NAT'L INST. OF STAND. & TECH. R.I.C.
A11103 398630

NIST
PUBLICATIONS

NIST SPECIAL PUBLICATION 400-85

U.S. DEPARTMENT OF COMMERCE/National Institute of Standards and Technology

Semiconductor Measurement Technology:

EPROP: An Interactive FORTRAN Program for Computing Selected Electronic Properties of Gallium Arsenide and Silicon

A. C. Seabaugh, J. J. Mathias, and M. I. Bell

QC
100
.U57
400-85
1990
C.2

NATIONAL INSTITUTE OF STANDARDS &
TECHNOLOGY
Research Information Center
Gaithersburg, MD 20899

DATE DUE

Demco, Inc. 38-293

QC100
.U57
NO.400-8
1990
C.2

Semiconductor Measurement Technology:

EPROP: An Interactive FORTRAN Program for Computing Selected Electronic Properties of Gallium Arsenide and Silicon

Alan C. Seabaugh,* John J. Mathias,† and Michael I. Bell

Semiconductor Electronics Division
Center for Electronics and Electrical Engineering
National Institute of Standards and Technology
Gaithersburg, MD 20899

* Texas Instruments, Incorporated
Dallas, TX 75265

† 8812 Connecticut Avenue
Chevy Chase, MD 20815

May 1990

U.S. DEPARTMENT OF COMMERCE, Robert A. Mosbacher, Secretary
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY, John W. Lyons, Director

National Institute of Standards and Technology Special Publication 400-85
Natl. Inst. Stand. Technol. Spec. Publ. 400-85, 121 pages (May 1990)
CODEN: NSPUE2

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1990

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

Table of Contents

List of Tables

List of Figures

List of Figures (continued)

Semiconductor Measurement Technology:

EPROP: An Interactive FORTRAN Program for Computing Selected Electronic Properties of Gallium Arsenide and Silicon

A. C. Seabaugh*, J. J. Mathias†, and M. I. Bell

Semiconductor Electronics Division
National Institute of Standards and Technology
Gaithersburg, MD 20899

Abstract

A new computer program, EPROP (an acronym for **E**lectronic **PROP**erties) is presented for use in interpreting measurements and experiments on gallium arsenide and silicon. EPROP computes a solution of the charge balance equation in thermodynamic equilibrium for up to six different impurities. The user supplies the density, energy level, and degeneracy for each impurity, and in response the program returns as many as 28 output parameters, such as the Fermi level, carrier density, and ionized impurity densities. These can be computed as functions of the temperature (or reciprocal temperature) or the density, energy, or degeneracy of any of the six possible impurities. Listings can also be obtained of various temperature-dependent parameters, such as the bandgap, densities of states, and effective masses. The interactive features of the program allow the user to send the output data to any combination of destinations: a terminal, a listing file, and/or up to four graphic output files, all at the user's direction. The user is also given freedom and ability to customize the data output to these destinations through menu-driven controls. The program is written in ANSI standard FORTRAN 77 and has been successfully compiled and run on both mainframe and microcomputers. Documentation is provided to assist the interested user in customizing the program for special applications, extracting portions for use elsewhere, or modifying the code to treat semiconductors other than silicon and gallium arsenide.

Keywords: electronic properties; Fermi level; FORTRAN; GaAs; gallium arsenide; Si; silicon.

* Present address, Central Research Laboratories, Texas Instruments Incorporated, Dallas, TX 75265

† Present address, 8812 Connecticut Ave., Chevy Chase, MD 20815

1. Introduction

In analyzing the transport properties of semiconductors, it is often useful to know the Fermi energy as a function of temperature, or as a function of the density, binding energy, or degeneracy of a particular impurity. Once the Fermi level is known, the electron and hole densities can be calculated, as well as the degree of ionization of each of the impurities. This can be useful in designing experiments and in interpreting measurements of resistivity, Hall effect, and capacitance. Such calculations can also be useful in materials and device design, e.g., for estimating the influence of a particular dopant configuration on the free carrier and ionized impurity densities.

The computer program described here, EPROP (an acronym for Electronic **PROP**erties) originated, in concept, with a program written by Larrabee, Thurber, and Bullis [1], which calculates the temperature dependence of certain electronic properties of silicon (Si), such as the free carrier and ionized impurity densities and the mobility. EPROP has expanded on this calculation, incorporating new data and computing properties of both Si and gallium arsenide (GaAs). The mobility calculation of reference [1] has not been included in EPROP, since it is not reliable over the wide range of impurities and temperatures allowed by this program. In addition to computing the temperature dependence of the Fermi level, EPROP can also compute the Fermi-level position as a function of the density, energy, or degeneracy of a particular impurity.

Often a user would like to control where the output of the program is directed, whether to a terminal, printer, or listing file. EPROP accommodates these needs. In addition, as many as four x-y graphic output files can be created, which the user can then customize for output to his particular graphics software and devices.

The purpose of this program is to provide a fast, first-order calculation of the Fermi level as a function of temperature and monovalent impurity parameters. Heavy doping effects, impurity excited states, and multiply-ionized centers are not included. In the case of GaAs, a density of states is used which takes into account the nonparabolicity of the Γ_6 conduction band minimum and the existence of the L_6 and X_6 upper conduction band minima. Inclusion of the upper conduction bands is shown (sec. 2.5.2) to be useful in interpreting high-temperature Hall-effect measurements on GaAs. An attempt has been made to provide enough documentation so that the user can readily "cannibalize" useful subroutines in EPROP or customize the program as desired.

EPROP has been compiled and run successfully on mainfraime, mini-, and microcomputers. A discussion of general considerations related to the portability of the code, as well as specific modifications required for several of the machines on which it has been tested, are given in appendix 1. Appendix 2 describes the changes required in order to include semiconductor materials other than Si and GaAs. Appendix 3 contains a complete listing of the source code. This listing is available in machine-readable form; inquiries should be addressed to the third author.

2. Theory

To simplify the presentation, we use the dimensionless energy scale of Blakemore [2]. A reduced energy variable, $\epsilon = (E - E_c)/kT$, and corresponding reduced Fermi energy, $\eta = (E_F - E_c)/kT$, are defined with respect to the conduction band edge E_c, where E and E_F are the usual energy variable and Fermi energy, respectively. The temperature in Kelvin is represented by T and Boltzmann's constant by k. The reduced Fermi function, giving the occupation probability of the available states, is $f(\epsilon) = [1 + \exp(\epsilon - \eta)]^{-1}$.

2.1. Charge-Balance Equation

In thermodynamic equilibrium, the position of the Fermi level can be determined by assuming that charge neutrality holds; i.e., the sum of all positive and negative charges (electrons, holes, and ionized impurities) is zero,

$$n_o + \sum_i N_{a_i}^- - p_o - \sum_j N_{d_j}^+ = 0, \tag{1}$$

where n_o and p_o are the electron and hole densities, and $N_{a_i}^-$ and $N_{d_j}^+$ are the ionized acceptor and ionized donor densities to be summed over the donor impurities (i) and acceptor impurities (j).

For a parabolic conduction band, the electron density in thermodynamic equilibrium is given by

$$n_o = N_c F_{1/2}(\eta), \tag{2}$$

where N_c is the density of conduction band states,

$$N_c = 2\left(\frac{2\pi \overline{m}_c kT}{h^2}\right)^{3/2}. \tag{3}$$

Here, $\overline{m}_c$ is the average, density-of-states effective mass, and h is Planck's constant. The Fermi-Dirac (F-D) integral is

$$F_j(\eta) = \frac{1}{\Gamma(j+1)} \int_0^\infty \frac{\epsilon^j \, d\epsilon}{1 + \exp(\epsilon - \eta)}, \tag{4}$$

where $\Gamma(i)$ is the gamma function.

For Si, the density of states (3), corresponding to a parabolic band, proves to be adequate for computing the electron carrier density. For GaAs, as will be discussed in section 2.2.2, a different density of states is needed in order to account for the nonparabolic conduction band.

An analogous set of relations exists to describe the hole density in thermodynamic equilibrium,

$$p_o = N_v F_{1/2}(\zeta), \tag{5}$$

where the reduced energy parameter is $\zeta = (E_v - E_F)/kT$, and the valence band density of states is given by,

$$N_v = 2\left(\frac{2\pi \overline{m}_v kT}{h^2}\right)^{3/2}. \tag{6}$$

Here, $\overline{m}_v$ is the average density-of-states effective mass for the valence band. Again, the assumption of parabolic bands appears adequate for Si, but a more accurate treatment is required for GaAs (see sec. 2.2.3).

The donor and acceptor impurities are assumed to be single, monovalent centers located at the discrete energies E_{a_i} and E_{d_j}, respectively. Each ionized acceptor impurity has a density, $N_{a_i}^-$, given by

$$N_{a_i}^- = N_{a_i}\left[1 + g_{v_i} \exp\left(\frac{E_{a_i} - E_F}{kT}\right)\right]^{-1}, \tag{7}$$

and each ionized donor impurity, $N_{d_j}^+$, has a density given by

$$N_{d_j}^+ = N_{d_j}\left[1 + g_{c_j} \exp\left(\frac{E_F - E_{d_j}}{kT}\right)\right]^{-1}. \tag{8}$$

The total impurity density for each acceptor and donor is given by N_{d_j} andN_{a_i}, respectively; g_{v_i} and g_{c_j} are the donor and acceptor degeneracies, respectively.

2.2. Temperature-Dependent Parameters – Gallium Arsenide

Blakemore [3] has reviewed the relevant literature. This section provides an annotated summary of the relationships used in the program, most of which have been obtained from reference [3].

2.2.1. Bandgap – GaAs

The empirical equation of Varshni [4] predicts the temperature dependence of the bandgap in GaAs,

$$E_g(T) = E_g(0) - \frac{\alpha T^2}{T+\beta}, \tag{9}$$

where E_g is the bandgap energy, and α and β are constants determined empirically from optical absorption measurements. At low temperatures, the bandgap has a quadratic temperature dependence, while at high temperatures the dependence becomes linear. As fitted to optical absorption measurements by Thurmond [5], the constants in eq (9) are $E_g(0) = 1.519\,\text{eV}$, $\alpha = 5.405 \times 10^{-4}\,\text{eV/K}$, and $\beta = 204\,\text{K}$. The standard deviation of the fit, five measurements over the range 297 to 973 K, was found to be 2.6 meV. This fit of the Varshni equation was also checked against other optical absorption, photoluminescence, and injection luminescence measurements for temperatures ranging down to 4.2 K and was found to deviate in the worst case by 6.7 meV. Provided that the bandgap is not perturbed by heavy doping effects, the temperature dependence of the bandgap given by this fit of the Varshni equation represents the experimental data very closely.

2.2.2. Conduction Band Density of States – GaAs

The question arises, does the simple density of states in eq (3), corresponding to a parabolic band, adequately describe the density of conduction states in GaAs? Under nondegenerate conditions, the answer is yes, but for many temperatures and doping configurations of interest in GaAs, the condition of nondegeneracy does not hold. For this reason, the program incorporates a modified density of states.

Blakemore has examined a number of attempts to determine the intrinsic carrier density in GaAs [6], and in his review article [3] proposes a modification of the density of states to account for the nonparabolicity of the conduction band. This proposal is based on Kane's [7] $k \cdot p$ perturbation approach to determining the band structure and Vrehen's [8] approximation for nonparabolicity in the energy range $(E - E_c) \ll E_g$. The modified density of states N_c' is given by

$$N_c' = N_c\left[1 - \left(\frac{15\alpha' kT}{4E_g}\right)\frac{F_{3/2}(\eta)}{F_{1/2}(\eta)}\right], \tag{10}$$

with

$$\alpha' = -\left(1 - \frac{\overline{m}_c}{m_o}\right)^2 \frac{3E_g^2 + 4E_g\Delta + 2\Delta^2}{(E_g+\Delta)(3E_g+2\Delta)}, \tag{11}$$

where Δ is the energy difference between the Γ_8 valence band maximum and the Γ_7 split-off band, and m_o is the free electron mass. The spin-orbit splitting Δ is 0.341 eV at room temperature and is assumed to be temperature-independent for lack of any experimental evidence to the contrary. The temperature dependence of α' is influenced by that of the

effective mass (sec. 2.2.4) and the energy gap. In fact, α' is not strongly temperature-dependent, varying between -0.824 at low temperature and -0.854 at the melting point. These modifications to the density of states act to increase the number of states at room temperature by about 6% and to inflate the density monotonically with increasing temperature to nearly 26% at 1000 K.

Long before the temperature reaches 1000 K, however, electrons begin to populate the L_6 and X_6 conduction band minima. As Blakemore shows [3], this effect is important as the temperature increases above 400 K. For temperatures exceeding 900 K, the L_6 conduction band contains more than half of all conduction electrons. It is possible to express the total conduction band population as the sum of the electron densities in each of the three bands [3], Γ_6, L_6, and X_6,

$$n_o = n_\Gamma + n_L + n_X. \tag{12}$$

The nonparabolicity and statistical occupation of the central Γ_6 minimum are accounted for by writing $n_\Gamma = N_c' F_{1/2}(\eta)$. Under nondegenerate conditions, we may assume Boltzmann distributions for the carriers in the L_6 and X_6 bands. Their electron densities (n_L and n_X, respectively) are given by

$$n_L = 2\left(\frac{2\pi \overline{m}_L kT}{h^2}\right)^{3/2} \exp\left(\eta - \frac{\Delta_{\Gamma L}}{kT}\right), \tag{13}$$

and

$$n_X = 2\left(\frac{2\pi \overline{m}_X kT}{h^2}\right)^{3/2} \exp\left(\eta - \frac{\Delta_{\Gamma X}}{kT}\right). \tag{14}$$

As in eqs (3) and (6), the average density-of-states electron masses in the bands, L_6 and X_6, are represented by $\overline{m}_L$ and $\overline{m}_X$. The temperature-independent values used for the effective masses are, following Blakemore [3], $\overline{m}_L = 0.52 m_o$ and $\overline{m}_X = 0.85 m_o$. The temperature-dependent energy difference between the L_6 and Γ_6 conduction bands, $\Delta_{\Gamma L}$, and between the X_6 and Γ_6 conduction bands, $\Delta_{\Gamma X}$, as fit to the Varshni equation (see Blakemore [3] for references) are,

$$\Delta_{\Gamma L} = 0.296\,\mathrm{eV} - (6.45 \times 10^{-5}\,\mathrm{eV/K})\frac{\mathrm{T}^2}{\mathrm{T} + \beta}, \tag{15}$$

and

$$\Delta_{\Gamma X} = 0.462\,\mathrm{eV} + (8.05 \times 10^{-5}\,\mathrm{eV/K})\frac{\mathrm{T}^2}{\mathrm{T} + \beta}, \tag{16}$$

where $\beta = 204\,\mathrm{K}$, as in eq (9).

2.2.3. Valence Band Density of States – GaAs

Blakemore [3] has proposed a companion density of states to eq (10) to account for the nonparabolicity of the light- and heavy-hole valence bands in GaAs:

$$N_v' = 2\left(\frac{2\pi kT}{h^2}\right)^{3/2}\left\{m_h^{3/2} + m_l^{3/2}\left[1 - \left(\frac{15\beta' kT}{4E_g}\right)\frac{F_{3/2}(\zeta)}{F_{1/2}(\zeta)}\right]\right\}, \tag{17}$$

where m_h is the heavy hole effective mass and m_l is the light hole effective mass.

In this treatment, only the light-hole band is nonparabolic, and its nonparabolicity is due entirely to interaction with the conduction band. The temperature-dependent parameter β' which describes the contribution of this nonparabolicity to the density of states is given by

$$\beta' = -\frac{1 + E_g/2\Delta}{\left(1 - E_g/2\chi_l\right)^2}, \tag{18}$$

where $\chi_l = 10.0\,\mathrm{eV}$. The parabolic heavy-hole band is not coupled to any other band in this model. Its mass must be determined entirely by reference to experiment.

The valence band density of states described by eqs (17) and (18) was used in early versions of EPROP and remains available as part of the computer code. A more accurate description of the density of states is provided, however, by the recent work of Lowney and Kahn [9]. This treatment includes the effects of two more distant conduction bands (lying above the primary conduction band at the center of the Brillouin zone) and is able to describe correctly the behavior of the heavy-hole band. The additional conduction bands also produce an effective interaction between the light- and heavy-hole bands which is absent in Blakemore's treatment. Without this interaction, Blakemore's light-hole band turns upward at large wavevector, an erroneous and potentially misleading result. The density-of-states effective masses obtained by Lowney and Kahn differ significantly from those of Blakemore, as can be seen in figures 2 and 3 of reference [9]. Somewhat fortuitously, the density-of-states expansion (17) is in good agreement with the results of Lowney and Kahn for energies less than 300 meV into the band.

Lowney and Kahn [9] provide numerical results for the density-of-states masses as a function of hole energy E. In order to calculate the density of states, these masses have been raised to the 3/2 power and the values fitted with cubic polynomials of the form $A + BE + CE^2 + DE^3$. For the heavy-hole band, the fitting coefficients are $A_h = 0.4009$, $B_h = 0.2637$, $C_h = 0.05909$, and $D_h = -0.3141$. For the light-hole band, the coefficients are $A_l = 0.03396$, $B_l = -0.5439$, $C_l = 5.203$, and $D_l = -6.488$. The density of states is then given by

$$N_v' = 2\left(\frac{2\pi kT}{h^2}\right)^{3/2}\left[A + BkT + C(kT)^2 + D(kT)^3\right], \tag{19}$$

where

$$A = A_h + A_l,$$

$$B = (B_h + B_l)\Gamma(5/2)F_{3/2}(\zeta)/G,$$
$$C = (C_h + C_l)\Gamma(7/2)F_{5/2}(\zeta)/G,$$
$$D = (D_h + D_l)\Gamma(9/2)F_{7/2}(\zeta)/G,$$

and

$$G = \Gamma(3/2)F_{1/2}(\zeta).$$

The F-D integrals of order 5/2 and 7/2 are now used, in addition to those of order 1/2 and 3/2. Note that because the heavy-hole band (as well as the light-hole band) is non-parabolic, its mass requires an expansion in powers of the energy. The total density of states is somewhat greater with this new model, and the results of the calculations are correspondingly changed. EPROP is delivered with the more accurate valence band density of states of Lowney and Kahn; the Blakemore formulation is readily computed, however, by making a simple modification of the source code.

2.2.4. Electron Effective Mass – GaAs

The temperature dependence of the conduction band effective mass is computed, again from $k \cdot p$ theory, [3]

$$\overline{m}_c = \frac{m_o}{1 + \chi_c\left(\frac{2}{E_g} + \frac{1}{E_g + \Delta}\right)}, \tag{20}$$

where $\chi_c = 7.51\,\mathrm{eV}$.

2.2.5. Hole Effective Mass – GaAs

When eq (17) is used for the valence band density of states, the heavy hole mass is set to $m_h = 0.5m_o$, independent of temperature. This corresponds to its room temperature value; at low temperatures ($T < 100\,\mathrm{K}$), Blakemore [3] reports $0.51 \pm 0.02m_o$ as the consensus of various experiments. This is an increase of only 2% from the room temperature value. The temperature dependence of the light hole mass is more significant; from $k \cdot p$ theory [3] it can be described by

$$m_l = \frac{E_g m_o}{2\chi_l - E_g}. \tag{21}$$

When eq (19) is used, the hole effective masses do not appear explicitly in the density of states. Instead, the temperature dependence of the effective masses is reflected in the coefficients A, B, C, and D. The values given in section 2.2.3 apply at low temperature. Since it is not presently possible to determine the temperature dependence of these coefficients [9], EPROP treats them as independent of temperature.

2.3. Temperature-Dependent Parameters – Silicon

The parameters used in EPROP are essentially identical to those employed in the program of reference [1], except that the bandgap is obtained from a fit to the Varshni equation.

2.3.1. Bandgap – Si

The temperature dependence of the bandgap of Si is also described well by the Varshni [4] eq (9). Thurmond's [5] fit to experimental data yields the values: $\alpha = 4.730 \times 10^{-4}\,\mathrm{eV/K}$ and $\beta = 636\,\mathrm{K}$, with $E_g(0) = 1.1700\,\mathrm{eV}$.

2.3.2. Conduction and Valence Band Density of States – Si

For Si, the assumption of parabolic bands is used, leading to the densities of states (3) and (6). No formulation for nonparabolic bands equivalent to that for GaAs is available. The degenerate valence bands and split-off bands have not been included, although formulations for their densities of states have been proposed [10]. These should be important at high temperatures and/or high doping.

2.3.3. Electron Effective Mass – Si

The measurements of electron effective mass of Ukhanov and Mal'tsev [11] and Stradling and Zhukov [12] were summarized by Barber [13]. A polynomial fit to these data in the temperature range 50 to 600 K was obtained by Larrabee et al. [1] and is used in EPROP,

$$\frac{\overline{m}_c}{m_o} = 1.0627 - \sum_{i=1}^{6} a_i T^i. \tag{22}$$

The coefficients a_i are listed in table 2.1.

2.3.4. Hole Effective Mass – Si

The hole effective mass, again fitted by Larrabee et al. [1] to the curves presented in Barber [13], is

$$\frac{\overline{m}_v}{m_o} = 0.590525 - \sum_{i=1}^{6} b_i T^i, \tag{23}$$

where the coefficients b_i are also listed in table 2.1. The fit is for the temperature range 0 to 500 K.

Table 2.1. Effective Mass Coefficients for Silicon

Coefficient i	Electrons (a_i) Eq (22)	Holes (b_i) Eq (23)
1	1.61708×10^{-4}	5.23548×10^{-4}
2	-6.83008×10^{-6}	-1.85678×10^{-5}
3	3.32013×10^{-8}	9.67212×10^{-8}
4	-8.04032×10^{-11}	-2.30049×10^{-10}
5	9.66067×10^{-14}	2.59673×10^{-13}
6	-4.54649×10^{-17}	-1.11997×10^{-16}

2.4. Fermi-Dirac Integral

Numerical integration of the Fermi-Dirac function is unnecessary due to the availability of accurate analytic approximations [14]. While approximations with relative accuracies of 10^{-9} or better [15] are available, a particularly short and simple approximation reported recently by Van Halen and Pulfrey [16] is used here. This set of short series approximations to the integral gives better than two parts in 10^4 accuracy over the entire range of the argument, $-\infty$ to $+\infty$.

For the F-D integrals of order 1/2, 3/2, 5/2, and 7/2, three separate series expansions are used. These can be written as follows:

$$F_j(x) = \sum_{r=1}^{7} a_{jr} G_{jr}(x),$$

where

$$G_{jr} = \begin{cases} (-1)^{r+1} e^{rx} & , \quad x \leq 0 \\ x^{r-1} & , \quad 0 < x < 3.7 \\ x^{j+1-2(r-1)} & , \quad x \geq 3.7 \end{cases} . \tag{24}$$

The coefficients a_{jr} are given by Van Halen and Pulfrey [16].

Equation (24) defines the ranges of validity for the series expansions somewhat differently than Van Halen and Pulfrey [16]. Our evaluation of the series in the range $3.7 \leq x \leq 4$ diverged from the published tables in Blakemore [2] and from the results of the series

approximation of Cody and Thacher [15]. Agreement to within less than five parts in 10^4 is achieved, however, by changing the ranges to those given above (24).

2.5. Justifications

Several computations which were performed to test the program are discussed below. The results serve to document the reasoning behind the analytical formulation and to illustrate the range of parameters for which the results have been examined.

2.5.1. Intrinsic Carrier Density

The nonparabolicity of the valence and conduction bands in GaAs is taken into account by modifying the density-of-states relations as previously described in sections 2.2.2-3. The nonparabolicity results in an increase in the conduction and valence band densities of states. The density of conduction states is further increased by the contributions from the L_6 and X_6 bands as described in eqs (12-14). The importance of these effects on the intrinsic carrier density is shown in figures 2.1 and 2.2. Neither the nonparabolicity nor the contribution of the upper conduction band states significantly affects the intrinsic carrier density for temperatures less than 600 K, as seen in figure 2.1. For temperatures greater than this, however, the upper conduction band states become occupied and modify the intrinsic carrier density. At 1000 K, the intrinsic carrier density computed including all three conduction bands is roughly twice that obtained with only a single conduction band. On the scale of the plot of figure 2.1, the difference between including and excluding the conduction and valence band nonparabolicity is negligible. Figure 2.2, however, shows an expansion of the temperature range between 500 and 1000 K which demonstrates that the nonparabolicity does in fact increase the intrinsic carrier density relative to the parabolic case. EPROP computes the properties of GaAs using the nonparabolic density of states and the upper conduction bands. The value of including the upper conduction band in the computation is seen in the next section.

2.5.2. High-Temperature Hall-Effect Measurements on GaAs

The ability to compute the occupation of the upper conduction band states is important in interpreting high-temperature Hall-effect measurements. This can be seen in an analysis of the high-temperature Hall-effect measurements of Nichols, Yee, and Wolfe [17]. The results show good agreement between experiment and theory using the program output. Incorporation of the nonparabolic density of states has negligible effect on the fit to the data.

The Hall coefficient data for one of the three n-type GaAs specimens measured by Nichols et al. [17] are shown in figure 2.3. For single-carrier conduction in an n-type semiconductor, the reciprocal of the Hall coefficient is proportional to the electron density, and if the scattering factor is unity, then the electron density is just $1/(qR_H)$. This quantity is plotted

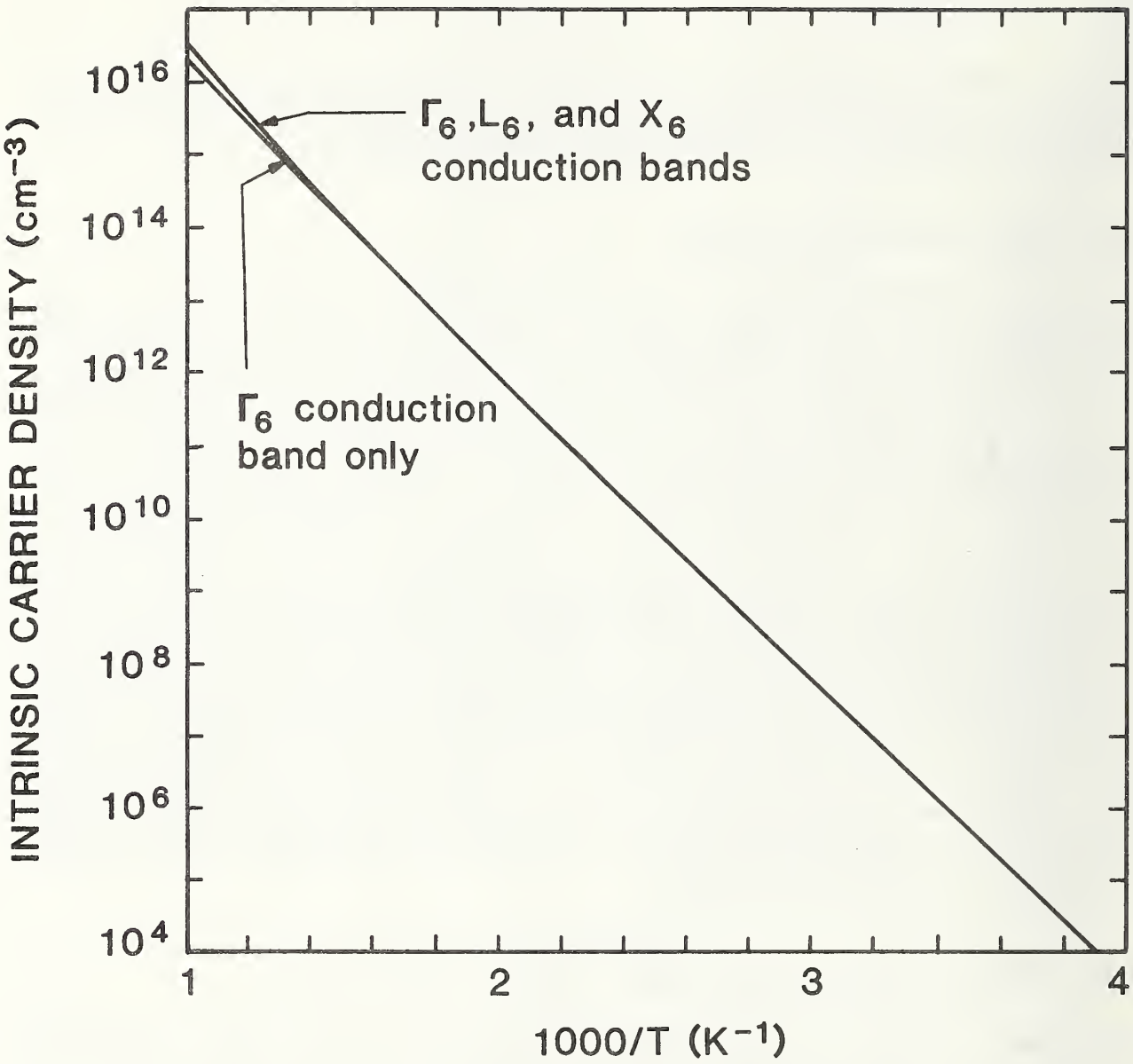

Figure 2.1. Intrinsic Carrier Density in GaAs. Dependence on reciprocal temperature. Inclusion of the upper conduction bands increases the density of states and the intrinsic carrier density at high temperatures.

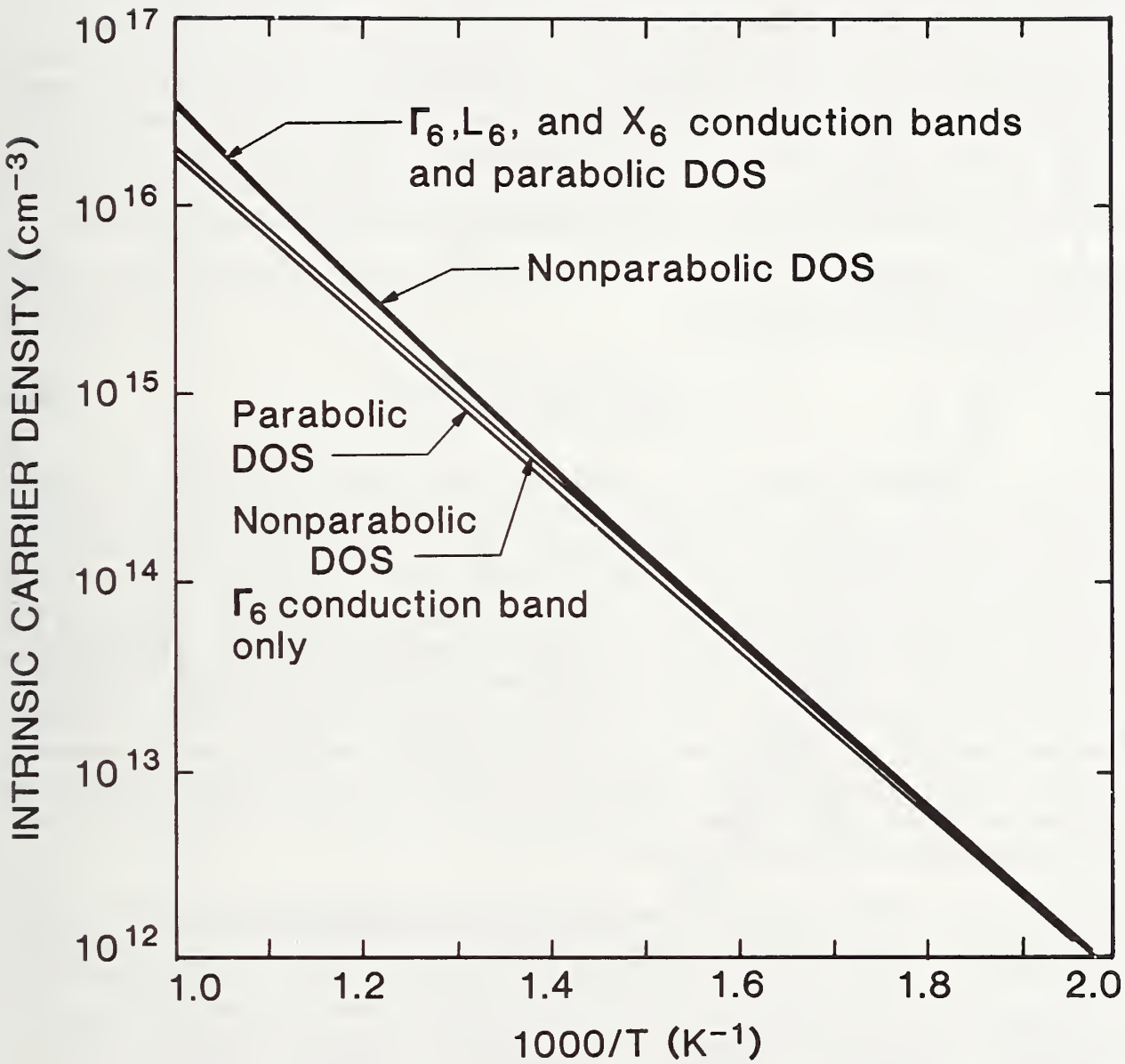

Figure 2.2. Effect of Band Nonparabolicity in GaAs. Influence of conduction band nonparabolicity and the upper conduction bands on the intrinsic carrier density of GaAs at high temperatures. The two lower curves include only the Γ_6 conduction band; the upper two curves (virtually indistinguishable) include the two higher conduction bands.

in figure 2.3. EPROP was used to compute the temperature dependence of the electron density for various formulations of the density of states. There is general agreement using a single donor impurity with a concentration of 2×10^{15} cm^{-3}, an activation energy of 6 meV, and a degeneracy of 2, but this computation does not describe all the features apparent in the experimental data. Curves (a) and (b) are computed without incorporating the upper conduction bands; (a) assumes parabolic bands, while (b) includes nonparabolicity. Curve (c) includes the upper conduction band density of states; the nonparabolic and parabolic density-of-states cases including the upper conduction bands are indistinguishable. It can be seen from all of these curves that the dip in the apparent electron density in the temperature range 500 to 800 K is not accounted for by this interpretation of the data.

