PROCEDURE ANALOGOUT(VOUT:REAL;D2A:DEVICE);
16    9:D   4
16   10:D   1    PROCEDURE INITECI;
16   10:D   1
```

```
16   10:D    1
16    1:D    1 IMPLEMENTATION
16    1:D    5
16    2:D    1 PROCEDURE I2V;EXTERNAL;
16    2:D    2 (* CONTROLS THE RANGE SETTING OF THE CURRENT TO VOLTAGE CONVERTER.
16    2:D    2    THE PARM SCALE IS SET TO THE ABS OF THE MAGNITUDE OF THE DESIRED
16    2:D    2    FULL-RANGE CURRENT READING IN AMPS.
16    2:D    2    E.G. FOR FULL-SCALE (V=10 V) OF 1.0E-4 AMPS THEN SCALE=4. *)
16    2:D    2
16    2:D    2
16    3:D    3 FUNCTION TIME;EXTERNAL;
16    3:D    4 (* RETURNS THE CURRENT VALUE OF THE SPECIFIED COUNTER.
16    3:D    4    SECTION=1 COUNTS MICROSECONDS;
16    3:D    4        "     2    "     MILLISECONDS, AND
16    3:D    4        "     3    "     SECONDS. *)
16    3:D    4
16    3:D    4
16    4:D    3 FUNCTION SINPUT;EXTERNAL;
16    4:D    4 (* POLLS THE S1 OR S2 INPUT.
16    4:D    4    A '1' WILL BE RETURNED IF THE LINE IS ACTIVE (OTHERWISE '0').
16    4:D    4    S1 IS ACTIVE ON A LO TO HI TRANSITION.
16    4:D    4    S2  "    "     " " HI TO LO       "         *)
16    4:D    4
16    4:D    4
16    5:D    1 PROCEDURE DOUT;EXTERNAL;
16    5:D    3 (* CONTROLS THE DIGITAL OUTPUTS D0,D1,D2,D3.
16    5:D    3    LINENUM IS THE NUMBER OF THE LINE TO BE ACCESSED (0,1,2,OR 3).
16    5:D    3    HIORLO IS THE DESIRED STATE (1 OR 0). *)
16    5:D    3
16    5:D    3
16    6:D    3 FUNCTION ASSAMPLE;EXTERNAL;
16    6:D    4 (* ASSEMBLED PROC TO READ THE ADC.
16    6:D    4    CHANNEL SPECIFIES THE AMUX CHANNEL TO BE ACTIVATED DURING CONVERSION.
16    6:D    4    RESULT IS THE 12-BIT SIGNED INTEGER VALUE READ DIRECTLY FROM THE ADC.*)
16    6:D    4
16    6:D    4
16   11:D    1 PROCEDURE INITPIA;EXTERNAL;
16   11:D    1
16   11:D    1
16   12:D    1 PROCEDURE DIG2AN(VOUTI,CONVERTER:INTEGER);EXTERNAL;
16   12:D    3
16   12:D    3
16    7:D    1 PROCEDURE ZEROT;EXTERNAL;
16    7:D    1 (* RESETS THE TIMER'S COUNTERS *)
16    7:D    1
16    7:D    1
16    9:D    1 PROCEDURE ANALOGOUT;
16    9:D    4 (* CONTROLS THE DIGITAL TO ANALOG CONVERTERS.
16    9:D    4    VOUT=DESIRED OUTPUT VOLTAGE (REAL #(-10.240 TO +10.220)). THIS PROC
16    9:D    4         ROUNDS VOUT TO THE NEAREST VOLTAGE LIMITED BY THE CONVERTER
16    9:D    4         RESOLUTION WHICH IS 0.020 VOLTS
16    9:D    4    D2A=CONVERTER TO BE WRITTEN INTO. THIS IS A VAR OF TYPE DEVICE
16    9:D    4         WHICH IS ONE OF X,Y,W,Z.
16    9:D    4 E.G. PROCEDURE ANALOGOUT(-0.13,Y);
16    9:D    4       WOULD PRODUCE AN OUTPUT OF -0.140 VOLTS AT Y. *)
16    9:D    4
16    9:D    4 VAR VOUTI,CONVERTER:INTEGER;
16    9:D    6
16    9:0    0 BEGIN
16    9:1    0 VOUTI:=ROUND(VOUT/0.02+512.0);
16    9:1   23 IF D2A=Z THEN CONVERTER:=0;
16    9:1   31 IF D2A=W THEN CONVERTER:=8;
16    9:1   39 IF D2A=Y THEN CONVERTER:=16;
16    9:1   47 IF D2A=X THEN CONVERTER:=24;
16    9:1   55 DIG2AN(VOUTI,CONVERTER);
16    9:0   59 END;
16    9:0   72
16    9:0   72
16    8:D    1 PROCEDURE SAMPLE;
16    8:D    3 (* SAME AS ASSAMPLE EXCEPT THAT THE RESULT IS RETURNED AS A REAL-VALUED
16    8:D    3    VOLTAGE (-10.24 TO +10.235). *)
16    8:D    3
16    8:D    3 VAR RESULT:INTEGER;
16    8:D    4
16    8:0    0 BEGIN
16    8:1    0 RRESULT:=(ASSAMPLE(CHANNEL)-2047)*0.005;
16    8:0   20 END;
16    8:0   32
```

```
16    8:0    32
16   10:D     1 PROCEDURE INITECI;
16   10:D     1 (* INITECI SHOULD BE CALLED BEFORE ANY OTHER ECI PROC TO ASSURE
16   10:D     1    THE SYSTEM IS PROPERLY CONFIGURED. INITECI ALSO INITIALIZES THE
16   10:D     1    VARIOUS DEVICES TO THE FOLOWING:
16   10:D     1    X,Y,W,Z:=0.0 VOLTS
16   10:D     1    D0,D1,D2,D3=0:=LO;
16   10:D     1    S1=ACTIVE HI,S2=ACTIVE LO *)
16   10:D     1
16   10:0     0 BEGIN
16   10:0     0 (* SET-UP PIA *)
16   10:0     0 (* SET TIMER TO MICROSEC,MILLISEC,SECS *)
16   10:1     0 INITPIA;
16   10:1     2 (* ZERO X,Y,W,Z *)
16   10:1     2 ANALOGOUT(0.0,X);
16   10:1    11 ANALOGOUT(0.0,Y);
16   10:1    21 ANALOGOUT(0.0,W);
16   10:1    31 ANALOGOUT(0.0,Z);
16   10:1    41 (* SET D0..D3 LO *)
16   10:1    41 DOUT(0,0);
16   10:1    45 DOUT(0,1);
16   10:1    49 DOUT(0,2);
16   10:1    53 DOUT(0,3);
16   10:1    57 (* SET I2V TO LEAST-SENSITIVE RANGE *)
16   10:1    57 I2V(2);
16   10:0    60 END;
16   10:0    72
 1    1:0     0 END.
```

PAGE -   1 ASSAMPLE   FILE:ASSAMPLE

```
0000:                                          .FUNC ASSAMPLE,1
Current memory available:       B132
0000:                                          .PUBLIC RETAD
0000:                                  ;THIS FUNC SELECTS THE INPUT CHANNEL
0000:                                  ;THRU THE AMUX (0..7),GENERATES A START
0000:                                  ;COVERSION PULSE (SC),WAITS FOR END OF
0000:                                  ;CONVERSION, AND RETURNS THE 12-BIT
0000:                                  ;RESULT .
0000:                                  
0000: C320                             DRA     .EQU 0C320   ;PIA
0000: C321                             CRA     .EQU 0C321
0000: C322                             DRB     .EQU 0C322
0000: C323                             CRB     .EQU 0C323
0000: C328                             AMUX    .EQU 0C328   ;CHANNEL SELECT
0000: 68                                       PLA          ;SAVE RET ADR
0001: 8D 0000                                  STA RETAD
0004: 68                                       PLA
0005: 8D 0100                                  STA RETAD+1
0008: 68                                       PLA
0009: 68                                       PLA
000A: 68                                       PLA
000B: 68                                       PLA
000C: 68                                       PLA
000D: 8D 28C3                                  STA AMUX     ;SEL INPUT CHANNEL
0010: 68                                       PLA          ;DISCARD MSB
0011: A9 04                                    LDA #4
0013: 8D 23C3                                  STA CRB
0016: A9 36                                    LDA #36H     ;SC PULSE LO
0018: 8D 21C3                                  STA CRA
001B: EA                                       NOP          ;KEEP SC LO 1 CLOCK
001C: EA                                       NOP
001D: EA                                       NOP
001E: EA                                       NOP
001F: EA                                       NOP
0020: EA                                       NOP
0021: EA                                       NOP
0022: EA                                       NOP
0023: A9 3E                                    LDA #3E      ;SC PULSE HI
0025: 8D 21C3                                  STA CRA      ;CONVERSION BEGINS
0028: A9 80                                    LDA #80      ;WAIT FOR COMPLETION
002A: 2C 21C3                   $01            BIT CRA
002D: F0FB                                     BEQ $01
002F: A9 0F                                    LDA #0F      ;MASK 4 HI BITS
0031: 2D 22C3                                  AND DRB
0034: 48                                       PHA          ;STORE HI ORDER DATA
0035: AD 20C3                                  LDA DRA
0038: 48                                       PHA          ;STORE LO ORDER DATA
0039: AD 0100                                  LDA RETAD+1  ;PUSH RET ADR
003C: 48                                       PHA
003D: AD 0000                                  LDA RETAD
0040: 48                                       PHA
0041: 60                                       RTS          ;RETURN
0042:                                          .END
```