Nichols et al. [17] show that these data can be fit to an expression for three-carrier transport: the Hall coefficient is given by

$$R_H = \frac{n_\Gamma \mu_\Gamma^2 + n_L \mu_L^2 - p\mu_h^2}{q(n_\Gamma \mu_\Gamma + n_L \mu_L - p\mu_h)^2}, \tag{25}$$

where the mobilities in the Γ_6 and L_6 conduction bands and the valence band are represented by μ_Γ, μ_L, and μ_h, respectively. For carrier densities greater than the intrinsic carrier density, the holes can be neglected [17]. In this range, with the additional assumption that $qR_H = 1/(n_\Gamma + n_L)$ at 400 K, the data can be fit using eq (25). The only adjustable parameter in this case is μ_L/μ_Γ, the ratio of the conduction band mobility at L_6 to that at Γ_6. The extracted values for this ratio are reproduced from the data of Nichols et al. in figure 2.4. Using a fit† to these data, shown as the solid line in figure 2.4, and assuming a constant ratio between the Γ_6 conduction band electron mobility and the valence band hole mobility, the temperature dependence of the Hall coefficient can be computed over the entire temperature range. EPROP was used to provide the electron density in the two conduction bands and the hole density in the valence band. The resulting fit to the experimental data is shown in figure 2.5.

The experimental data in figure 2.5 are the same as in figure 2.3. The ordinate is now labeled $1/qR_H$, which is what is actually measured and computed. The computation now predicts the Hall coefficient increase in the range 500 to 700 K. The agreement is not surprising since the temperature dependence of μ_L/μ_Γ was computed to account for this effect. The values of this ratio are reasonable, however, and lend credence to this interpretation [17]. At the highest temperatures, the intrinsic carrier density overestimates the experimental findings, but good agreement with experiment is found for temperatures less than 950 K.

† $\mu_L/\mu_\Gamma = 1.2068 - 1.7590 \times 10^{-3}\ T + 6.7235 \times 10^{-7}\ T^2$.

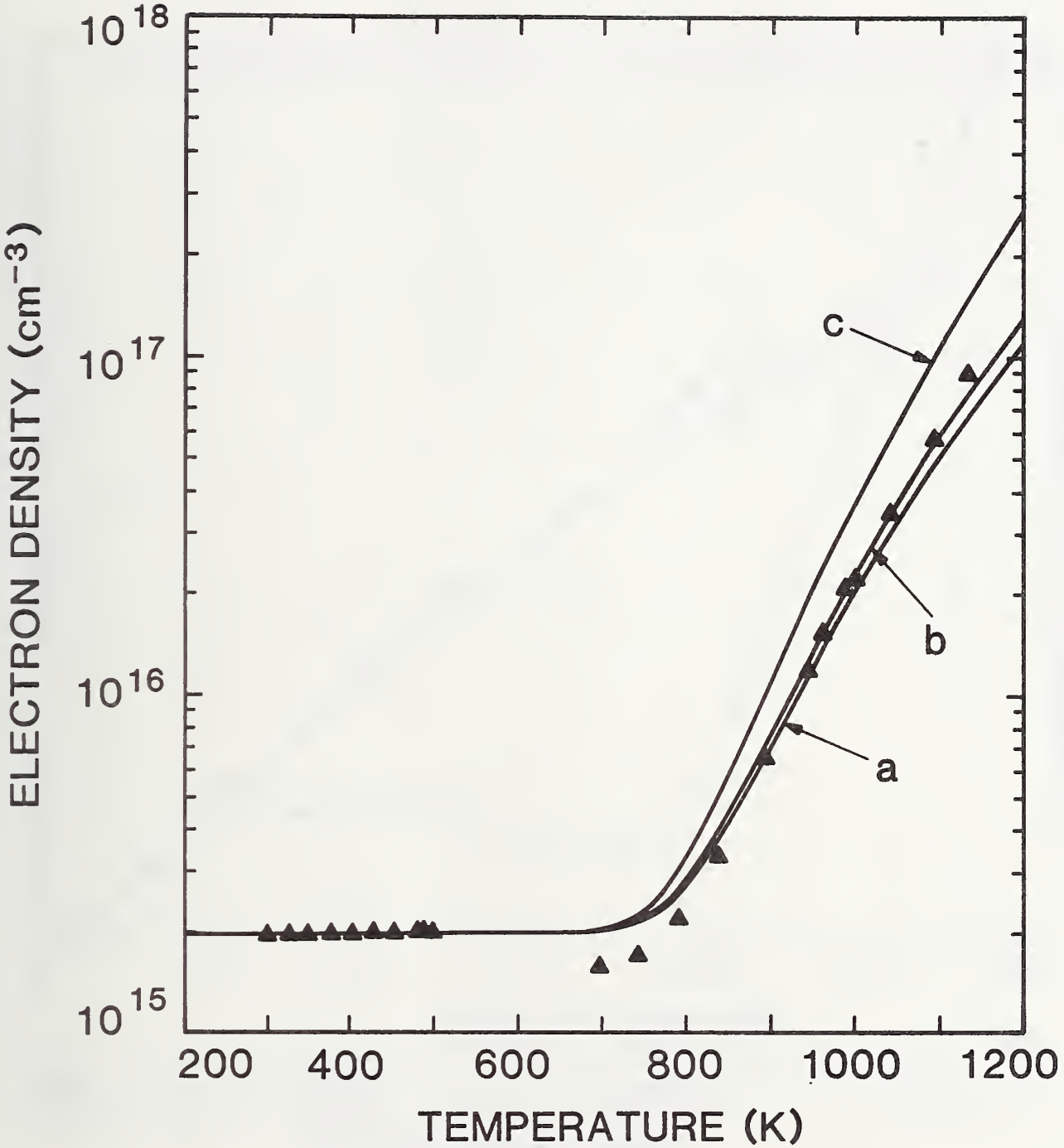

Figure 2.3. Apparent Electron Density in GaAs. Temperature dependence, based on the data (triangles) of reference [17]. Fits were obtained from EPROP using equations for the density of states based on (a) a single, parabolic conduction band, (b) a single, nonparabolic conduction band, and (c) multiple conduction bands.

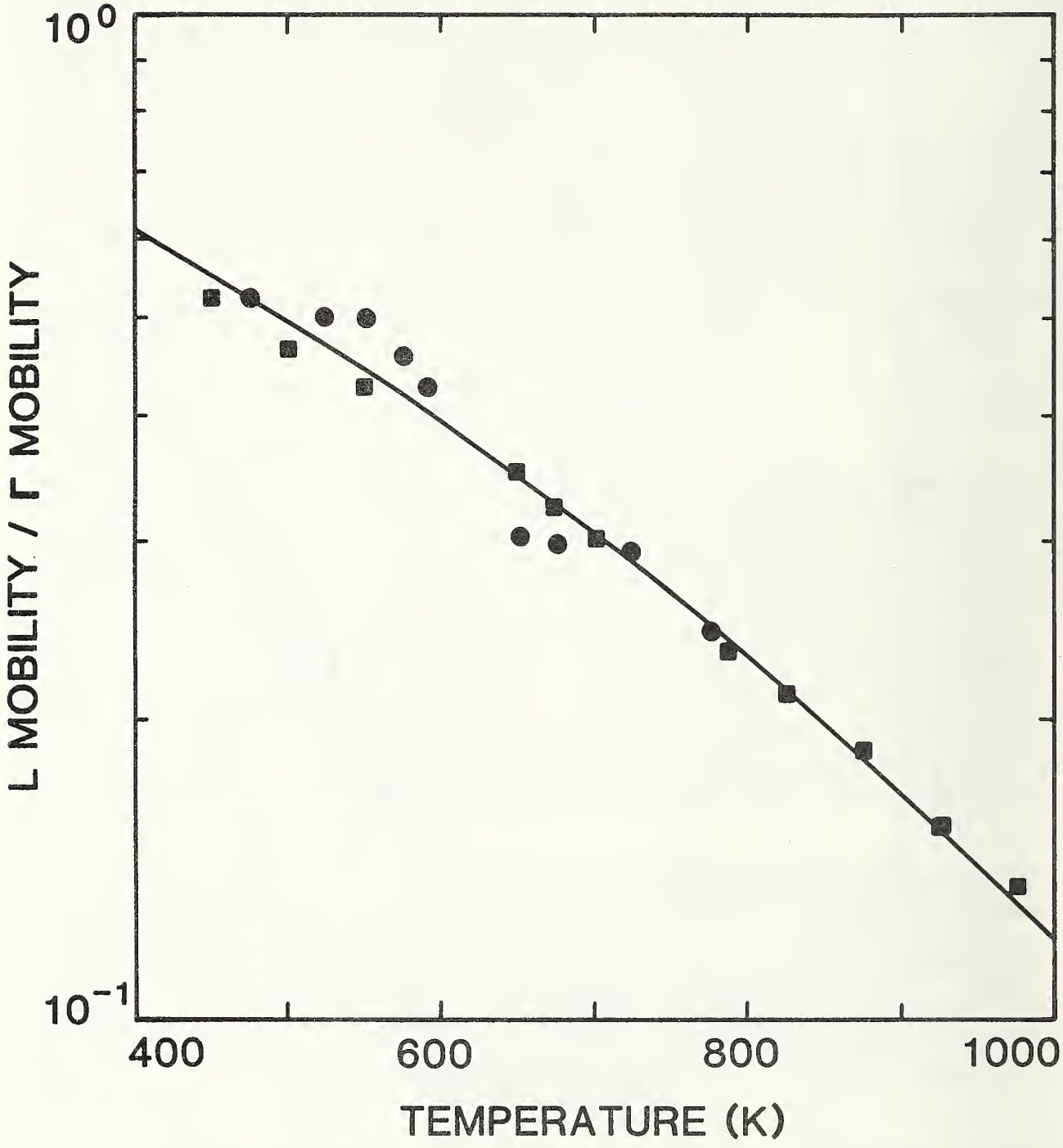

Figure 2.4. Electron Mobility of GaAs. Temperature dependence of the mobility ratio μ_L/μ_Γ [17]; squares and circles are for two different GaAs specimens. Solid line is a power series fit to the data.

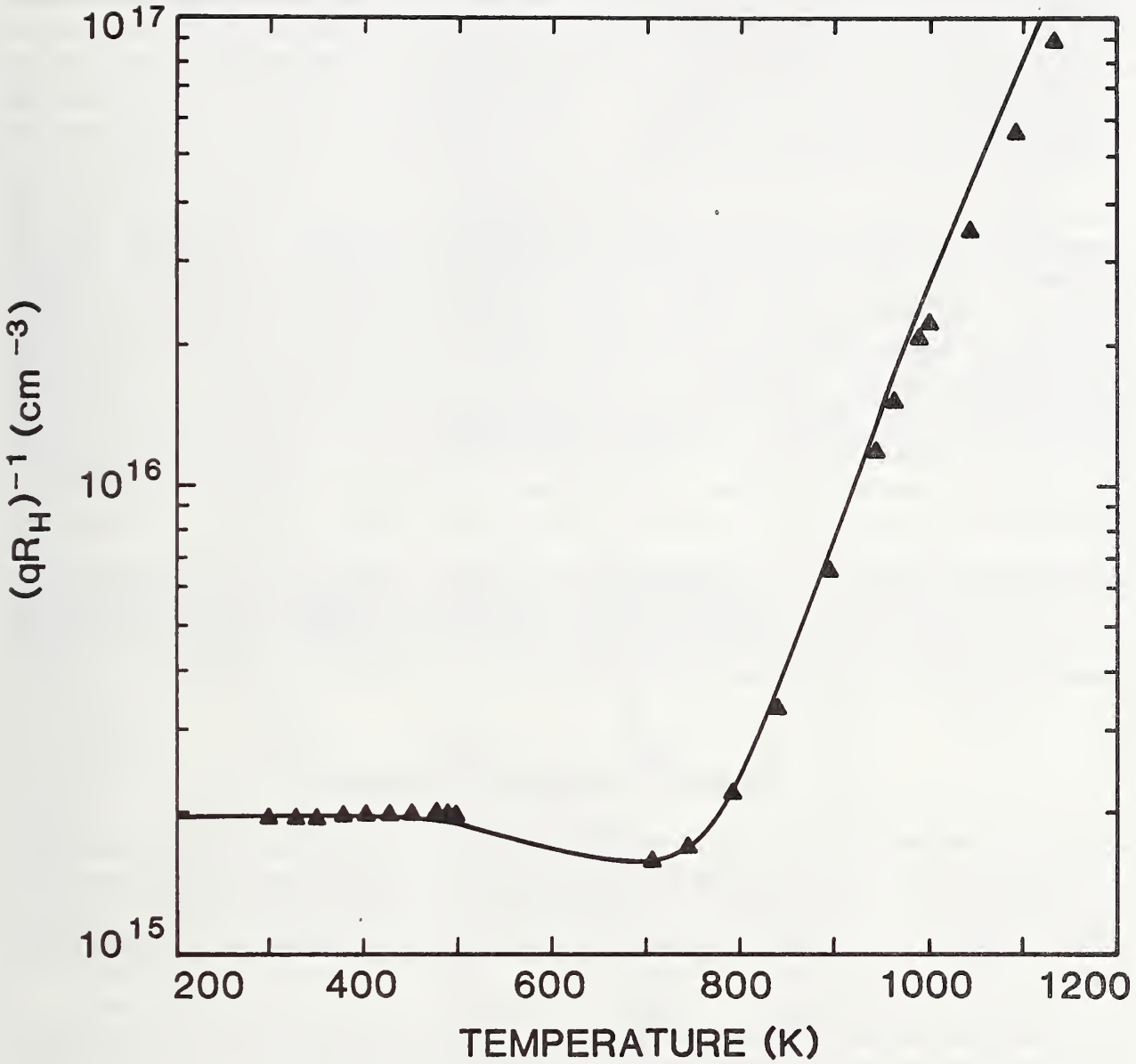

Figure 2.5. Hall Effect in GaAs. Fit to experimental data of Nichols et al. [17] using EPROP to predict occupancy of the conduction bands as a function of temperature.

2.6. Limitations

Changes in the temperature or in the energy or density of the impurity states can cause the argument of the F-D integral to swing from large negative values to large positive ones. It is possible, at certain extremes, to make calculations for situations in which the validity of the model is in question. Important physical intuition can often be gained from pondering these extreme cases, however, so this output is not suppressed. Appropriate warning messages are provided to the user; care must be taken in interpreting the results.

The program is clearly limited by what it does not take into account. For example, no provision is made for divalent or multiply-charged centers, although the statistics of these centers have been formulated [18-23]. Neither does the program include any heavy doping effects [24]. Heavy doping results in a modification of the density of states, decreasing the bandgap and the impurity activation energy. This has been shown to be a significant effect in Si [25-26] for impurity densities greater than 5×10^{17} cm^{-3}. Similarly, for GaAs [27-28], these effects can be significant for acceptor impurity densities exceeding 5×10^{17} cm^{-3} and donor densities exceeding 10^{16} cm^{-3}. Results from EPROP may deviate from experimental data when the impurity or carrier concentrations become large, and appropriate warnings are given.

The numerical accuracy of the calculation is considerably greater than the physical accuracy of the models used. Numerical accuracy is roughly five parts in 10^5 (worst case), as determined by the F-D integral computation. For most computers and compilers, single-precision arithmetic should prove adequate.

3. Program Operation and Output

In this chapter, four different sample sessions are presented. These demonstrate the interactive features of the program and illustrate a few of the many ways that the output can be organized. The user may find it instructive to execute the program while reading this chapter. This is not necessary, however, since both the input and output for each sample session are shown.

The numerical results for GaAs were obtained using eq (17) for the valence band density of states. The final version of the program (see listing in appendix 3) uses eq (19), but the statements required to implement eq (17) are included as comment lines in the source code. If the user wishes to reproduce exactly the numerical results of this chapter (perhaps as a test of modifications to the program), these lines can be substituted for the appropriate active sections of the code.

3.1. Description

To run the program requires only the executable file, but after execution, several other files will be created. One of these is named `EPROP.DAT` and contains all the specifications of the previous run. With `EPROP.DAT` present, the previous input data and plotting attributes are recalled, and the user can make modifications for the next run without wholesale re-entry of every parameter. If the user has selected an output listing file, the output is sent to the file `LIST.DAT`. Similarly, if plot files are specified, these appear as the files `PLOT1.DAT`, `PLOT2.DAT`, `PLOT3.DAT`, and/or `PLOT4.DAT`. When errors occur in the computation, e.g., the solution of the charge balance equation does not achieve the required accuracy, error messages are written to the file `ERROR.LIS`. If messages are sent to this file, the user is notified at the conclusion of the computations. The only other file that the program creates is the file `TEMPOR.ARY`. This file is used to house the data temporarily prior to the final data output. After all the computations are completed, data are selected from `TEMPOR.ARY` and written to the selected output devices according to the output option then in effect. Upon normal termination of the program, the file `TEMPOR.ARY` is deleted and is therefore not visible to the user.

EPROP allows almost any arrangement of the output data. Three internally defined output sets are available. These output options are called **`short set`**, **`full set`**, and **`constants`**. The **`short set`** lists 10 parameters as shown below in figure 3.1. Figures 3.2 and 3.3 show the output parameters under the full set option for GaAs and Si, respectively.

Temp	EF	n	p	Nd1+	Na1-
Nd2+	Na2-	Nd3+	Na3-		

Figure 3.1. Short Set Output Parameters (Si and GaAs).

Temp	1000/T	EF	n	p	Nd1+
Nd2+	Nd3+	Na1-	Na2-	Na3-	Eg
Sum Charges	kT	Ec-EF	diel	Temp**3/2	efme
efmh	Nc	Nv	Nc'	Nv'	nG
nL	nX	Delta EGL	Delta EGX		

Figure 3.2. Full Set Output Parameters (GaAs).

The **`constants`** output option also differs slightly depending on whether the computation is being made for GaAs or Si. These two output sets are listed in figures 3.4 and 3.5 for GaAs and Si, respectively.

A fourth option, the **`user-specified`** set, allows the user to designate any of the output parameters for listing or display. For GaAs, up to 28 parameters can be selected for output; in the case of Si, 21 parameters are available.

Temp	1000/T	EF	n	p	Nd1+
Nd2+	Nd3+	Na1-	Na2-	Na3-	Eg
Sum Charges	kT	Ec-EF	diel	Temp**3/2	efme
efmh	Nc	Nv			

Figure 3.3. Full Set Output Parameters (Si).

Temp	1000/T	kT	Eg	diel	Temp**3/2
efme	efmh	Nc	Nv	Delta EGL	Delta EGX

Figure 3.4. Constants Option (GaAs).

Temp	1000/T	kT	Eg	diel	Temp**3/2
efme	efmh	Nc	Nv		

Figure 3.5. Constants Option (Si).

Correspondences between the output parameter names used in figures 3.1-3.5, the variables appearing in the equations of chapter 2, and the names used in the FORTRAN program (appendix 3) can be obtained from tables 3.1-3.

Table 3.1. Symbols (Band-Structure Parameters)

Equation Symbol	Eq Nos.	Output Symbol	Program Variable
$\overline{m}_c/m_0$	3	efme	EFME
$\overline{m}_v/m_0$	6	efmh	EFMH
N_c'	10	Nc1	FNCP
N_v'	17,19	Nv1	FNVP
n_Γ	12	nG	CNG
n_L	13	nL	CNL
n_x	14	nX	CNX
α	11		ALPHA
β	18		BETA
Δ	11		DEL
$\Delta_{\Gamma L}$	15	Delta EGL	DELGL
$\Delta_{\Gamma X}$	16	Delta EGX	DELGX

The output can be directed to any combination of three different destinations: terminal (or printer), listing file, and plot files. If terminal output is selected, the first six parameters of the output set and the input information about the impurities are sent to the terminal.

Table 3.2. Symbols (Dopant Parameters)

Equation Symbol	Eq Nos.	Output Symbol	Program Variable
n_0	2	n	CN
p_0	5	p	CP
N_{a_1}	7	Na1	DN3
N_{a_2}	7	Na2	DN4
N_{a_3}	7	Na3	DN5
N_{d_1}	8	Nd1	DN1
N_{d_2}	8	Nd2	DN2
N_{d_3}	8	Nd3	DN3
$N_{a_1}^-$	7	Na1-	DI3
$N_{a_2}^-$	7	Na2-	DI4
$N_{a_3}^-$	7	Na3-	DI5
$N_{d_1}^+$	8	Nd1+	DI1
$N_{d_2}^+$	8	Nd2+	DI2
$N_{d_3}^+$	8	Nd3+	DI3
$E_c - E_{d_1}$	8	Ed1	EN1
$E_c - E_{d_2}$	8	Ed2	EN2
$E_c - E_{d_3}$	8	Ed3	EN3
$E_{a_1} - E_v$	7	Ea1	EN4
$E_{a_2} - E_v$	7	Ea2	EN5
$E_{a_3} - E_v$	7	Ea3	EN6
N_c	2,3	Nc	FNC
N_v	5,6	Nv	FNV
(see equation)	1	Sum Charges	SUM

Table 3.3. Symbols (Miscellaneous Parameters)

Equation Symbol	Eq Nos.	Output Symbol	Program Variable
F_j	4		FD(J,ARG)
η			ETA
ζ			ZETA
T		Temp	T
kT		kT/q	CAYT
$E_c - E_F$		Ec-EF	ECMEF
$T^{3/2}$		Temp**3/2	TPOWR
$E_F - E_v$		EF	EF
$E_c - E_F$		Ec-EF	ECMEF
h			H

The listing file, if selected, will contain a complete listing of the output set as defined above, along with the impurity information. If plot files are desired, the user can specify up to four pairs of parameters to be tabulated in separate files.

The file EPROP.DAT, which contains the attributes entered during the previous program execution, is normally present and read by the main program. If EPROP.DAT does not exist, it is created through a default start-up sequence. Upon execution, EPROP attempts to read EPROP.DAT. If it is not present, the program presents the start-up header and instructions shown in figure 3.6. If a carriage return is given, a menu allowing input of impurity parameters will appear (fig. 3.7). The six slots for impurity information are listed by type: three donors, D1, D2, and D3, and three acceptors, A1, A2, and A3. The names of the impurities would normally appear under the Dopants header. The word none indicates that no impurities have been defined.

```
                    EPROP   Version 1.0

          An Interactive Program for Computing  the Electrical
                        Properties of GaAs and Si

                                by
          John J. Mathias, Alan C. Seabaugh, Michael I. Bell
                    Semiconductor Electronics Division
                National Institute of Standards and Technology
                        Gaithersburg, MD 20899
                                MCMXC

      Instructions: You will be asked to respond to a
series of menus. If an input is incorrectly entered, do
not abort the program; you will have an opportunity to
make changes. The file, EPROP.DAT (normally containing
parameters from the previous run) is not present. Press
<RETURN> to create a new input file or enter "Q" to quit. >
```

Figure 3.6. EPROP Start-Up Menu.

At this point, the user may enter impurity information by entering one of the six Type designators or, if a computation for the intrinsic (no impurity) case is desired, the return key can be pressed to exit the menu. All of the menus, except the Main Menu, can be exited by pressing a carriage return.

For now, assume that the computation is to be made for the intrinsic case. Entering a carriage return causes the Independent Variable Selection Menu to appear (fig. 3.8). Upon start-up with no EPROP.DAT file, the independent variable defaults to temperature with its initial value (Start) and final value (Stop) set to room temperature (300 K); the

```
                        Impurity Selection
Type    Dopants     Density(cm-3)  Energy(eV)  Degeneracy
----    -------     -------------  ---------   ----------
 D1     none
 D2     none
 D3     none
 A1     none
 A2     none
 A3     none
 Enter type >
```

Figure 3.7. Impurity Selection Menu.

incremental step (**Step**) is set to 1.0 (the step cannot be zero). Assume this is satisfactory, i.e., only a single computation, at room temperature, is required. Pressing the return key causes the **Main Menu** to be displayed. The **Main Menu** contains all the input information and the desired output attributes (fig. 3.9). At the top of the menu, **Material** indicates that the computation is to be made for GaAs (the default start-up selection). The second item directs that the output be sent to the user's **terminal** only and the third item tells which output option is selected. Here, the **short set** option is listed. Under item 4, the independent variable and its previously defined attributes are listed. Under the **Dopants** header, the label none is displayed to indicate that no dopants are specified.

```
          Independent Variable Selection Menu

          Independent variable >   Temp
                         Start >     300.0000
                         Stop  >     300.0000
                         Step  >       1.0000

               1   New independent variable

               2   New starting value

               3   New final value

               4   New step (log or linear)

                   Enter number(s) >
```

Figure 3.8. Independent Variable Selection Menu.

```
                        Main Menu
   1  Material      >  GaAs
   2  Output to     >  terminal only
   3  Output        >  short set
   4  Input data    :
      Independent variable >    Temp
                     Start >     300.0000
                     Stop  >     300.0000
                     Step  >       1.0000

      Dopants        Density(cm-3)      Energy(eV)       Degeneracy
      ----------     -------------      ----------       ----------
      none

   5  Execute
   6  Exit                        Enter number >
```

Figure 3.9. Main Menu.

To make changes to any of the data or output attributes, enter the number corresponding to the item which should be changed. A new menu will appear which can be manipulated in the same way or exited to return to the **Main Menu**. If the number **5** is entered without making changes, the **short set** output to the terminal, a single room temperature computation, is displayed (fig. 3.10). The ionized donor density **Nd1+** and the ionized acceptor density **Na1-** are listed as equal to zero when they are undefined. At this point, the user has enough of an introduction to be able to experiment with the program. The user directs the program flow and can return to any point as often as necessary before beginning the computation. The next four sections detail the manipulation of the menus.

```
Electrical Properties of Gallium Arsenide

Dopants        Density(cm-3)     Energy(eV)     Degeneracy
----------     -------------     ----------     ----------
none

      Temp           EF          n           p          Nd1+         Na1-
    300.0000      0.751551  2.2582E+06  2.2582E+06  0.0000E+00  0.0000E+00
            --------------  Computation Complete  --------------
FORTRAN STOP
```

Figure 3.10. Default Start-Up Computation.

3.2. Sample Session – Short Set

This section describes the **short set** output option. In addition, the plot file output is demonstrated. As an example, we examine the Hall-effect data and computer fit of Ta et al. [29] on undoped p-type GaAs.

Figure 3.11 reproduces the Hall-effect data of reference [29], showing the temperature dependence of the hole density in an undoped, liquid-encapsulated-Czochralski (LEC) GaAs crystal. Ta et al. reported a fit to the data as listed in the figure. As is often the case, due perhaps to the large number of parameters required to describe the density of states, effective masses, bandgap, and impurity energies and degeneracies, the model used was not completely specified. It will be shown that for the model embodied in EPROP, the fitted parameters are sensitive to the values of the model parameters used. This leads to an uncertainty in the fitting parameters greater than the statistical uncertainty of the experiment. Hence, to compare the results of these fits with those made at other laboratories requires that the impurity parameters be extracted using the same program (i.e., the same model and input parameters). EPROP can serve as this standard for data comparison. It can also be useful in testing the sensitivity of a fit to changes in the model parameters.

Upon running the program again, the main menu appears with the data and attributes of the previous run (fig. 3.9). Suppose a plot file is desired for input to a graphics package. Entering the number 2 brings up the **Output Device Selection Menu** (fig. 3.12), which lists the possible output destinations. This menu shows that the present selection, output to the terminal, is on with all other outputs off. The listing file option directs the data to an output file, `LIST.DAT`, which the user can then print, edit, or archive as desired. Entering the number or numbers with or without separators (spaces or commas) will enable or disable the corresponding output.

When the plot file option is selected, the entire set of possible plot parameters is listed (fig. 3.13). The prompt asks for the x-axis input which is specified by the number corresponding to the desired x-axis parameter in the menu. To plot hole density p as a function of $1000/T$, type 2 to select $1000/T$ and enter a carriage return; then enter 5 to select hole density, and press return. The **Plot File Menu** appears, allowing the specification of additional plot files (fig. 3.14). If a single plot file is all that is needed, press the return key. The **Main Menu** is then displayed (lower half of fig. 3.14) with the selected plot files listed under the **Output to** selection. The **Main Menu** always gives a complete description of the output attributes.

It should be noted that while the program checks the user's input for everything that might prove fatal to the computation, it does not check for every logical inconsistency. For example, if the user were to select the degeneracy of acceptor 3 (number 39, **Deg. A3**, the degeneracy of **A3**) for output to a plot file (fig. 3.13), the program would not check to make certain that acceptor 3 is defined. This is because the user could enter the impurity data at any time before computation. Errors which can be detected unambiguously are reported, however, as when the user tries to plot a variable against itself. No entry is truly

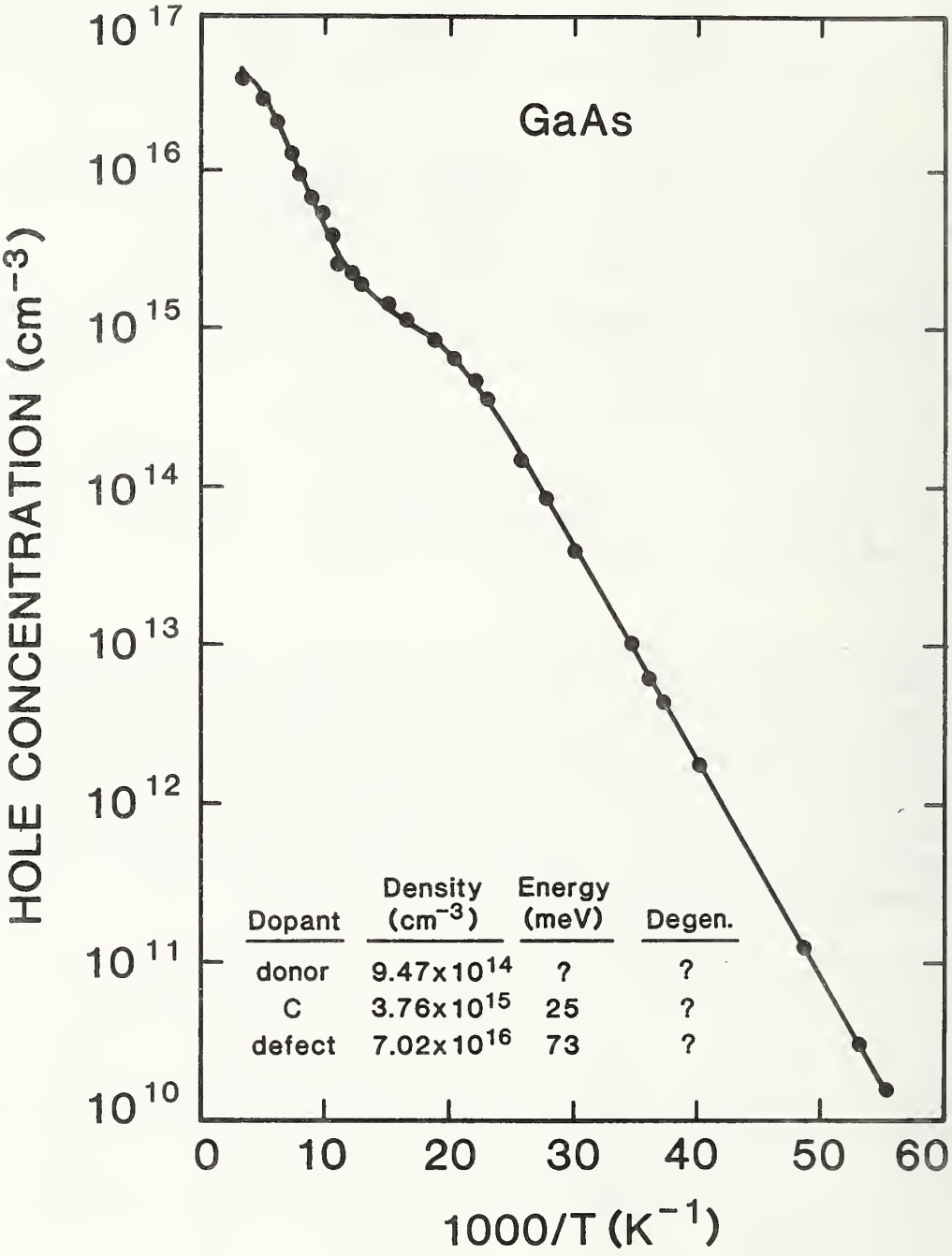

Figure 3.11. Hole Density in GaAs (Experimental). Hall-effect measurements of Ta et al. on an LEC GaAs wafer. The numbers reported in the table are the energies and densities of the impurities used to obtain the fit (solid line); the energy of the donor and the three degeneracies were not reported.

```
Output Device Selection Menu

1   Terminal      >       on

2   Listing file  >       off

3   Plot file(s)  >       off

To toggle, enter number(s) > 3
```

Figure 3.12. Output Device Selection Menu.

```
                              Plot Parameter Menu
 1>     Temp      2>     1000/T    3>        EF     4>       n       5>       p
 6>      Nd1+     7>      Nd2+     8>      Nd3+     9>      Na1-    10>      Na2-
11>      Na3-    12>       Eg     13> Sum Charges14>      kT      15>     Ec-EF
16>     diel     17>  Temp**3/2   18>     efme     19>     efmh    20>      Nc
21>       Nv     22>      Nd1     23>      Ed1     24>   Deg. D1   25>     Nd2
26>      Ed2     27>   Deg. D2    28>      Nd3     29>      Ed3    30>   Deg. D3
31>      Na1     32>      Ea1     33>   Deg. A1    34>      Na2    35>      Ea2
36>   Deg. A2    37>      Na3     38>      Ea3     39>   Deg. A3   40>      Nc'
41>      Nv'     42>       nG     43>       nL     44>       nX    45>  Delta EGL
46>  Delta EGX
Enter parameter number for the X-axis > 2
Enter parameter number for the Y-axis > 5
```

Figure 3.13. Plot Parameter Menu.

fatal, since all attributes are saved (in `EPROP.DAT`) when the program is executed, and these attributes are recalled upon re-running the program. In this way, the user can easily recover from errors.

Next, in order to fit the data of figure 3.11, the independent variable and its range must be changed and the dopant parameters must be entered. To make a change to the input data, enter 4. With this, the **Input Data Selection Menu** appears (fig. 3.15). Select 1 to change the independent variable and a new menu appears (fig. 3.16). Above the selections, the present independent variable and its range are listed. Entering 1 brings up the **New Independent Variable Selection Menu**, which lists the possible independent variables (fig. 3.17). To fit the present data, the independent variable must be $1000/T$, so from the table in figure 3.17, 2 is entered. With a change in the independent variable, the starting value, stopping value, and the step increment are all requested. In addition, the user can

```
                          Plot File Menu
                 1         p      vs.    1000/T
                 2          not selected
                 3          not selected
                 4          not selected
Enter number >

                              Main Menu
    1  Material      >  GaAs
    2  Output to     >  terminal and plot file(s)
       Plot file(s) :
              p      vs.    1000/T
    3  Output        >  short set
    4  Input data    :
       Independent variable >    Temp
                      Start >   300.0000
                      Stop  >   300.0000
                      Step  >     1.0000
       Dopants        Density(cm-3)     Energy(eV)      Degeneracy
       ----------     -------------     ----------      ----------
       none
    5  Execute
    6  Exit                      Enter number > 4
```

Figure 3.14. Plot File Menu.

```
                      Input Data Selection Menu

    1  Independent variable >    Temp
                      Start >     300.0000
                      Stop  >     300.0000
                      Step  >       1.0000
    2  Impurity data:

       Dopants        Density(cm-3)     Energy(eV)      Degeneracy
       -------        -------------     ----------      ----------
       none

       Enter number > 1
```

Figure 3.15. Input Data Selection Menu.

change between linear and logarithmic stepping of the independent variable.

```
                Independent Variable Selection Menu

Independent variable >    Temp
                Start >     300.0000
                Stop  >     300.0000
                Step  >       1.0000


     1   New independent variable

     2   New starting value

     3   New final value

     4   New step (log or linear)

         Enter number(s) > 1
```

Figure 3.16. Independent Variable Selection Menu.

```
                New Independent Variable Selection
                      1   Temperature (K)
                      2   1000/Temp
  Impurity        Density(cm-3)       Energy(eV)          Degeneracy
 ----------       -------------       ----------        ------------
Not Selected            3                 4                 5
Not Selected            6                 7                 8
Not Selected            9                10                11
Not Selected           12                13                14
Not Selected           15                16                17
Not Selected           18                19                20
                Enter number > 2
 Enter the starting value > 4
 Enter the final value > 56
 Do you want to change to log scale ? (y/n) > n
 Enter the step value  > 4
```

Figure 3.17. New Independent Variable Selection Menu.

With this completed as shown, the user is returned to the **Input Data Selection Menu** (fig. 3.18). To enter the impurity parameters, select 2 and choose the impurity type from the **Impurity Selection** shown previously in figure 3.7. The choices are, for donors, **D1**, **D2**, and **D3**, and for acceptors, **A1**, **A2**, and **A3**. Here, **d1** is entered, signifying that

parameters for a donor are to be entered. The Impurity Data Entry prompts are issued as shown in figure 3.19.

```
                    Input Data Selection Menu

  1  Independent variable >    1000/T
                     Start >       4.0000
                     Stop  >      56.0000
                     Step  >       4.0000
  2  Impurity data:

     Dopants        Density(cm-3)     Energy(eV)      Degeneracy
     -------        -------------     ----------      ----------
     none

     Enter number > 2
```

Figure 3.18. Input Data Selection Menu.

```
                          Impurity Data Entry
                                    123456789012
Enter the name (up to 12 characters) > Silicon
Enter the density > 9.47e14
Enter the energy > .006
Enter the degeneracy > 2
                          Impurity Selection
   Type    Dopants     Density(cm-3)  Energy(eV)  Degeneracy
   ----    -------     -------------  ---------   ----------
    D1     Silicon       9.470E+14     0.0060       2.0000
    D2     none
    D3     none
    A1     none
    A2     none
    A3     none
    Enter type > a1
```

Figure 3.19. Impurity Selection Inputs.

Following the instructions, the parameters listed in figure 3.11 can be entered. The impurity energy requested is the difference between the conduction band and the donor energy level or between the acceptor energy and the valence band. The **Impurity Selection**

table is redrawn (lower half of fig. 3.19), and the parameters of the next dopant can be entered. Upon completion of the data entry for all three impurities, the `Impurity Selection` menu should look like figure 3.20, and pressing return brings back the `Input Data Selection Menu` (fig. 3.21).

```
                        Impurity Selection
Type    Dopants     Density(cm-3)  Energy(eV)  Degeneracy
----    -------     -------------  ---------   ----------
 D1     Silicon       9.470E+14      0.0060       2.0000
 D2     none
 D3     none
 A1     Carbon        3.760E+15      0.0250       4.0000
 A2     defect        7.020E+16      0.0730       4.0000
 A3     none
 Enter type >
```

Figure 3.20. Completed Dopant Menu.

```
                    Input Data Selection Menu

  1  Independent variable >    1000/T
                     Start >      4.0000
                     Stop  >     56.0000
                     Step  >      4.0000
  2  Impurity data:

     Dopants        Density(cm-3)      Energy(eV)      Degeneracy
     -------        -------------      ----------      ----------
     Silicon          9.4700E+14          0.0060          2.0
     Carbon           3.7600E+15          0.0250          4.0
     Defect           7.0200E+16          0.0730          4.0

     Enter number >
```

Figure 3.21. Input Data Selection Menu.

Enter a carriage return to bring up the `Main Menu`, followed by 5 to execute. The terminal output is shown in figure 3.22, and the plot file output (`PLOT1.DAT`) in figure 3.23.

With the aid of a graphics program, this plot file can be compared with the data, as shown in figure 3.24. It is immediately apparent that the fit obtained using EPROP and the parameters reported by Ta et al. (dashed line) is not as tight as the fit shown in

```
Electrical Properties of Gallium Arsenide

Dopants        Density(cm-3)     Energy(eV)     Degeneracy
---------      -------------     ----------     ----------
Silicon          9.470E+14         0.0060         2.0000
Carbon           3.760E+15         0.0250         4.0000
Defect           7.020E+16         0.0730         4.0000

   1000/T          EF          n            p          Nd1+         Na1-
   4.0000       0.110151   4.0681E-10   4.3524E+16   9.4700E+14   3.4918E+15
   8.0000       0.061920   0.0000E+00   8.1402E+15   9.4700E+14   3.3278E+15
  12.0000       0.045441   0.0000E+00   2.4806E+15   9.4700E+14   3.0515E+15
  16.0000       0.034941   0.0000E+00   1.3726E+15   9.4700E+14   2.3045E+15
  20.0000       0.029691   0.0000E+00   6.5600E+14   9.4700E+14   1.6022E+15
  24.0000       0.027239   0.0000E+00   2.4893E+14   9.4700E+14   1.1959E+15
  28.0000       0.026242   0.0000E+00   7.7113E+13   9.4700E+14   1.0241E+15
  32.0000       0.025882   0.0000E+00   2.1340E+13   9.4700E+14   9.6833E+14
  36.0000       0.025732   0.0000E+00   5.7264E+12   9.4700E+14   9.5275E+14
  40.0000       0.025646   0.0000E+00   1.5410E+12   9.4700E+14   9.4852E+14
  44.0000       0.025584   0.0000E+00   4.1915E+11   9.4700E+14   9.4741E+14
  48.0000       0.025535   0.0000E+00   1.1531E+11   9.4700E+14   9.4713E+14
  52.0000       0.025493   0.0000E+00   3.2045E+10   9.4700E+14   9.4703E+14
  56.0000       0.025458   0.0000E+00   8.9819E+09   9.4700E+14   9.4719E+14
          --------------  Computation Complete  --------------
FORTRAN STOP
```

Figure 3.22. Terminal Output.

figure 3.11. This could be due to differences in the temperature-dependent quantities (bandgaps, densities of states, and effective masses) and illustrates the fact that when the model is incompletely specified, the impurity parameters have a greater degree of uncertainty than the statistical deviations in the fit. Differences can also arise from inaccuracy in calculating the Fermi-Dirac integral, although this is unlikely in cases such as this where the Fermi level is not close to either band edge. In situations where it is of interest to make accurate comparisons, EPROP can be used as a benchmark to compare published impurity parameters against reported data using known temperature-dependent constants.

A closer fit can be obtained by using EPROP to vary the impurity parameters at a single temperature. For example, since the slope of the data at low temperatures is due to the thermal activation of the carbon impurity, the carbon activation energy was varied in a single EPROP computation to fit the lowest temperature data point. Similarly, since the slope of the line in the 100- to 200-K range is controlled by the defect, its activation energy can be varied to fit a data point in this range, and so on. By making informed guesses, the impurity parameters listed in the upper part of figure 3.24 were obtained. The fit which

```
1000/T       p
 4.0000  4.3524E+16
 8.0000  8.1402E+15
12.0000  2.4806E+15
16.0000  1.3726E+15
20.0000  6.5600E+14
24.0000  2.4893E+14
28.0000  7.7113E+13
32.0000  2.1340E+13
36.0000  5.7264E+12
40.0000  1.5410E+12
44.0000  4.1915E+11
48.0000  1.1531E+11
52.0000  3.2045E+10
56.0000  8.9819E+09
```

Figure 3.23. PLOT1.DAT Output File Format.

resulted is shown as the solid line. This does not represent an optimum solution, but the fit is visibly improved. These changes amount to a decrease in all the reported impurity parameters (carbon density −15%, defect density −37%, carbon activation energy −4%, and defect activation energy −14%) and show the extent to which the impurity parameters can be sensitive to the model and temperature-dependent parameters used.

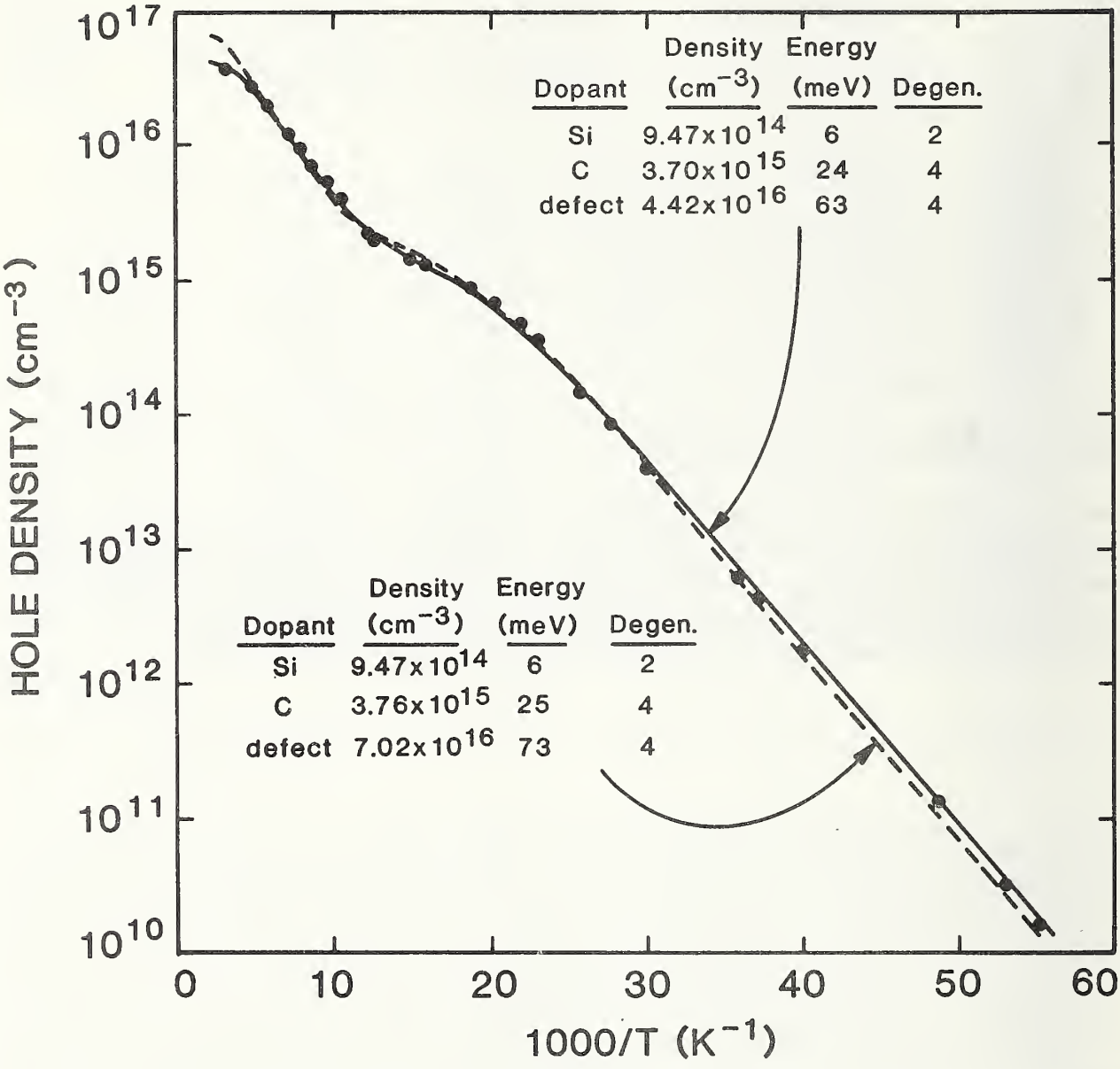

Figure 3.24. Hole Density in GaAs (Calculated). Comparison of published fit (dashed curve) with modified fit (solid curves). Both curves were computed using EPROP.

3.3. Sample Session – Full Set

As an example of the `full set` option, the impurity parameters used by Larrabee et al. [1] in their electrical properties program are reproduced. They computed the temperature dependence for closely compensated Si doped with phosphorus and boron. To change to the `full set` option, EPROP is run, bringing up the **Main Menu** obtained during the previous run, figure 3.25. To change the material to Si, simply enter 1 and the **Material Selection** table is presented, figure 3.26. (This menu has been provided in case other materials are incorporated.)

```
                              Main Menu
1  Material      >  GaAs
2  Output to     >  terminal and plot file(s)
   Plot file(s)  :
           p       vs.    1000/T
3  Output        >  short set
4  Input data    :
   Independent variable >    1000/T
                  Start >       4.0000
                  Final >      56.0000
                  Step  >       4.0000

   Dopants        Density(cm-3)     Energy(eV)      Degeneracy
   ----------     -------------     ----------      ----------
   Silicon          9.4700E+14        0.0060          2.0
   Carbon           3.7600E+15        0.0250          4.0
   Defect           7.0200E+16        0.0730          4.0

5  Execute
6  Exit                      Enter number > 1
```

Figure 3.25. Main Menu.

```
                        Material Selection

                        S   Silicon

                        G   Gallium Arsenide

                        Enter letter > s
```

Figure 3.26. Material Selection Menu.

The Main Menu is then re-displayed with the Material entry now indicating Si. The output destinations can be modified to eliminate the plot file by entering 2 to bring up the Output Device Selection Menu, figure 3.27. If the number 123 is entered, all three output devices are changed. Here, the terminal and plot file output are turned off, and the listing file is turned on. Next, from the Main Menu, the number 3 is input to choose the output set. The Output Set Selection Menu is then displayed as shown in figure 3.28. In this example, the number 2 for the full set option is chosen.

```
                Output Device Selection Menu

1   Terminal      >        on

2   Listing file  >        off

3   Plot file(s)  >        on

4   Change plot file(s)

To toggle, enter number(s) > 123
```

Figure 3.27. Output Device Selection Menu.

```
                Output Set Selection Menu

                    1   short set

                    2   full set

                    3   constants

                    4   user specified

          The current selection is short set

                    Enter number > 2
```

Figure 3.28. Output Set Selection Menu.

As in the previous example, the input data for the impurity parameters and the independent variables can be entered by first selecting 4 from the Main Menu and then working through the menus described in the previous section. The impurity configuration to be computed is described by the resulting Main Menu, figure 3.29.

```
                              Main Menu
1  Material      >  Si
2  Output to     >  listing file only
3  Output        >  full set
4  Input data    :
   Independent variable >    1000/T
                  Start >      50.0000
                  Final >       2.0000
                  Step  >      -2.0000

   Dopants        Density(cm-3)      Energy(eV)       Degeneracy
   ----------     -------------      ----------       ----------
   Phosphorus       1.0000E+15         0.0286           2.0
   Boron            1.0000E+13         0.0090           4.0

5  Execute
6  Exit                        Enter number >
```

Figure 3.29. Si Main Menu for Full Set Output Computation.

The `full set` output data are listed in figure 3.30. Comparison of the output with the results of reference [1] reveals some differences. These are due to the different approximations used for the Fermi-Dirac integral. The program of reference [1] uses the approximations recommended by Blakemore [2], which are accurate to roughly 3%, while EPROP uses approximations with an accuracy of better than 5 parts in 10^5.

Electrical Properties of Silicon

Dopants	Density(cm-3)	Energy(eV)	Degeneracy
Phosphorus	1.000E+15	0.0286	2.0000
Boron	1.000E+13	0.0090	4.0000

1000/T	Temp	EF	n	p	Nd1+
50.0000	20.0000	1.133781	1.2857E+12	0.0000E+00	1.1287E+13
48.0000	20.8333	1.133878	2.4112E+12	0.0000E+00	1.2414E+13
46.0000	21.7391	1.133892	4.3201E+12	0.0000E+00	1.4317E+13
44.0000	22.7273	1.133804	7.3694E+12	0.0000E+00	1.7372E+13
42.0000	23.8095	1.133611	1.2043E+13	0.0000E+00	2.2035E+13
40.0000	25.0000	1.133311	1.8986E+13	0.0000E+00	2.9005E+13
38.0000	26.3158	1.132911	2.9179E+13	0.0000E+00	3.9179E+13
36.0000	27.7778	1.132405	4.3940E+13	0.0000E+00	5.3970E+13
34.0000	29.4118	1.131787	6.5223E+13	0.0000E+00	7.5231E+13
32.0000	31.2500	1.131038	9.5592E+13	0.0000E+00	1.0564E+14
30.0000	33.3333	1.130133	1.3855E+14	0.0000E+00	1.4854E+14
28.0000	35.7143	1.129030	1.9815E+14	0.0000E+00	2.0817E+14
26.0000	38.4615	1.127665	2.7882E+14	0.0000E+00	2.8870E+14
24.0000	41.6667	1.125940	3.8301E+14	0.0000E+00	3.9310E+14
22.0000	45.4545	1.123710	5.0917E+14	0.0000E+00	5.1917E+14
20.0000	50.0000	1.120752	6.4633E+14	0.0000E+00	6.5629E+14
18.0000	55.5556	1.116743	7.7420E+14	0.0000E+00	7.8425E+14
16.0000	62.5000	1.111235	8.7237E+14	0.0000E+00	8.8238E+14
14.0000	71.4286	1.103585	9.3371E+14	0.0000E+00	9.4355E+14
12.0000	83.3333	1.092734	9.6552E+14	0.0000E+00	9.7552E+14
10.0000	100.0000	1.076687	9.8023E+14	0.0000E+00	9.9022E+14
8.0000	125.0000	1.051182	9.8642E+14	0.0000E+00	9.9641E+14
6.0000	166.6667	1.005619	9.8879E+14	2.4657E-12	9.9883E+14
4.0000	250.0000	0.905928	9.8968E+14	7.1978E+00	9.9969E+14
2.0000	500.0000	0.573313	1.0659E+15	7.5932E+13	9.9995E+14

Nd2+	Nd3+	Na1-	Na2-	Na3-	Eg
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155867
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155866
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155863
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155858
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155851
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155841
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155827
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155809
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155784
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155752
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155709
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155651
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155574
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155469
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155326
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.155125
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.154840
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.154422
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.153786
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.152773
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.151053
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.147862
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.141098
0.0000E+00	0.0000E+00	1.0000E+13	0.0000E+00	0.0000E+00	1.123330
0.0000E+00	0.0000E+00	9.9999E+12	0.0000E+00	0.0000E+00	1.054783

Sum Charges	kT	Ec-EF	diel	Temp**3/2	efme
-1.4869E+09	1.7235E-03	0.022086	11.4472	8.9443E+01	1.0619E+00
-2.4526E+09	1.7953E-03	0.021987	11.4480	9.5091E+01	1.0620E+00
3.3775E+09	1.8733E-03	0.021971	11.4488	1.0136E+02	1.0621E+00
-2.4264E+09	1.9585E-03	0.022054	11.4497	1.0835E+02	1.0622E+00
7.7951E+09	2.0517E-03	0.022240	11.4506	1.1618E+02	1.0623E+00
-1.8587E+10	2.1543E-03	0.022530	11.4517	1.2500E+02	1.0624E+00
5.9559E+08	2.2677E-03	0.022916	11.4529	1.3500E+02	1.0626E+00

```
-3.0291E+10  2.3937E-03    0.023404   11.4542    1.4640E+02  1.0628E+00
-8.6822E+09  2.5345E-03    0.023998   11.4557    1.5951E+02  1.0631E+00
-4.3814E+10  2.6929E-03    0.024714   11.4573    1.7469E+02  1.0634E+00
 1.2382E+10  2.8724E-03    0.025575   11.4592    1.9245E+02  1.0638E+00
-1.7952E+10  3.0776E-03    0.026622   11.4613    2.1343E+02  1.0642E+00
 1.2516E+11  3.3143E-03    0.027909   11.4637    2.3853E+02  1.0649E+00
-9.4925E+10  3.5905E-03    0.029529   11.4666    2.6896E+02  1.0656E+00
-5.3687E+08  3.9170E-03    0.031615   11.4700    3.0645E+02  1.0667E+00
 3.4226E+10  4.3086E-03    0.034373   11.4741    3.5355E+02  1.0680E+00
-5.3419E+10  4.7874E-03    0.038097   11.4790    4.1409E+02  1.0698E+00
-1.8522E+10  5.3858E-03    0.043187   11.4853    4.9411E+02  1.0723E+00
 1.6106E+11  6.1552E-03    0.050201   11.4933    6.0368E+02  1.0758E+00
 1.8119E+09  7.1811E-03    0.060039   11.5039    7.6073E+02  1.0809E+00
 1.0469E+10  8.6173E-03    0.074366   11.5189    1.0000E+03  1.0887E+00
 1.0737E+09  1.0772E-02    0.096680   11.5414    1.3975E+03  1.1012E+00
-3.4360E+10  1.4362E-02    0.135479   11.5790    2.1517E+03  1.1224E+00
-3.0870E+09  2.1543E-02    0.217403   11.6545    3.9528E+03  1.1612E+00
-2.6844E+08  4.3086E-02    0.481470   11.8840    1.1180E+04  1.2558E+00
   efmh          Nc           Nv
 5.8674E-01  4.7270E+17  1.9413E+17
 5.8684E-01  5.0259E+17  2.0645E+17
 5.8697E-01  5.3578E+17  2.2013E+17
 5.8714E-01  5.7280E+17  2.3541E+17
 5.8735E-01  6.1430E+17  2.5256E+17
 5.8762E-01  6.6107E+17  2.7192E+17
 5.8795E-01  7.1411E+17  2.9391E+17
 5.8837E-01  7.7467E+17  3.1908E+17
 5.8889E-01  8.4432E+17  3.4811E+17
 5.8956E-01  9.2510E+17  3.8190E+17
 5.9040E-01  1.0197E+18  4.2162E+17
 5.9146E-01  1.1317E+18  4.6886E+17
 5.9283E-01  1.2658E+18  5.2580E+17
 5.9461E-01  1.4289E+18  5.9555E+17
 5.9694E-01  1.6304E+18  6.8257E+17
 6.0004E-01  1.8845E+18  7.9360E+17
 6.0422E-01  2.2128E+18  9.3922E+17
 6.0999E-01  2.6496E+18  1.1368E+18
 6.1813E-01  3.2531E+18  1.4168E+18
 6.2995E-01  4.1287E+18  1.8369E+18
 6.4765E-01  5.4862E+18  2.5171E+18
 6.7496E-01  7.7993E+18  3.7426E+18
 7.1777E-01  1.2355E+19  6.3188E+18
 7.8124E-01  2.3887E+19  1.3182E+19
 8.9379E-01  7.5987E+19  4.5624E+19
```

Figure 3.30. Si Full Set Output Listing File LIST.DAT

3.4. Sample Session – Constants

Here the constants used in the program for both GaAs and Si are listed using the `constants` option. Since all of the menu manipulations necessary to produce this output have already been described, the results are simply listed.

The `Main Menu` for the Si output set is shown in figure 3.31. If impurity parameters had been present, they would not influence the output, since parameters output under the `constants` option depend only on temperature. The terminal output, which is not a complete listing of all the constants, is shown in figure 3.32, and the listing file showing all of the constants is listed in figure 3.33. Similarly, the `Main Menu` used to produce the `constants` output for GaAs is shown in figure 3.34, followed by the output listing file shown in figure 3.35.

```
                              Main Menu
1  Material       >  Si
2  Output to      >  terminal and listing file
3  Output         >  constants
4  Input data     :
   Independent variable >    Temp
                  Start >      25.0000
                  Final >    1000.0000
                  Step  >      25.0000

   Dopants        Density(cm-3)      Energy(eV)       Degeneracy
   ----------     -------------      ----------       ----------
   none

5  Execute
6  Exit                        Enter number >
```

Figure 3.31. Main Menu for Si Constants Option.

```
Electrical Properties of Silicon

Dopants        Density(cm-3)     Energy(eV)     Degeneracy
----------     -------------     ----------     ----------
none

    Temp         1000/T          kT             Eg          diel       Temp**3/2
    25.0000      40.0000    2.1543E-03     1.155841      11.4517     1.2500E+02
    50.0000      20.0000    4.3086E-03     1.155125      11.4741     3.5355E+02
    75.0000      13.3333    6.4630E-03     1.153501      11.4965     6.4952E+02
   100.0000      10.0000    8.6173E-03     1.151053      11.5189     1.0000E+03
   125.0000       8.0000    1.0772E-02     1.147862      11.5414     1.3975E+03
   150.0000       6.6667    1.2926E-02     1.144004      11.5639     1.8371E+03
   175.0000       5.7143    1.5080E-02     1.139554      11.5865     2.3150E+03
   200.0000       5.0000    1.7235E-02     1.134581      11.6091     2.8284E+03
   225.0000       4.4444    1.9389E-02     1.129152      11.6318     3.3750E+03
   250.0000       4.0000    2.1543E-02     1.123330      11.6545     3.9528E+03
   275.0000       3.6364    2.3698E-02     1.117175      11.6772     4.5604E+03
   300.0000       3.3333    2.5852E-02     1.110741      11.7000     5.1962E+03
   325.0000       3.0769    2.8006E-02     1.104082      11.7228     5.8590E+03
   350.0000       2.8571    3.0161E-02     1.097246      11.7570     6.5479E+03
   375.0000       2.6667    3.2315E-02     1.090278      11.7686     7.2618E+03
   400.0000       2.5000    3.4469E-02     1.083219      11.7916     8.0000E+03
   425.0000       2.3529    3.6624E-02     1.076108      11.8146     8.7616E+03
   450.0000       2.2222    3.8778E-02     1.068978      11.8377     9.5459E+03
   475.0000       2.1053    4.0932E-02     1.061861      11.8608     1.0352E+04
   500.0000       2.0000    4.3086E-02     1.054783      11.8840     1.1180E+04
   525.0000       1.9048    4.5241E-02     1.047769      11.9071     1.2029E+04
   550.0000       1.8182    4.7395E-02     1.040839      11.9304     1.2899E+04
   575.0000       1.7391    4.9549E-02     1.034008      11.9537     1.3788E+04
   600.0000       1.6667    5.1704E-02     1.027290      11.9770     1.4697E+04
   625.0000       1.6000    5.3858E-02     1.020695      12.0004     1.5625E+04
   650.0000       1.5385    5.6012E-02     1.014227      12.0238     1.6572E+04
   675.0000       1.4815    5.8167E-02     1.007889      12.0473     1.7537E+04
   700.0000       1.4286    6.0321E-02     1.001680      12.0708     1.8520E+04
   725.0000       1.3793    6.2475E-02     0.995595      12.0944     1.9521E+04
   750.0000       1.3333    6.4630E-02     0.989626      12.1180     2.0540E+04
   775.0000       1.2903    6.6784E-02     0.983760      12.1416     2.1575E+04
   800.0000       1.2500    6.8938E-02     0.977981      12.1653     2.2627E+04
   825.0000       1.2121    7.1093E-02     0.972270      12.1891     2.3696E+04
   850.0000       1.1765    7.3247E-02     0.966606      12.2129     2.4782E+04
   875.0000       1.1429    7.5401E-02     0.960960      12.2367     2.5883E+04
   900.0000       1.1111    7.7556E-02     0.955304      12.2606     2.7000E+04
   925.0000       1.0811    7.9710E-02     0.949603      12.2845     2.8133E+04
   950.0000       1.0526    8.1864E-02     0.943821      12.3085     2.9281E+04
   975.0000       1.0256    8.4019E-02     0.937917      12.3325     3.0444E+04
  1000.0000       1.0000    8.6173E-02     0.931847      12.3566     3.1623E+04
                ------------  Computation Complete  ---------------
FORTRAN STOP
```

Figure 3.32. Si Terminal Output Using the Constants Option.