```
PAGE -   1  SINPUT    FILE:SINPUT

0000!                                   .FUNC SINPUT,1
Current memory available:     8132
0000!                                   .PRIVATE RETAD
0000! C323                      CRB     .EQU 0C323
0000! C322                      DRB     .EQU 0C322
0000! 68                                PLA             ;SAVE RETURN ADR
0001! 8D 0000                           STA RETAD
0004! 68                                PLA
0005! 8D 0100                           STA RETAD+1
0008! 68                                PLA
0009! 68                                PLA
000A! 68                                PLA
000B! 68                                PLA
000C! 68                                PLA             ;GET 1 OR 2 INPUT..
000D! C9 01                             CMP #1          ;..AND LEAVE 00 ON TOP
000F! F0**                              BEQ S1          ;LOOK AT S1
0011! C9 02                             CMP #2
0013! F0**                              BEQ S2          ;LOOK AT S2
0015! D0**                              BNE NONE        ;LOOK AT NONE
000F* 06
0017! A9 80                     S1      LDA #80         ;MASK CRB-7
0019! 4C ****                           JMP CNT
0013* 07
001C! A9 40                     S2      LDA #40         ;MASK CRB-6
001A* 1E00
001E! 2C 23C3                   CNT     BIT CRB
0021! F0**                              BEQ LO
0023! A9 01                             LDA #1          ;SINPUT=1
0025! 4C ****                           JMP EXIT
0021* 05
0028! A9 00                     LO      LDA #0          ;SINPUT=0
002A! 4C ****                           JMP EXIT
0015* 16
002D! A9 02                     NONE    LDA #2          ;SINPUT=2
002B* 2F00
0026* 2F00
002F! 48                        EXIT    PHA             ;PUT SINPUT ON STACK
0030! AD 0100                           LDA RETAD+1     ;RETURN
0033! 48                                PHA
0034! AD 0000                           LDA RETAD
0037! 48                                PHA
0038! AD 22C3                           LDA DRB         ;CLEAR INPUT FLAGS
003B! 60                                RTS
003C!                                   .END
```

```
PAGE -   1  I2V        FILE:I2V
0000!                                   .PROC I2V,1
Current memory available:      8132
0000!                              ;THIS PROC SELECTS THE ACTIVE CHANNEL
0000!                              ;ON THE CURRENT/VOLTAGE CONVERTER. PIA
0000!                              ;OUTPUTS PB4-PB6 ADR THE AMUX.
0000!                              ;THE INPUT PARM "SCALE" IS AN INTEGER
0000!                              ;VALUE THAT IS CALC BY TAKING THE ABS
0000!                              ;VALUE OF THE MAGNITUDE OF THE DESIRED
0000!                              ;CURRENT AT FULL-SCALE OUTPUT (10 VOLTS).
0000!                              ;E.G. SCALE=4 WOULD LEAD TO A FULL
0000!                              ;SCALE OUTPUT @ I=1.0E-4 AMPS.
0000!                                   .PRIVATE RETAD,SCALE
0000! C322                     DRB      .EQU 0C322
0000! 68                                PLA           ;SAVE RETURN ADR
0001! 8D 0000                           STA RETAD
0004! 68                                PLA
0005! 8D 0100                           STA RETAD+1
0008! 68                                PLA
0009! 8D 0000                           STA SCALE     ;SAVE SCALE FACTOR
000C! A9 0A                             LDA #0A
000E! 18                                CLC
000F! ED 0000                           SBC SCALE     ;CALC AMUX CHANNEL
0012! 18                                CLC
0013! 2A                                ROL A         ;SHIFT BEFORE
0014! 2A                                ROL A         ;WRITING TO PIA
0015! 2A                                ROL A
0016! 2A                                ROL A
0017! 8D 0000                           STA SCALE
001A! AD 22C3                           LDA DRB       ;MAINTAIN DRB-7
001D! 29 80                             AND #80
001F! 0D 0000                           ORA SCALE
0022! 8D 22C3                           STA DRB       ;WRITE TO PIA
0025! 68                                PLA           ;CLEAN-UP STACK
0026! AD 0100                           LDA RETAD+1   ;RETURN
0029! 48                                PHA
002A! AD 0000                           LDA RETAD
002D! 48                                PHA
002E! 60                                RTS
002F!                                   .END


PAGE -   1  INITPIA   FILE:INITPIA
0000!                                   .PROC INITPIA
Current memory available:      8132
0000!                              ;CONFIGURES THE PIA AND PTM
0000!                              ;FOR STANDARD ECI OPERATION.
0000! C320                     DRA      .EQU 0C320   ;PIA
0000! C321                     CRA      .EQU 0C321    ;
0000! C322                     DRB      .EQU 0C322    ;
0000! C323                     CRB      .EQU 0C323    ;
0000! C330                     WCR13    .EQU 0C330   ;PTM
0000! C331                     WCR2     .EQU 0C331    ;
0000! C332                     RT1C     .EQU 0C332    ;
0000! C334                     RT2C     .EQU 0C334    ;
0000! C336                     RT3C     .EQU 0C336    ;
0000! A9 00                             LDA #0       ;RESET CONTROL REGS.
0002! 8D 21C3                           STA CRA
0005! 8D 23C3                           STA CRB
0008! 8D 20C3                           STA DRA      ;SET A-SIDE TO INPUTS
000B! A9 3E                             LDA #3E      ;CA1=IN,CA2=OUT&HI
000D! 8D 21C3                           STA CRA
0010! A9 F0                             LDA #0F0     ;PB0..3=IN
0012! 8D 22C3                           STA DRB      ;PB4..7=OUT
0015! A9 06                             LDA #6       ;CB1=ACTIVE(HI)
0017! 8D 23C3                           STA CRB      ;CB2=ACTIVE(LO)
001A! A9 03                             LDA #3       ;START T1 COUNT @ 999
001C! 8D 32C3                           STA RT1C      ;
001F! A9 F9                             LDA #0F9      ;
0021! 8D 33C3                           STA RT1C+1    ;
0024! A9 03                             LDA #3       ;START T2 COUNT @ 999
0026! 8D 34C3                           STA RT2C      ;
0029! A9 F9                             LDA #0F9      ;
002B! 8D 35C3                           STA RT2C+1    ;
002E! A9 7F                             LDA #07F     ;START T3 COUNT @ 32K
0030! 8D 36C3                           STA RT3C      ;
0033! A9 FF                             LDA #0FF      ;
0035! 8D 37C3                           STA RT3C+1    ;
0038! A9 00                             LDA #0       ;INIT PTM CR
003A! 8D 31C3                           STA WCR2     ;POINT TO WCR13
003D! A9 10                             LDA #10H     ;PUT T3 IN 16-BIT MODE
003F! 8D 30C3                           STA WCR13     ;& EXTERNAL CLOCK.
0042! A9 95                             LDA #95H     ;PUT T2 IN 8-BIT MODE
0044! 8D 31C3                           STA WCR2      ;& EXTERNAL CLOCK.
0047! A9 96                             LDA #96H     ;PUT T3 IN 8-BIT MODE
0049! 8D 30C3                           STA WCR13     ;INTERNAL 1.023 MHZ CLOCK.
004C! 60                                RTS
004D!                                   .END
```

```
0000!                                    .FUNC TIME,1
Current memory available:       8132
0000!                           ;THE TIMER WILL BE CONFUGURED BY INITECI
0000!                           ;TO COUNT MICROSECS,MILLISECS,AND SECS.
0000!                           ;THE INPUT PARM SELECTS WHICH WILL BE READ.
0000!                           ;PARM  = 1 --> TIMER1 --> MICROSECS,
0000!                           ;        2 --> TIMER2 --> MILLISECS,
0000!                           ;        3 --> TIMER3 --> SECONDS.
0000!                           ;TIMER3 MAX=32767.
0000!                                    .PRIVATE RETAD,MSB,LSB
0000! C330                     WCR13    .EQU 0C330
0000! C331                     WCR2     .EQU 0C331
0000! C332                     RT1C     .EQU 0C332
0000! C334                     RT2C     .EQU 0C334
0000! C336                     RT3C     .EQU 0C336
0000! 68                                PLA             ;SAVE RETURN ADR
0001! 8D 0000                           STA RETAD
0004! 68                                PLA
0005! 8D 0100                           STA RETAD+1
0008! 68                                PLA             ;FIX STACK
0009! 68                                PLA              ;
000A! 68                                PLA              ;
000B! 68                                PLA              ;
000C! 68                                PLA             ;GET TIMER # TO BE READ
000D! C9 01                             CMP #1          ;LOOK AT T1
000F! F0**                              BEQ T1
0011! C9 02                             CMP #2          ;LOOK AT T2
0013! F0**                              BEQ T2          ;ELSE LOOK AT T3
0015! AD 36C3                           LDA RT3C        ;READ TIMER3
0018! 8D 0000                           STA MSB
001B! AD 37C3                           LDA RT3C+1
001E! 8D 0000                           STA LSB
0021! A9 00                             LDA #0          ;CONVERT T3 RESULT TO THE
0023! 18                                CLC             ;  APPROPRIATE DATA
0024! ED 0000                           SBC LSB         ;  FORMAT
0027! 8D 0000                           STA LSB          ;
002A! A9 80                             LDA #80          ;
002C! ED 0000                           SBC MSB          ;
002F! 8D 0000                           STA MSB          ;
0032! 4C ****                           JMP OUT
0013* 20
0035! AD 34C3                  T2       LDA RT2C        ;READ TIMER2
0038! 8D 0000                           STA MSB
003B! AD 35C3                           LDA RT2C+1
003E! 4C ****                           JMP OUT2
000F* 30
0041! AD 32C3                  T1       LDA RT1C        ;READ TIMER1
0044! 8D 0000                           STA MSB
0047! AD 33C3                           LDA RT1C+1
004A!
003F* 4A00
004A! 8D 0000                  OUT2     STA LSB         ;CONVERT T2 OR T1 RESULT
004D! AD 0000                           LDA MSB         ;  TO THE APPROPRIATE
0050! F0**                              BEQ C0          ;  DATA FORMAT.
0052! C9 02                             CMP #2
0054! 30**                              BMI C1
0056! F0**                              BEQ C2
0058! A9 12                             LDA #18.
005A! 4C ****                           JMP CNT
0056* 05
005D! A9 0C                    C2       LDA #12.
005F! 4C ****                           JMP CNT
0054* 0C
0062! A9 06                    C1       LDA #6.
0064! 4C ****                           JMP CNT
0050* 15
0067! A9 00                    C0       LDA #0
0065* 6900
0060* 6900
005B* 6900
0069! 18                       CNT      CLC
006A! 69 E7                             ADC #0E7
006C! 38                                SEC
006D! ED 0000                           SBC LSB
0070! 8D 0000                           STA LSB
0073! A9 03                             LDA #3
0075! ED 0000                           SBC MSB
0078! 8D 0000                           STA MSB
0033* 7B00
007B! 68                       OUT      PLA             ;STACK CLEAN-UP
007C! AD 0000                           LDA MSB         ;RETURN RESULT
007F! 48                                PHA              ;
0080! AD 0000                           LDA LSB          ;
0083! 48                                PHA              ;
0084! AD 0100                           LDA RETAD+1     ;RETURN
0087! 48                                PHA
0088! AD 0000                           LDA RETAD
008B! 48                                PHA
008C! 60                                RTS
008D!                                   .END
```