Electrical Properties of Silicon

Dopants	Density(cm-3)	Energy(eV)	Degeneracy
none			

Temp	1000/T	kT	Eg	diel	Temp**3/2
25.0000	40.0000	2.1543E-03	1.155841	11.4517	1.2500E+02
50.0000	20.0000	4.3086E-03	1.155125	11.4741	3.5355E+02
75.0000	13.3333	6.4630E-03	1.153501	11.4965	6.4952E+02
100.0000	10.0000	8.6173E-03	1.151053	11.5189	1.0000E+03
125.0000	8.0000	1.0772E-02	1.147862	11.5414	1.3975E+03
150.0000	6.6667	1.2926E-02	1.144004	11.5639	1.8371E+03
175.0000	5.7143	1.5080E-02	1.139554	11.5865	2.3150E+03
200.0000	5.0000	1.7235E-02	1.134581	11.6091	2.8284E+03
225.0000	4.4444	1.9389E-02	1.129152	11.6318	3.3750E+03
250.0000	4.0000	2.1543E-02	1.123330	11.6545	3.9528E+03
275.0000	3.6364	2.3698E-02	1.117175	11.6772	4.5604E+03
300.0000	3.3333	2.5852E-02	1.110741	11.7000	5.1962E+03
325.0000	3.0769	2.8006E-02	1.104082	11.7228	5.8590E+03
350.0000	2.8571	3.0161E-02	1.097246	11.7457	6.5479E+03
375.0000	2.6667	3.2315E-02	1.090278	11.7686	7.2618E+03
400.0000	2.5000	3.4469E-02	1.083219	11.7916	8.0000E+03
425.0000	2.3529	3.6624E-02	1.076108	11.8146	8.7616E+03
450.0000	2.2222	3.8778E-02	1.068978	11.8377	9.5459E+03
475.0000	2.1053	4.0932E-02	1.061861	11.8608	1.0352E+04
500.0000	2.0000	4.3086E-02	1.054783	11.8840	1.1180E+04
525.0000	1.9048	4.5241E-02	1.047769	11.9071	1.2029E+04
550.0000	1.8182	4.7395E-02	1.040839	11.9304	1.2899E+04
575.0000	1.7391	4.9549E-02	1.034008	11.9537	1.3788E+04
600.0000	1.6667	5.1704E-02	1.027290	11.9770	1.4697E+04
625.0000	1.6000	5.3858E-02	1.020695	12.0004	1.5625E+04
650.0000	1.5385	5.6012E-02	1.014227	12.0238	1.6572E+04
675.0000	1.4815	5.8167E-02	1.007889	12.0473	1.7537E+04
700.0000	1.4286	6.0321E-02	1.001680	12.0708	1.8520E+04
725.0000	1.3793	6.2475E-02	0.995595	12.0944	1.9521E+04
750.0000	1.3333	6.4630E-02	0.989626	12.1180	2.0540E+04
775.0000	1.2903	6.6784E-02	0.983760	12.1416	2.1575E+04
800.0000	1.2500	6.8938E-02	0.977981	12.1653	2.2627E+04
825.0000	1.2121	7.1093E-02	0.972270	12.1891	2.3696E+04
850.0000	1.1765	7.3247E-02	0.966606	12.2129	2.4782E+04
875.0000	1.1429	7.5401E-02	0.960960	12.2367	2.5883E+04
900.0000	1.1111	7.7556E-02	0.955304	12.2606	2.7000E+04
925.0000	1.0811	7.9710E-02	0.949603	12.2845	2.8133E+04
950.0000	1.0526	8.1864E-02	0.943821	12.3085	2.9281E+04
975.0000	1.0256	8.4019E-02	0.937917	12.3325	3.0444E+04
1000.0000	1.0000	8.6173E-02	0.931847	12.3566	3.1623E+04

efme	efmh	Nc	Nv
1.0624E+00	5.8762E-01	6.6107E+17	2.7192E+17
1.0680E+00	6.0004E-01	1.8845E+18	7.9360E+17
1.0773E+00	6.2158E-01	3.5074E+18	1.5372E+18
1.0887E+00	6.4765E-01	5.4862E+18	2.5171E+18
1.1012E+00	6.7496E-01	7.7993E+18	3.7426E+18
1.1140E+00	7.0135E-01	1.0431E+19	5.2111E+18
1.1265E+00	7.2554E-01	1.3367E+19	6.9092E+18
1.1386E+00	7.4691E-01	1.6595E+19	8.8172E+18
1.1502E+00	7.6539E-01	2.0105E+19	1.0914E+19
1.1612E+00	7.8124E-01	2.3887E+19	1.3182E+19
1.1718E+00	7.9496E-01	2.7937E+19	1.5610E+19
1.1821E+00	8.0713E-01	3.2252E+19	1.8196E+19
1.1922E+00	8.1834E-01	3.6830E+19	2.0946E+19
1.2020E+00	8.2912E-01	4.1672E+19	2.3873E+19
1.2117E+00	8.3985E-01	4.6777E+19	2.6992E+19
1.2212E+00	8.5074E-01	5.2141E+19	3.0316E+19
1.2305E+00	8.6182E-01	5.7756E+19	3.3853E+19

```
1.2395E+00   8.7293E-01   6.3613E+19   3.7599E+19
1.2479E+00   8.8374E-01   6.9696E+19   4.1535E+19
1.2558E+00   8.9379E-01   7.5987E+19   4.5624E+19
1.2631E+00   9.0254E-01   8.2469E+19   4.9810E+19
1.2698E+00   9.0948E-01   8.9134E+19   5.4027E+19
1.2761E+00   9.1420E-01   9.5988E+19   5.8204E+19
1.2824E+00   9.1653E-01   1.0307E+20   6.2278E+19
1.2892E+00   9.1667E-01   1.1045E+20   6.6225E+19
1.2976E+00   9.1531E-01   1.1829E+20   7.0082E+19
1.3090E+00   9.1390E-01   1.2683E+20   7.3992E+19
1.3252E+00   9.1475E-01   1.3645E+20   7.8250E+19
1.3489E+00   9.2130E-01   1.4769E+20   8.3367E+19
1.3832E+00   9.3837E-01   1.6136E+20   9.0165E+19
1.4322E+00   9.7236E-01   1.7858E+20   9.9903E+19
1.5009E+00   1.0316E+00   2.0094E+20   1.1449E+20
1.5956E+00   1.1265E+00   2.3064E+20   1.3683E+20
1.7234E+00   1.2702E+00   2.7076E+20   1.7132E+20
1.8931E+00   1.4784E+00   3.2557E+20   2.2468E+20
2.1149E+00   1.7701E+00   4.0105E+20   3.0708E+20
2.4009E+00   2.1680E+00   5.0541E+20   4.3371E+20
2.7646E+00   2.6987E+00   6.5002E+20   6.2689E+20
3.2220E+00   3.3929E+00   8.5033E+20   9.1887E+20
3.7912E+00   4.2866E+00   1.1273E+21   1.3553E+21
```

Figure 3.33. Si LIST.DAT File Using the Constants Option.

```
                              Main Menu
1  Material      >  GaAs
2  Output to     >  listing file only
3  Output        >  constants
4  Input data    :
   Independent variable >    Temp
                   Start >      25.0000
                   Stop  >    1000.0000
                   Step  >      25.0000

   Dopants         Density(cm-3)      Energy(eV)       Degeneracy
   ----------      -------------      ----------       ----------
   none

5  Execute
6  Exit                        Enter number > 5
```

Figure 3.34. Main Menu for GaAs Constants Option.

Electrical Properties of Gallium Arsenide

Dopants	Density(cm-3)	Energy(eV)	Degeneracy
none			

Temp	1000/T	kT	Eg	diel	Temp**3/2
25.0000	40.0000	2.1543E-03	1.517525	12.4372	1.2500E+02
50.0000	20.0000	4.3086E-03	1.513680	12.4744	3.5355E+02
75.0000	13.3333	6.4630E-03	1.508103	12.5116	6.4952E+02
100.0000	10.0000	8.6173E-03	1.501220	12.5488	1.0000E+03
125.0000	8.0000	1.0772E-02	1.493330	12.5860	1.3975E+03
150.0000	6.6667	1.2926E-02	1.484646	12.6232	1.8371E+03
175.0000	5.7143	1.5080E-02	1.475325	12.6604	2.3150E+03
200.0000	5.0000	1.7235E-02	1.465485	12.6976	2.8284E+03
225.0000	4.4444	1.9389E-02	1.455217	12.7348	3.3750E+03
250.0000	4.0000	2.1543E-02	1.444592	12.7720	3.9528E+03
275.0000	3.6364	2.3698E-02	1.433665	12.8092	4.5604E+03
300.0000	3.3333	2.5852E-02	1.422482	12.8464	5.1962E+03
325.0000	3.0769	2.8006E-02	1.411079	12.8836	5.8590E+03
350.0000	2.8571	3.0161E-02	1.399485	12.9208	6.5479E+03
375.0000	2.6667	3.2315E-02	1.387726	12.9580	7.2618E+03
400.0000	2.5000	3.4469E-02	1.375821	12.9952	8.0000E+03
425.0000	2.3529	3.6624E-02	1.363789	13.0324	8.7616E+03
450.0000	2.2222	3.8778E-02	1.351643	13.0696	9.5459E+03
475.0000	2.1053	4.0932E-02	1.339397	13.1068	1.0352E+04
500.0000	2.0000	4.3086E-02	1.327061	13.1440	1.1180E+04
525.0000	1.9048	4.5241E-02	1.314644	13.1812	1.2029E+04
550.0000	1.8182	4.7395E-02	1.302155	13.2184	1.2899E+04
575.0000	1.7391	4.9549E-02	1.289600	13.2556	1.3788E+04
600.0000	1.6667	5.1704E-02	1.276985	13.2928	1.4697E+04
625.0000	1.6000	5.3858E-02	1.264316	13.3300	1.5625E+04
650.0000	1.5385	5.6012E-02	1.251598	13.3672	1.6572E+04
675.0000	1.4815	5.8167E-02	1.238835	13.4044	1.7537E+04
700.0000	1.4286	6.0321E-02	1.226030	13.4416	1.8520E+04
725.0000	1.3793	6.2475E-02	1.213187	13.4788	1.9521E+04
750.0000	1.3333	6.4630E-02	1.200309	13.5160	2.0540E+04
775.0000	1.2903	6.6784E-02	1.187399	13.5532	2.1575E+04
800.0000	1.2500	6.8938E-02	1.174458	13.5904	2.2627E+04
825.0000	1.2121	7.1093E-02	1.161490	13.6276	2.3696E+04
850.0000	1.1765	7.3247E-02	1.148496	13.6648	2.4782E+04
875.0000	1.1429	7.5401E-02	1.135478	13.7020	2.5883E+04
900.0000	1.1111	7.7556E-02	1.122438	13.7392	2.7000E+04
925.0000	1.0811	7.9710E-02	1.109376	13.7764	2.8133E+04
950.0000	1.0526	8.1864E-02	1.096295	13.8136	2.9281E+04
975.0000	1.0256	8.4019E-02	1.083196	13.8508	3.0444E+04
1000.0000	1.0000	8.6173E-02	1.070080	13.8880	3.1623E+04

efme	efmh	Nc	Nv	Delta EGL	Delta EGX
6.6941E-02	5.2194E-01	1.0455E+16	2.2763E+17	2.9582E-01	4.6222E-01
6.6791E-02	5.2185E-01	2.9473E+16	6.4367E+17	2.9537E-01	4.6279E-01
6.6574E-02	5.2172E-01	5.3880E+16	1.1821E+18	2.9470E-01	4.6362E-01
6.6305E-02	5.2157E-01	8.2453E+16	1.8191E+18	2.9388E-01	4.6465E-01
6.5997E-02	5.2138E-01	1.1443E+17	2.5409E+18	2.9294E-01	4.6582E-01
6.5658E-02	5.2118E-01	1.4926E+17	3.3382E+18	2.9190E-01	4.6712E-01
6.5293E-02	5.2097E-01	1.8653E+17	4.2040E+18	2.9079E-01	4.6850E-01
6.4908E-02	5.2075E-01	2.2588E+17	5.1330E+18	2.8961E-01	4.6997E-01
6.4506E-02	5.2052E-01	2.6703E+17	6.1208E+18	2.8839E-01	4.7150E-01
6.4089E-02	5.2028E-01	3.0972E+17	7.1638E+18	2.8712E-01	4.7308E-01
6.3660E-02	5.2003E-01	3.5374E+17	8.2590E+18	2.8582E-01	4.7471E-01
6.3221E-02	5.1978E-01	3.9889E+17	9.4037E+18	2.8448E-01	4.7638E-01
6.2772E-02	5.1953E-01	4.4500E+17	1.0596E+19	2.8312E-01	4.7807E-01
6.2315E-02	5.1927E-01	4.9190E+17	1.1833E+19	2.8174E-01	4.7980E-01
6.1852E-02	5.1902E-01	5.3946E+17	1.3113E+19	2.8033E-01	4.8155E-01
6.1382E-02	5.1876E-01	5.8753E+17	1.4435E+19	2.7891E-01	4.8332E-01
6.0906E-02	5.1849E-01	6.3600E+17	1.5797E+19	2.7748E-01	4.8512E-01

```
6.0426E-02  5.1823E-01  6.8476E+17  1.7198E+19  2.7603E-01  4.8693E-01
5.9941E-02  5.1797E-01  7.3368E+17  1.8637E+19  2.7457E-01  4.8875E-01
5.9451E-02  5.1771E-01  7.8268E+17  2.0112E+19  2.7310E-01  4.9059E-01
5.8958E-02  5.1744E-01  8.3165E+17  2.1623E+19  2.7161E-01  4.9244E-01
5.8462E-02  5.1718E-01  8.8051E+17  2.3168E+19  2.7012E-01  4.9430E-01
5.7962E-02  5.1692E-01  9.2918E+17  2.4747E+19  2.6862E-01  4.9617E-01
5.7459E-02  5.1666E-01  9.7758E+17  2.6358E+19  2.6712E-01  4.9804E-01
5.6954E-02  5.1639E-01  1.0256E+18  2.8001E+19  2.6561E-01  4.9993E-01
5.6445E-02  5.1613E-01  1.0732E+18  2.9675E+19  2.6409E-01  5.0183E-01
5.5935E-02  5.1587E-01  1.1204E+18  3.1380E+19  2.6257E-01  5.0373E-01
5.5422E-02  5.1561E-01  1.1670E+18  3.3114E+19  2.6104E-01  5.0563E-01
5.4907E-02  5.1535E-01  1.2129E+18  3.4878E+19  2.5951E-01  5.0755E-01
5.4389E-02  5.1510E-01  1.2582E+18  3.6670E+19  2.5797E-01  5.0946E-01
5.3870E-02  5.1484E-01  1.3027E+18  3.8490E+19  2.5643E-01  5.1139E-01
5.3349E-02  5.1459E-01  1.3465E+18  4.0337E+19  2.5488E-01  5.1331E-01
5.2826E-02  5.1433E-01  1.3894E+18  4.2211E+19  2.5334E-01  5.1525E-01
5.2301E-02  5.1408E-01  1.4315E+18  4.4112E+19  2.5179E-01  5.1718E-01
5.1775E-02  5.1383E-01  1.4726E+18  4.6039E+19  2.5023E-01  5.1912E-01
5.1247E-02  5.1358E-01  1.5127E+18  4.7991E+19  2.4868E-01  5.2106E-01
5.0717E-02  5.1333E-01  1.5518E+18  4.9968E+19  2.4712E-01  5.2301E-01
5.0186E-02  5.1308E-01  1.5898E+18  5.1970E+19  2.4556E-01  5.2496E-01
4.9654E-02  5.1284E-01  1.6267E+18  5.3996E+19  2.4399E-01  5.2691E-01
4.9120E-02  5.1259E-01  1.6625E+18  5.6046E+19  2.4243E-01  5.2886E-01
```

Figure 3.35. GaAs `LIST.DAT` File Using the Constants Option.

3.5. Sample Session – User-Specified Set

A particularly useful feature of EPROP is the `user-specified` output set. When this output option is in effect, the output can be put into almost any format desired. This can be used to limit the data output to only what is desired, or it can be used to create data for input to another program, as was done to fit the data of Nichols et al. in section 2.5.

As an example, a `user-specified` set will be created to examine the room-temperature electrical properties of GaAs doped with carbon and a monovalent deep level with an energy appropriate for EL2 [30]. Here, the position of the Fermi level will be studied as a function of the EL2 and carbon densities.

If the `user-specified` output set is selected (through selection 3, the `Output to` option in the `Main Menu`), the `User-Specified Output Set` menu is displayed (see fig. 3.36). The screen is divided into two parts, with instructions at the bottom. The `Current Set` is what would be output if the program were run; in this case, it would give only the reciprocal temperature. The first entry is always the independent variable and will be changed automatically if the independent variable is changed. It cannot be changed in any other way from this menu.

The second listing, `Optional Output Parameters`, gives the complete list from which the user-defined output can be selected. To create an output which contains `EF, p, Nd1+,` and `Na1-`, the response is `A3 5 6 8` (or `C3 5 6 8`). Pressing carriage return rewrites the menu with the `user-specified` output listed under current set. Changes can be made if necessary using the `R` command for removing, the `A` command for adding, and `C` to start over (creating). A carriage return brings back the `Main Menu`.

Using the menus previously described, the independent variable and dopant parameters are entered (fig. 3.37). Here, for a background silicon density of $10^{15}\,\mathrm{cm}^{-3}$, and an EL2 concentration of $10^{15}\,\mathrm{cm}^{-3}$, the carbon density is stepped from 10^{14} to $10^{18}\,\mathrm{cm}^{-3}$. The user-defined terminal output which results when the program is run is shown in figure 3.38. This dependence of the Fermi energy on the carbon and EL2 concentrations, with the background silicon concentration fixed at $10^{15}\,\mathrm{cm}^{-3}$ is shown in figure 3.39. With no EL2 present, the resistivity changes abruptly from n-type to p-type as the carbon density becomes greater than the silicon density. If high resistivity material is desired, a broader range in carbon density can be tolerated as the deep electron trap density is increased. The minimum carrier densities are obtained for a Fermi-level position near the intrinsic Fermi energy; this is best achieved using large trap densities or by introducing traps with energies closer to the intrinsic Fermi energy.

```
                    User-Specified Output Set

Current Set :
 1>    1000/T

Optional Output Parameters :
 1>    Temp      2>    1000/T    3>       EF     4>      n       5>      p
 6>     Nd1+     7>     Nd2+     8>     Nd3+     9>     Na1-     10>     Na2-
11>     Na3-    12>     kT       13>     Eg      14>    Ec-EF    15> Sum Charges
16>  Temp**3/2  17>    diel      18>    efme     19>    efmh     20>     Nc
21>     Nv      22>     Nc'      23>     Nv'     24>     nG      25>     nL
26>     nX      27>  Delta EGL   28>  Delta EGX

        Enter R[number(s)] to remove from the current output set,
              A[number(s)] to add to the current output set, or
              C[number(s)] to create a new set.
        Separate numbers by commas or spaces, press return to exit.
         (Note: Use numbers from the Current set for a removal)

  Input > a 3 5 6,8

                    User-Specified Output Set

Current Set :
 1>    1000/T    2>       EF     3>      p       4>     Nd1+     5>     Nd3+

Optional Output Parameters :
 1>    Temp      2>    1000/T    3>       EF     4>      n       5>      p
 6>     Nd1+     7>     Nd2+     8>     Nd3+     9>     Na1-     10>     Na2-
11>     Na3-    12>     kT       13>     Eg      14>    Ec-EF    15> Sum Charges
16>  Temp**3/2  17>    diel      18>    efme     19>    efmh     20>     Nc
21>     Nv      22>     Nc'      23>     Nv'     24>     nG      25>     nL
26>     nX      27>  Delta EGL   28>  Delta EGX

        Enter R[number(s)] to remove from the current output set,
              A[number(s)] to add to the current output set, or
              C[number(s)] to create a new set.
        Separate numbers by commas or spaces, press return to exit.
         (Note: Use numbers from the Current set for a removal)

  Input >
```

Figure 3.36. User-Specified Output Menu.

```
                              Main Menu
1  Material      >  GaAs
2  Output to     >  terminal only
3  Output        >  user specified
4  Input data    :
   Independent variable >     Na1
                  Start >   1.000E+14
                  Final >   1.000E+18
      Divisions/Decade  >   5.000E+00
   Temperature (K)      >     300.0000

   Dopants         Density(cm-3)      Energy(eV)       Degeneracy
   ----------      -------------      ----------       ----------
   EL2               1.0000E+15         0.8300           2.0
   Silicon           1.0000E+15         0.0060           2.0
   Carbon            ind. var.          0.0250           4.0

5  Execute
6  Exit                        Enter number >
```

Figure 3.37. Main Menu for User-Specified Output.

Electrical Properties of Gallium Arsenide

Dopants	Density(cm-3)	Energy(eV)	Degeneracy
EL2	1.000E+15	0.8300	2.0000
Silicon	1.000E+15	0.0060	2.0000
Carbon	1.000E+14	0.0250	4.0000

Na1	EF	p	Nd1+	Na1-
1.000E+14	1.263375	5.6937E-03	2.6818E+03	1.0000E+14
1.585E+14	1.261636	6.0898E-03	2.8684E+03	1.5849E+14
2.512E+14	1.258617	6.8442E-03	3.2237E+03	2.5119E+14
3.981E+14	1.252968	8.5159E-03	4.0111E+03	3.9811E+14
6.310E+14	1.240317	1.3892E-02	6.5432E+03	6.3096E+14
1.000E+15	0.920335	3.2979E+03	1.5533E+09	1.0000E+15
1.585E+15	0.565699	2.9914E+09	5.8489E+14	1.5849E+15
2.512E+15	0.254252	5.1047E+14	1.0000E+15	2.5105E+15
3.981E+15	0.219307	1.9724E+15	1.0000E+15	3.9724E+15
6.310E+15	0.199279	4.2798E+15	1.0000E+15	6.2799E+15
1.000E+16	0.183385	7.9135E+15	1.0000E+15	9.9134E+15
1.585E+16	0.169354	1.3614E+16	1.0000E+15	1.5614E+16
2.512E+16	0.156346	2.2510E+16	1.0000E+15	2.4509E+16
3.981E+16	0.143996	3.6276E+16	1.0000E+15	3.8276E+16
6.310E+16	0.132143	5.7334E+16	1.0000E+15	5.9334E+16
1.000E+17	0.120737	8.9026E+16	1.0000E+15	9.1027E+16
1.585E+17	0.109786	1.3575E+17	1.0000E+15	1.3775E+17
2.512E+17	0.099327	2.0294E+17	1.0000E+15	2.0494E+17
3.981E+17	0.089391	2.9702E+17	1.0000E+15	2.9902E+17
6.310E+17	0.079990	4.2527E+17	1.0000E+15	4.2727E+17
1.000E+18	0.071103	5.9599E+17	1.0000E+15	5.9798E+17
1.585E+18	0.062683	8.1871E+17	1.0000E+15	8.2071E+17

---------- Computation Complete ----------

FORTRAN STOP

Figure 3.38. User-Defined Terminal Output.

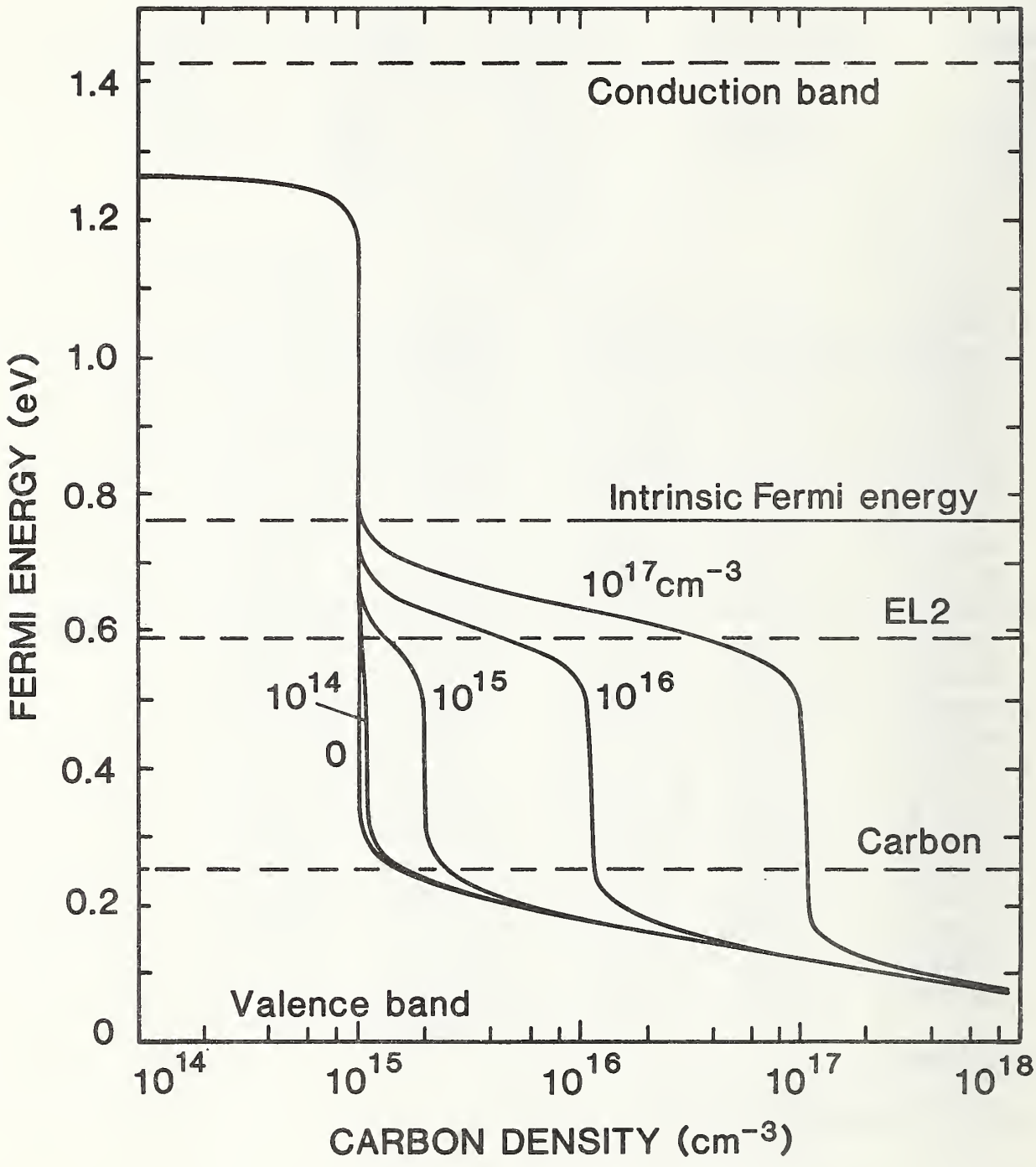

Figure 3.39. Fermi Energy of GaAs. Dependence on carbon and EL2 density for a fixed silicon background density of 10^{15} cm^{-3}.

4. Program Operations and Output

The purpose of this section is to describe the EPROP program in enough detail that the interested user can modify it to fit a particular purpose. To make this section more readable and easier to reference, all names of variables and constants are in boldface. The variables and constants used in the program are also described in the following sections.

4.1. Program Structure and Flow Description

The program executes in two parts, an interactive part and a computational part. First, the parameters from the previous run are read from the file `EPROP.DAT`, and the main menu is presented. The main menu is the dispatcher for the interactive part of the program. The user can gain access to any of the menus and submenus starting from the main menu. In general, each of the menus and submenus is a single subroutine. The routines handling submenus are called from the routines handling more general menus, the highest menu being the main menu which is the subroutine MENU. The way these menus interconnect is described by the diagram of figure 4.1, which further illustrates the hierarchy of the routines. The arrows show the paths which the program flow can follow. The individual routines are described in the next section.

If, for example, the independent variable needs to be changed, the program flow is directed to INPSEL (the INput Parameters SELection menu) and from there to INDSEL (INDependent-variable SELection). If the independent variable is no longer temperature, TSEL (Temperature SELection) is called to receive the temperature for the computation. The return trip to the main menu subroutine MENU is made by retracing the same path.

Note that the diagram above represents the flow of control among the menu routines; it does not always reflect the order in which the menus can be presented. The transition from MENU to a lower subroutine always involves display of the intermediate menus. However, in several cases, a low order subroutine will appear to return to the level above its calling routine. This is true of TSEL, PLTSEL (PLoT SELection), and USPEC (User-SPECified output).

In one special case, not shown in figure 4.1, the subroutines INDSEL, DOPSEL (DOPant SELection), and TSEL can be called directly from MENU. This occurs when the input file, `EPROP.DAT` is not available. (The actual sequence of events in such a case is discussed in sec. 3.1.)

The computational part of the program starts when the user exits the main menu. While the interactive part of the program has a flexible flow pattern, the computational part has a structured control flow as shown in figure 4.2. The capitalized names in the small boxes are individual routines. The large boxes contain the names of routines which themselves call other routines. The content of these larger boxes is charted elsewhere (figs. 4.1 and 4.3). Upon completing the interactive part (MENU) the program generates an impurity table (IMPTBL) to be output to the destinations selected earlier. The HEADER routine then generates the column headings. If the independent variable is not related to temperature,

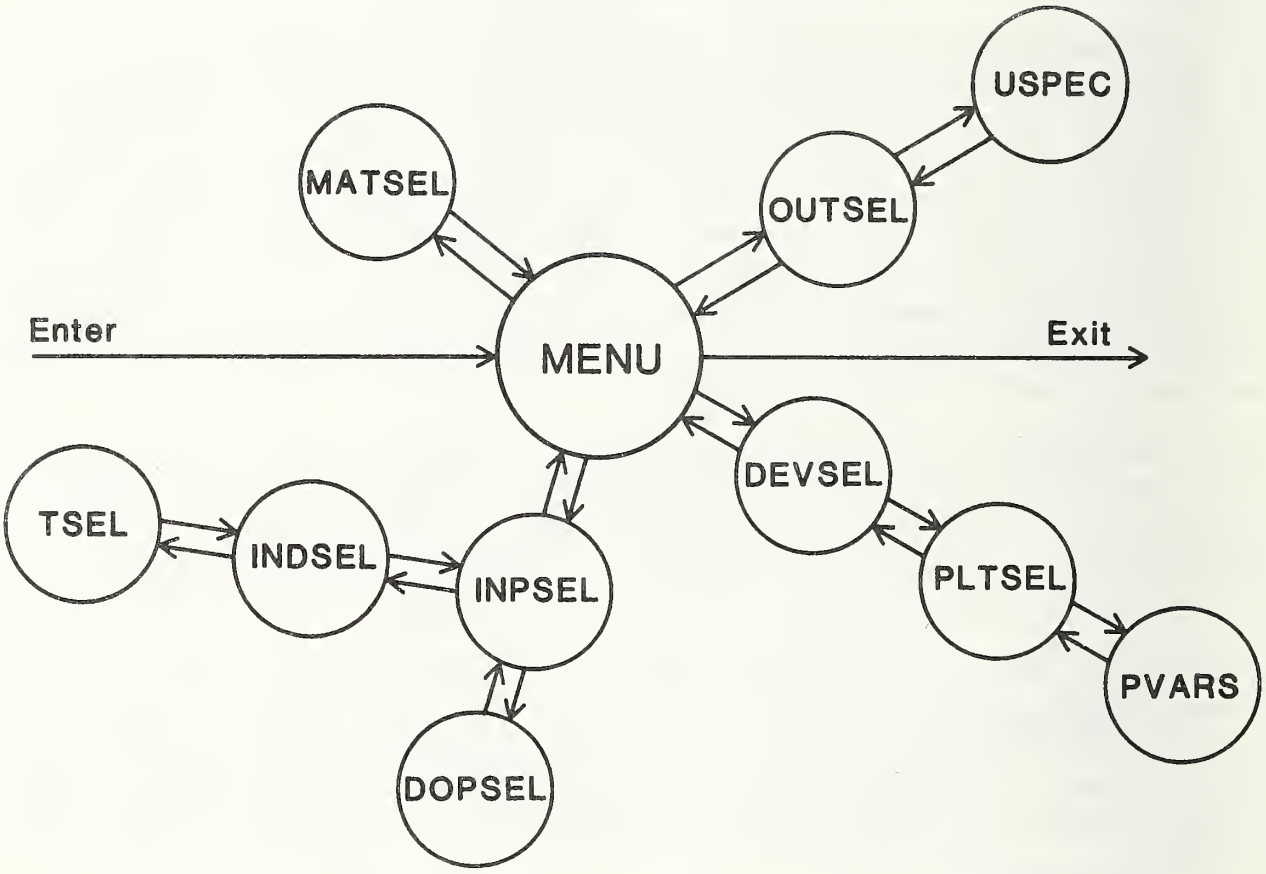

Figure 4.1. Program Flow For Menu Manipulation. The labels refer to subroutines, and the arrows between the subroutines show the calling and returning paths.

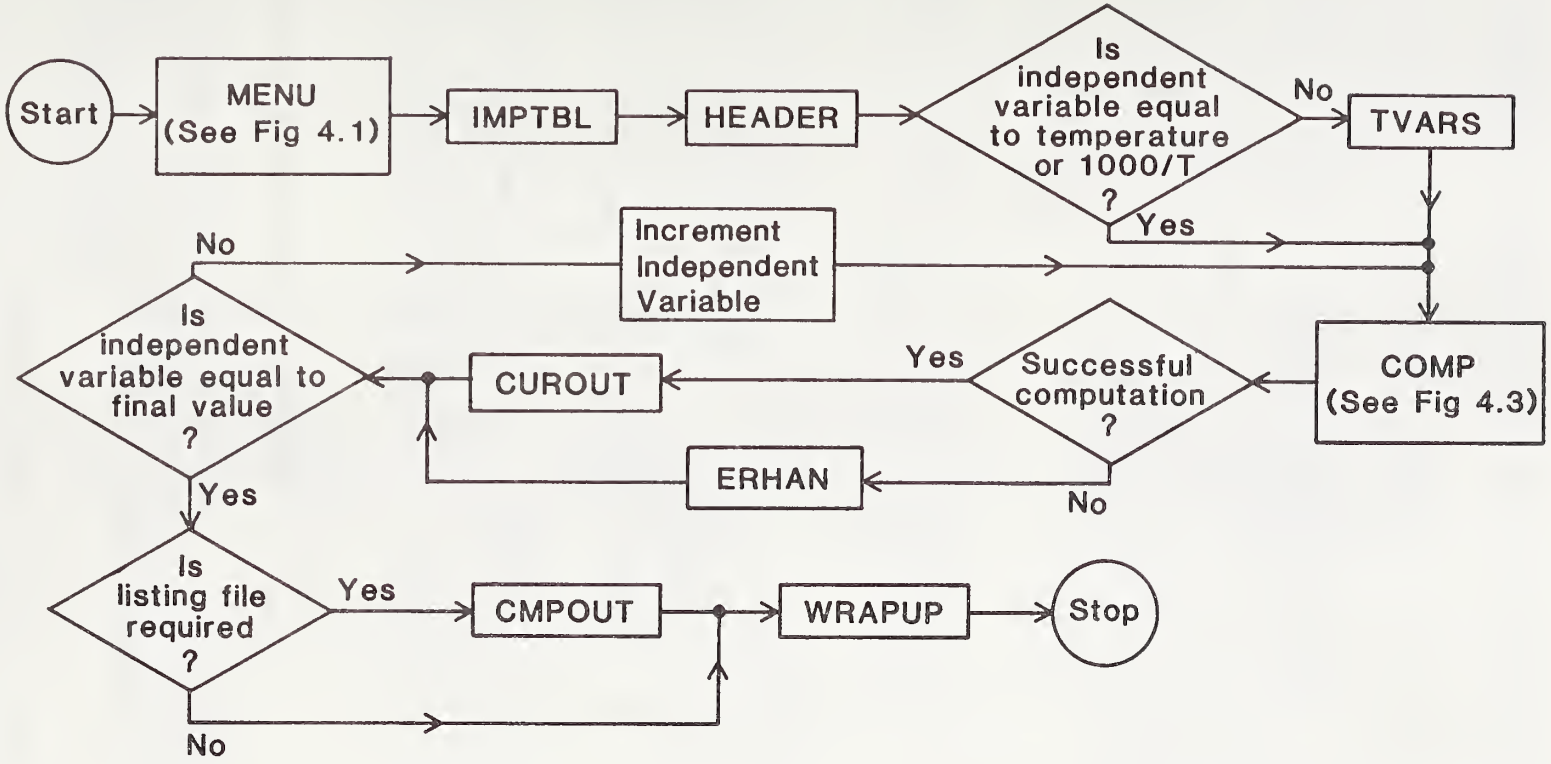

Figure 4.2. EPROP Program Flow Diagram.