```
PAGE -    1  ZEROT      FILE:ZEROT

0000:                                   .PROC ZEROT
Current memory available:     8132
0000:                          ;RESETS THE TIMER COUNTERS.
0000:                          ;USES THE INTERNAL-RESET BIT (CR1-0).
0000:                                   .PRIVATE RETAD
0000: C330                     WCR13   .EQU 0C330
0000: C331                     WCR2    .EQU 0C331
0000: C332                     RT1C    .EQU 0C332
0000: C334                     RT2C    .EQU 0C334
0000: C336                     RT3C    .EQU 0C336
0000: 68                               PLA          ;SAVE RETURN ADR
0001: 8D 0000                          STA RETAD
0004: 68                               PLA
0005: 8D 0100                          STA RETAD+1
0008: A9 97                            LDA #97H     ;INTERNAL RESET
000A: 8D 30C3                          STA WCR13
000D: A9 96                            LDA #96H
000F: 8D 30C3                          STA WCR13
0012: AD 0100                          LDA RETAD+1  ;RETURN
0015: 48                               PHA
0016: AD 0000                          LDA RETAD
0019: 48                               PHA
001A: 60                               RTS
001B:                                  .END
```

```
PAGE -    1  DIG2AN     FILE:DIG2AN

0000:                                   .PROC DIG2AN ,2
Current memory available:     8132
0000:                          ;CONTROLS THE D/A CONVERTERS.
0000:                          ;CALLED BY ANALOGOUT IN ECI.
0000:                                   .PRIVATE RETAD,V,YTEMP,XTEMP
0000: C300                     SLOT3   .EQU 0C300
0000: 8C 0000                           STY YTEMP    ;SAVE Y STATUS
0003: 8E 0000                           STX XTEMP    ;SAVE X STATUS
0006: 68                                PLA          ;SAVE RETURN ADR
0007: 8D 0000                           STA RETAD
000A: 68                                PLA
000B: 8D 0100                           STA RETAD+1
000E: 68                                PLA
000F: AA                                TAX          ;GET D2A OFFSET
0010: 68                                PLA
0011: 68                                PLA
0012: A8                                TAY          ;SAVE LO ORDER VOLTS
0013: 8D 0000                           STA V
0016: 68                                PLA
0017: 8D 0100                           STA V+1
001A: 6E 0100                           ROR V+1      ;PUT THE 8 M.S.BITS IN V
001D: 6E 0000                           ROR V
0020: 6E 0100                           ROR V+1
0023: 6E 0000                           ROR V
0026: AD 0000                           LDA V
0029: 9D 01C3                           STA SLOT3+1,X;WRITE MSBS
002C: 98                                TYA
002D: 9D 00C3                           STA SLOT3,X  ;WRITE L.S.BITS
0030: AD 0100                           LDA RETAD+1  ;RETURN
0033: 48                                PHA
0034: AD 0000                           LDA RETAD
0037: 48                                PHA
0038: AC 0000                           LDY YTEMP    ;RESTORE Y STATUS
003B: AE 0000                           LDX XTEMP    ;RESTORE X STATUS
003E: 60                                RTS
003F:                                   .END
```

```
  1      1:D   1 (*$L #8:*)
  1      1:D   1 (*$S+*)
 17      1:D   1 UNIT PLOTAIDS;INTRINSIC CODE 17;
 17      1:D   1
 17      1:D   1 (*********************** PLOTAIDS *************************)
 17      1:D   1 (*                                                        *)
 17      1:D   1 (*                 WRITTEN BY G.C. MCGONIGAL              *)
 17      1:D   1 (*                                                        *)
 17      1:D   1 (*              MATERIALS & DEVICES RESEARCH LAB          *)
 17      1:D   1 (*              DEPT. OF  ELECTRICAL ENGINEERING          *)
 17      1:D   1 (*                    UNIVERSITY OF MANITOBA              *)
 17      1:D   1 (*                                                        *)
 17      1:D   1 (*                         VER I.0/1982                   *)
 17      1:D   1 (*                                                        *)
 17      1:D   1 (**********************************************************)
 17      1:D   1 (*                                                        *)
 17      1:D   1 (*   PLOTAIDS IS A PACKAGE OF PASCAL SUBROUTINES DESIGNED *)
 17      1:D   1 (* TO REDUCE THE COMPLEXITY OF PROGRAMS THAT CONSTRUCT    *)
 17      1:D   1 (* GRAPHICAL DISPLAYS ON THE APPLE MONITOR.               *)
 17      1:D   1 (*                                                        *)
 17      1:D   1 (*   PLOTAIDS IS ACCESSED FROM SYSTEM.LIBRARY BY THE DECL.*)
 17      1:D   1 (*            USES TRANSCEND,TURTLEGRAPHICS,PLOTAIDS;     *)
 17      1:D   1 (*                                                        *)
 17      1:D   1 (* SUMMARY OF PROC. PROVIDED:                             *)
 17      1:D   1 (* NAME:      TYPE/FUNCTION:                              *)
 17      1:D   1 (*                                                        *)
 17      1:D   1 (* PLOTPOINT PROC WRITES A SYMBOL AT THE SPECIFIED PT.    *)
 17      1:D   1 (* DRAWGRID  PROC GENERATES AND LABELS THE GRIDLINES      *)
 17      1:D   1 (*                                                        *)
 17      1:D   1 (**********************************************************)
 17      1:D   1
 17      1:D   1   INTERFACE
 17      1:D   1
 29      1:D   1
 29      2:D   3   FUNCTION SIN(X:REAL):REAL;
 29      3:D   3   FUNCTION COS(X:REAL):REAL;
 29      4:D   3   FUNCTION EXP(X:REAL):REAL;
 29      5:D   3   FUNCTION ATAN(X:REAL):REAL;
 29      6:D   3   FUNCTION LN(X:REAL):REAL;
 29      7:D   3   FUNCTION LOG(X:REAL):REAL;
 29      8:D   3   FUNCTION SQRT(X:REAL):REAL;
 29      8:D   5
 20      1:D   5
 20      1:D   1     TYPE
 20      1:D   1       SCREENCOLOR=(none,white,black,reverse,radar,
 20      1:D   1                    black1,green,violet,white1,black2,orange,blue,white2);
 20      1:D   1
 20      2:D   1     PROCEDURE INITTURTLE;
 20      3:D   1     PROCEDURE TURN(ANGLE: INTEGER);
 20      4:D   1     PROCEDURE TURNTO(ANGLE: INTEGER);
 20      5:D   1     PROCEDURE MOVE(DIST: INTEGER);
 20      6:D   1     PROCEDURE MOVETO(X,Y: INTEGER);
 20      7:D   1     PROCEDURE PENCOLOR(PENMODE: SCREENCOLOR);
 20      8:D   1     PROCEDURE TEXTMODE;
 20      9:D   1     PROCEDURE GRAFMODE;
 20     10:D   1     PROCEDURE FILLSCREEN(FILLCOLOR: SCREENCOLOR);
 20     11:D   1     PROCEDURE VIEWPORT(LEFT,RIGHT,BOTTOM,TOP: INTEGER);
 20     12:D   3     FUNCTION  TURTLEX: INTEGER;
 20     13:D   3     FUNCTION  TURTLEY: INTEGER;
 20     14:D   3     FUNCTION  TURTLEANG: INTEGER;
 20     15:D   3     FUNCTION  SCREENBIT(X,Y: INTEGER): BOOLEAN;
 20     16:D   1     PROCEDURE DRAWBLOCK(VAR SOURCE; ROWSIZE,XSKIP,YSKIP,WIDTH,HEIGHT,
 20     16:D   2                         XSCREEN,YSCREEN,MODE: INTEGER);
 20     17:D   1     PROCEDURE WCHAR(CH: CHAR);
 20     18:D   1     PROCEDURE WSTRING(S: STRING);
 20     19:D   1     PROCEDURE CHARTYPE(MODE: INTEGER);
 20     19:D   2
 17      1:D   2     USES TRANSCEND,TURTLEGRAPHICS;
 17      1:D   1
 17      1:D   1     TYPE POINTMARKER= ( POINT,CROSS,SQUARE );
 17      1:D   1
 17      1:D   1          VECTOR= ARRAY[0..7] OF REAL;
```

```
17    1:D     1
17    2:D     1       PROCEDURE PLOTPOINT(XNUM,YNUM,
17    2:D     1                            XBOTTOMLINE,YBOTTOMLINE,XDIVSIZE,YDIVSIZE:REAL;
17    2:D    13                            CONNECTPOINTS:BOOLEAN;POINTSYMBOL:POINTMARKER);
17    2:D    15
17    3:D     1       PROCEDURE DRAWGRID(XMAX,XMIN,YMAX,YMIN:REAL;
17    3:D     9                           VAR XBOTTOMLINE,YBOTTOMLINE,
17    3:D     9                           XDIVSIZE,YDIVSIZE:REAL;DOTTEDLINES:BOOLEAN);
17    3:D    14
17    3:D    14
17    1:D    14     IMPLEMENTATION
17    1:D     1
17    4:D     3 FUNCTION EXPO(X:REAL):INTEGER;
17    4:D     5 (* RETURNS THE EXPONENTIAL PART OF X *)
17    4:0     0 BEGIN
17    4:1     0   IF X=0.0 THEN
17    4:2    14     EXPO:=0
17    4:1    14   ELSE
17    4:2    19     EXPO:=TRUNC(LOG(ABS(X))+100)-100;
17    4:0    38 END;
17    4:0    50
17    4:0    50
17    5:D     3 FUNCTION MANTISSA(X:REAL):REAL;
17    5:D     5 (* RETURNS THE MANTISSA OF X *)
17    5:0     0 BEGIN
17    5:1     0   MANTISSA:=X/EXP(EXPO(X)*2.30259);
17    5:0    30 END;
17    5:0    42
17    5:0    42
17    6:D     3 FUNCTION XNOR(A,B:BOOLEAN):BOOLEAN;
17    6:D     5 (* EXECUTES THE EXCLUSIVE NOR FCN ON (A,B) *)
17    6:0     0 BEGIN
17    6:1     0   XNOR:=(A AND B) OR ((NOT A) AND (NOT B));
17    6:0    11 END;
17    6:0    24
17    6:0    24
17    2:D     1 PROCEDURE PLOTPOINT;
17    2:D    15 (* WRITES A SYMBOL AT THE REAL # CO-ORD. (XNUM,YNUM)
17    2:D    15    XBOTTOMLINE,YBOTTOMLINE,XDIVSIZE,YDIVSIZE ARE REAL PARMS GENERATED
17    2:D    15    BY DRAWGRID. THE POINT WILL BE CONNECTED BY A STRAIGHT LINE TO THE
17    2:D    15    PREVIOUS POINT IF CONNECTPOINTS=TRUE. POINTSYMBOL IS OF TYPE
17    2:D    15    POINTMARKER (EITHER CROSS,SQUARE,OR POINT). *)
17    2:D    15
17    2:D    15 VAR C:INTEGER;
17    2:D    16
17    2:0     0 BEGIN
17    2:1     0 PENCOLOR(BLACK);
17    2:1     4 IF NOT CONNECTPOINTS THEN
17    2:2     8   PENCOLOR(NONE);
17    2:2    12 (* CALC POSITION ON SCREEN *)
17    2:1    12 MOVETO(ROUND(21+(XNUM-XBOTTOMLINE)/XDIVSIZE*32.0),
17    2:1    39        ROUND(9+(YNUM-YBOTTOMLINE)/YDIVSIZE*23.0));
17    2:1    68 IF POINTSYMBOL = CROSS THEN
17    2:2    73   BEGIN
17    2:3    73   TURNTO(0);
17    2:3    77   MOVE(1);
17    2:3    81   PENCOLOR(BLACK);
17    2:3    85   TURN(135);
17    2:3    91   MOVE(1);
17    2:3    95   TURN(90);
17    2:3    99   MOVE(1);
17    2:3   103   TURN(90);
17    2:3   107   MOVE(1);
17    2:3   111   TURN(135);
17    2:3   117   MOVE(1);
17    2:3   121   PENCOLOR(WHITE);
17    2:3   125   MOVE(0);
17    2:2   129   END;
17    2:1   129 IF POINTSYMBOL = SQUARE THEN
17    2:2   134   BEGIN
17    2:3   134   TURNTO(45);
17    2:3   138   MOVE(1);
17    2:3   142   TURNTO(180);
17    2:3   148   PENCOLOR(BLACK);
```

```
17    2:3   152    FOR C:= 1 TO 4 DO
17    2:4   163      BEGIN
17    2:5   163      MOVE(2);
17    2:5   167      TURN(90);
17    2:4   171      END;
17    2:3   178    TURNTO(225);
17    2:3   184    MOVE(1);
17    2:3   188    PENCOLOR(WHITE);
17    2:3   192    MOVE(0);
17    2:2   196    END;
17    2:1   196 IF POINTSYMBOL = POINT THEN
17    2:2   201    BEGIN
17    2:3   201    PENCOLOR(BLACK);
17    2:3   205    MOVE(0);
17    2:2   209    END;
17    2:1   209 PENCOLOR(NONE);
17    2:1   213 (* MOVE BACK TO PT. *)
17    2:1   213 MOVETO(ROUND(21+(XNUM-XBOTTOMLINE)/XDIVSIZE*32.0),
17    2:1   239          ROUND(9+(YNUM-YBOTTOMLINE)/YDIVSIZE*23.0));
17    2:1   268 PENCOLOR(BLACK);
17    2:0   272 END;
17    2:0   286
17    2:0   286
17    7:D     3 FUNCTION LABELODDS(VAR LINEVALUE:VECTOR):BOOLEAN;
17    7:D     4 (* USED BY DRAWGRID TO DETERMINE IF THE 4 ODD OR EVEN NUMBERED
17    7:D     4     GRIDLINES SHOULD BE LABELED *)
17    7:D     4
17    7:D     4 VAR I:INTEGER;
17    7:D     5
17    7:0     0 BEGIN
17    7:1     0 LABELODDS:=FALSE;
17    7:1     3
17    7:1     3 (* CHECK FOR WHOLE NUMBERS *)
17    7:1     3 IF ABS(ROUND(MANTISSA(LINEVALUE[0]))-MANTISSA(LINEVALUE[0]))
17    7:1    34     > 0.01 THEN
17    7:2    44   LABELODDS:=TRUE;
17    7:2    47
17    7:2    47 (* CHECK FOR 'ZERO' AXIS *)
17    7:1    47 FOR I:=0 TO 7 DO
17    7:2    58   BEGIN
17    7:3    58   IF ABS(LINEVALUE[I]/(LINEVALUE[7]-LINEVALUE[0])) < 0.01 THEN
17    7:4    98     LABELODDS:=ODD(I);
17    7:2   101   END;
17    7:0   108 END;
17    7:0   122
17    7:0   122
17    8:D     1 PROCEDURE LINEARAXIS(VAR LINEVALUE:VECTOR;MAX,MIN:REAL);
17    8:D     6
17    8:D     6 VAR I,RGPTR: INTEGER;
17    8:D     8      DIVSIZE,RANGE,RGMANTISSA: REAL;
17    8:D    14      FULLRANGE:ARRAY[0..10] OF REAL;
17    8:D    36
17    8:0     0 BEGIN
17    8:0     0 (* SET-UP RANGE POSSIBILITIES *)
17    8:1     0 FULLRANGE[0]:=1.2;
17    8:1    16 FULLRANGE[1]:=1.6;
17    8:1    32 FULLRANGE[2]:=2.0;
17    8:1    48 FULLRANGE[3]:=2.4;
17    8:1    64 FULLRANGE[4]:=3.2;
17    8:1    80 FULLRANGE[5]:=4.0;
17    8:1    96 FULLRANGE[6]:=4.8;
17    8:1   112 FULLRANGE[7]:=6.4;
17    8:1   128 FULLRANGE[8]:=8.0;
17    8:1   144 FULLRANGE[9]:=12.0;
17    8:1   160 FULLRANGE[10]:=16.0;
17    8:1   176
17    8:1   176 RANGE:=MAX-MIN;
17    8:1   189
17    8:1   189 (* CHECK FOR A CONDITION OF ZERO EXTENT *)
17    8:1   189 IF RANGE=0.0 THEN
17    8:2   204   BEGIN
17    8:3   204   MAX:=ABS(MAX*1.1);
17    8:3   220   MIN:=-ABS(MIN*1.1);
17    8:2   237   END;
17    8:2   237
```

```
17      8:1    237 REPEAT
17      8:2    237   BEGIN
17      8:2    237   (* FIND A SUITABLE RANGE *)
17      8:3    237   RGMANTISSA:=MANTISSA(RANGE);
17      8:3    249   RGPTR:=-1;
17      8:3    253   REPEAT
17      8:4    253     RGPTR:=RGPTR+1;
17      8:3    258   UNTIL RGMANTISSA <= FULLRANGE[RGPTR];
17      8:3    276
17      8:3    276   (* MAKE EACH DIV 1/8 OF FULLRANGE *)
17      8:3    276   DIVSIZE:=FULLRANGE[RGPTR]/8.0
17      8:3    288            *EXP(EXPO(RANGE)*2.30259);
17      8:3    320
17      8:3    320   (* DETERMINE SUITABLE BOTTOMLINE *)
17      8:3    320   IF MIN >= 0.0 THEN
17      8:4    334     BEGIN
17      8:5    334     LINEVALUE[0]:=0.0;
17      8:5    350     WHILE MIN > LINEVALUE[0]+DIVSIZE DO
17      8:6    372       LINEVALUE[0]:=LINEVALUE[0]+DIVSIZE;
17      8:4    397     END
17      8:3    397   ELSE
17      8:4    399     BEGIN
17      8:5    399     LINEVALUE[0]:=0.0;
17      8:5    414     WHILE LINEVALUE[0]*1.001 > MIN DO
17      8:6    439       LINEVALUE[0]:=LINEVALUE[0]-DIVSIZE;
17      8:4    464     END;
17      8:4    464
17      8:4    464   (* DERIVE 8 GRIDLINES *)
17      8:3    464   FOR I:=1 TO 7 DO
17      8:4    476     BEGIN
17      8:5    476     LINEVALUE[I]:=LINEVALUE[I-1]+DIVSIZE;
17      8:4    501     END;
17      8:2    508   END
17      8:2    508 (* CHECK THAT THIS RESULT WILL BE SUITABLE *)
17      8:1    508 UNTIL LINEVALUE[7]-LINEVALUE[0]+DIVSIZE >= RANGE;
17      8:0    540 END;
17      8:0    562
17      8:0    562
17      3:D      1 PROCEDURE DRAWGRID;
17      3:D     14 (* DRAWS & LABELS A GRID SO AS TO ACCOMODATE A SET OF POINTS OF THE
17      3:D     14    EXTENT XMAX DOWNTO XMIN ON THE X-AXIS AND YMAX DOWNTO YMIN ON
17      3:D     14    THE Y-AXIS. THESE DIMENSIONS CAN BE GENERATED BY DATASCAN OR
17      3:D     14    FIXED BY THE PROGRAMMER.
17      3:D     14    XBOTTOMLINE IS THE REAL VALUE OF THE LEFT-MOST VERT. LINE,
17      3:D     14    YBOTTOMLINE  "  "   "    "    "  "     LOWEST  HORZ. LINE,
17      3:D     14    XDIVSIZE=1/8 OF THE RANGE IN X-VALUES THAT CAN BE PLOTTED,
17      3:D     14    YDIVSIZE=1/8  "  "    "    " Y-VALUES   "   "   " PLOTTED. *)
17      3:D     14
17      3:D     14 VAR RANGE: REAL;
17      3:D     16     ODDLINES:BOOLEAN;
17      3:D     17     EXPLABEL,I,J,LINELABEL:INTEGER;
17      3:D     21     LABELSTR:STRING[30];
17      3:D     37     LINEVALUE:ARRAY[0..7] OF REAL;
17      3:D     53
17      3:0      0 BEGIN
17      3:1      0 INITTURTLE;
17      3:1      3 (* DEFINE PLOT AREA *)
17      3:1      3 VIEWPORT(21,276,9,192);
17      3:1     14 FILLSCREEN(WHITE);
17      3:1     18 VIEWPORT(0,282,0,193);
17      3:1     29
17      3:1     29 (***** DO Y-LINES *****)
17      3:1     29 LINEARAXIS(LINEVALUE,YMAX,YMIN);
17      3:1     41
17      3:1     41 (* PRINT Y-AXIS EXPONENT *)
17      3:1     41 MOVETO(0,183);
17      3:1     48 WSTRING('E');
17      3:1     52 EXPLABEL:=EXPO(LINEVALUE[1]-LINEVALUE[0]);
17      3:1     79 STR(EXPLABEL,LABELSTR);
17      3:1     92 WSTRING(LABELSTR);
17      3:1     97
17      3:1     97 (* CHECK LINES TO BE LABELED *)
17      3:1     97 ODDLINES:=LABELODDS(LINEVALUE);
17      3:1    105
17      3:1    105 (* WRITE LABEL & GRID LINE *)
17      3:1    105
```