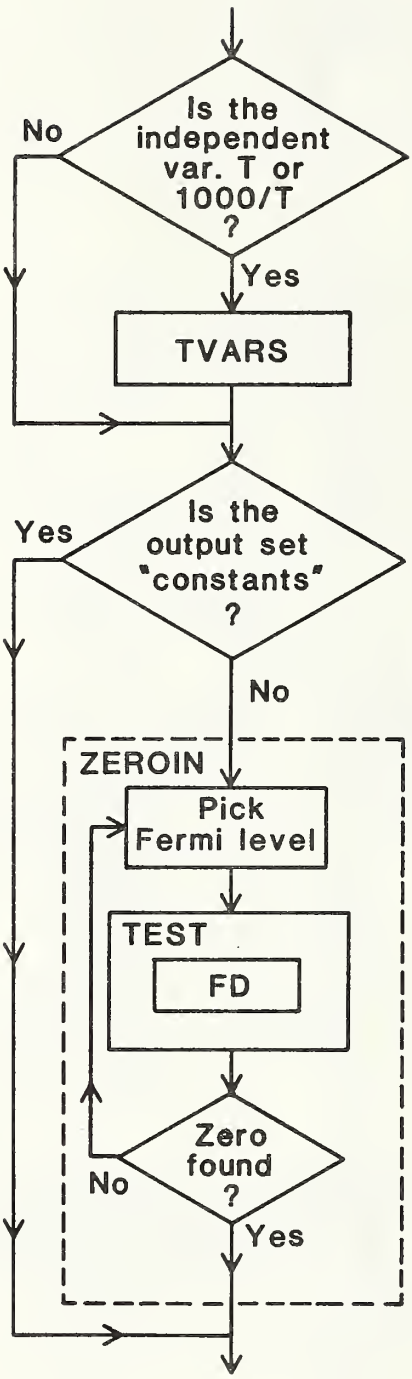

Figure 4.3. Program Flow In The Computation Subroutine, COMP.

then the temperature-dependent constants can be computed once and for all in TVARS. Otherwise, they will have to be computed for each value of temperature.

The computation, COMP, which will be described later, is then performed for a single value of the independent variable. EPROP then tests to see whether the computation was successful; this usually depends on the ability of the zero finding routine to converge. For a successful computation, CUROUT (CURrent OUTput) writes the selected output to one of the output devices. For an unsuccessful computation, the error handling routine, ERHAN, is called. If a listing file is selected, all computations are stored in the scratch file `TEMPOR.ARY`. Once all values of the independent variable are evaluated, the final output, CMPOUT (CoMPlete OUTput), is written to the listing file, and if terminal output is also selected, the first six output variables are sent to the terminal. The WRAPUP routine then closes all open files, sends out any error messages generated during the computation, and terminates the program.

Referring to figure 4.3, the computational program flow can be described. If the independent variable is temperature or $1000/T$, the temperature-dependent constants need to be reevaluated each time through the loop, in TVARS. Next, if the constants output option is selected, no solution of the charge balance equation is required.

The heart of the computation is the ZEROIN subroutine [31], which computes the roots of the equation, $F(x) = 0$. This function, $F(x)$ is passed to ZEROIN as an external function TEST. In EPROP, TEST is the charge neutrality condition, eq (1). ZEROIN selects a value for the Fermi level within the bounds defined in the subroutine call, using bisection and the secant rule to select new values. If a zero is found for a particular Fermi-level selection, within the accuracy required, ZEROIN is exited.

For those interested in making changes to EPROP, a map of the common blocks and the program segments (subroutines and functions) which contain them is provided in table 4.1. The left-hand column lists the main program EPROP followed by the subroutines and functions which pass or use common variables. The common variables are shown across the top of the table, with the common block names bracketing the variables.

4.2. Functions, Subroutines, and Files

Two logical functions and two real functions are used in EPROP. They are defined as shown in table 4.2. Twenty-one subroutines are called; their names and functions are listed in table 4.3. Up to eight files may be opened during program execution; these are listed and described in table 4.4.

Table 4.1 Common Blocks and Program Variables – Closed circles indicate that the variable is necessary to the segment (subroutine or function) and is made available through the common block labeled at the top. Open circles indicate that the variable is passed to the segment through an argument in a subroutine or function call.

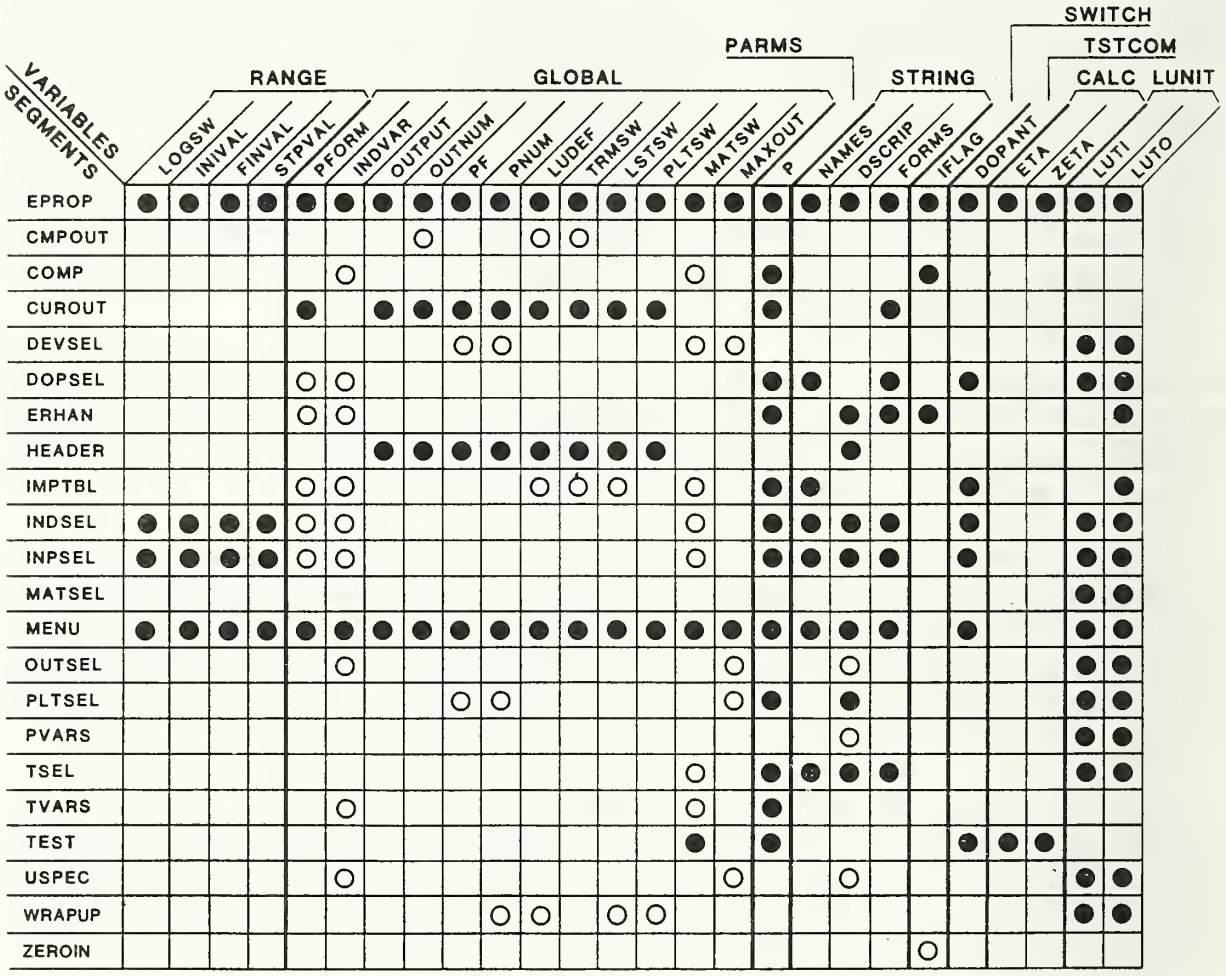

Table 4.2. Functions

DIGIT( )	returns true if the argument is an ASCII character between 0 and 9
INRNGE(**LOW**,**UP**,**VAR**)	returns true if the integer **VAR** is between the lower bound **LOW** and the upper bound **UP**
FD(**J**, **ARG**)	returns the value of the Fermi-Dirac integral of order *J*
TEST(**FERMI**)	returns the real value of the sum of all positive and negative charges for a Fermi energy passed via **FERMI**

Table 4.3. Subroutines

CMPOUT	completes output, produces terminal and listing-file output from the scratch file, TEMPOR.ARY
COMP	computation – directs the computation flow
CUROUT	current output – directs data output concurrent with the computation
DEVSEL	device select – produces the menu for output device selection
DOPSEL	dopant select – produces the menu for selection of dopant impurities
ERHAN	error handler – writes error messages to error file
HEADER	produces column headings for output
IMPTBL	impurity table – produces table of impurity data for output listings
INDSEL	independent variable selection – produces the menu for selection of independent variable
INPSEL	input data selection – calls INDSEL and DOPSEL
MATSEL	material selection – menu for choice of Si or GaAs
MENU	main menu – directs menu calls
OUTSEL	output set selection – provides menu to choose from among three predefined output sets or a user-defined output
PLTSEL	plot file selection – provides menu to select plot file attributes
PROMPT	sends a prompt string to the terminal (without carriage return), preceded by a specified number of blank lines and spaces
PVARS	plot variables – provides listing of all possible x- and y-axis plot parameters
TSEL	temperature selection – provides menu for selecting temperature or $1000/T$ and specifying its value
TVARS	temperature-dependent variables – computes the temperature-dependent variables
USPEC	user-specified output set selection – allows selection of a desired output set
WRAPUP	closes all opened files and displays error messages from the computation
ZEROIN	searches for a zero of the function TEST

Table 4.4. Files

EPROP.DAT	contains current data
ERROR.LIS	listing of computation errors, deleted if no errors are detected
LIST.DAT	contains output data
PLOT1.DAT	first plot file
PLOT2.DAT	second plot file
PLOT3.DAT	third plot file
PLOT4.DAT	fourth plot file
TEMPOR.ARY	scratch file, deleted after normal program completion

4.3. Input and Output Parameters

The values of all of the input and output parameters are stored in a single real vector **P**. This vector is indexed by integer constants which have been given names descriptive of the parameters they represent (e.g., **T** for temperature). These integer constants are also used to index two other vectors, one containing the string descriptors (**DSCRIP**) and the other indicating the type of output format to be used with each parameter (**PFORM**). This technique simplifies the bookkeeping for input and output and for switching independent variables, and greatly reduces the number of decisions the program must make. These three vectors and the format vector are defined in table 4.5.

Table 4.5. Vectors

P( )	real vector containing the values of all input and output parameters
PFORM( )	integer vector containing indices for the character vector **FORMS( )**. **FORMS( )** holds the actual string format
DSCRIP( )	character vector containing the labels for all parameters that can be input or output
FORMS( )	character vector containing the format for all parameters that can be input or output

4.3.1. Independent Variables

The program allows the user to select the independent variable from among 17 different parameters: temperature, reciprocal temperature, or, for any of six impurities: density, energy level, or degeneracy. Up to three acceptors and three donors can be specified. The possible independent variables are listed in table 4.6; the physical units (dimensions) for each variable are given in parentheses.

4.3.2. Dependent Variables

In addition to the independent variables, the parameters listed in table 4.7 are also available for output.

4.4. Other Variables

Other variables and constants used in the program are defined in table 4.8.

Table 4.6. Independent Variables – Physical units (dimensions) are in parentheses.

Parameter	Integer Value	Refers to
DG1	23	degeneracy of donor impurity 1
DG2	26	degeneracy of donor impurity 2
DG3	29	degeneracy of acceptor impurity 1
DG4	32	degeneracy of acceptor impurity 2
DG5	35	degeneracy of acceptor impurity 3
DG6	46	degeneracy of donor impurity 3
DN1	21	density of donor impurity 1 (cm^{-3})
DN2	24	density of donor impurity 2 (cm^{-3})
DN3	27	density of acceptor impurity 1 (cm^{-3})
DN4	30	density of acceptor impurity 2 (cm^{-3})
DN5	33	density of acceptor impurity 3 (cm^{-3})
DN6	44	density of donor impurity 3 (cm^{-3})
EN1	22	binding energy of donor 1 (eV)
EN2	25	binding energy of donor 2 (eV)
EN3	28	binding energy of acceptor (eV)
EN4	31	binding energy of acceptor 2 (eV)
EN5	34	binding energy of acceptor 3 (eV)
EN6	45	binding energy of donor 3 (eV)
RTEMP	2	reciprocal temperature (K^{-1})
T	1	temperature (K)

Table 4.7. Dependent Variables – Physical units (dimensions) are in parentheses.

Parameter	Integer Value	Refers to
CAYT	13	thermal energy, kT/q , (eV) where k is Boltzmann's constant, 1.380662×10^{-23} J/K, q is the elemental charge, 1.602189×10^{-19} C
CN	4	free electron density (cm^{-3})
CP	5	free hole density (cm^{-3})
CNG	38	electron density in the Γ_6 conduction band (cm^{-3})
CNL	39	electron density in the L_6 conduction band (cm^{-3})
CNX	40	electron density in the X_6 conduction band (cm^{-3})
DELGL	41	energy difference between Γ_6 and L_6 conduction bands (eV)
DELGX	42	energy difference between Γ_6 and X_6 conduction bands (eV)
DI1	6	density of ionized donor 1 (cm^{-3})
DI2	7	density of ionized donor 2 (cm^{-3})
DI3	8	density of ionized acceptor 1 (cm^{-3})
DI4	9	density of ionized acceptor 2 (cm^{-3})
DI5	10	density of ionized acceptor 3 (cm^{-3})
DI6	43	density of ionized donor 3 (cm^{-3})
ECMEF	14	difference between the conduction band and Fermi energies
EF	3	Fermi level (eV)
EFME	17	electron effective mass
EFMH	18	hole effective mass
EG	11	energy gap (eV)
FNC	19	conduction band density of states (cm^{-3})
FNCP	36	conduction band density of states (cm^{-3}) (nonparabolic, GaAs only)
FNV	20	valence band density of states (cm^{-3})
FNVP	37	valence band density of states (cm^{-3}) (nonparabolic, GaAs only)
RDIEL	15	relative dielectric constant (EPROP computes the temperature dependence of the dielectric constant independent of the rest of the calculations. References for the formulas used can be found in the program source.)
SUM	12	sum of all charged species
TPOWR	16	$T^{3/2}$ ($K^{3/2}$)

Table 4.8. Other Program Variables

Name	Type	Description
AE	real	absolute error, referred to in ZEROIN subroutine
ARG	real	dummy argument for FD subroutine
ALPHA	real	defined by eq (10), $(E_F - E_g)/(kT/q)$
BETA	real	defined by eq (17), $E_F/(kT/q)$
CONST	real constant	$2(2\pi m_o k/h^2)^{3/2} = 4.8293 \times 10^{15}\ \mathrm{K}^{-3/2}\,\mathrm{cm}^{-3}$, where $m_o = 9.1095 \times 10^{-31}$ kg is the electron mass and $h = 6.6262 \times 10^{-34}$ Js is Planck's constant
CSTSET( )	integer	pointers to the ten parameters in the constants output set
DEV( )	logical	indicates where output will be directed, where true directs output to the terminal (1), a listing file (2), or up to four plot files (3)
DI( )	integer	pointers to the ionized donor densities
DG( )	integer	pointers to the dopant degeneracies
DN( )	integer	pointers to the dopant densities
DOPANT( )	logical	switches for which dopants are selected
DUMB	real	used to calculate the instantaneous value of the independent variable
EGAP	real	upper limit of Fermi-level energy used in ZEROIN
EN( )	integer	pointers for the energies of the dopant impurities (up to five)
ETA	real	reduced Fermi energy (relative to conduction band edge), $(E_F - E_c)/kT$
FERMI	real	instantaneous Fermi energy
FINVAL	real	final value of independent variable
FTNUM	real	first value of independent variable when logarithmic stepping is selected
FULSET( )	integer	pointers to the 20 full-set output parameters
HYBNUM	integer	number of parameters in the hybrid (user-specified) output set
HYBSET( )	integer	pointers to hybrid (user-specified) output set variables
INDSET( )	integer	pointers to possible independent variables
INDVAR	integer	pointer to the independent variable
INIVAL	real	initial value of independent variable
IFLAG	integer	flag returned by ZEROIN subroutine
J	real	order of Fermi-Dirac integral
LOGSW	logical	true indicates log scale output
LOW	integer	lower bound used by logical function INRNGE
LOWLIM	real	low temperature limit of computation defined by machine exponentiation overflow

LSTSW	logical	list switch, true indicates output should go to the listing file, `LIST.DAT`
LUDEF	integer	default logical unit number
LUER	integer constant	logical unit number (27) for storing error messages
LUIN	integer constant	logical unit number (20) for input data file, `EPROP.DAT`
LULST	integer constant	logical unit number (21) for datalisting file, `LIST.DAT`
LUPL( )	integer	contains the four plot file logical unit numbers
LUPL1	integer constant	logical unit number (22) for plot file 1 output
LUPL2	integer constant	logical unit number (23) for plot file 2 output
LUPL3	integer constant	logical unit number (24) for plot file 3 output
LUPL4	integer constant	logical unit number (25) for plot file 4 output
LUTI	integer	logical unit number (5) for terminal input
LUTO	integer	logical unit number (6) for terminal output
LUTEMP	integer constant	logical unit number (26) for temporary storage of output data
MATSW	integer	global materials switch, indicates Si (1), GaAs (2), other, user defined material (3 or greater)
MAXOUT	integer	maximum number of output parameters
MBL	real	L_6 conduction band effective mass
MH	real	Γ_8 valence band heavy hole effective mass
MLL	real	Γ_8 valence band light hole effective mass
MSEL	integer	local materials switch like **MATSW**
MX	real	X_6 conduction band effective mass
NAMES( )	character	names of dopant impurities (up to six)
NDV	integer	number of divisions per decade
NPT	integer	number of values of the independent variable
NUMER	integer	number of computation errors
OUTNUM	integer	number of parameters in selected output set
OUTPUT( )	integer	pointers for selected output set (up to 20)
OUTSET( )	character	labels for the four possible output sets, short set, full set, constants set, and user specified
PF(,)	integer	integer pointers to the x- and y-axis variables for up to four plot files
PLNAM( )	character	names of four plot files
PLTSW	logical	plot switch, true indicates output should go to the plot files
PNUM	integer	number of plot files selected
POWER	real	argument of ionized impurity density exponent; see eqs (7) and (8)
RE	real	relative error, referred to in ZEROIN subroutine
SETNUM	integer	argument for **OUTSET** vector indicating

		set of parameters being output
SHTSET( )	integer	pointers to the nine short set output parameters
START	integer	lower limit for computation loop
STEP	integer	step for computation loop
STOPIT	integer	upper limit for computation loop
STPVAL	real	incremental step of independent variable for linear scale, number of divisions per decade for log scale
TRMSW	logical	terminal switch, true indicates output should go to the terminal
UP	integer	upper bound used by logical function INRNGE
UPLIM	real	melting point of semiconductor
VAR	integer	variable tested by logical function INRNGE
X	real	value of the Fermi-Dirac integral in FD
ZETA	real	reduced energy for holes, $(E_v - E_F)/kT$

Acknowledgments

The authors wish to thank R. D. Larrabee, W. R. Thurber, and W. M. Bullis whose calculation of the electrical properties of silicon was the basis for this work. J. R. Lowney provided the results of reference [9] prior to publication and converted them to a form which could be used in our program. H. S. Bennett, and C. L. Wilson made helpful comments and suggestions. C. E. Bouldin and D. Kahaner provided help in improving the portability of the code. Thanks are also due to P. Baker, J. Walters, and W. Gladden for their assistance in producing the final manuscript.

References

1. Larrabee, R. D., Thurber, W. R., and Bullis, W. M., *Semiconductor Measurement Technology:* A FORTRAN Program for Calculating the Electrical Properties of Extrinsic Silicon, NBS Special Publication 400-63 (October 1980).

2. Blakemore, J. S., *Semiconductor Statistics* (Pergamon Press, New York, NY, 1962).

3. Blakemore, J. S., Semiconducting and Other Major Properties of Gallium Arsenide, *J. Appl. Phys.* **53**, R123-R181 (1982).

4. Varshni, Y. P., Temperature Dependence of the Energy Band Gap in Semiconductors, *Physica* **34**, 149-154 (1967).

5. Thurmond, C. D., The Standard Thermodynamic Functions for the Formation of Electrons and Holes in Ge, Si, GaAs, and GaP, *J. Electrochem. Soc.* **122**, 1133-1141 (1975).

6. Blakemore, J. S., Intrinsic Density $n_i(T)$ in GaAs: Deduced from Band Gap and Effective Mass Parameters and Derived Independently from Cr Acceptor Capture and Emission Coefficients, *J. Appl. Phys.* **53**, 520-531 (1982).

7. Kane, E. O., Band Structure of Indium Antimonide, *J. Phys. Chem. Solids* **1**, 249-261 (1957).

8. Vrehen, Q. H. F., Interband Magneto-Optical Absorption in Gallium Arsenide, *J. Phys. Chem. Solids* **29**, 129-141 (1968).

9. Lowney, J. R. and Kahn, A. H., Valence-band effective masses of GaAs, *J. Appl. Phys.* **64**, 447-450 (1988).

10. Heasel, E. L., The Calculation of Carrier Concentrations in Silicon in the Medium and High Temperature Regions, *Solid State Electronics* **16**, 651-655 (1972).

11. Ukhanov, Yu. I., and Mal'tsev, Yu. V., Investigation of the Temperature Dependence of the Electron Effective Mass in Semiconductors, *Soviet Physics – Solid State* **5**, 2144-2149 (1964).

12. Stradling, R. A. and Zhukov, V. V., Cyclotron Resonance of Electrons in Silicon at Temperatures up to 200°K, *Proc. Phys. Soc.* **87**, 263-271 (1966).

13. Barber, H. D., Effective Mass and Intrinsic Concentration in Silicon, *Solid State Electronics* **10**, 1039-1051 (1967).

14. Blakemore, J. S., Review Paper, Approximations for Fermi-Dirac Integrals, Especially the Function $F_{\frac{1}{2}}(\eta)$ Used to Describe the Electron Density in a Semiconductor, *Solid State Electronics* **25**, 1067-1076 (1982).

15. Cody, W. J. and Thacher, Jr., H.C., Rational Chebyshev Approximations for Fermi-Dirac Integrals of Orders $-1/2, 1/2$, and $3/2$, *Mathematics of Computation* **21**, 30-40 (1967).

16. Van Halen, P. and Pulfrey, D. L., Accurate Short Series Approximations to Fermi-Dirac Integrals of Order $-1/2, 1/2, 1, 3/2, 2, 5/2, 3$, and $7/2$, *J. Appl. Phys.* **57**, 5271-5274 (1985); Erratum, *J. Appl. Phys.* **59**, 2264-2265 (1986).

17. Nichols, K. H., Yee, C. M. L., and Wolfe, C. M., High-Temperature Carrier Transport in $n-$Type Epitaxial GaAs, *Solid State Electronics* **23**, 109-116 (1980).

18. Landsberg, P. T., Defect With Several Trapping Levels in Semiconductors, *Proc. Phys. Soc. London* **B69**, 1056-1059 (1956).

19. Champness, C. H., The Statistics of Divalent Impurity Centres in a Semiconductor, *Proc. Phys. Soc. London* **B69**, 1335-1339 (1956).

20. Shockley, W., and Last, J. T., Statistics of the Charge Distribution for a Localized Flaw in a Semiconductor, *Phys. Rev.* **107**, 392-396 (1957).

21. Sah, C. T. and Shockley, W., Electron-Hole Recombination Statistics in Semiconductors Through Flaws With Many Charge Conditions, *Phys. Rev.* **109**, 1103-1115 (1958).

22. Blakemore, J. S., Modeling of a Multi-Valent Impurity, Such as GaAs:Cr, *Semi-Insulating III-V Materials Nottingham 1980* (Imprint Editions, Fort Collins, CO, 1980), pp. 29-40.

23. Look, D. C., Statistics of Multicharged Centers in Semiconductors: Applications, *Phys. Rev. B* **24**, 5852-5862 (1981).

24. Bonch-Bruevich, V. L., Effect of Heavy Doping on the Semiconductor Band Structure, *Semiconductors and Semimetals* **1**, R. K. Willardson and A. C. Beer, Editors (Academic Press, New York, NY, 1966), pp. 101-142.

25. Bennett, H. S., Heavy Doping Effects on Bandgaps, Effective Intrinsic Carrier Concentrations and Carrier Mobilities and Lifetimes, *Solid State Electronics* **28**, 193-200 (1985).

26. Lowney, J. R., Impurity Bands and Band Tailing in Moderately Doped Silicon, *J. Appl. Phys.* **59**, 2048-2053 (1986).

27. Bennett, H. S., High Dopant and Carrier Concentration Effects in Gallium Arsenide: Band Structure and Effective Intrinsic Carrier Concentrations, *J. Appl. Phys.* **60**, 2866-2874 (1986).

28. Lowney, J. R., Impurity Bands and Band Tailing in n-Type GaAs, *J. Appl. Phys.* **60**, 2854-2859 (1986).

29. Ta, L. B., Thomas, R. N., Eldridge, G. W., and Hobgood, H. M., Reproducibility and Uniformity Considerations in LEC Growth of Undoped Semi-Insulating GaAs for Large-Area, Direct Implantation Technology, *Inst. Phys. Conf. Ser. No.* **65**, 31-39 (1982).

30. Milnes, A. G., Impurity and Defect Levels (Experimental) in Gallium Arsenide, *Advances in Electronics and Electron Physics* **61**, 63-160 (1983).

31. Shampine, L. F., and Allen, R. C., Numerical Computing: An Introduction (W. B. Saunders Co., Philadelphia, PA, 1973).

Appendix 1. Portability Considerations

EPROP is written in ANSI standard FORTRAN 77 and has been successfully compiled and run on a Digital Equipment Corporation (DEC) VAX 11/785, an International Business Machines (IBM) PC/AT, and on Apple Computer Macintosh Plus and Macintosh II computers.* The source code listed in Appendix 3 of this document was developed on a DEC VAX 11/785 with VMS operating system using the VAX FORTRAN compiler. It was compiled without modification on an IBM PC/AT, using the Layhey FORTRAN compiler V2.0 and IBM linker V2.3 and using Ryan-McFarland RM/FORTRAN V2.00 and linker V1.51RM. It has also been run with minor modifications (described below) on the Apple Macintosh Plus using Absoft MacFortran V2.4 and on the Apple Macintosh II using Absoft MacFortran/020.

Two modifications are necessary in order to produce a working application on the Macintosh. The logical unit number for reading and writing to the Macintosh screen is 9, as opposed to the usual assignments of logical unit 5 for reading and 6 for writing. In the main program, the statements `LUTI=5` and `LUTO=6` must be changed to `LUTI=9` and `LUTO=9`; these values are then made available to all the subroutines which require them. Second, upon completion of an application, the Macintosh operating system returns the user to the *Finder*, erasing any terminal output. To prevent this, a `PAUSE` statement should be inserted just prior to `STOP` in the main program and in the subroutine MENU. Early versions of the FORTRAN compiler allocated too little memory (in the application heap) to permit the simultaneous use of all eight output files allowed by EPROP. The linker could be used, however, to increase this default allocation, or a dummy subroutine could be added to EPROP containing a `DIMENSION` statement allocating a large, unused block of memory. The current release of the compiler appears to have eliminated this problem.

In the VAX/VMS operating environment, when an `OPEN` statement is issued with the qualifier `STATUS=NEW` and a file already exists with this file name, VMS creates a new file with a new version number. EPROP uses `OPEN` statements with `STATUS=UNKNOWN` so that the VAX does not create a new version number. If it is desired to use this VAX VMS feature, all such `OPEN` statements should be changed to `STATUS=NEW`.

In adapting the program to other computers, several factors may need to be considered, some related to the operating system and some to the FORTRAN compiler used. EPROP makes extensive use of one nonstandard feature of FORTRAN. This is so-called "$ editing," in which the end-of-line output (typically carriage return/linefeed) is suppressed in output if a "$" string appears in the first space of the `FORMAT` statement. The VAX and IBM compilers, as well as the current release of MacFortran support this extension, but other compilers may not. For this reason, all output requiring end-of-line suppression is sent

* Certain commercial equipment, instruments, or materials are identified in this paper in order to adequately specify the procedures used. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

through the subroutine PROMPT (see table 4.3). The VAX and IBM versions of PROMPT use $ editing; the Macintosh version uses the TYPE statement which also suppresses the end-of-line output and was available in earlier versions of the compiler. Users of other compilers will have to supply appropriate code in this subroutine.

Under operating systems which do not support file name extensions (as in EPROP.DAT, ERROR.LIS, and TEMPOR.ARY), the eight output file names listed in section 4.2 must be changed globally to names compatible with the target system. The DEC, IBM, and Apple systems mentioned above support file name extensions.

Appendix 2. Adding a New Semiconductor

It is not difficult to modify EPROP to perform calculations for additional semiconductor materials, especially if the same formulation as for GaAs or Si can be used. The following description will aid the user in adding a third material. There is no reason not to add as many as desired; a total of of 26 will, however, exhaust the letters of the alphabet available for menu selections.

The first step is to add another option to the **Material Selection Menu** by modifying the MATSEL subroutine. Add the following to the Block-IF structure in the routine:

```
ELSEIF ((CH.EQ.'U.C.').OR(CH.EQ.'l.c.'))  THEN
       MATSW=3
```

where `U.C.` and `l.c.` represent the upper and lower case, respectively, of the letter chosen to indicate the material being added.

The rest of the modifications involve either adding a third option to Block-IF structures to account for the third material, or adding `.OR.(MATSW.EQ.3)` to an existing condition in a Block-IF. In general, a third option will be necessary, but where the new material has enough in common with silicon or gallium arsenide (such as a formula or number of output parameters), the user may be able to make use of one of the two existing options.

There are 12 places in the code where the logic branches on the basis of material. They are listed below; their location in the source code can be readily found by searching for the string "`--- option ---`."

SUBROUTINE OR FUNCTION	TYPE OF CHANGE
COMP	Define EGAP and UPLIM (energy gap approximation and melting point).
IMPTBL	Define the string: "Electrical Properties of XXXX"; this occurs in two places in IMPTBL.
INDSEL	Define UPLIM (same as for COMP).
MENU	1. Add a third option in addition to "Si" and "GaAs". 2. Define MAXOUT (number of output parameters). Note: this could well be the same for the new material as it is for Si or GaAs. 3. Define OUTNUM for the constants output set (number of output parameters). Note: this also could be the same as for Si or GaAs. 4. Check for undefined parameters. This checking will need to be changed only if a new formulation and new output parameters are required.
PVARS	Define MAX (number of possible plot file parameters).
TEST	The formulation for electron P(N) and hole density

	P(CP) must be changed to use the appropriate density of states for the new material.
TSEL	Define UPLIM (same as COMP).
TVARS	The temperature-dependent constants, P(EFME), P(EFMH), and P(EG) must be defined appropriately for the new semiconductor.