```
17    3:1    105 FOR I:=7 DOWNTO 0 DO
17    3:2    118   BEGIN
17    3:3    118   MOVETO(0,I*23+9);
17    3:3    128   IF XNOR(ODD(I),ODDLINES) THEN
17    3:4    137     BEGIN
17    3:5    137     LINELABEL:=ROUND(LINEVALUE[I]
17    3:5    146       /EXP(EXPLABEL*2.30259));
17    3:5    168     STR(LINELABEL,LABELSTR);
17    3:5    181     WSTRING(LABELSTR);
17    3:5    186     WRITELN(LINEVALUE[I],'   ',LINELABEL);
17    3:5    240     (*CHECK FOR ZERO AXIS*)
17    3:5    240     IF ABS(LINEVALUE[I]/(LINEVALUE[7]-LINEVALUE[0])) < 0.01 THEN
17    3:6    284       BEGIN
17    3:7    284       TURNTO(0);
17    3:7    288       FOR J:=0 TO 9 DO
17    3:8    301         BEGIN
17    3:9    301         MOVETO(21+J*32,9+I*23);
17    3:9    316         TURN(90);
17    3:9    320         PENCOLOR(BLACK);
17    3:9    324         MOVE(3);
17    3:9    328         TURN(180);
17    3:9    334         MOVE(6);
17    3:9    338         MOVETO(21+J*32,9+I*23);
17    3:9    353         PENCOLOR(WHITE);
17    3:9    357         TURNTO(0);
17    3:9    361         MOVE(1);
17    3:9    365         PENCOLOR(BLACK);
17    3:8    369         END;
17    3:7    377       TURNTO(180);
17    3:7    383       MOVE(40);
17    3:6    387       END
17    3:5    387     ELSE
17    3:6    389       BEGIN
17    3:7    389       TURNTO(0);
17    3:7    393       MOVETO(21,9+I*23);
17    3:7    403       PENCOLOR(BLACK);
17    3:7    407       MOVE(3);
17    3:7    411       IF DOTTEDLINES THEN
17    3:8    414         BEGIN
17    3:9    414         FOR J:=0 TO 64 DO
17    3:0    427           BEGIN
17    3:1    427           PENCOLOR(WHITE);
17    3:1    431           MOVE(1);
17    3:1    435           PENCOLOR(BLACK);
17    3:1    439           MOVE(1);
17    3:1    443           PENCOLOR(WHITE);
17    3:1    447           MOVE(1);
17    3:1    451           PENCOLOR(BLACK);
17    3:1    455           MOVE(1);
17    3:0    459           END;
17    3:8    467         END
17    3:7    467       ELSE
17    3:8    469         BEGIN
17    3:9    469         PENCOLOR(WHITE);
17    3:9    473         MOVE(260);
17    3:8    479         END;
17    3:7    479       TURN(180);
17    3:7    485       PENCOLOR(BLACK);
17    3:7    489       MOVE(11);
17    3:6    493       END;
17    3:4    493     END;
17    3:3    493   PENCOLOR(NONE);
17    3:2    497   END;
17    3:1    505 YBOTTOMLINE:=LINEVALUE[0];
17    3:1    518 YDIVSIZE:=LINEVALUE[1]-YBOTTOMLINE;
17    3:1    535
17    3:1    535 (***** DO X-LINES *****)
17    3:1    535 LINEARAXIS(LINEVALUE,XMAX,XMIN);
17    3:1    547
17    3:1    547 (* PRINT X-AXIS EXPONENT *)
17    3:1    547 MOVETO(252,0);
17    3:1    554 WSTRING('E');
17    3:1    558 EXPLABEL:=EXPO(LINEVALUE[1]-LINEVALUE[0]);
17    3:1    585 STR(EXPLABEL,LABELSTR);
17    3:1    598 WSTRING(LABELSTR);
17    3:1    603
```

```
 17   3:1   603 (* CHECK LINES TO BE LABELED *)
 17   3:1   603 ODDLINES:=LABELODDS(LINEVALUE);
 17   3:1   611
 17   3:1   611 (* WRITE LABEL & GRID LINE *)
 17   3:1   611
 17   3:1   611 FOR I:=7 DOWNTO 0 DO
 17   3:2   624   BEGIN
 17   3:3   624   MOVETO(8+I*32,0);
 17   3:3   634   IF XNOR(ODD(I),ODDLINES) THEN
 17   3:4   643     BEGIN
 17   3:5   643     LINELABEL:=ROUND(LINEVALUE[I]
 17   3:5   652       /EXP(EXPLABEL*2.30259));
 17   3:5   674     STR(LINELABEL,LABELSTR);
 17   3:5   687     WSTRING(LABELSTR);
 17   3:5   692     WRITELN(LINEVALUE[I],'  ',LINELABEL);
 17   3:5   746     (*CHECK FOR AXIS*)
 17   3:5   746     IF ABS(LINEVALUE[I]/(LINEVALUE[7]-LINEVALUE[0])) < 0.01 THEN
 17   3:6   790       BEGIN
 17   3:7   790       TURNTO(90);
 17   3:7   794       FOR J:=0 TO 9 DO
 17   3:8   807         BEGIN
 17   3:9   807         MOVETO(21+I*32,9+J*23);
 17   3:9   822         TURN(90);
 17   3:9   826         PENCOLOR(BLACK);
 17   3:9   830         MOVE(3);
 17   3:9   834         TURN(180);
 17   3:9   840         MOVE(6);
 17   3:9   844         MOVETO(21+I*32,9+J*23);
 17   3:9   859         PENCOLOR(WHITE);
 17   3:9   863         TURNTO(90);
 17   3:9   867         MOVE(1);
 17   3:9   871         PENCOLOR(BLACK);
 17   3:8   875         END;
 17   3:7   883       TURNTO(270);
 17   3:7   889       MOVE(40);
 17   3:6   893       END
 17   3:5   893     ELSE
 17   3:6   895       BEGIN
 17   3:7   895       TURNTO(90);
 17   3:7   899       MOVETO(21+I*32,9);
 17   3:7   909       PENCOLOR(BLACK);
 17   3:7   913       MOVE(3);
 17   3:7   917       IF DOTTEDLINES THEN
 17   3:8   920         BEGIN
 17   3:9   920         FOR J:=0 TO 45 DO
 17   3:0   933           BEGIN
 17   3:1   933           PENCOLOR(WHITE);
 17   3:1   937           MOVE(1);
 17   3:1   941           PENCOLOR(BLACK);
 17   3:1   945           MOVE(1);
 17   3:1   949           PENCOLOR(WHITE);
 17   3:1   953           MOVE(1);
 17   3:1   957           PENCOLOR(BLACK);
 17   3:1   961           MOVE(1);
 17   3:0   965           END;
 17   3:8   973         END
 17   3:7   973       ELSE
 17   3:8   975         BEGIN
 17   3:9   975         PENCOLOR(WHITE);
 17   3:9   979         MOVE(184);
 17   3:8   985         END;
 17   3:7   985       TURN(180);
 17   3:7   991       PENCOLOR(BLACK);
 17   3:7   995       MOVE(6);
 17   3:6   999       END;
 17   3:4   999     END;
 17   3:3   999   PENCOLOR(NONE);
 17   3:2  1003   END;
 17   3:1  1011 XBOTTOMLINE:=LINEVALUE[0];
 17   3:1  1024 XDIVSIZE:=LINEVALUE[1]-XBOTTOMLINE;
 17   3:0  1041 END;
 17   3:0  1074
 17   3:0  1074
  1   1:0     0 END.
```

```
 1    1:D    1 (*$L #8:*)
 1    1:D    1 (*$S+*)
 1    1:D    1
18    1:D    1 UNIT PLOTTERGRAPHICS;INTRINSIC CODE 18 DATA 19;
18    1:D    1
18    1:D    1 (************************** PLOTTERGRAPHICS ************************)
18    1:D    1 (*                                                                  *)
18    1:D    1 (*                    WRITTEN BY G.C. MCGONIGAL                     *)
18    1:D    1 (*                                                                  *)
18    1:D    1 (*                  MATERIALS & DEVICES RESEARCH LAB                *)
18    1:D    1 (*                  DEPT. OF  ELECTRICAL ENGINEERING                *)
18    1:D    1 (*                      UNIVERSITY OF MANITOBA                      *)
18    1:D    1 (*                                                                  *)
18    1:D    1 (*                          VER I.0/1982                            *)
18    1:D    1 (*                                                                  *)
18    1:D    1 (********************************************************************)
18    1:D    1 (*                                                                  *)
18    1:D    1 (*    PLOTTERGRAPHICS IS A GRAPHICS PACKAGE DESIGNED AS THE PLOTTER *)
18    1:D    1 (* IMPLEMENTATION OF UCSD TURTLEGRAPHICS. MOST TURTLEGRAPHICS      *)
18    1:D    1 (* SUBROUTINES ARE SUPPORTTED.                                      *)
18    1:D    1 (*                                                                  *)
18    1:D    1 (*    PROGRAMS ARE CONVERTED FROM TURTLEGRAPHICS TO PLOTTERGRAPHICS *)
18    1:D    1 (* BY PREFIXING THE SUBROUTINES WITH 'H'.                           *)
18    1:D    1 (*                                                                  *)
18    1:D    1 (*    NOTE THAT THE PLOT DIMENSIONS REMAIN 192 X 282 BUT ALL        *)
18    1:D    1 (* POSITIONAL & ANGULAR PARAMETERS ARE NOW OF TYPE REAL. ALSO,     *)
18    1:D    1 (* THE HPENCOLOR PARAMETERS HAVE BEEN MODIFIED.                     *)
18    1:D    1 (*                                                                  *)
18    1:D    1 (* SUMMARY OF AVAILABLE PROCEDURES:                                 *)
18    1:D    1 (*                                                                  *)
18    1:D    1 (* HINITTURTLE                                                      *)
18    1:D    1 (* HTURN                                                            *)
18    1:D    1 (* HTURNTO                                                          *)
18    1:D    1 (* HMOVE                                                            *)
18    1:D    1 (* HMOVETO                                                          *)
18    1:D    1 (* HPENCOLOR                                                        *)
18    1:D    1 (* HVIEWPORT                                                        *)
18    1:D    1 (* HWSTRING                                                         *)
18    1:D    1 (*                                                                  *)
18    1:D    1 (********************************************************************)
18    1:D    1
18    1:D    1 INTERFACE
18    1:D    1
29    1:D    1
29    2:D    3   FUNCTION SIN(X:REAL):REAL;
29    3:D    3   FUNCTION COS(X:REAL):REAL;
29    4:D    3   FUNCTION EXP(X:REAL):REAL;
29    5:D    3   FUNCTION ATAN(X:REAL):REAL;
29    6:D    3   FUNCTION LN(X:REAL):REAL;
29    7:D    3   FUNCTION LOG(X:REAL):REAL;
29    8:D    3   FUNCTION SQRT(X:REAL):REAL;
29    8:D    5
18    1:D    5   USES TRANSCEND;
18    1:D    1
18    1:D    1   TYPE HSCREENCOLOR=(NO,YES,ONE,TWO,AWAY);
18    1:D    1
18    1:D    1   VAR PLOTTER:TEXT;
18    1:D  302
18    2:D    1   PROCEDURE HINITTURTLE;
18    2:D    1
18    3:D    1   PROCEDURE HTURN(ANGLE:REAL);
18    3:D    3
18    4:D    1   PROCEDURE HTURNTO(ANGLE:REAL);
18    4:D    3
18    5:D    1   PROCEDURE HMOVE(DIST:REAL);
18    5:D    3
18    6:D    1   PROCEDURE HMOVETO(X,Y:REAL);
18    6:D    5
18    7:D    1   PROCEDURE HPENCOLOR(PENMODE:HSCREENCOLOR);
18    7:D    2
18    8:D    1   PROCEDURE HVIEWPORT(LEFT,RIGHT,BOTTOM,TOP:REAL);
18    8:D    9
18    9:D    1   PROCEDURE HWSTRING(S:STRING);
18    9:D   43
```

```
 18   9:D    43
 18   1:D    43 IMPLEMENTATION
 18   1:D   302
 18   1:D   302   VAR CURRENTANGLE:REAL;
 18   1:D   304
 18   2:D     1   PROCEDURE HINITTURTLE;
 18   2:0     0   BEGIN
 18   2:1     0   REWRITE(PLOTTER,'REMOUT:');
 18   2:1    20   (* SET SCALING SAME AS SCREEN *)
 18   2:1    20   WRITE(PLOTTER,'IN;IP;SC0,282,0,192;');
 18   2:1    52   (* PENCOLOR=NO,PEN:=LLH CORNER  *)
 18   2:1    52   WRITE(PLOTTER,'PU;SP0;PA0.0,0.0;');
 18   2:1    81   CURRENTANGLE:=0.0;
 18   2:0    94   END;
 18   2:0   106
 18   2:0   106
 18   3:D     1   PROCEDURE HTURN;
 18   3:0     0   BEGIN
 18   3:1     0   CURRENTANGLE:=CURRENTANGLE+ANGLE;
 18   3:0    17   END;
 18   3:0    30
 18   3:0    30
 18   4:D     1   PROCEDURE HTURNTO;
 18   4:0     0   BEGIN
 18   4:1     0   CURRENTANGLE:=ANGLE;
 18   4:0    10   END;
 18   4:0    22
 18   4:0    22
 18   5:D     1   PROCEDURE HMOVE;
 18   5:D     3   VAR XMOVE,YMOVE:REAL;
 18   5:0     0   BEGIN
 18   5:1     0   XMOVE:=DIST*COS(CURRENTANGLE*1.74533E-2);
 18   5:1    27   YMOVE:=DIST*SIN(CURRENTANGLE*1.74533E-2);
 18   5:1    55   WRITE(PLOTTER,'PR',XMOVE:9:4,',',YMOVE:9:4,';');
 18   5:0   117   END;
 18   5:0   130
 18   5:0   130
 18   6:D     1   PROCEDURE HMOVETO;
 18   6:0     0   BEGIN
 18   6:1     0   WRITE(PLOTTER,'PA',X:9:4,',',Y:9:4,';');
 18   6:0    62   END;
 18   6:0    74
 18   6:0    74
 18   7:D     1   PROCEDURE HPENCOLOR;
 18   7:0     0   BEGIN
 18   7:1     0   CASE PENMODE OF
 18   7:1     3     NO:  WRITE(PLOTTER,'PU;');
 18   7:1    20     ONE: WRITE(PLOTTER,'SP1;');
 18   7:1    38     TWO: WRITE(PLOTTER,'SP2;');
 18   7:1    56     YES: WRITE(PLOTTER,'PD;');
 18   7:1    73     AWAY:WRITE(PLOTTER,'SP0;');
 18   7:1    91     END;
 18   7:0   108   END;
 18   7:0   120
 18   7:0   120
 18   8:D     1   PROCEDURE HVIEWPORT;
 18   8:0     0   BEGIN
 18   8:0     0   (* IW USES ONLY PLOTTER UNITS *)
 18   8:1     0   WRITE(PLOTTER,'IW',250+ROUND(LEFT*36.5248):6,',',279+ROUND(BOTTOM*39.375):6,',',
 18   8:1    86         250+ROUND(RIGHT*36.5248):6,',',279+ROUND(TOP*39.375):6,';');
 18   8:0   158   END;
 18   8:0   170
 18   9:D     1   PROCEDURE HWSTRING;
 18   9:0     0   BEGIN
 18   9:1     0   WRITE(PLOTTER,'DT',CHR(3),';LB',S,CHR(3));
 18   9:0    65   END;
 18   9:0    78
  1   1:0     0 END.
```

```
  1     1:D     1 (*$L #B:*)
  1     1:D     1 (*$S++*)
  1     1:D     1
 23     1:D     1 UNIT PENPLOTAIDS;INTRINSIC CODE 23;
 23     1:D     1
 23     1:D     1 (********************** PENPLOTAIDS **********************)
 23     1:D     1 (*                                                        *)
 23     1:D     1 (*                 WRITTEN BY G.C. MCGONIGAL              *)
 23     1:D     1 (*                                                        *)
 23     1:D     1 (*            MATERIALS & DEVICES RESEARCH LAB            *)
 23     1:D     1 (*            DEPT. OF  ELECTRICAL ENGINEERING            *)
 23     1:D     1 (*                 UNIVERSITY OF MANITOBA                 *)
 23     1:D     1 (*                                                        *)
 23     1:D     1 (*                      VER I.0/1982                      *)
 23     1:D     1 (*                                                        *)
 23     1:D     1 (**********************************************************)
 23     1:D     1 (*                                                        *)
 23     1:D     1 (* PENPLOTAIDS IS A PACKAGE OF PASCAL SUBROUTINES DESIGNED*)
 23     1:D     1 (* TO REDUCE THE COMPLEXITY OF PROGRAMS THAT CONSTRUCT    *)
 23     1:D     1 (* GRAPHICAL DISPLAYS ON THE HP7470A PLOTTER.             *)
 23     1:D     1 (*                                                        *)
 23     1:D     1 (* PENPLOTAIDS IS ACCESSED FROM SYSTEM.LIBRARY BY THE DECL*)
 23     1:D     1 (*            USES TRANSCEND,TURTLEGRAPHICS,PENPLOTAIDS;  *)
 23     1:D     1 (*                                                        *)
 23     1:D     1 (* SUMMARY OF PROC. & FUNC. PROVIDED:                     *)
 23     1:D     1 (* NAME:      TYPE/FUNCTION:                              *)
 23     1:D     1 (*                                                        *)
 23     1:D     1 (* HPLOTPOINT PROC WRITES A SYMBOL AT THE SPECIFIED PT.   *)
 23     1:D     1 (* HDRAWGRID  PROC GENERATES AND LABELS THE GRIDLINES.    *)
 23     1:D     1 (* HAXISLABEL PROC WRITES TITLES FOR THE X- & Y-AXIS.     *)
 23     1:D     1 (*                                                        *)
 23     1:D     1 (**********************************************************)
 23     1:D     1
 23     1:D     1   INTERFACE
 23     1:D     1
 29     1:D     1
 29     2:D     3   FUNCTION SIN(X:REAL):REAL;
 29     3:D     3   FUNCTION COS(X:REAL):REAL;
 29     4:D     3   FUNCTION EXP(X:REAL):REAL;
 29     5:D     3   FUNCTION ATAN(X:REAL):REAL;
 29     6:D     3   FUNCTION LN(X:REAL):REAL;
 29     7:D     3   FUNCTION LOG(X:REAL):REAL;
 29     8:D     3   FUNCTION SQRT(X:REAL):REAL;
 29     8:D     5
 18     1:D     5
 18     1:D     1
 18     1:D     1   USES TRANSCEND;
 18     1:D     1
 18     1:D     1   TYPE HSCREENCOLOR=(NO,YES,ONE,TWO,AWAY);
 18     1:D     1
 18     1:D     1   VAR PLOTTER:TEXT;
 18     1:D   302
 18     2:D     1   PROCEDURE HINITTURTLE;
 18     2:D     1
 18     3:D     1   PROCEDURE HTURN(ANGLE:REAL);
 18     3:D     3
 18     4:D     1   PROCEDURE HTURNTO(ANGLE:REAL);
 18     4:D     3
 18     5:D     1   PROCEDURE HMOVE(DIST:REAL);
 18     5:D     3
 18     6:D     1   PROCEDURE HMOVETO(X,Y:REAL);
 18     6:D     5
 18     7:D     1   PROCEDURE HPENCOLOR(PENMODE:HSCREENCOLOR);
 18     7:D     2
 18     8:D     1   PROCEDURE HVIEWPORT(LEFT,RIGHT,BOTTOM,TOP:REAL);
 18     8:D     9
 18     9:D     1   PROCEDURE HWSTRING(S:STRING);
 18     9:D    43
 18     9:D    43
 23     1:D    43     USES TRANSCEND,PLOTTERGRAPHICS;
 23     1:D     1
 23     1:D     1     TYPE HPOINTMARKER=(HPOINT,HDOT,HCROSS,HSQUARE);
 23     1:D     1
 23     1:D     1          HVECTOR= ARRAY[0..7] OF REAL;
```