Appendix 3. Program Listing

```
      PROGRAM EPROP
C
C Written by John J. Mathias and Alan C. Seabaugh
C Modified by Michael I. Bell and Jeremiah R. Lowney
C Semiconductor Electronics Division
C National Institute of Standards and Technology
C (Formerly the National Bureau of Standards)
C Gaithersburg, MD  20899
C
C January 1990, Version 1.0
C
C This program solves the charge balance equation to compute
C selected electrical properties of gallium arsenide and silicon.
C The user interface allows the selection of independent variables
C from the list:  temperature, reciprocal temperature, and
C concentration, energy level, and degeneracy for up to three donor
C and three acceptor impurities.  The output can be selected from
C a list including:  Fermi energy level, electron and hole densities,
C donor or acceptor ionized impurity densities, energy band gap,
C valence and conduction band density of states, and
C electron and hole effective masses.
C
      REAL INIVAL,FINVAL,STPVAL
      LOGICAL LOGSW
      COMMON /RANGE/ INIVAL,FINVAL,STPVAL,LOGSW
      INTEGER PFORM(46),INDVAR,OUTPUT(28),OUTNUM,PF(4,2),
     * PNUM,LUDEF,MAXOUT,MATSW
      LOGICAL TRMSW,LSTSW,PLTSW
      COMMON /GLOBAL/ PFORM,INDVAR,OUTPUT,OUTNUM,
     * PF,PNUM,LUDEF,TRMSW,LSTSW,PLTSW,MATSW,MAXOUT
      REAL P(46)
      COMMON /PARMS/ P
      CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
      COMMON /STRING/ NAMES,DSCRIP,FORMS
      INTEGER IFLAG
      COMMON /SWITCH/ IFLAG
      REAL ETA,ZETA
      COMMON /CALC/ ETA,ZETA
      LOGICAL DOPANT(6)
      COMMON /TSTCOM/ DOPANT
      INTEGER LUTI,LUTO
      COMMON /LUNIT/ LUTI,LUTO
C
      INTEGER T,RTEMP,EF,CN,CP,DI1,DI2,DI3,DI4,DI5,EG,SUM,CAYT,ECMEF,
     * RDIEL,TPOWR,EFME,EFMH,FNC,FNV,DN1,EN1,DG1,DN2,EN2,DG2,DN3,EN3,
     * DG3,DN4,EN4,DG4,DN5,EN5,DG5,FNCP,FNVP,CNG,CNL,CNX,DELGL,DELGX
     * DI6,DN6,EN6,DG6
      PARAMETER (T=1,RTEMP=2,EF=3,CN=4,CP=5,DI1=6,DI2=7,DI3=8,DI4=9,
     * DI5=10,DI6=43,EG=11,SUM=12,CAYT=13,ECMEF=14,RDIEL=15,TPOWR=16,
     * EFME=17,EFMH=18,FNC=19,FNV=20,DN1=21,EN1=22,DG1=23,DN2=24,
     * EN2=25,DG2=26,DN3=27,EN3=28,DG3=29,DN4=30,EN4=31,DG4=32,DN5=33,
     * EN5=34,DG5=35,DN6=44,EN6=45,DG6=46,FNCP=36,FNVP=37,CNG=38,
     * CNL=39,CNX=40,DELGL=41,DELGX=42)
      INTEGER NUMER,SETNUM,NPT
      REAL DUMB,START,STOPIT,STEP,NDV,FTNUM
C
C     *** Format specifier specification ***
      PFORM(T)=      1
      PFORM(RTEMP)=  1
      PFORM(EF)=     2
      PFORM(CN)=     3
      PFORM(CP)=     3
      PFORM(DI1)=    3
```

```
      PFORM(DI2)=     3
      PFORM(DI3)=     3
      PFORM(DI4)=     3
      PFORM(DI5)=     3
      PFORM(DI6)=     3
      PFORM(EG)=      2
      PFORM(SUM)=     3
      PFORM(CAYT)=    3
      PFORM(ECMEF)=   2
      PFORM(RDIEL)=   1
      PFORM(TPOWR)=   3
      PFORM(EFME)=    3
      PFORM(EFMH)=    3
      PFORM(FNC)=     3
      PFORM(FNV)=     3
      PFORM(DN1)=     4
      PFORM(EN1)=     6
      PFORM(DG1)=     5
      PFORM(DN2)=     4
      PFORM(EN2)=     6
      PFORM(DG2)=     5
      PFORM(DN3)=     4
      PFORM(EN3)=     6
      PFORM(DG3)=     5
      PFORM(DN4)=     4
      PFORM(EN4)=     6
      PFORM(DG4)=     5
      PFORM(DN5)=     4
      PFORM(EN5)=     6
      PFORM(DG5)=     5
      PFORM(DN6)=     4
      PFORM(EN6)=     6
      PFORM(DG6)=     5
      PFORM(FNCP)=    3
      PFORM(FNVP)=    3
      PFORM(CNG)=     3
      PFORM(CNL)=     3
      PFORM(CNX)=     3
      PFORM(DELGL)=   3
      PFORM(DELGX)=   3
C     *** Variable description specification ***
      DSCRIP(T)=        '    Temp    '
      DSCRIP(RTEMP)=    '    1000/T  '
      DSCRIP(EF)=       '       EF   '
      DSCRIP(CN)=       '       n    '
      DSCRIP(CP)=       '       p    '
      DSCRIP(DI1)=      '     Nd1+   '
      DSCRIP(DI2)=      '     Nd2+   '
      DSCRIP(DI3)=      '     Nd3+   '
      DSCRIP(DI4)=      '     Na1-   '
      DSCRIP(DI5)=      '     Na2-   '
      DSCRIP(DI6)=      '     Na3-   '
      DSCRIP(EG)=       '     Eg     '
      DSCRIP(SUM)=      'Sum Charges'
      DSCRIP(CAYT)=     '     kT     '
      DSCRIP(ECMEF)=    '    Ec-EF   '
      DSCRIP(RDIEL)=    '    diel    '
      DSCRIP(TPOWR)=    ' Temp**3/2  '
      DSCRIP(EFME)=     '    efme    '
      DSCRIP(EFMH)=     '    efmh    '
      DSCRIP(FNC)=      '     Nc     '
      DSCRIP(FNV)=      '     Nv     '
      DSCRIP(DN1)=      '     Nd1    '
      DSCRIP(EN1)=      '     Ed1    '
      DSCRIP(DG1)=      '  Deg. D1   '
      DSCRIP(DN2)=      '     Nd2    '
```

```
      DSCRIP(EN2)=     '    Ed2    '
      DSCRIP(DG2)=     '  Deg. D2  '
      DSCRIP(DN3)=     '    Nd3    '
      DSCRIP(EN3)=     '    Ed3    '
      DSCRIP(DG3)=     '  Deg. D3  '
      DSCRIP(DN4)=     '    Na1    '
      DSCRIP(EN4)=     '    Ea1    '
      DSCRIP(DG4)=     '  Deg. A1  '
      DSCRIP(DN5)=     '    Na2    '
      DSCRIP(EN5)=     '    Ea2    '
      DSCRIP(DG5)=     '  Deg. A2  '
      DSCRIP(DN6)=     '    Na3    '
      DSCRIP(EN6)=     '    Ea3    '
      DSCRIP(DG6)=     '  Deg. A3  '
      DSCRIP(FNCP)=    '    Nc''     '
      DSCRIP(FNVP)=    '    Nv''     '
      DSCRIP(CNG)=     '    nG     '
      DSCRIP(CNL)=     '    nL     '
      DSCRIP(CNX)=     '    nX     '
      DSCRIP(DELGL)=   ' Delta EGL '
      DSCRIP(DELGX)=   ' Delta EGX '
C     *** Format specifiers ***
      FORMS(1)=       '(F10.4,2X)      '
      FORMS(2)=       '(1X,F11.6)      '
      FORMS(3)=       '(1X,1PE11.4)    '
      FORMS(4)=       '(1X,1PE10.3,1X)'
      FORMS(5)=       '(2X,F8.4,2X)    '
      FORMS(6)=       '(2X,F7.4,3X)    '
C
C     *** Logical unit specification (terminal) ***
CALL UNITS(LUTI,LUTO)
C
      NUMER=0
C
C     *** Main Program loop ***
      CALL MENU(SETNUM)
      CALL IMPTBL(PFORM,INDVAR,LUDEF,LSTSW,MATSW,TRMSW)
      CALL HEADER
      IF (((INDVAR.NE.T).AND.(INDVAR.NE.RTEMP)))
     *     CALL TVARS(INDVAR,MATSW)
      IF (LOGSW) THEN
           START=1.
           STEP=1.
           NDV=ABS(STPVAL)
           FTNUM=INT(ALOG10(AMIN1(FINVAL,INIVAL))*NDV)
           NPT=INT(ALOG10(AMAX1(FINVAL,INIVAL))*NDV)-FTNUM+2
           IF (ABS(FLOAT(INT(ALOG10(MAX(FINVAL,INIVAL))))-
     *         ALOG10(MAX(FINVAL,INIVAL))).LT.0.0002) NPT=NPT-1
           STOPIT=NPT
      ELSE
           START=INIVAL
           STOPIT=FINVAL
           STEP=STPVAL
      ENDIF
      DO 100 DUMB=START,STOPIT,STEP
           IF (LOGSW) THEN
                P(INDVAR)=10.**((FTNUM+DUMB-1)/NDV)
           ELSE
                P(INDVAR)=DUMB
           ENDIF
           CALL COMP(INDVAR,MATSW,SETNUM)
           IF (IFLAG.GT.2) THEN
                CALL ERHAN(INDVAR,NUMER,PFORM)
           ELSE
                CALL CUROUT
           ENDIF
```

```
100   CONTINUE
      IF (LSTSW) CALL CMPOUT(LUDEF,TRMSW,OUTNUM,START,STOPIT,STEP,NUMER)
      CALL WRAPUP(LUDEF,LSTSW,NUMER,PLTSW,PNUM)
C     Pause needed in MacFORTRAN
C     PAUSE
      STOP
      END
C
C Subroutines and functions follow in alphabetical order.
C
C Subroutine CMPOUT provides terminal output if a listing file is
C selected and a clean listing file if the number of output
C parameters is more than six (six fills the screen).
C
      SUBROUTINE CMPOUT(LUDEF,TRMSW,OUTNUM,START,STOPIT,STEP,NUMER)
C
      INTEGER LULST,LUTMP,LUER
      PARAMETER (LULST=21,LUTMP=26,LUER=27)
C
      INTEGER LUDEF,OUTNUM,NUMER
      LOGICAL TRMSW
      REAL DUMB,START,STOPIT,STEP
      CHARACTER*72 OUTSTR(6),TEMP*12
      INTEGER LUTO,LUTI
      COMMON /LUNIT/ LUTI,LUTO
      DATA OUTSTR /' ',' ',' ',' ',' ',' '/
      DATA TEMP /' '/
      STOPIT=STOPIT-NUMER*STEP
      IF (LUDEF.NE.LUTMP) THEN
           IF (TRMSW) THEN
                REWIND (UNIT=LUDEF)
10              READ (LUDEF,'(A)',ERR=60,END=60) OUTSTR(1)
                WRITE (LUTO,'(A)') OUTSTR(1)
                GOTO 10
           ENDIF
      ELSE
           N=OUTNUM/6
           ITEST=N*6
           IF (ITEST.NE.OUTNUM) N=N+1
           DO 40 L=1,N
                REWIND (UNIT=LUDEF)
                DO 20 I=1,N
                     LAST=6
                     IF (I.EQ.N) LAST=OUTNUM+6-N*6
                     DO 20 J=1,LAST
15                        READ (LUDEF,'(A)') TEMP
                          IF (TEMP.EQ.'            ') GOTO 15
                          OUTSTR(I)(12*(J-1)+1:12*J)=TEMP
20              CONTINUE
                LAST=72
                IF (L.EQ.N) LAST=OUTNUM*12-N*72+72
                WRITE (LULST,*) OUTSTR(L)(:LAST)
                IF (TRMSW.AND.(L.EQ.1)) WRITE (LUTO,*)
     *               OUTSTR(L)(:LAST)
                DO 40 DUMB=START,STOPIT,STEP
                     DO 30 I=1,N
                          LAST2=6
                          IF (I.EQ.N) LAST2=OUTNUM+6-N*6
                          DO 30 J=1,LAST2
                               READ (LUDEF,'(A)',ERR=40,END=40) TEMP
                               OUTSTR(I)(12*(J-1)+1:12*J)=TEMP
30                   CONTINUE
                     LAST=72
                     IF (L.EQ.N) LAST=OUTNUM*12+72-N*72
                     WRITE (LULST,*) OUTSTR(L)(:LAST)
                     IF (TRMSW.AND.(L.EQ.1)) WRITE (LUTO,*)
```

```
     *                    OUTSTR(L)(:LAST)
40         CONTINUE
45         IF (NUMER.NE.0) THEN
               REWIND (UNIT=LUER)
50             READ (LUER,'(A)',ERR=55,END=55) OUTSTR(1)
               IF (TRMSW) WRITE (LUTO,*) OUTSTR(1)
               WRITE(LULST,*) OUTSTR(1)
               GOTO 50
           ENDIF
       ENDIF
55     RETURN
60     IF (NUMER.NE.0) THEN
           REWIND (UNIT=LUER)
65         READ (LUER,'(A)',ERR=55,END=55) OUTSTR(1)
           WRITE(LUTO,*) OUTSTR(1)
           GOTO 65
       ENDIF
       END
C
C Subroutine COMP computes the electrical properties for a
C fixed value of one independent variable.
C
       SUBROUTINE COMP(INDVAR,MATSW,SETNUM)
C
       EXTERNAL TEST
       REAL P(46)
       COMMON /PARMS/ P
       INTEGER IFLAG
       COMMON /SWITCH/ IFLAG
C
       INTEGER T,RTEMP,EF,EG,SUM,ECMEF
       PARAMETER (T=1,RTEMP=2,EF=3,EG=11,SUM=12,ECMEF=14)
       REAL UPLIM
C
       REAL EGAP,FERMI,RE,AE
       INTEGER INDVAR,SETNUM,MATSW
C
       FERMI=-0.2
       IFLAG=0
       RE=1.0E-06
       AE=1.0E-06
C
       IF (MATSW.EQ.1) THEN
C      *** Silicon ***
           EGAP=1.4
           UPLIM=1688.
       ELSEIF (MATSW.EQ.2) THEN
C      *** Gallium Arsenide ***
           EGAP=1.8
           UPLIM=1511.
       ELSE
C ---option--- This option has been added for InP, but could be
C          changed to work for any semiconductor.
           EGAP=1.35
           UPLIM=1343.
       ENDIF
       IF ((INDVAR.EQ.T).OR.(INDVAR.EQ.RTEMP)) CALL TVARS(INDVAR,MATSW)
       IF (P(T).GT.UPLIM) THEN
           IFLAG=6
           RETURN
       ENDIF
       IF (SETNUM.NE.3) THEN
           CALL ZEROIN (TEST,FERMI,EGAP,RE,AE,IFLAG)
           P(EF)=FERMI
           IF (IFLAG.LE.2) THEN
               P(SUM)=TEST(FERMI)
```

```
                P(ECMEF)=P(EG)-P(EF)
            ENDIF
        ENDIF
        RETURN
        END
C
C Subroutine CUROUT outputs data concurrent with the computation.
C
        SUBROUTINE CUROUT
C
        INTEGER PFORM(46),INDVAR,OUTPUT(28),OUTNUM,PF(4,2),
       * PNUM,LUDEF,MAXOUT,MATSW
        LOGICAL TRMSW,LSTSW,PLTSW
        COMMON /GLOBAL/ PFORM,INDVAR,OUTPUT,OUTNUM,
       * PF,PNUM,LUDEF,TRMSW,LSTSW,PLTSW,MATSW,MAXOUT
        REAL P(46)
        COMMON /PARMS/ P
        CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
        COMMON /STRING/ NAMES,DSCRIP,FORMS
C
        INTEGER LUPL1,LUPL2,LUPL3,LUPL4
        PARAMETER (LUPL1=22,LUPL2=23,LUPL3=24,LUPL4=25,LUTMP=26)
C
        CHARACTER*72 OUTSTR(5)
        INTEGER LUPL(4)
        DATA OUTSTR /' ',' ',' ',' ',' '/
        DATA LUPL /LUPL1,LUPL2,LUPL3,LUPL4/
C
        IF (LUDEF.NE.LUTMP) THEN
            IF (TRMSW.OR.LSTSW) THEN
                DO 10 I=1,OUTNUM
                    WRITE (OUTSTR(1)(12*(I-1)+1:12*I),
     *                      FORMS(PFORM(OUTPUT(I)))) P(OUTPUT(I))
10              CONTINUE
                WRITE (LUDEF,*) OUTSTR(1)(:12*OUTNUM)
            ENDIF
        ELSE
            DO 20 I=1,OUTNUM
                WRITE (LUDEF,FORMS(PFORM(OUTPUT(I)))) P(OUTPUT(I))
20          CONTINUE
        ENDIF
        IF (PLTSW) THEN
            DO 30 I=1,PNUM
                WRITE (OUTSTR(1)(:12),FORMS(PFORM(PF(I,1)))) P(PF(I,1))
                WRITE (OUTSTR(1)(13:24),FORMS(PFORM(PF(I,2))))
     *                                                       P(PF(I,2))
                WRITE (LUPL(I),*) OUTSTR(1)(:24)
30          CONTINUE
        ENDIF
        RETURN
        END
C
C Subroutine DEVSEL obtains user's output device selection.
C
        SUBROUTINE DEVSEL(DEV,PNUM,PF,MAXOUT,MATSW)
C
        INTEGER LUTO,LUTI
        COMMON /LUNIT/ LUTI,LUTO
C
        CHARACTER OUTSTR(3)*80,INSTR*5
        INTEGER I,J,K,PF(4,2),PNUM,MAXOUT,MATSW
        LOGICAL DEV(3),DIGIT,INRNGE
C       *** Note that DIGIT and INRNGE are logical functions ***
        DATA OUTSTR /3*' '/
        DATA INSTR /' '/
C       ***  Create output device selection menu ***
```

```
      WRITE (LUTO,'(////)')
5     WRITE (LUTO,'(21X,''Output Device Selection Menu'',///)')
      OUTSTR(1) (:47)='                    1   Terminal      >        '
      OUTSTR(2) (:47)='                    2   Listing file  >        '
      OUTSTR(3) (:47)='                    3   Plot file(s)  >        '
      DO 10 I=1,3
           IF (DEV(I)) THEN
                OUTSTR(I) (48:50)='on '
           ELSE
                OUTSTR(I) (48:50)='off'
           ENDIF
           WRITE (LUTO,'(A,/)') OUTSTR(I)
10    CONTINUE
      IF (DEV(3).OR.(PNUM.GT.0)) WRITE (LUTO,
     *'(/,20X,''4   Change plot file(s)'',/)')
      CALL PROMPT(' To toggle, enter number(s) > ',2,19)
      READ (LUTI,'(A)') INSTR
C     *** Count the number of characters in the response ***
      DO 20 I=1,5
           IF (DIGIT(INSTR(I:I))) J=I
20    CONTINUE
C     *** Toggle logical switches ***
      DO 30 I=1,J
           IF (DIGIT(INSTR(I:I))) THEN
                READ (INSTR(I:I),'(I1)') K
                IF (INRNGE(1,3,K)) THEN
                     DEV(K)=(.NOT.DEV(K))
                ELSEIF (K.EQ.4) THEN
                     CALL PLTSEL(DEV,PNUM,PF,MAXOUT,MATSW)
                ENDIF
           ENDIF
30    CONTINUE
      IF (.NOT.((DEV(1)).OR.(DEV(2)).OR.(DEV(3)))) THEN
           WRITE (LUTO,'(//,'' At least one output device must'',
     *          '' be selected.'',//)')
           GOTO 5
      ENDIF
      IF (DEV(3).AND.(PNUM.EQ.0)) CALL PLTSEL(DEV,PNUM,PF,
     *                                        MAXOUT,MATSW)
      RETURN
      END
C
C Function DIGIT returns TRUE if the character (C) is a digit
C between 0 and 9.
C
      LOGICAL FUNCTION DIGIT (C)
      CHARACTER C
      DIGIT = (('0'.LE.C).AND.(C.LE.'9'))
      RETURN
      END
C
C Subroutine DOPSEL obtains the impurity information from the user.
C
      SUBROUTINE DOPSEL(INDVAR,PFORM)
C
      REAL P(46)
      COMMON /PARMS/ P
      CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
      COMMON /STRING/ NAMES,DSCRIP,FORMS
      LOGICAL DOPANT(6)
      COMMON /TSTCOM/ DOPANT
C
      INTEGER DN1,EN1,DG1,DN2,EN2,DG2,DN3,EN3,
     * DG3,DN4,EN4,DG4,DN5,EN5,DG5,DN6,EN6,DG6
      PARAMETER (DN1=21,EN1=22,DG1=23,DN2=24,EN2=25,DG2=26,DN3=27,
     * EN3=28,DG3=29,DN4=30,EN4=31,DG4=32,DN5=33,EN5=34,DG5=35,DN6=44,
```

```
     * EN6=45,DG6=46)
      INTEGER LUTO,LUTI
      COMMON /LUNIT/ LUTI,LUTO
C
      INTEGER DG(6),DN(6),EN(6),INDVAR,PFORM(46),RSPON,SELECT
      CHARACTER*2 TYPE(12),OUTSTR*80,INSTR*2,INSTR2*40
      LOGICAL DONE,ACTION(6),DIGIT,INRNGE
      DATA DN /DN1,DN2,DN3,DN4,DN5,DN6/
      DATA EN /EN1,EN2,EN3,EN4,EN5,EN6/
      DATA DG /DG1,DG2,DG3,DG4,DG5,DG6/
      DATA TYPE /'D1','D2','D3','A1','A2','A3',
     *           'd1','d2','d3','a1','a2','a3'/
      DATA OUTSTR,INSTR,INSTR2 /' ',' ',' '/
      DONE=.FALSE.
      WRITE (LUTO,'(////)')
1     IF (.NOT.DONE) THEN
          DO 5 I=1,6
               ACTION(I)=.FALSE.
5         CONTINUE
          WRITE (LUTO,'(30X,''Impurity Selection'',///)')
          WRITE (LUTO,'(5X,''Type    Dopants     Density(cm-3)'',
     *           ''  Energy(eV)  Degeneracy'')')
          WRITE (LUTO,'(5X,''----    -------     -------------'',
     *           ''  ---------   ----------'',/)')
          OUTSTR(:37)='                                      '
          DO 10 L=1,6
               OUTSTR(6:7)=TYPE(L)
               IF (DOPANT(L)) THEN
                    OUTSTR(13:25)=NAMES(L)
                    OUTSTR(62:62)=' '
                    IF (INDVAR.EQ.DN(L)) THEN
                         OUTSTR(25:36)='  ind. var.'
                         WRITE (OUTSTR(38:49),FORMS(PFORM(EN1)))
     *                        P(EN(L))
                         WRITE (OUTSTR(50:61),FORMS(PFORM(DG1)))
     *                        P(DG(L))
                    ELSEIF (INDVAR.EQ.EN(L)) THEN
                         WRITE (OUTSTR(25:36),FORMS(PFORM(DN1)))
     *                        P(DN(L))
                         OUTSTR(38:49)=' ind. var. '
                         WRITE (OUTSTR(50:61),FORMS(PFORM(DG1)))
     *                        P(DG(L))
                    ELSEIF (INDVAR.EQ.DG(L)) THEN
                         WRITE (OUTSTR(25:36),FORMS(PFORM(DN1)))
     *                        P(DN(L))
                        WRITE (OUTSTR(38:49),FORMS(PFORM(EN1)))
     *                        P(EN(L))
                         OUTSTR(50:62)='   ind. var.'
                    ELSE
                        WRITE (OUTSTR(25:36),FORMS(PFORM(DN1)))
     *                      P(DN(L))
                        WRITE (OUTSTR(38:49),FORMS(PFORM(EN1)))
     *                      P(EN(L))
                        WRITE (OUTSTR(50:61),FORMS(PFORM(DG1)))
     *                      P(DG(L))
                    ENDIF
               ELSE
                    OUTSTR(10:30)='   none                       '
                    OUTSTR(31:61)='                               '
               ENDIF
               WRITE (LUTO,'(1X,A61,/)') OUTSTR(:62)
10        CONTINUE
20        CALL PROMPT(' Enter type > ',2,5)
          READ (LUTI,'(A)') INSTR
          IF (INSTR.EQ.'  ') THEN
               DONE = .TRUE.
```

```
          ELSE
               SELECT=0
               DO 30 I=1,6
                    IF ((INSTR.EQ.TYPE(I)).OR.
     *                   (INSTR.EQ.TYPE(I+6))) SELECT=I
30             CONTINUE
               IF (SELECT.EQ.0) THEN
                    WRITE (LUTO,*)' Error: improper response. ',
     *                   ' Enter D1, D2, A1, A2, or A3'
                    GOTO 20
               END IF
          ENDIF
40        IF ((DOPANT(SELECT)).AND..NOT.DONE) THEN
               WRITE (LUTO,'(////,32X,'' Impurity Data Entry'',///)')
               WRITE (LUTO,'(5X,''Type    Dopants     Density(cm-3)'',
     *          ''  Energy(eV)  Degeneracy'')')
               WRITE (LUTO,'(5X,''----    -------     ------------'',
     *          ''  ---------   ----------'',/)')
               OUTSTR(6:7)=TYPE(SELECT)
               OUTSTR(13:25)=NAMES(SELECT)
               WRITE (OUTSTR(25:36),FORMS(PFORM(DN1))) P(DN(SELECT))
               WRITE (OUTSTR(38:49),FORMS(PFORM(EN1))) P(EN(SELECT))
               WRITE (OUTSTR(50:61),FORMS(PFORM(DG1))) P(DG(SELECT))
               WRITE (LUTO,*) OUTSTR(:61)
               WRITE (LUTO,2000)
2000                FORMAT(//,30X,'1   Name',/,30X,
     *                   '2   Density(cm-3)',/,30X,
     *                   '3   Energy(eV)',/,30X,
     *                   '4   Degeneracy',/,30X,
     *                   '5   Replace entry',/,30X,
     *                   '6   Delete entry',/)
               CALL PROMPT(' Enter number(s) > ',2,4)
               READ (LUTI,'(A)') INSTR2
C              *** count number of significant
C              characters in the response ***
               DO 50 I=1,40
               IF (DIGIT(INSTR2(I:I))) J=I
50             CONTINUE
C              *** toggle logical switches ***
               DO 60 I=1,J
               IF (DIGIT(INSTR2(I:I))) THEN
                    READ (INSTR2(I:I),'(I1)') K
                    IF (INRNGE(1,6,K)) ACTION(K)=.TRUE.
               ENDIF
60             CONTINUE
          ELSEIF (.NOT.DONE) THEN
               DOPANT(SELECT)=.TRUE.
               DO 65 I=1,4
65                  ACTION(I)=.TRUE.
          ENDIF
          IF (ACTION(6)) THEN
               DO 70 I=1,5
                    ACTION(I)=.FALSE.
70             CONTINUE
               DOPANT(SELECT)=.FALSE.
               NAMES(SELECT)='Not Selected '
          ENDIF
          IF (ACTION(5)) THEN
               DO 80 I=1,4
                    ACTION(I)=.TRUE.
80             CONTINUE
          ENDIF
          IF (ACTION(1)) THEN
               WRITE (LUTO,'(//,30X,''Impurity Data Entry'',/)')
               WRITE (LUTO,'(41X,''123456789012'')')
               CALL PROMPT(' Enter the name (up to 12 characters) > ',
```

```
     *                    0,0)
               READ (LUTI,'(A)') NAMES(SELECT)
          ENDIF
4005      FORMAT ('  Error: Response must be a positive, ',
     *    'real number')
90        IF (ACTION(2)) THEN
               CALL PROMPT(' Enter the density > ',2,0)
               READ (LUTI,*,ERR=9510) P(DN(SELECT))
               IF (P(DN(SELECT)).LT.0.) THEN
                    WRITE (LUTO,4005)
                    GOTO 90
               ENDIF
               IF (P(DN(SELECT)).GT.1.0E+21) THEN
                    WRITE (LUTO,*)' Error: The upper limit for ',
     *              'density is 1.0E+21'
                    GOTO 90
               ENDIF
          ENDIF
100       IF (ACTION(3)) THEN
               CALL PROMPT(' Enter the energy > ',2,0)
               READ (LUTI,*,ERR=9520) P(EN(SELECT))
               IF (P(EN(SELECT)).LT.0.) THEN
                    WRITE (LUTO,4005)
                    GOTO 100
               ENDIF
               IF (P(EN(SELECT)).GT.1.6) THEN
                    WRITE (LUTO,*)' Error: The upper limit for ',
     *              'energy is 1.6'
                    GOTO 100
               ENDIF
          ENDIF
110       IF (ACTION(4)) THEN
               CALL PROMPT(' Enter the degeneracy > ',2,0)
               READ (LUTI,*,ERR=9530) P(DG(SELECT))
               IF (P(DG(SELECT)).LT.0.) THEN
                    WRITE (LUTO,4005)
                    GOTO 110
               ENDIF
               IF (P(DG(SELECT)).GT.50.) THEN
                    WRITE (LUTO,*)' Error: The upper limit for ',
     *              'degeneracy is 50.'
                    GOTO 110
               ENDIF
          ENDIF
          GOTO 1
9510      WRITE (LUTO,*)' Error: response must be a real number'
          GOTO 90
9520      WRITE (LUTO,*)' Error: response must be a real number'
          GOTO 100
9530      WRITE (LUTO,*)' Error: response must be a real number'
          GOTO 110
       ENDIF
       RETURN
       END
C
C Subroutine ERHAN updates the error file.
C
       SUBROUTINE ERHAN(INDVAR,NUMER,PFORM)
C
       REAL P(46)
       COMMON /PARMS/ P
       CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
       COMMON /STRING/ NAMES,DSCRIP,FORMS
       INTEGER IFLAG
       COMMON /SWITCH/ IFLAG
C
```

```
      INTEGER LUTO,LUTI,LUER
      COMMON /LUNIT/ LUTI,LUTO
      PARAMETER (LUER=27)
      INTEGER INDVAR,PFORM(46)
      CHARACTER*80 OUTSTR
C
      IF (IFLAG.EQ.3) THEN
C           OUTSTR(:39)=' IFLAG = 3, At a value of 123456789012 '
           OUTSTR(:39)='            At a value of 123456789012 '
           OUTSTR(40:55)='for 12345678901:'
           WRITE (OUTSTR(27:38),FORMS(PFORM(INDVAR))) P(INDVAR)
           OUTSTR(44:54)=DSCRIP(INDVAR)
           WRITE (LUER,'(/)')
           WRITE (LUER,*) OUTSTR(:55)
           WRITE (LUER,10)
10         FORMAT(1X,' the convergence criterion was met, but ',
     *          'the true zero was')
           WRITE (LUER,*) ' not found.  The output is suppressed.'
           NUMER=NUMER+1
      ELSEIF (IFLAG.EQ.4) THEN
C           OUTSTR(:39)=' IFLAG = 4, At a value of 123456789012 '
           OUTSTR(:39)='            At a value of 123456789012 '
           OUTSTR(40:55)='for 12345678901:'
           WRITE (OUTSTR(27:38),FORMS(PFORM(INDVAR))) P(INDVAR)
           OUTSTR(44:54)=DSCRIP(INDVAR)
           WRITE (LUER,'(/)')
           WRITE (LUER,*) OUTSTR(:55)
           WRITE (LUER,20)
20         FORMAT(1X,' no root was found in the search interval.',
     *          '   The output is suppressed.')
           NUMER=NUMER+1
      ELSEIF (IFLAG.EQ.5) THEN
C           OUTSTR(:39)=' IFLAG = 5, At a value of 123456789012 '
           OUTSTR(:39)='            At a value of 123456789012 '
           OUTSTR(40:55)='for 12345678901:'
           WRITE (OUTSTR(27:38),FORMS(PFORM(INDVAR))) P(INDVAR)
           OUTSTR(44:54)=DSCRIP(INDVAR)
           WRITE (LUER,'(/)')
           WRITE (LUER,*) OUTSTR(:55)
           WRITE (LUER,30)
30         FORMAT(1X,' no zero was found after 500 iterations.',
     *          '   The output is suppressed.')
           NUMER=NUMER+1
      ELSEIF (IFLAG.EQ.6) THEN
C           OUTSTR(:39)=' IFLAG = 6, At a value of 123456789012 '
           OUTSTR(:39)='            At a value of 123456789012 '
           OUTSTR(40:55)='for 12345678901:'
           WRITE (OUTSTR(27:38),FORMS(PFORM(INDVAR))) P(INDVAR)
           OUTSTR(44:54)=DSCRIP(INDVAR)
           WRITE (LUER,'(/)')
           WRITE (LUER,*) OUTSTR(:55)
           WRITE (LUER,40)
40         FORMAT(1X,' Temperature exceeded the melting point ',
     *          'of the material.',/,1X,' The output is suppressed.')
           NUMER=NUMER+1
      ENDIF
      RETURN
      END
C
C Real Function FD performs the Fermi Dirac integral calculation.
C
      FUNCTION FD(J,ARG)
C
C Function FD defines the Fermi-Dirac integral of order J.
C The short series expansion of P. Van Halen and D. L. Pulfrey, Appl.
C Phys. Lett. 57, 5271-5274 (1985) is used to approximate the integral
```

```
C to within two parts in ten thousand for all real arguments.
C Corrected coefficients were received from David Pulfrey.
C
      INTEGER R
      REAL A(7),B(7),C(7),D(7)
      REAL A1(7),B1(7),C1(7),D1(7)
      REAL A2(7),B2(7),D2(7)
      REAL A3(7),B3(7),D3(7)
      REAL A4(7),B4(7),D4(7)
      REAL ARG,J,X
      DATA A1 /1.,.353568,.192439,.122973,.077134,.036228,.008346/
      DATA B1 /.765147,.604911,.189885,.020307,-.00438,-.000366,.000133/
      DATA C1 /.777114,.581307,.206132,.01768,-.006549,.000784,-.000036/
      DATA D1/.752253,.928195,.680839,25.7829,-553.636,3531.43,-3254.65/