```
 23    1:D     1
 23    2:D     1       PROCEDURE HPLOTPOINT(XNUM,YNUM,
 23    2:D     1                             XBOTTOMLINE,YBOTTOMLINE,XDIVSIZE,YDIVSIZE:REAL;
 23    2:D    13                             CONNECTPOINTS:BOOLEAN;POINTSYMBOL:HPOINTMARKER);
 23    2:D    15
 23    3:D     1       PROCEDURE HDRAWGRID(XMAX,XMIN,YMAX,YMIN:REAL;
 23    3:D     9                           VAR XBOTTOMLINE,YBOTTOMLINE,
 23    3:D     9                           XDIVSIZE,YDIVSIZE:REAL;DOTTEDLINES:BOOLEAN);
 23    3:D    14
 23    4:D     1       PROCEDURE HAXISLABEL(XLABEL,YLABEL:STRING);
 23    4:D    85
 23    4:D    85
 23    1:D    85    IMPLEMENTATION
 23    1:D     1
 23    1:D     1
 23    5:D     3 FUNCTION HEXPO(X:REAL):INTEGER;
 23    5:D     5 (* RETURNS THE EXPONENTIAL PART OF X *)
 23    5:0     0 BEGIN
 23    5:1     0    IF X=0.0 THEN
 23    5:2    14      HEXPO:=0
 23    5:1    14    ELSE
 23    5:2    19      HEXPO:=TRUNC(LOG(ABS(X))+100)-100;
 23    5:0    38 END;
 23    5:0    50
 23    5:0    50
 23    6:D     3 FUNCTION HMANTISSA(X:REAL):REAL;
 23    6:D     5 (* RETURNS THE MANTISSA OF X *)
 23    6:0     0 BEGIN
 23    6:1     0    HMANTISSA:=X/EXP(HEXPO(X)*2.30259);
 23    6:0    30 END;
 23    6:0    42
 23    6:0    42
 23    7:D     1 PROCEDURE HDRAWDARK;
 23    7:D     1 (* DRAWS GRIDLINES*)
 23    7:D     1 VAR I:INTEGER;
 23    7:0     0 BEGIN
 23    7:1     0 HPENCOLOR(NO);
 23    7:1     4 FOR I:=1 TO 7 DO
 23    7:2    15    BEGIN
 23    7:3    15    HMOVETO(21.0,9+I*23.0);
 23    7:3    37    HPENCOLOR(YES);
 23    7:3    41    HMOVETO(276.0,9+I*23.0);
 23    7:3    63    HPENCOLOR(NO);
 23    7:3    67    HMOVETO(21+I*32.0,9.0);
 23    7:3    89    HPENCOLOR(YES);
 23    7:3    93    HMOVETO(21+I*32.0,192.0);
 23    7:3   115    HPENCOLOR(NO);
 23    7:2   119    END;
 23    7:0   126 END;
 23    7:0   140
 23    7:0   140
 23    8:D     1 PROCEDURE HLINEARAXIS(VAR LINEVALUE:HVECTOR;MAX,MIN:REAL);
 23    8:D     6 (* USED BY HDRAWGRID TO ESTABLISH AXIS SCALING*)
 23    8:D     6 VAR I,RGPTR: INTEGER;
 23    8:D     8      DIVSIZE,RANGE,RGMANTISSA: REAL;
 23    8:D    14      FULLRANGE:ARRAY[0..10] OF REAL;
 23    8:D    36
 23    8:0     0 BEGIN
 23    8:0     0 (* SET-UP RANGE POSSIBILITIES *)
 23    8:1     0 FULLRANGE[0]:=1.2;
 23    8:1    16 FULLRANGE[1]:=1.6;
 23    8:1    32 FULLRANGE[2]:=2.0;
 23    8:1    48 FULLRANGE[3]:=2.4;
 23    8:1    64 FULLRANGE[4]:=3.2;
 23    8:1    80 FULLRANGE[5]:=4.0;
 23    8:1    96 FULLRANGE[6]:=4.8;
 23    8:1   112 FULLRANGE[7]:=6.4;
 23    8:1   128 FULLRANGE[8]:=8.0;
 23    8:1   144 FULLRANGE[9]:=12.0;
 23    8:1   160 FULLRANGE[10]:=16.0;
 23    8:1   176
 23    8:1   176 RANGE:=MAX-MIN;
 23    8:1   189
 23    8:1   189 (* CHECK FOR A CONDITION OF ZERO EXTENT *)
 23    8:1   189 IF RANGE=0.0 THEN
 23    8:2   204    BEGIN
 23    8:3   204    MAX:=ABS(MAX*1.1);
 23    8:3   220    MIN:=-ABS(MIN*1.1);
 23    8:2   237    END;
```

```
 23    8:2   237
 23    8:1   237 REPEAT
 23    8:2   237   BEGIN
 23    8:2   237   (* FIND A SUITABLE RANGE *)
 23    8:3   237   RGMANTISSA:=HMANTISSA(RANGE);
 23    8:3   249   RGPTR:=-1;
 23    8:3   253   REPEAT
 23    8:4   253     RGPTR:=RGPTR+1;
 23    8:3   258   UNTIL RGMANTISSA <= FULLRANGE[RGPTR];
 23    8:3   276
 23    8:3   276   (* MAKE EACH DIV 1/8 OF FULLRANGE *)
 23    8:3   276   DIVSIZE:=FULLRANGE[RGPTR]/8.0
 23    8:3   288             *EXP(HEXPO(RANGE)*2.30259);
 23    8:3   320
 23    8:3   320   (* DETERMINE SUITABLE BOTTOMLINE *)
 23    8:3   320   IF MIN >= 0.0 THEN
 23    8:4   334     BEGIN
 23    8:5   334     LINEVALUE[0]:=0.0;
 23    8:5   350     WHILE MIN > LINEVALUE[0]+DIVSIZE DO
 23    8:6   372       LINEVALUE[0]:=LINEVALUE[0]+DIVSIZE;
 23    8:4   397     END
 23    8:3   397   ELSE
 23    8:4   399     BEGIN
 23    8:5   399     LINEVALUE[0]:=0.0;
 23    8:5   414     WHILE LINEVALUE[0]*1.001 > MIN DO
 23    8:6   439       LINEVALUE[0]:=LINEVALUE[0]-DIVSIZE;
 23    8:4   464     END;
 23    8:4   464
 23    8:4   464   (* DERIVE 8 GRIDLINES *)
 23    8:3   464   FOR I:=1 TO 7 DO
 23    8:4   476     BEGIN
 23    8:5   476     LINEVALUE[I]:=LINEVALUE[I-1]+DIVSIZE;
 23    8:4   501     END;
 23    8:2   508   END
 23    8:2   508 (* CHECK THAT THIS RESULT WILL BE SUITABLE *)
 23    8:1   508 UNTIL LINEVALUE[7]-LINEVALUE[0]+DIVSIZE >= RANGE;
 23    8:0   540 END;
 23    8:0   562
 23    8:0   562
 23    4:D     1 PROCEDURE HAXISLABEL;
 23    4:D    85 (* STRING XPRE IS PRINTED BEFORE THE EXPONENT,
 23    4:D    85              XPOST AFTER EXP *)
 23    4:D    85
 23    4:0     0 BEGIN
 23    4:1     0 HPENCOLOR(NO);
 23    4:1    14 HMOVETO(120.0,-1.0);
 23    4:1    30 HWSTRING(XLABEL);
 23    4:1    35 HMOVETO(2.0,74.0);
 23    4:1    51 WRITE(PLOTTER,'DI0.0,1.0;');
 23    4:1    73 HWSTRING(YLABEL);
 23    4:1    78 WRITE(PLOTTER,'DI;');
 23    4:0    93 END;
 23    4:0   106
 23    4:0   106
 23    2:D     1 PROCEDURE HPLOTPOINT;
 23    2:D    15 (* WRITES A SYMBOL AT THE REAL # CO-ORD. (XNUM,YNUM)
 23    2:D    15    XBOTTOMLINE,YBOTTOMLINE,XDIVSIZE,YDIVSIZE ARE REAL PARMS GENERATED
 23    2:D    15    BY DRAWGRID. THE POINT WILL BE CONNECTED BY A STRAIGHT LINE TO THE
 23    2:D    15    PREVIOUS POINT IF CONNECTPOINTS=TRUE. POINTSYMBOL IS OF TYPE
 23    2:D    15    POINTMARKER (EITHER CROSS,SQUARE,OR POINT). *)
 23    2:D    15
 23    2:D    15 VAR C:INTEGER;
 23    2:D    16
 23    2:0     0 BEGIN
 23    2:1     0 IF NOT CONNECTPOINTS THEN
 23    2:2     4   HPENCOLOR(NO)
 23    2:1     5 ELSE
 23    2:2    10   HPENCOLOR(YES);
 23    2:2    14 (* CALC POSITION ON SCREEN *)
 23    2:1    14 HMOVETO(21.0+(XNUM-XBOTTOMLINE)/XDIVSIZE*32.0,
 23    2:1    42         9.0+(YNUM-YBOTTOMLINE)/YDIVSIZE*23.0);
 23    2:1    73 HPENCOLOR(YES);
 23    2:1    77 IF POINTSYMBOL = HCROSS THEN
 23    2:2    82   WRITE(PLOTTER,'PR0.0,1.3,0.0,-2.6,0.0,1.3,1.3,0.0,-2.6,0.0,1.3,0.0;');
 23    2:1   146 IF POINTSYMBOL = HSQUARE THEN
 23    2:2   151   BEGIN
 23    2:3   151   WRITE(PLOTTER,'PR0.0,1.0,-1.0,0.0,0.0,-2.0,2.0,0.0,');
 23    2:3   199   WRITE(PLOTTER,'0.0,2.0,-1.0,0.0,0.0,-1.0;');
 23    2:2   237   END;
```

```
23     2:1   237 IF POINTSYMBOL = HPOINT THEN
23     2:2   242   BEGIN
23     2:3   242   HPENCOLOR(YES);
23     2:3   246   WRITE(PLOTTER,'PR0.0,0.1,0.0,-0.2,0.0,0.1;');
23     2:2   285   END;
23     2:0   285 END;
23     2:0   298
23     2:0   298
23     9:D     1 PROCEDURE DRAWAXIS(J,I:INTEGER);
23     9:0     0 BEGIN
23     9:1     0 HMOVETO(21+J*32.0,9+I*23.0);
23     9:1    27 HTURN(90.0);
23     9:1    37 HPENCOLOR(YES);
23     9:1    41 WRITE(PLOTTER,'PR0.0,3.0,0.0,-6.0,0.0,3.0;');
23     9:0    80 END;
23     9:0    92
23     9:0    92
23    10:D     1 PROCEDURE OUTLINE;
23    10:D     1 VAR I:INTEGER;
23    10:0     0 BEGIN
23    10:1     0 HMOVETO(21.0,9.0);
23    10:1    15 HPENCOLOR(TWO);
23    10:1    19 HPENCOLOR(YES);
23    10:1    23 FOR I:=0 TO 7 DO
23    10:2    34   BEGIN
23    10:3    34   WRITE(PLOTTER,'PR0.0,23.0,3.0,0.0,-3.0,0.0;');
23    10:2    74   END;
23    10:1    81 FOR I:=0 TO 7 DO
23    10:2    92   BEGIN
23    10:3    92   WRITE(PLOTTER,'PR32.0,0.0,0.0,-3.0,0.0,3.0;');
23    10:2   132   END;
23    10:1   139 FOR I:=0 TO 7 DO
23    10:2   150   BEGIN
23    10:3   150   WRITE(PLOTTER,'PR0.0,-23.0,-3.0,0.0,3.0,0.0;');
23    10:2   191   END;
23    10:1   198 FOR I:=0 TO 6 DO
23    10:2   209   BEGIN
23    10:3   209   WRITE(PLOTTER,'PR-32.0,0.0,0.0,3.0,0.0,-3.0;');
23    10:2   250   END;
23    10:1   257 HMOVETO(22.0,9.0);
23    10:1   273 HPENCOLOR(NO);
23    10:1   277 HPENCOLOR(ONE);
23    10:0   281 END;
23    10:0   302
23    10:0   302
23     3:D     1 PROCEDURE HDRAWGRID;
23     3:D    14 (* DRAWS & LABELS A GRID SO AS TO ACCOMODATE A SET OF POINTS OF THE
23     3:D    14      EXTENT XMAX DOWNTO XMIN ON THE X-AXIS AND YMAX DOWNTO YMIN ON
23     3:D    14      THE Y-AXIS. THESE DIMENSIONS CAN BE GENERATED BY DATASCAN OR
23     3:D    14      FIXED BY THE PROGRAMMER.
23     3:D    14      XBOTTOMLINE IS THE REAL VALUE OF THE LEFT-MOST VERT. LINE,
23     3:D    14      YBOTTOMLINE  "  "   "     "    "  "    LOWEST  HORZ. LINE,
23     3:D    14      XDIVSIZE=1/8 OF THE RANGE IN X-VALUES THAT CAN BE PLOTTED,
23     3:D    14      YDIVSIZE=1/8  "  "    "    " Y-VALUES   "   "    " PLOTTED  *)
23     3:D    14
23     3:D    14 VAR RANGE,LINELABEL: REAL;
23     3:D    18      EXPLABEL,I,J:INTEGER;
23     3:D    21      LABELSTR:STRING[30];
23     3:D    37      LINEVALUE:ARRAY[0..7] OF REAL;
23     3:D    53
23     3:0     0 BEGIN
23     3:1     0 HINITTURTLE;
23     3:1     3 OUTLINE;
23     3:1     5
23     3:1     5 (***** DO Y-LINES *****)
23     3:1     5 HLINEARAXIS(LINEVALUE,YMAX,YMIN);
23     3:1    17
23     3:1    17 (* PRINT Y-AXIS EXPONENT *)
23     3:1    17 EXPLABEL:=HEXPO(LINEVALUE[1]-LINEVALUE[0]);
23     3:1    44 IF (EXPLABEL > 2) OR (EXPLABEL < -1) THEN
23     3:2    56   BEGIN
23     3:3    56   HMOVETO(2.0,123.0);
23     3:3    71   WRITE(PLOTTER,'DI0.0,1.0;SR0.5,1.0;');
23     3:3   103   HWSTRING('X');
23     3:3   107   WRITE(PLOTTER,'SR;');
23     3:3   122   HWSTRING('10');
23     3:3   130   WRITE(PLOTTER,'PR-2.0,0.0;');
23     3:3   153   STR(EXPLABEL,LABELSTR);
23     3:3   166   HWSTRING(LABELSTR);
```

```
23   3:3   171     WRITE(PLOTTER,'DI;');
23   3:2   186     END
23   3:1   186 ELSE
23   3:2   188     EXPLABEL:=0;
23   3:2   191
23   3:2   191 (* WRITE LABEL & GRID LINE *)
23   3:1   191 FOR I:=7 DOWNTO 0 DO
23   3:2   204     BEGIN
23   3:3   204     HMOVETO(0.0,I*23.0+8);
23   3:3   226     LINELABEL:=LINEVALUE[I]/EXP(EXPLABEL*2.30259);
23   3:3   258     WRITE(PLOTTER,'SR0.6,1.2;LB',LINELABEL:8:2,CHR(3),'SR;');
23   3:3   321     (*CHECK FOR ZERO AXIS*)
23   3:3   321     IF (ABS(LINEVALUE[I]/(LINEVALUE[7]-LINEVALUE[0])) < 0.01)   AND
23   3:3   364       (I<>0) THEN
23   3:4   371       BEGIN
23   3:5   371       HTURNTO(0.0);
23   3:5   381       FOR J:=0 TO 8 DO
23   3:6   394         BEGIN
23   3:7   394         HVIEWPORT(21.0,290.0,9.0,192.0);
23   3:7   421         DRAWAXIS(J,I);
23   3:7   427         HVIEWPORT(0.0,292.0,0.0,192.0);
23   3:6   455         END;
23   3:4   463       END;
23   3:3   463     HPENCOLOR(NO);
23   3:2   467     END;
23   3:1   475 YBOTTOMLINE:=LINEVALUE[0];
23   3:1   488 YDIVSIZE:=LINEVALUE[1]-YBOTTOMLINE;
23   3:1   505
23   3:1   505 (***** DO X-LINES *****)
23   3:1   505 HLINEARAXIS(LINEVALUE,XMAX,XMIN);
23   3:1   517
23   3:1   517 (* PRINT X-AXIS EXPONENT *)
23   3:1   517 EXPLABEL:=HEXPO(LINEVALUE[1]-LINEVALUE[0]);
23   3:1   544 IF (EXPLABEL > 2) OR (EXPLABEL < -1) THEN
23   3:2   556     BEGIN
23   3:3   556     HMOVETO(185.0,-1.0);
23   3:3   572     WRITE(PLOTTER,'SR0.5,1.0;');
23   3:3   594     HWSTRING('X');
23   3:3   598     WRITE(PLOTTER,'SR;');
23   3:3   613     HWSTRING('10');
23   3:3   621     WRITE(PLOTTER,'PR0.0,2.0;');
23   3:3   643     STR(EXPLABEL,LABELSTR);
23   3:3   656     HWSTRING(LABELSTR);
23   3:3   661     WRITE(PLOTTER,'DI;');
23   3:2   676     END
23   3:1   676 ELSE
23   3:2   678     EXPLABEL:=0;
23   3:2   681
23   3:2   681 (* WRITE LABEL & GRID LINE *)
23   3:2   681
23   3:1   681 FOR I:=7 DOWNTO 0 DO
23   3:2   694     BEGIN
23   3:3   694     HMOVETO(13+I*32.0,5.0);
23   3:3   717     LINELABEL:=LINEVALUE[I]/EXP(EXPLABEL*2.30259);
23   3:3   748     WRITE(PLOTTER,'SR0.6,1.2;LB',LINELABEL:6:2,CHR(3),'SR;');
23   3:3   811     (*CHECK FOR AXIS*)
23   3:3   811     IF (ABS(LINEVALUE[I]/(LINEVALUE[7]-LINEVALUE[0])) < 0.01) AND
23   3:3   854       (I<>0) THEN
23   3:4   861       BEGIN
23   3:5   861       HVIEWPORT(21.0,290.0,8.0,192.0);
23   3:5   889       HTURNTO(90.0);
23   3:5   899       FOR J:=0 TO 8 DO
23   3:6   912         BEGIN
23   3:7   912         HMOVETO(21+I*32.0,9+J*23.0);
23   3:7   943         HPENCOLOR(YES);
23   3:7   947         WRITE(PLOTTER,'PR3.0,0.0,-6.0,0.0,3.0,0.0;');
23   3:6   986         END;
23   3:5   994       HVIEWPORT(0.0,292.0,0.0,192.0);
23   3:4  1021       END;
23   3:3  1021     HPENCOLOR(NO);
23   3:2  1025     END;
23   3:1  1033 IF DOTTEDLINES THEN HDRAWDARK;
23   3:1  1038 XBOTTOMLINE:=LINEVALUE[0];
23   3:1  1051 XDIVSIZE:=LINEVALUE[1]-XBOTTOMLINE;
23   3:0  1068 END;
23   3:0  1096
23   3:0  1096
 1   1:0     0 END.
```