      DATA A2/1.,.176826,.064722,.033677,.021353,.011451,.003032/
      DATA B2/.8672,.765101,.302693,.062718,.005793,-.001342,.000089/
      DATA D2/.300901,1.85581,-.466432,-7.71648,120.535,
     *-800.702,2189.84/
      DATA A3/1.,.088392,.021407,.007917,.003723,.001716,.000451/
      DATA B3/.927560,.866971,.383690,.098863,.017398,.000418,
     *-.000067/
      DATA D3/.085972,1.23738,1.07293,.362030,38.7579,-750.718,
     *4378.70/
      DATA A4/1.,.044203,.007157,.001976,.000719,.000317,.000106/
      DATA B4/.961478,.927751,.432494,.129617,.023308,.004067,
     *-.000051/
      DATA D4/.019105,.494958,2.13722,-.503902,-6.99243,96.6031,
     *-426.046/
C
      IF(J.EQ.0.5) THEN
           DO 5 R=1,7
                A(R)=A1(R)
                B(R)=B1(R)
                C(R)=C1(R)
                D(R)=D1(R)
5          CONTINUE
      ELSEIF (J.EQ.1.5) THEN
           DO 6 R=1,7
                A(R)=A2(R)
                B(R)=B2(R)
                C(R)=B2(R)
                D(R)=D2(R)
6          CONTINUE
      ELSEIF (J.EQ.2.5) THEN
           DO 7 R=1,7
              A(R)=A3(R)
              B(R)=B3(R)
              C(R)=B3(R)
              D(R)=D3(R)
7          CONTINUE
      ELSEIF (J.EQ.3.5) THEN
           DO 8 R=1,7
              A(R)=A4(R)
              B(R)=B4(R)
              C(R)=B4(R)
              D(R)=D4(R)
8          CONTINUE
      ELSE
           STOP
      ENDIF
C
      X=0
      IF(ARG.LE.0) THEN
           DO 10 R=1,7
                IF (R*ARG.GT.-85.) X=X+(-1)**(R+1)*A(R)*EXP(ARG*R)
10         CONTINUE
```

```
      ELSEIF((ARG.GT.0).AND.(ARG.LE.2.)) THEN
           DO 20 R=1,7
                X=X+B(R)*ARG**(R-1)
20         CONTINUE
      ELSEIF((ARG.GT.2.).AND.(ARG.LT.3.6999)) THEN
           DO 30 R=1,7
                X=X+C(R)*ARG**(R-1)
30         CONTINUE
      ELSE
           DO 40 R=1,7
                X=X+D(R)/ARG**(2*(R-1))
40         CONTINUE
           X=X*ARG**(J+1)
      ENDIF
C
      FD=X
      RETURN
      END
C
C Subroutine HEADER produces output headings.
C
      SUBROUTINE HEADER
C
      INTEGER PFORM(46),INDVAR,OUTPUT(28),OUTNUM,PF(4,2),
     * PNUM,LUDEF,MAXOUT,MATSW
      LOGICAL TRMSW,LSTSW,PLTSW
      COMMON /GLOBAL/ PFORM,INDVAR,OUTPUT,OUTNUM,
     * PF,PNUM,LUDEF,TRMSW,LSTSW,PLTSW,MATSW,MAXOUT
      CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
      COMMON /STRING/ NAMES,DSCRIP,FORMS
C
      INTEGER LUPL1,LUPL2,LUPL3,LUPL4,LUTMP
      PARAMETER (LUPL1=22,LUPL2=23,LUPL3=24,LUPL4=25,LUTMP=26)
C
      CHARACTER*72 OUTSTR
      INTEGER LUPL(4)
      DATA LUPL /LUPL1,LUPL2,LUPL3,LUPL4/
      DATA OUTSTR /' '/
      IF (LUDEF.NE.LUTMP) THEN
           IF (TRMSW.OR.LSTSW) THEN
                DO 10 I=1,OUTNUM
                     WRITE (OUTSTR(12*(I-1)+1:12*(I)),'(1X,A11)')
     *                    DSCRIP(OUTPUT(I))
10              CONTINUE
                WRITE (LUDEF,'(/)')
                WRITE (LUDEF,*) OUTSTR(:12*OUTNUM)
                DO 15 I=1,OUTNUM
                     WRITE (OUTSTR(12*(I-1)+1:12*(I)),'(1X,A11)')
     *                    '-----------'
15              CONTINUE
           ENDIF
      ELSE
           DO 20 I=1,OUTNUM
                WRITE (LUDEF,*) DSCRIP(OUTPUT(I))
20         CONTINUE
      ENDIF
      IF (PLTSW) THEN
           DO 30 I=1,PNUM
                OUTSTR(:12)=DSCRIP(PF(I,1))
                OUTSTR(13:24)=DSCRIP(PF(I,2))
                WRITE (LUPL(I),*) OUTSTR(:24)
30         CONTINUE
      ENDIF
      RETURN
      END
C
```

```
C Subroutine IMPTBL produces a table of impurity information.
C
      SUBROUTINE IMPTBL(PFORM,INDVAR,LUDEF,LSTSW,MATSW,TRMSW)
C
      REAL P(46)
      COMMON /PARMS/ P
      CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
      COMMON /STRING/ NAMES,DSCRIP,FORMS
      LOGICAL DOPANT(6)
      COMMON /TSTCOM/ DOPANT
C
      INTEGER T,RTEMP,DN1,EN1,DG1,DN2,EN2,DG2,DN3,EN3,
     * DG3,DN4,EN4,DG4,DN5,EN5,DG5,DN6,EN6,DG6
      PARAMETER (T=1,RTEMP=2,DN1=21,EN1=22,DG1=23,DN2=24,EN2=25,DG2=26,
     * DN3=27,EN3=28,DG3=29,DN4=30,EN4=31,DG4=32,DN5=33,EN5=34,DG5=35,
     * DN6=44,EN6=45,DG6=46)
      INTEGER LULST
      PARAMETER (LULST=21)
      INTEGER LUTI,LUTO
      COMMON /LUNIT/ LUTI,LUTO
C
      CHARACTER OUTSTR*80
      INTEGER DN(6),EN(6),DG(6),LUDEF,PFORM(46),INDVAR,MATSW
      LOGICAL LSTSW,TRMSW
      DATA DN /DN1,DN2,DN3,DN4,DN5,DN6/
      DATA EN /EN1,EN2,EN3,EN4,EN5,EN6/
      DATA DG /DG1,DG2,DG3,DG4,DG5,DG6/
      DATA OUTSTR /' '/
C
      IF (INDVAR.GT.RTEMP) THEN
           OUTSTR(:26)='Measurement Temperature = '
           WRITE (OUTSTR(27:38),FORMS(PFORM(T))) P(T)
           OUTSTR(39:53)='      1000/T = '
           WRITE (OUTSTR(54:65),FORMS(PFORM(RTEMP))) P(RTEMP)
      ENDIF
      IF (TRMSW.AND.(LUDEF.NE.LULST)) THEN
           IF (MATSW.EQ.1) THEN
                WRITE (LUTO,'(////,'' Electrical Properties of '',
     *               ''Silicon'',/)')
           ELSEIF (MATSW.EQ.2) THEN
                WRITE (LUTO,'(////,'' Electrical Properties of '',
     *               ''Gallium Arsenide'',/)')
           ELSE
C ---option---  This option has been added for InP, but could
C               be changed to any semiconductor name.
                WRITE (LUTO,'(////,'' Electrical Properties of '',
     *               ''Indium Phosphide'',/)')
           ENDIF
           IF (INDVAR.GT.RTEMP) THEN
                WRITE (LUTO,*) OUTSTR(:65)
           ENDIF
           WRITE (LUTO,150)
150        FORMAT(/,1X,'Dopants',7X,'Density(cm-3)',4X,'Energy(eV)',
     *          4X,'Degeneracy')
           WRITE (LUTO,155)
155        FORMAT(1X,'----------',4X,'-------------',4X,'----------',
     *          4X,'----------')
           DO 15 I=1,6
                IF (DOPANT(I)) THEN
                     IF (DN(I).EQ.INDVAR) THEN
                          WRITE (LUTO,251) NAMES(I),
     *                         ' ind. var. ',P(EN(I)),P(DG(I))
                     ELSEIF (EN(I).EQ.INDVAR) THEN
                          WRITE (LUTO,252) NAMES(I),
     *                         P(DN(I)),' ind. var. ',P(DG(I))
                     ELSEIF (DG(I).EQ.INDVAR) THEN
```

```
                         WRITE (LUTO,253) NAMES(I),
     *                   P(DN(I)),P(EN(I)),'   ind. var'
                    ELSE
                         WRITE (LUTO,250) NAMES(I),
     *                   P(DN(I)),P(EN(I)),P(DG(I))
                    ENDIF
               ENDIF
15        CONTINUE
250       FORMAT(1X,A13,2X,1PE10.3,0P,6X,F8.4,6X,F8.4)
251       FORMAT(1X,A13,2X,A11,0P,5X,F8.4,6X,F8.4)
252       FORMAT(1X,A13,2X,1PE10.3,0P,6X,A11,3X,F8.4)
253       FORMAT(1X,A13,2X,1PE10.3,0P,6X,F8.4,3X,A12)
          IF(.NOT.((DOPANT(1)).OR.(DOPANT(2)).OR.(DOPANT(3)).OR.
     *         (DOPANT(4)).OR.(DOPANT(5)).OR.(DOPANT(6))))
     *         WRITE (LUTO,'(1X,''none'')')
          IF (LSTSW) WRITE (LUTO,'(/)')
      ENDIF
      IF (LSTSW) THEN
           IF (MATSW.EQ.1) THEN
                WRITE (LULST,'('' Electrical Properties of '',
     *               ''Silicon'',/)')
           ELSEIF (MATSW.EQ.2) THEN
                WRITE (LULST,'(////,'' Electrical Properties of '',
     *               ''Gallium Arsenide'',/)')
           ELSE
C ---option---  This option has been added for InP, but could be
C               changed to any semiconductor name.
                WRITE (LULST,'(////,'' Electrical Properties of '',
     *               ''Indium Phosphide'',/)')
           ENDIF
           IF (INDVAR.GT.RTEMP) THEN
                WRITE (LULST,*) OUTSTR(:65)
           ENDIF
           WRITE (LULST,150)
           WRITE (LULST,155)
           DO 25 I=1,6
                IF (DOPANT(I)) THEN
                     IF (DN(I).EQ.INDVAR) THEN
                          WRITE (LULST,251) NAMES(I),
     *                    ' ind. var. ',P(EN(I)),P(DG(I))
                     ELSEIF (EN(I).EQ.INDVAR) THEN
                          WRITE (LULST,252) NAMES(I),
     *                    P(DN(I)),' ind. var. ',P(DG(I))
                     ELSEIF (DG(I).EQ.INDVAR) THEN
                          WRITE (LULST,253) NAMES(I),
     *                    P(DN(I)),P(EN(I)),'   ind. var'
                     ELSE
                          WRITE (LULST,250) NAMES(I),
     *                    P(DN(I)),P(EN(I)),P(DG(I))
                     ENDIF
                ENDIF
25         CONTINUE
           IF(.NOT.((DOPANT(1)).OR.(DOPANT(2)).OR.(DOPANT(3)).OR.
     *         (DOPANT(4)).OR.(DOPANT(5)).OR.(DOPANT(6))))
     *         WRITE (LULST,'(1X,''none'')')
      ENDIF
      RETURN
      END
C
C Subroutine INDSEL obtains information about independent variable.
C
      SUBROUTINE INDSEL(INDVAR,PFORM,MATSW)
C
      REAL INIVAL,FINVAL,STPVAL
      LOGICAL LOGSW
      COMMON /RANGE/ INIVAL,FINVAL,STPVAL,LOGSW
```

```
      REAL P(46)
      COMMON /PARMS/ P
      CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
      COMMON /STRING/ NAMES,DSCRIP,FORMS
      LOGICAL DOPANT(6)
      COMMON /TSTCOM/ DOPANT
      INTEGER LUTO,LUTI
      COMMON /LUNIT/ LUTI,LUTO
C
      INTEGER T,RTEMP,EN1,DG1,DN2,EN2,DG2,DN3,EN3,
     * DG3,DN4,EN4,DG4,DN5,EN5,DG5,DN6,EN6,DG6
      PARAMETER (T=1,RTEMP=2,DN1=21,EN1=22,DG1=23,DN2=24,EN2=25,DG2=26,
     * DN3=27,EN3=28,DG3=29,DN4=30,EN4=31,DG4=32,DN5=33,EN5=34,DG5=35,
     * DN6=44,EN6=45,DG6=46)
C
      INTEGER INDSET(20),INDVAR,ITEST,PFORM(46),RSPON,NDV,OTHER,
     * DN(6),EN(6),DG(6),MATSW
C     *** Note that DIGIT and INRNGE are logical functions ***
      LOGICAL MOD(5),DIGIT,INRNGE,TOOLO,TOOHI
      CHARACTER OUTSTR*80,INSTR*15,YES*1,CH
      REAL UPLIM,LOWLIM,T01,T02,UPCHK,LOWCHK,SIZE
      DATA INDSET /T,RTEMP,DN1,EN1,DG1,DN2,EN2,DG2,DN3,
     *     EN3,DG3,DN4,EN4,DG4,DN5,EN5,DG5,DN6,EN6,DG6/
      DATA DN /DN1,DN2,DN3,DN4,DN5,DN6/
      DATA EN /EN1,EN2,EN3,EN4,EN5,EN6/
      DATA DG /DG1,DG2,DG3,DG4,DG5,DG6/
      DATA OUTSTR /' '/
C
      DO 5 I=1,5
           MOD(I)=.FALSE.
5     CONTINUE
      IF (MATSW.EQ.1) THEN
C     *** Silicon ***
           UPLIM=1688.0
      ELSEIF (MATSW.EQ.2) THEN
C     *** Gallium Arsenide ***
           UPLIM=1511.0
      ELSE
C ---option--- Option added for melting point of InP, this could be
C          changed to work for any semiconductor.
           UPLIM=1343.0
      ENDIF
      LOWLIM=2.5
C     *** Produce the independent variable selection menu ***
      WRITE (LUTO,'(////)')
15    INSTR='               '
      WRITE (LUTO,1000)
1000  FORMAT(20X,'Independent Variable Selection Menu',//)
      OUTSTR(:42)='                    Independent variable > '
      OUTSTR(43:54) = DSCRIP(INDVAR)
      WRITE (LUTO,*) OUTSTR(:54)
      OUTSTR(:42)='                                   Start > '
      WRITE (OUTSTR(43:54),FORMS(PFORM(INDVAR))) INIVAL
      WRITE (LUTO,*) OUTSTR(:54)
      OUTSTR(:42)='                                    Stop > '
      WRITE (OUTSTR(43:54),FORMS(PFORM(INDVAR))) FINVAL
      WRITE (LUTO,*) OUTSTR(:54)
      IF (LOGSW) THEN
           OUTSTR(:42)='                       Divisions/Decade > '
      ELSE
           OUTSTR(:42)='                                   Step > '
      ENDIF
      WRITE (OUTSTR(43:54),FORMS(PFORM(INDVAR))) STPVAL
      WRITE (LUTO,*) OUTSTR(:54)
      IF ((INDVAR.NE.T).AND.(INDVAR.NE.RTEMP)) THEN
           OUTSTR(:42)='                    Temperature (K)     > '
```

```
          WRITE (OUTSTR(43:54),FORMS(PFORM(T))) P(T)
          WRITE (LUTO,'(1X,/,A54)') OUTSTR(:54)
     ENDIF
     WRITE (LUTO,'(//,25X,''1   New independent variable'',/)')
     WRITE (LUTO,'(25X,''2   New starting value'',/)')
     WRITE (LUTO,'(25X,''3   New final value'',/)')
     WRITE (LUTO,'(25X,''4   New step (log or linear)  '',/)')
     IF (((INDVAR.NE.T).AND.(INDVAR.NE.RTEMP))) THEN
          WRITE(LUTO,'(25X,''5   New temperature '',/)')
     ENDIF
     CALL PROMPT(' Enter number(s) > ',1,28)
     READ (LUTI,'(A)') INSTR
C    *** Count the number of characters in the response ***
     J=0
     DO 10 I=1,15
          IF (DIGIT (INSTR(I:I))) J=I
10   CONTINUE
C    *** Flag true, the menu item selected for change ***
     DO 20 I=1,J
          IF (DIGIT (INSTR(I:I))) THEN
               READ (INSTR(I:I),'(I1)') K
               IF (INRNGE(1,5,K)) MOD(K)=.TRUE.
          ENDIF
20   CONTINUE
C    *** Execute options selected ***
     WRITE (LUTO,'(////)')
40   IF (MOD(1)) THEN
          DO 30 I=2,4
               MOD(I)=.TRUE.
30        CONTINUE
          IF (INDVAR.LE.RTEMP) MOD(5)=.TRUE.
C         *** Just turned other options on ***
C
C         *** Create variable menu ***
C
          WRITE (LUTO,'(25X,''New'',
     *         '' Independent Variable Selection'',///)')
          WRITE (LUTO,'(33X,''1   Temperature (K)'',/)')
          WRITE (LUTO,'(33X,''2   1000/Temp  '',//)')
          WRITE (LUTO,'(1X,''  Impurity   '',6X,''Density(cm-3)'',
     *         6X,'' Energy(eV)  '',6X,''  Degeneracy '')')
          WRITE (LUTO,'(1X,'' ----------  '',6X,''-------------'',
     *         6X,'' ----------  '',6X,'' ------------'',/)')
          WRITE (LUTO,'(1X,A,T28,''3'',
     *         T47,''4'',T66,''5'',/)') NAMES(1)
          WRITE (LUTO,'(1X,A,T28,''6'',
     *         T47,''7'',T66,''8'',/)') NAMES(2)
          WRITE (LUTO,'(1X,A,T28,''9'',
     *         T46,''10'',T65,''11'',/)') NAMES(3)
          WRITE (LUTO,'(1X,A,T27,''12'',
     *         T46,''13'',T65,''14'',/)') NAMES(4)
          WRITE (LUTO,'(1X,A,T27,''15'',
     *         T46,''16'',T65,''17'',/)') NAMES(5)
          WRITE (LUTO,'(1X,A,T27,''18'',
     *         T46,''19'',T65,''20'',/)') NAMES(6)
C         *** Install new independent variable ***
          CALL PROMPT(' Enter number > ',1,19)
          READ (LUTI,*,ERR=9500) RSPON
          IF (INRNGE (1,20,RSPON)) THEN
               ITEST=RSPON/3
               IF ((ITEST.GE.1).AND.(.NOT.DOPANT(ITEST)))THEN
                    WRITE (LUTO,'(1X,'' Error:  parameter is '',
     *                   ''inactive, reselect. '')')
                    WRITE (LUTO,'(1X,''Hit <RETURN> to continue'')')
                    READ (LUTI,'(A)') YES
                    GOTO 40
```

```
                ENDIF
                INDVAR=INDSET(RSPON)
           ELSE
                WRITE(LUTO,*)' Error: Response out of range'
                WRITE (LUTO,'(1X,''Hit <RETURN> to continue'')')
                READ (LUTI,'(A)') YES
                GOTO 40
           ENDIF
      ENDIF
C     *** Starting value... ***
60    TOOLO=.FALSE.
      TOOHI=.FALSE.
      IF (MOD(2)) THEN
           CALL PROMPT(' Enter the starting value > ',1,0)
           READ (LUTI,*,ERR=9600) INIVAL
           P(INDVAR)=INIVAL
           IF (INDVAR.EQ.T) P(RTEMP) = 1000./P(T)
           IF (INDVAR.EQ.RTEMP) P(T)= 1000./P(RTEMP)
      ENDIF
C     *** Final value... ***
70    IF (MOD(3)) THEN
           CALL PROMPT(' Enter the final value > ',1,0)
           READ (LUTI,*,ERR=9700) FINVAL
      ENDIF
C     *** Make sure Temperature is in an acceptable range ***
      IF (INDVAR.LE.RTEMP) THEN
           IF (INDVAR.EQ.T) THEN
                OTHER=RTEMP
           ELSE
                OTHER=T
           ENDIF
           P(INDVAR)=INIVAL
           P(OTHER)=1000./P(INDVAR)
           T01=P(T)
           P(INDVAR)=FINVAL
           P(OTHER)=1000./P(INDVAR)
           T02=P(T)
           UPCHK=AMAX1(T01,T02)
           LOWCHK=AMIN1(T01,T02)
           IF (LOWCHK.LT.LOWLIM) THEN
                WRITE (LUTO,*)' Error: Temperature must always be ',
     *          'greater than'
                WRITE (LUTO,*)'        2.5 degrees (absolute) for ',
     *          'computational'
                WRITE (LUTO,*)'        reasons'
                TOOLO=.TRUE.
           ENDIF
           IF (UPCHK.GT.UPLIM) THEN
                WRITE (LUTO,*)' Error: Calculation attempted past the ',
     *          'melting point'
                WRITE (LUTO,'(9X,''of the material, '',F6.1,
     *          '' degrees (absolute)'')') UPLIM
                TOOHI=.TRUE.
           ENDIF
           IF (TOOHI.OR.TOOLO) GOTO 60
           P(INDVAR)=INIVAL
           P(OTHER)=1000./P(INDVAR)
      ELSE
C     *** Check for unreasonable impurity parameters ***
          DO 83 I=1,6
          IF ((INDVAR.EQ.DN(I)).AND.(AMAX1(FINVAL,INIVAL).GT.1.0E+21))
     *    THEN
               WRITE (LUTO,*)' Error: Maximum allowed density',
     *         ' is 1.0E+21'
               TOOHI=.TRUE.
          ENDIF
```

```
          IF ((INDVAR.EQ.EN(I)).AND.(AMAX1(FINVAL,INIVAL).GT.1.6)) THEN
               WRITE (LUTO,*)' Error: Maximum allowed energy is 1.6'
               TOOHI=.TRUE.
          ENDIF
          IF ((INDVAR.EQ.DG(I)).AND.(AMAX1(FINVAL,INIVAL).GT.50.)) THEN
               WRITE (LUTO,*)' Error: Maximum allowed degeneracy is 50'
               TOOHI=.TRUE.
          ENDIF
83        CONTINUE
          IF ((INIVAL.LT.0.).OR.(FINVAL.LT.0.)) THEN
               WRITE (LUTO,*)' Error: Starting and Final values must'
               WRITE (LUTO,*)'        be positive real numbers.'
               TOOLO=.TRUE.
          ENDIF
          IF (TOOLO.OR.TOOHI) GOTO 60
       ENDIF
C      *** Step... ***
72     IF (MOD(4)) THEN
            IF (LOGSW) THEN
                 CALL PROMPT(' Do you want to change '//
     *                       'to linear scale ? (y/n) > ',1,0)
            ELSE
                 CALL PROMPT(' Do you want to change '//
     *                       'to log scale ? (y/n) > ',1,0)
            ENDIF
            READ (LUTI,'(A)') YES
            IF ((YES.EQ.'Y').OR.(YES.EQ.'y')) LOGSW=.NOT.LOGSW
            IF (LOGSW) THEN
75               CALL PROMPT(' Enter the integer number of'//
     *                       ' divisions/decade > ',1,0)
                 READ (LUTI,*,ERR=75) NDV
                 STPVAL=NDV
            ELSE
80               CALL PROMPT(' Enter the step value  > ',1,0)
                 READ (LUTI,*,ERR=80) STPVAL
            ENDIF
       ENDIF
C      ***  Make sure step is reasonable ***
       IF (STPVAL.EQ.0) THEN
            WRITE (LUTO,*)' Error: Improper response or zero entered'
            WRITE (LUTO,*)'        for step or div./decade'
            GOTO 72
       ENDIF
       IF (((INIVAL.GT.FINVAL).AND.(STPVAL.GT.0.0)).OR.
     *     ((INIVAL.LT.FINVAL).AND.(STPVAL.LT.0.0))) STPVAL=-STPVAL
       IF (LOGSW.AND.((INIVAL.EQ.0.).OR.(FINVAL.EQ.0.))) THEN
            WRITE (LUTO,*)' Error: Logarithmic stepping is not possible'
            WRITE (LUTO,*)'        when starting or final value is zero'
            MOD(4)=.TRUE.
            GOTO 72
       ENDIF
       IF (LOGSW) STPVAL=ABS(STPVAL)
       I1=0
       I2=0
       J=PFORM(INDVAR)
       DO 85 I=1,15
            CH = FORMS(J)(I:I)
            IF (CH.EQ.'F') I1=I
            IF (CH.EQ.'.') I2=I
85     CONTINUE
       IF (I1.NE.0) THEN
            I1=I1+1
            I3=I2+1
            I2=I2-1
            READ (FORMS(J)(I1:I2),'(I2)') K1
            READ (FORMS(J)(I3:I3),'(I1)') K2
```

```
            SIZE=10**(K1-K2-1)-1
            IF (STPVAL.LT.0.) SIZE=SIZE/10.
            IF (ABS(STPVAL).GT.SIZE) THEN
                 WRITE (LUTO,*)' Error: Step or div/decade is too large'
                 IF (STPVAL.GT.0)
     *            WRITE (LUTO,'(9X,''a positive number for'',
     *            A)') DSCRIP(INDVAR)
                 IF (STPVAL.LT.0)
     *            WRITE (LUTO,'(9X,''a negative number for'',
     *            A)') DSCRIP(INDVAR)
                 MOD(4)=.TRUE.
                 GOTO 72
            ENDIF
       ENDIF
C      *** Temperature... ***
       IF ((MOD(5)).AND.(INDVAR.GT.RTEMP)) CALL TSEL(MATSW)
       DO 90 I=1,5
            MOD(I)=.FALSE.
90     CONTINUE
       RETURN
9500   WRITE (LUTO,*)' Error: response must be an integer'
       WRITE (LUTO,'(1X,''Hit <RETURN> to continue'')')
       READ (LUTI,'(A)') YES
       GOTO 40
9600   WRITE (LUTO,*)' Error: response must be in F, D, or E format'
       GOTO 60
9700   WRITE (LUTO,*)' Error: response must be in F, D, or E format'
       GOTO 70
       END
C
C Subroutine INPSEL allows user to select various input parameters.
C
       SUBROUTINE INPSEL(INDVAR,PFORM,MATSW)
C
       REAL INIVAL,FINVAL,STPVAL
       LOGICAL LOGSW
       COMMON /RANGE/ INIVAL,FINVAL,STPVAL,LOGSW
       REAL P(46)
       COMMON /PARMS/ P
       CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
       COMMON /STRING/ NAMES,DSCRIP,FORMS
       LOGICAL DOPANT(6)
       COMMON /TSTCOM/ DOPANT
C
       INTEGER T,RTEMP,DN1,EN1,DG1,DN2,EN2,DG2,DN3,EN3,
      * DG3,DN4,EN4,DG4,DN5,EN5,DG5,DN6,EN6,DG6
       PARAMETER (T=1,RTEMP=2,DN1=21,EN1=22,DG1=23,DN2=24,EN2=25,DG2=26,
      * DN3=27,EN3=28,DG3=29,DN4=30,EN4=31,DG4=32,DN5=33,EN5=34,DG5=35,
      * DN6=44,EN6=45,DG6=46)
       INTEGER LUTI,LUTO
       COMMON /LUNIT/ LUTI,LUTO
C
       INTEGER DN(6),EN(6),DG(6),INDVAR,PFORM(46),RSPON,MATSW
       CHARACTER OUTSTR*80,CH*1
       LOGICAL DONE,DIGIT
       DATA OUTSTR /' '/
       DATA CH /' '/
       DATA DN /DN1,DN2,DN3,DN4,DN5,DN6/
       DATA EN /EN1,EN2,EN3,EN4,EN5,EN6/
       DATA DG /DG1,DG2,DG3,DG4,DG5,DG6/
       DONE=.FALSE.
       WRITE (LUTO,'(////)')
10     IF (.NOT.DONE) THEN
            WRITE (LUTO,1000)
1000        FORMAT(/,25X,'Input Data Selection Menu',///)
            OUTSTR(:32)='      1  Independent variable > '
```

```
      OUTSTR(33:44)=DSCRIP(INDVAR)
      WRITE (LUTO,*) OUTSTR(:44)
      OUTSTR(:32)='                              Start > '
      WRITE (OUTSTR(33:44),FORMS(PFORM(INDVAR))) INIVAL
      WRITE (LUTO,*) OUTSTR(:44)
      OUTSTR(:32)='                              Stop  > '
      WRITE (OUTSTR(33:44),FORMS(PFORM(INDVAR))) FINVAL
      WRITE (LUTO,*) OUTSTR(:44)
      IF (LOGSW) THEN
           OUTSTR(:32)='            Divisions/Decade  > '
      ELSE
           OUTSTR(:32)='                         Step  > '
      ENDIF
      WRITE (OUTSTR(33:44),FORMS(PFORM(INDVAR))) STPVAL
      WRITE (LUTO,*) OUTSTR(:44)
      IF (((INDVAR.LT.1).OR.(INDVAR.GT.2))) THEN
           OUTSTR(:32)='          Temperature (K)     > '
           WRITE (OUTSTR(33:44),FORMS(PFORM(T))) P(T)
           WRITE (LUTO,'(1X,/,A44,/)') OUTSTR(:44)
      ENDIF
      WRITE (LUTO,'(7X,''2  Impurity data:  '')')
      WRITE (LUTO,2000)
2000  FORMAT(1X,/,10X,'Dopants',8X,'Density(cm-3)',5X,
     *     'Energy(eV)',6X,'Degeneracy')
      WRITE (LUTO,2005)
2005  FORMAT(10X,'-------',8X,'-------------',5X,
     *     '----------',6X,'----------')
      DO 50 L=1,6
           IF (DOPANT(L)) THEN
                IF (INDVAR.EQ.DN(L)) THEN
                     WRITE (LUTO,3000)NAMES(L),
     *                    ' ind. var. ',P(EN(L)),P(DG(L))
                ELSE
                     IF (INDVAR.EQ.EN(L)) THEN
                          WRITE (LUTO,3010)NAMES(L),
     *                         P(DN(L)),' ind. var. ',P(DG(L))
                     ELSE
                          IF (INDVAR.EQ.DG(L)) THEN
                               WRITE (LUTO,3020)NAMES(L),
     *                              P(DN(L)),P(EN(L)),'  ind. var.'
                          ELSE
                               WRITE (LUTO,3030)NAMES(L),
     *                              P(DN(L)),P(EN(L)),P(DG(L))
                          ENDIF
                     ENDIF
                ENDIF
           ENDIF
50    CONTINUE
3000  FORMAT(10X,A13,3X,A11,5X,F8.4,8X,F8.1)
3010  FORMAT(10X,A13,3X,1PE11.4,5X,A11,5X,0PF8.1)
3020  FORMAT(10X,A13,3X,1PE11.4,5X,0PF8.4,8X,A11)
3030  FORMAT(10X,A13,3X,1PE11.4,5X,0PF8.4,8X,F8.1)
      IF (.NOT.((DOPANT(1)).OR.(DOPANT(2)).OR.(DOPANT(3))
     *     .OR.(DOPANT(4)).OR.(DOPANT(5)).OR.(DOPANT(6))))
     *     WRITE (LUTO,'(10X,''none'')')
C      WRITE (LUTO,'(/,7X,''3  Exit'',/)')
      CALL PROMPT(' Enter number > ',2,9)
      READ (LUTI,'(A)',ERR=9500) CH
      IF (DIGIT(CH)) THEN
           READ (CH,'(I1)') RSPON
      ELSEIF (CH.EQ.' ') THEN
           RSPON=3
      ELSE
           GOTO 9500
      ENDIF
      IF (RSPON.EQ.0) RSPON=3
```

```
            IF (RSPON.EQ.1) THEN
                  CALL INDSEL(INDVAR,PFORM,MATSW)
            ELSEIF (RSPON.EQ.2) THEN
                  CALL DOPSEL(INDVAR,PFORM)
            ELSEIF (RSPON.EQ.3) THEN
                  DONE=.TRUE.
            ELSE
                  WRITE (LUTO,*)' Error: improper response, try again'
                  WRITE (LUTO,*)'        Hit <RETURN> to continue.'
                  READ (LUTI,'(A)') CH
            ENDIF
            GOTO 10
      ENDIF
      RETURN
9500  WRITE (LUTO,*)' Error: response must be an integer'
      WRITE (LUTO,*)'        Hit <RETURN> to continue.'
      READ (LUTI,'(A)') CH
      GOTO 10
      END
C
C Logical function INRNGE returns true if VAR is in the range [LOW,UP].
C
      LOGICAL FUNCTION INRNGE (LOW,UP,VAR)
C
      INTEGER LOW,UP,VAR
      INRNGE=((VAR.GE.LOW).AND.(VAR.LE.UP))
      RETURN
      END
C
C Subroutine MATSEL selects gallium arsenide, silicon,
C or other semiconductor.
C
      SUBROUTINE MATSEL(MATSW)
C
      INTEGER LUTI,LUTO
      COMMON /LUNIT/ LUTI,LUTO
C
      INTEGER MATSW
      CHARACTER*1 CH
      DATA CH /' '/
10    WRITE (LUTO,'(////,30X,''Material Selection'',///)')
      WRITE (LUTO,'(30X,''S   Silicon'',/)')
      WRITE (LUTO,'(30X,''G   Gallium Arsenide'',/)')
C ---option--- Uncommenting the next statement will enable EPROP to
C     compute for InP.  No temperature dependent parameters
C     have been included, however, and the constants used are not
C     necessarily the best available.  The InP example illustrates
C     how other impurities can be added.
C     WRITE (LUTO,'(30X,''I   Indium Phosphide'',/)')
      CALL PROMPT('     Enter letter > ',2,25)
      READ (LUTI,'(A)') CH
      IF ((CH.EQ.'S').OR.(CH.EQ.'s')) THEN
            MATSW=1
      ELSEIF ((CH.EQ.'G').OR.(CH.EQ.'g')) THEN
            MATSW=2
      ELSEIF ((CH.EQ.'I').OR.(CH.EQ.'i')) THEN
            MATSW=3
      ELSEIF (CH.EQ.' ') THEN
            RETURN
      ELSE
            WRITE (LUTO,*)' Error: improper response'
            GOTO 10
      ENDIF
      RETURN
      END
C
```

```
C Subroutine MENU sets up all of the switches and parameters needed
C for the operation of the program. It produces the main menu and
C calls the routines which produce all of the submenus.
C
      SUBROUTINE MENU(SETNUM)
C
      REAL INIVAL,FINVAL,STPVAL
      LOGICAL LOGSW
      COMMON /RANGE/ INIVAL,FINVAL,STPVAL,LOGSW
      INTEGER PFORM(46),INDVAR,OUTPUT(28),OUTNUM,PF(4,2),
     * PNUM,LUDEF,MAXOUT,MATSW
      LOGICAL TRMSW,LSTSW,PLTSW
      COMMON /GLOBAL/ PFORM,INDVAR,OUTPUT,OUTNUM,
     * PF,PNUM,LUDEF,TRMSW,LSTSW,PLTSW,MATSW,MAXOUT
      REAL P(46)
      COMMON /PARMS/ P
      CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
      COMMON /STRING/ NAMES,DSCRIP,FORMS
      LOGICAL DOPANT(6)
      COMMON /TSTCOM/ DOPANT
      INTEGER LUTO,LUTI
      COMMON /LUNIT/ LUTI,LUTO
C
      INTEGER T,RTEMP,EF,CN,CP,DI1,DI2,DI3,DI4,DI5,EG,SUM,CAYT,ECMEF,
     * RDIEL,TPOWR,EFME,EFMH,FNC,FNV,DN1,EN1,DG1,DN2,EN2,DG2,DN3,EN3,
     * DG3,DN4,EN4,DG4,DN5,EN5,DG5,FNCP,FNVP,CNG,CNL,CNX,DELGL,DELGX,
     * DI6,DN6,EN6,DG6
      PARAMETER (T=1,RTEMP=2,EF=3,CN=4,CP=5,DI1=6,DI2=7,DI3=8,DI4=9,
     * DI5=10,EG=11,SUM=12,CAYT=13,ECMEF=14,RDIEL=15,TPOWR=16,EFME=17,
     * EFMH=18,FNC=19,FNV=20,DN1=21,EN1=22,DG1=23,DN2=24,EN2=25,DG2=26,
     * DN3=27,EN3=28,DG3=29,DN4=30,EN4=31,DG4=32,DN5=33,EN5=34,DG5=35,
     * FNCP=36,FNVP=37,CNG=38,CNL=39,CNX=40,DELGL=41,DELGX=42,DI6=43,
     * DN6=44,EN6=45,DG6=46)
      INTEGER LUIN,LULST,LUPL1,LUPL2,
     * LUPL3,LUPL4,LUTMP,LUER
      PARAMETER (LUIN=20,LULST=21,LUPL1=22,LUPL2=23,
     * LUPL3=24,LUPL4=25,LUTMP=26,LUER=27)
C
      INTEGER SETNUM,DN(6),EN(6),DG(6),RSPON,HYBSET(28),HYBNUM,
     * SHTSET(10),FULSET(28),CSTSET(12),LUPL(4)
      CHARACTER OUTSTR*80,CH*1,OUTSET(4)*14,PLNAM(4)*9,INSTR*4
      LOGICAL DEV(3),DONE,INRNGE,UNDEFP,UNDEFO,DIGIT
      DATA DN /DN1,DN2,DN3,DN4,DN5,DN6/
      DATA EN /EN1,EN2,EN3,EN4,EN5,EN6/
      DATA DG /DG1,DG2,DG3,DG4,DG5,DG6/
      DATA OUTSET /'short set     ','full set      ',
     * 'constants     ','user specified'/
      DATA SHTSET /RTEMP,EF,CN,CP,DI1,DI4,DI2,DI5,DI3,DI6/
      DATA FULSET /T,RTEMP,EF,CN,CP,DI1,DI2,DI3,DI4,DI5,DI6,EG,SUM,
     *           CAYT,ECMEF,RDIEL,TPOWR,EFME,EFMH,FNC,FNV,FNCP,FNVP,
     *           CNG,CNL,CNX,DELGL,DELGX/
      DATA CSTSET /T,RTEMP,CAYT,EG,RDIEL,TPOWR,EFME,
     *           EFMH,FNC,FNV,DELGL,DELGX/
      DATA LUPL /LUPL1,LUPL2,LUPL3,LUPL4/
      DATA PLNAM /'PLOT1.DAT','PLOT2.DAT','PLOT3.DAT','PLOT4.DAT'/
      DATA OUTSTR /' '/
      DATA CH /' '/
      DONE=.FALSE.
C     *** Read input file ***
      OPEN (UNIT=LUIN,FILE='EPROP.DAT',STATUS='OLD',ERR=9510)
      READ (LUIN,'(4L2)',ERR=9510,END=9510) DEV,LOGSW
      READ (LUIN,'(2I2)',ERR=9510,END=9510) SETNUM,MATSW
      READ (LUIN,'(9I4)',ERR=9510,END=9510)PNUM,((PF(I,J),I=1,4),J=1,2)
      READ (LUIN,'(29(1X,I2))',ERR=9510,END=9510) OUTNUM,OUTPUT
      READ (LUIN,'(29(1X,I2))',ERR=9510,END=9510) HYBNUM,HYBSET
      READ (LUIN,'(1X,I2)',ERR=9510,END=9510) INDVAR
```

```
      READ (LUIN,'(3(1X,E12.5))',ERR=9510,END=9510)INIVAL,FINVAL,STPVAL
      READ (LUIN,'(2(1X,E12.5))',ERR=9510,END=9510) P(T),P(RTEMP)
      READ (LUIN,'(6L2)',ERR=9510,END=9510) DOPANT
      DO 10 I=1,6
           IF (DOPANT(I)) THEN
                READ (LUIN,'(1X,A13,3(1X,E12.5))',ERR=9510,END=9510)
     *          NAMES(I),P(DN(I)),P(EN(I)),P(DG(I))
           ELSE
                P(DN(I))=0.0
                P(EN(I))=0.0
                P(DG(I))=0.0
                NAMES(I)='Not Selected '
           ENDIF
10    CONTINUE
C     *** Menu loop ***
20    IF (.NOT.DONE) THEN
C          *** Put the main menu on the screen ***
           WRITE (LUTO,'(//////////,35X,''Main Menu'')')
           OUTSTR(:25)='      1  Material     > '
           IF (MATSW.EQ.1) THEN
                OUTSTR(26:29)='Si  '
           ELSEIF (MATSW.EQ.2) THEN
                OUTSTR(26:29)='GaAs'
           ELSE
C ---option---  This option added for InP; but could represent
C               any semiconductor.
                OUTSTR(26:29)='InP'
           ENDIF
           WRITE (LUTO,*) OUTSTR(:29)
           OUTSTR(:25)='      2  Output to    > '
           OUTSTR(26:65)='                                                '
           IF (DEV(1)) THEN
                IF (DEV(2)) THEN
                     IF (DEV(3)) THEN
                          OUTSTR(26:48)='terminal, listing file,'
                          OUTSTR(49:65)=' and plot file(s)'
                     ELSE
                          OUTSTR(26:50)='terminal and listing file'
                     ENDIF
                ELSE
                     IF (DEV(3)) THEN
                          OUTSTR(26:50)='terminal and plot file(s)'
                     ELSE
                          OUTSTR(26:38)='terminal only'
                     ENDIF
                ENDIF
           ELSE
                IF (DEV(2)) THEN
                     IF (DEV(3)) THEN
                          OUTSTR(26:49)='listing and plot file(s)'
                     ELSE
                          OUTSTR(26:42)='listing file only'
                     ENDIF
                ELSE
                     OUTSTR(26:42)='plot file(s) only'
                ENDIF
           ENDIF
           WRITE (LUTO,*) OUTSTR(:65)
           IF (DEV(3)) THEN
                OUTSTR(:25)='         Plot file(s) :  '
                WRITE (LUTO,*) OUTSTR(:25)
                IF (PNUM.EQ.1) WRITE (LUTO,'(13X,A11,
     *               '' vs. '',A11)') DSCRIP(PF(1,2)),DSCRIP(PF(1,1))
                IF (PNUM.GE.2) WRITE (LUTO,'(10X,A11,
     *               '' vs. '',A11,4X,A11,'' vs. '',A11)')
     *               DSCRIP(PF(1,2)),DSCRIP(PF(1,1)),
```

```
*              DSCRIP(PF(2,2)),DSCRIP(PF(2,1))
               IF (PNUM.EQ.3) WRITE (LUTO,'(10X,A11,
*                   '' vs. '',A11)') DSCRIP(PF(3,2)),DSCRIP(PF(3,1))
               IF (PNUM.EQ.4) WRITE (LUTO,'(10X,A11,
*                   '' vs. '',A11,4X,A11,'' vs. '',A11)')
*              DSCRIP(PF(3,2)),DSCRIP(PF(3,1)),
*              DSCRIP(PF(4,2)),DSCRIP(PF(4,1))
          ENDIF
          OUTSTR(:25)='      3  Output       >  '
          OUTSTR(26:39)= OUTSET(SETNUM)
          WRITE (LUTO,*) OUTSTR(:39)
          OUTSTR(:25)='      4  Input data   :  '
          WRITE (LUTO,*) OUTSTR(:25)
          OUTSTR(:32)='          Independent variable > '
          OUTSTR(33:44) = DSCRIP(INDVAR)
          WRITE (LUTO,*) OUTSTR(:44)
          OUTSTR(:32)='                          Start > '
          WRITE (OUTSTR(33:44),FORMS(PFORM(INDVAR))) INIVAL
          WRITE (LUTO,*) OUTSTR(:44)
          OUTSTR(:32)='                          Stop  > '
          WRITE (OUTSTR(33:44),FORMS(PFORM(INDVAR))) FINVAL
          WRITE (LUTO,*) OUTSTR(:44)
          IF (LOGSW) THEN
               OUTSTR(:32)='            Divisions/Decade  > '
          ELSE
               OUTSTR(:32)='                       Step  > '
          ENDIF
          WRITE (OUTSTR(33:44),FORMS(PFORM(INDVAR))) STPVAL
          WRITE (LUTO,*) OUTSTR(:44)
          IF (((INDVAR.LT.1).OR.(INDVAR.GT.2))) THEN
               OUTSTR(:32)='         Temperature (K)      > '
               WRITE (OUTSTR(33:44),FORMS(PFORM(T))) P(T)
               WRITE (LUTO,*) OUTSTR(:44)
          ENDIF
          WRITE (LUTO,2000)
2000      FORMAT(1X,/,10X,'Dopants',8X,'Density(cm-3)',5X,
*              'Energy(eV)',6X,'Degeneracy')
          WRITE (LUTO,2005)
2005      FORMAT(10X,'----------',5X,'-------------',5X,
*              '----------',6X,'----------')
          DO 50 L=1,6
               IF (DOPANT(L)) THEN
                    IF (INDVAR.EQ.DN(L)) THEN
                         WRITE (LUTO,3000)NAMES(L),
*                             ' ind. var. ',P(EN(L)),P(DG(L))
                    ELSEIF (INDVAR.EQ.EN(L)) THEN
                         WRITE (LUTO,3010)NAMES(L),
*                              P(DN(L)),' ind. var. ',P(DG(L))
                    ELSEIF (INDVAR.EQ.DG(L)) THEN
                         WRITE (LUTO,3020)NAMES(L),
*                             P(DN(L)),P(EN(L)),'   ind. var.'
                    ELSE
                         WRITE (LUTO,3030)NAMES(L),
*                             P(DN(L)),P(EN(L)),P(DG(L))
                    ENDIF
               ENDIF
50        CONTINUE
3000      FORMAT(10X,A13,3X,A11,5X,F9.4,5X,F8.1)
3010      FORMAT(10X,A13,3X,1PE11.4,0P,5X,A11,3X,F8.1)
3020      FORMAT(10X,A13,3X,1PE11.4,0P,5X,F9.4,5X,A12)
3030      FORMAT(10X,A13,3X,1PE11.4,0P,5X,F9.4,5X,F8.1)
          IF (.NOT.((DOPANT(1)).OR.(DOPANT(2)).OR.(DOPANT(3))
*              .OR.(DOPANT(4)).OR.(DOPANT(5)).OR.(DOPANT(6))))
*              WRITE (LUTO,'(10X,''none'')')
          WRITE (LUTO,*)
          WRITE (LUTO,4000)
```

```
4000        FORMAT(7X,'5  Execute')
            CALL PROMPT(' 6  Exit',0,6)
            CALL PROMPT(' Enter number > ',0,27)
            READ (LUTI,'(A)',ERR=9500) INSTR
            DO 60 L=1,4
                 IF (DIGIT(INSTR(L:L))) THEN
                      READ (INSTR(L:L),'(I1)') RSPON
                      GOTO 65
                 ENDIF
60          CONTINUE
            GOTO 9500
65          IF (.NOT.INRNGE(1,6,RSPON)) THEN
                 WRITE (LUTO,*)' Error: input not in the range 1 to 6'
                 WRITE (LUTO,*)'        Hit <RETURN> to continue.'
                 READ (LUTI,'(A)') CH
            ELSE
                 IF (MATSW.EQ.1) THEN
C                *** Silicon ***
                      MAXOUT=21
                 ELSEIF (MATSW.EQ.2) THEN
C                *** Gallium Arsenide ***
                      MAXOUT=28
                 ELSE
C ---option---        This option specifies the maximum number of
C                     output parameters, taken here to be the same
C                     as for Si.
                      MAXOUT=21
                 ENDIF
                 IF (RSPON.EQ.1) THEN
                      CALL MATSEL (MATSW)
                 ELSEIF (RSPON.EQ.2) THEN
                      CALL DEVSEL (DEV,PNUM,PF,MAXOUT,MATSW)
                 ELSEIF (RSPON.EQ.3) THEN
                      CALL OUTSEL (DSCRIP,INDVAR,SETNUM,OUTSET,
     *                     HYBSET,HYBNUM,MAXOUT)
                 ELSEIF (RSPON.EQ.4) THEN
                      CALL INPSEL(INDVAR,PFORM,MATSW)
                 ELSEIF (RSPON.EQ.6) THEN
                      WRITE (LUTO,*)'         EPROP run aborted,',
     *                                ' no changes made.'
C                Pause needed in MacFortran
C                     PAUSE
                      STOP
                 ELSE
                      DONE = .TRUE.
                 ENDIF
            ENDIF
            GOTO 20
       ENDIF
C      *** Initialize certain parameters according
C                          to decisions made above ***
       OUTPUT(1)=INDVAR
       TRMSW=DEV(1)
       LSTSW=DEV(2)
       PLTSW=DEV(3)
       IF (SETNUM.EQ.1) THEN
            DO 70 I=2,10
                 OUTPUT(I)=SHTSET(I)
70          CONTINUE
            OUTNUM=9
       ELSEIF (SETNUM.EQ.2) THEN
            DO 80 I=2,MAXOUT
                 OUTPUT(I)=FULSET(I)
80          CONTINUE
            OUTNUM=MAXOUT
            IF (INDVAR.EQ.RTEMP) OUTPUT(2)=T
```

```
      ELSEIF (SETNUM.EQ.3) THEN
           IF (MATSW.EQ.1) THEN
C          *** Silicon ***
                OUTNUM=10
           ELSEIF (MATSW.EQ.2) THEN
C          *** Gallium Arsenide ***
                OUTNUM=12
           ELSE
C ---option---  Third material option for number of output
C               parameters under constants option, set here
C               to be the same as for Si.
                OUTNUM=10
           ENDIF
           DO 90 I=1,OUTNUM
                OUTPUT(I)=CSTSET(I)
90         CONTINUE
      ELSE
           DO 100 I=2,HYBNUM
                OUTPUT(I)=HYBSET(I)
100        CONTINUE
           OUTNUM=HYBNUM
      ENDIF
      WRITE (LUTO,'(/////////////////////////////)')
C      *** Warnings for unusual exit conditions ***
      IF ((SETNUM.EQ.3).AND.(INDVAR.NE.T).AND.(INDVAR.NE.RTEMP)) THEN
           WRITE (LUTO,102)
102        FORMAT(//,' Warning:  The constants option is',
     *          ' selected and the ',/,' temperature is fixed. ',
     *          ' Would you like to make temperature or')
           CALL PROMPT(' 1000/T the independent variable? (Y/N) > ',0,0)
           READ (LUTI,'(A)') CH
           IF (CH.EQ.'Y') THEN
                DONE=.FALSE.
                GOTO 20
           ENDIF
      ENDIF
      IF ((MATSW.EQ.1).OR.(MATSW.EQ.3)) THEN
           UNDEFP=.FALSE.
           UNDEFO=.FALSE.
           IF (DEV(3)) THEN
                DO 103 I=1,4
                DO 103 J=1,2
                     IF (INRNGE(FNCP,DELGX,PF(I,J))) UNDEFP=.TRUE.
103             CONTINUE
           ENDIF
           IF (SETNUM.EQ.4) THEN
                DO 104 I=1,HYBNUM
                     IF (INRNGE(FNCP,DELGX,HYBSET(I)))
     *                    UNDEFO=.TRUE.
104             CONTINUE
           ENDIF
           IF (UNDEFO.AND.UNDEFP) THEN
                WRITE (LUTO,*)' Warning: There are',
     *           ' parameters in both the plot file'
                WRITE (LUTO,*)' output and the user specified',
     *           ' output which are undefined'
                WRITE (LUTO,*)' for this material'
           ELSEIF (UNDEFO) THEN
                WRITE (LUTO,'('' Warning: There are'',
     *           '' parameters in the user specified'',/,
     *           '' output which are undefined for'',
     *           '' this material'')')
           ELSEIF (UNDEFP) THEN
                WRITE (LUTO,'('' Warning: There are'',
     *           '' parameters in the plot file output'',/,
     *           '' which are undefined for this material'')')
```

```
            ENDIF
            IF (UNDEFO.OR.UNDEFP) THEN
                 CALL PROMPT(' Do you wish to return to '//
     *                      'the main menu? <Y/N> ',0,0)
                 READ (LUTI,'(A1)') CH
                 IF ((CH.EQ.'Y').OR.(CH.EQ.'y')) THEN
                      DONE=.FALSE.
                      GOTO 20
                 ENDIF
            ENDIF
       ENDIF
       DO 1041 I=1,6
            IF (.NOT.DOPANT(I)) THEN
                 IF ((INDVAR.EQ.DN(I)).OR.(INDVAR.EQ.EN(I)).OR.
     *           (INDVAR.EQ.DG(I))) THEN
                      WRITE (LUTO,*)' Warning: The independent ',
     *                'variable is an unselected parameter'
                      CALL PROMPT(' Do you wish to return to'//
     *                      ' the main menu? <Y/N> ',0,0)
                      READ (LUTI,'(A1)') CH
                      IF ((CH.EQ.'Y').OR.(CH.EQ.'y')) THEN
                           DONE=.FALSE.
                           GOTO 20
                      ENDIF
                 ENDIF
            ENDIF
1041   CONTINUE
       IF ((.NOT.LSTSW).AND.(OUTNUM.GT.6)) OUTNUM=6
       IF (.NOT.LSTSW) THEN
            LUDEF=LUTO
       ELSE
            OPEN (UNIT=LULST,FILE='LIST.DAT',STATUS='UNKNOWN')
            IF (OUTNUM.GT.6) THEN
                 LUDEF=LUTMP
                 OPEN (UNIT=LUTMP,FILE='TEMPOR.ARY',STATUS='UNKNOWN')
            ELSE
                 LUDEF=LULST
            ENDIF
       ENDIF
       IF (PLTSW) THEN
            DO 105 I=1,PNUM
                 OPEN (UNIT=LUPL(I),FILE=PLNAM(I),STATUS='UNKNOWN')
105         CONTINUE
       ENDIF
       OPEN (UNIT=LUER,FILE='ERROR.LIS',STATUS='UNKNOWN')
       CLOSE (UNIT=LUIN,STATUS='DELETE')
C      *** Send modifications to the input file ***
       OPEN (UNIT=LUIN,FILE='EPROP.DAT',STATUS='NEW')
       WRITE (LUIN,'(4L2)') DEV,LOGSW
       WRITE (LUIN,'(2I2)') SETNUM,MATSW
       WRITE (LUIN,'(9I4)') PNUM,((PF(I,J),I=1,4),J=1,2)
       WRITE (LUIN,'(29(1X,I2))') OUTNUM,OUTPUT
       WRITE (LUIN,'(29(1X,I2))') HYBNUM,HYBSET
       WRITE (LUIN,'(1X,I2)') INDVAR
       WRITE (LUIN,'(3(1X,E12.5))') INIVAL,FINVAL,STPVAL
       WRITE (LUIN,'(2(1X,E12.5))') P(T),P(RTEMP)
       WRITE (LUIN,'(6L2)') DOPANT
       DO 110 I=1,6
            IF (DOPANT(I)) WRITE (LUIN,'(1X,A13,3(1X,E12.5))')
     *           NAMES(I),P(DN(I)),P(EN(I)),P(DG(I))
110    CONTINUE
       CLOSE (UNIT=LUIN)
200    RETURN
9500   WRITE (LUTO,*)' Error: response must be an integer'
       WRITE (LUTO,*)' Hit <RETURN> to continue.'
       READ (LUTI,'(A)') CH
```

```
      GOTO 20
C     *** Sets up new input file if one doesn't exist ***
9510  WRITE (LUTO,9520)
9520  FORMAT(////,27X,'EPROP   Version 1.0',///,
     *12X,'An Interactive Program for Computing the Electrical',/,
     *12X,'              Properties of GaAs and Si',///,
     *35X,'by',//,13X,'John J. Mathias, Alan C. Seabaugh, ',
     *'Michael I. Bell',/,21X,'Semiconductor Electronics Division',/,
     *14X,'National Institute of Standards and Technology',
     */,26X,'Gaithersburg, MD 20899',/,34X,'MCMXC')
      WRITE (LUTO,9521)
9521  FORMAT(//,18X,'Instructions: You will be asked to respond to a',/
     *11X,  'series of menus. If an input is incorrectly entered, do',
     */,11X,'not abort the program; you will have an opportunity to',/,
     *11X,  'make changes. The file, EPROP.DAT (normally containing',/,
     *11X,  'parameters from the previous run) is not present. Press')
      CALL PROMPT(' <RETURN> to create a new input file '//
     *            'or enter "Q" to quit. >',0,10)
      READ (LUTI,'(A)') CH
      IF ((CH.EQ.'Q').OR.(CH.EQ.'q')) STOP
      DEV(1)=.TRUE.
      DEV(2)=.FALSE.
      DEV(3)=.FALSE.
      LOGSW=.FALSE.
      SETNUM=1
      MATSW=2
      PNUM=0
      OUTNUM=10
      HYBNUM=1
      MAXOUT=28
      DO 9525 I=1,21
          HYBSET(I)=FULSET(I)
          OUTPUT(I)=FULSET(I)
9525  CONTINUE
      DO 9526 I=1,6
          NAMES(I)='Not Selected '
          DOPANT(I)=.FALSE.
9526  CONTINUE
      INDVAR=T
      INIVAL=300.
      FINVAL=300.
      STPVAL=1.
      P(T)=300.
      P(RTEMP)=1000./P(T)
      CALL DOPSEL(INDVAR,PFORM)
      CALL INDSEL(INDVAR,PFORM,MATSW)
      HYBSET(1)=INDVAR
      IF ((INDVAR.NE.T).AND.(INDVAR.NE.RTEMP)) CALL TSEL(MATSW)
      GOTO 20
      END
C
C Subroutine OUTSEL selects the output set.
C
      SUBROUTINE OUTSEL(DSCRIP,INDVAR,SETNUM,OUTSET,
     *                  HYBSET,HYBNUM,MAXOUT)
C
      INTEGER LUTI,LUTO
      COMMON /LUNIT/ LUTI,LUTO
C
      LOGICAL INRNGE,DIGIT
      INTEGER HYBSET(28),HYBNUM,SETNUM,RSPON,MAXOUT
      CHARACTER OUTSET(4)*14,OUTSTR*80,DSCRIP(46)*11,CH*1
      DATA OUTSTR /' '/
C     *** Create the output set selection menu ***
      WRITE (LUTO,'(////)')
1     WRITE (LUTO,'(25X,''Output Set Selection Menu'',///)')
```

```
      OUTSTR(:33)='                                    '
      DO 10 I=1,4
           WRITE (OUTSTR(30:30),'(I1)') I
           OUTSTR(34:48)=OUTSET(I)
           WRITE (LUTO,'(A,/)') OUTSTR(:48)
10    CONTINUE
      OUTSTR(:40)='               The current selection is '
      OUTSTR(41:55)=OUTSET(SETNUM)
      WRITE (LUTO,'(1X,A,//)') OUTSTR(:54)
15    CALL PROMPT(' Enter number > ',0,27)
C     *** Read the response from the terminal
C           and perform indicated function ***
      READ (LUTI,'(A)',ERR=9500) CH
      IF (DIGIT(CH)) THEN
           READ (CH,'(I1)') RSPON
      ELSEIF (CH.EQ.' ') THEN
           RSPON=5
      ELSE
           GOTO 9500
      ENDIF
      IF (RSPON.EQ.0) RSPON=5
      IF (INRNGE(1,5,RSPON)) THEN
           IF (RSPON.NE.5) THEN
                SETNUM=RSPON
                IF (SETNUM.EQ.4) CALL USPEC (DSCRIP,INDVAR,
     *               HYBNUM,HYBSET,MAXOUT)
           ENDIF
      ELSE
           WRITE (LUTO,*) ' Error: improper response, try again'
           WRITE (LUTO,*) '        Hit <RETURN> to continue.'
           READ (LUTI,'(A)') CH
           GOTO 1
      ENDIF
      RETURN
9500  WRITE (LUTO,*)' Error: response must be an integer'
      GOTO 15
      END
C
C Subroutine PLTSEL selects plot file pairs.
C
      SUBROUTINE PLTSEL(DEV,PNUM,PF,MAXOUT,MATSW)
C
      REAL P(46)
      COMMON /PARMS/ P
      CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
      COMMON /STRING/ NAMES,DSCRIP,FORMS
C
      INTEGER LUTI,LUTO
      COMMON /LUNIT/ LUTI,LUTO
C
      INTEGER T,RTEMP,EF,CN,CP,DI1,DI2,DI3,DI4,DI5,EG,SUM,CAYT,ECMEF,
     * RDIEL,TPOWR,EFME,EFMH,FNC,FNV,DN1,EN1,DG1,DN2,EN2,DG2,DN3,EN3,
     * DG3,DN4,EN4,DG4,DN5,EN5,DG5,FNCP,FNVP,CNG,CNL,CNX,DELGL,DELGX
     * DI6,DN6,EN6,DG6
      PARAMETER (T=1,RTEMP=2,EF=3,CN=4,CP=5,DI1=6,DI2=7,DI3=8,DI4=9,
     * DI5=10,DI6=43,EG=11,SUM=12,CAYT=13,ECMEF=14,RDIEL=15,TPOWR=16,
     * EFME=17,EFMH=18,FNC=19,FNV=20,DN1=21,EN1=22,DG1=23,DN2=24,
     * EN2=25,DG2=26,DN3=27,EN3=28,DG3=29,DN4=30,EN4=31,DG4=32,DN5=33,
     * EN5=34,DG5=35,DN6=44,EN6=45,DG6=46,FNCP=36,FNVP=37,CNG=38,
     * CNL=39,CNX=40,DELGL=41,DELGX=42)
C
      CHARACTER*80 OUTSTR
      INTEGER ITEMP,PF(4,2),PNUM,RSPON,RSPON2,MAXOUT,STOVER,PLST(46),
     * MATSW
      LOGICAL DEV(3),INRNGE
      DATA PLST /T,RTEMP,EF,CN,CP,DI1,DI2,DI3,DI4,DI5,DI6,EG,SUM,CAYT,
```

```
     * ECMEF,RDIEL,TPOWR,EFME,EFMH,FNC,FNV,DN1,EN1,DG1,DN2,EN2,DG2,DN3,
     * EN3,DG3,DN4,EN4,DG4,DN5,EN5,DG5,DN6,EN6,DG6,FNCP,FNVP,CNG,
     * CNL,CNX,DELGL,DELGX/
      DATA OUTSTR /' '/
C     *** Have plot file attributes been previously entered? ***
      WRITE (LUTO,'(////)')
1     IF (PNUM.GE.1) THEN
           WRITE (LUTO,'(31X,''Plot File Menu'',//)')
           OUTSTR(:24)='                         '
           DO 10 I=1,PNUM
                WRITE (OUTSTR(20:20),'(I1)') I
                OUTSTR(36:40)=' vs. '
                OUTSTR(25:35)=DSCRIP(PF(I,2))
                OUTSTR(41:51)=DSCRIP(PF(I,1))
                WRITE (LUTO,*) OUTSTR(:51)
                WRITE (LUTO,*)
10         CONTINUE
           DO 20 I=PNUM+1,4
                WRITE (LUTO,'(20X,I1,6X,''    not selected'',/)') I
20         CONTINUE
25         CALL PROMPT(' Enter number > ',2,2)
           READ (LUTI,'(A)') OUTSTR
           IF (OUTSTR(:20).EQ.'                    ') THEN
                GOTO 1000
           ELSE
                READ (OUTSTR(1:1),'(I1)',ERR=9500) RSPON
           ENDIF
           IF (.NOT.INRNGE(1,4,RSPON)) THEN
                WRITE (LUTO,*)' Error: response out of range'
                GOTO 25
           ENDIF
      ELSE
           RSPON=1
      ENDIF
      STOVER=PNUM
2     IF (RSPON.LE.PNUM) THEN
           WRITE (LUTO,'(////,28X,''Plot File Submenu'',//)')
           WRITE (LUTO,'(23X,''File'',5X,''X-axis'',6X,
     *          ''Y-axis'')')
           WRITE (LUTO,'(23X,''----'',5X,''------'',6X,
     *          ''------'')')
           WRITE (LUTO,'(24X,I1,5X,A11,2X,A11,//)')
     *          RSPON,DSCRIP(PF(RSPON,1)),DSCRIP(PF(RSPON,2))
           WRITE (LUTO,'(29X,''1   Change x-axis'')')
           WRITE (LUTO,'(29X,''2   Change y-axis'')')
           WRITE (LUTO,'(29X,''3   Change both  '')')
           WRITE (LUTO,'(29X,''4   Switch order '')')
           WRITE (LUTO,'(29X,''5   Delete entry '')')
30         CALL PROMPT(' Enter number > ',2,29)
           READ (LUTI,'(A)') OUTSTR
           IF (OUTSTR(:20).EQ.'                    ') THEN
                GOTO 1
           ELSE
                READ (OUTSTR(1:1),'(I1)',ERR=9505) RSPON2
           ENDIF
           IF (.NOT.INRNGE(1,5,RSPON2)) THEN
                WRITE (LUTO,*)' Error: response out of range'
                GOTO 30
           ENDIF
           IF (RSPON2.EQ.5) THEN
                DO 40 I=RSPON,PNUM-1
                     PF(I,1)=PF(I+1,1)
40                   PF(I,2)=PF(I+1,2)
                PNUM=PNUM-1
                IF (PNUM.EQ.0) THEN
                     DEV(3)=.FALSE.
```

```
                    GOTO 1000
               ENDIF
          ELSEIF (RSPON2.EQ.4) THEN
               ITEMP=PF(RSPON,1)
               PF(RSPON,1)=PF(RSPON,2)
               PF(RSPON,2)=ITEMP
          ELSE
               CALL PVARS(DSCRIP,MATSW,PLST)
               WRITE (LUTO,'(//)')
               IF ((RSPON2.EQ.1).OR.(RSPON2.EQ.3)) THEN
45                  CALL PROMPT(' Enter parameter number'//
     *                          ' for the X-axis > ',0,0)
                    READ (LUTI,*,ERR=9505) ITEMP
                    IF (INRNGE(1,MAXOUT+18,ITEMP)) THEN
                         PF(RSPON,1)=PLST(ITEMP)
                    ELSE
                         WRITE (LUTO,*)
     *                     ' Error: Response out of range'
                         GOTO 45
                    ENDIF
               ENDIF
               IF ((RSPON2.EQ.2).OR.(RSPON2.EQ.3)) THEN
50                  CALL PROMPT(' Enter parameter number'//
     *                          ' for the Y-axis > ',0,0)
                    READ (LUTI,*,ERR=9505) ITEMP
                    IF (INRNGE(1,MAXOUT+18,ITEMP)) THEN
                         PF(RSPON,2)=PLST(ITEMP)
                    ELSE
                         WRITE (LUTO,*)
     *                     ' Error: Response out of range'
                         GOTO 50
                    ENDIF
               ENDIF
          ENDIF
      ELSE
          PNUM=PNUM+1
          CALL PVARS(DSCRIP,MATSW,PLST)
55        CALL PROMPT(' Enter parameter number'//
     *                ' for the X-axis > ',0,0)
          READ (LUTI,*,ERR=9530) ITEMP
          IF (INRNGE(1,MAXOUT+18,ITEMP)) THEN
               PF(PNUM,1)=PLST(ITEMP)
          ELSE
               WRITE (LUTO,*)
     *           ' Error: Response out of range'
               GOTO 55
          ENDIF
60        CALL PROMPT(' Enter parameter number'//
     *                ' for the Y-axis > ',0,0)
          READ (LUTI,*,ERR=9530) ITEMP
          IF (INRNGE(1,MAXOUT+18,ITEMP)) THEN
               PF(PNUM,2)=PLST(ITEMP)
          ELSE
               WRITE (LUTO,*)
     *           ' Error: Response out of range'
               GOTO 60
          ENDIF
          IF (PF(PNUM,1).EQ.PF(PNUM,2)) THEN
               WRITE (LUTO,'(//,'' *** X and Y axes are the same,'',
     *              '' reselect.'',//)')
               GOTO 55
          ENDIF
      ENDIF
      WRITE (LUTO,'(//)')
      GOTO 1
1000  RETURN
```

```
9500  WRITE (LUTO,*)' Error: Response must be an integer'
      WRITE (LUTO,*)' Hit <RETURN> to continue.'
      READ (LUTI,'(A)') CH
      GOTO 1
9505  WRITE (LUTO,*)' Error: Response must be an integer'
      WRITE (LUTO,*)' Hit <RETURN> to continue.'
      READ (LUTI,'(A)') CH
      GOTO 2
9530  WRITE (LUTO,*)' Error: Response must be an integer'
      WRITE (LUTO,*)' Hit <RETURN> to continue.'
      READ (LUTI,'(A)') CH
      PNUM=STOVER
      GOTO 2
      END
C
C Subroutine PROMPT sends a prompt string to the terminal,
C preceeded by the specified number of blank lines and spaces.
C
      SUBROUTINE PROMPT(STRING,RTNS,TAB)
C
      IMPLICIT NONE
      CHARACTER*(*) STRING
      INTEGER RTNS,TAB
C
      INTEGER LUTI,LUTO
      COMMON /LUNIT/ LUTI,LUTO
C
      CHARACTER FMT*20,TABSTR*3
      INTEGER I
      IF (RTNS .GT. 0) THEN
            DO 10 I=1,RTNS
            WRITE (9,'(A)')
 10         CONTINUE
      END IF
      WRITE (TABSTR,'(I3)') TAB
C
C MacFortran version
C     FMT='('//TABSTR//'X,A)'
C     TYPE (LUTO,FMT) STRING
C
C VAX-11 Fortran version
      FMT='($,'//TABSTR//'X,A)'
      WRITE (LUTO,FMT) STRING
C
      RETURN
      END
C
C Subroutine PVARS provides a list of the possible file
C variables for PLTSEL
C
      SUBROUTINE PVARS(DSCRIP,MATSW,PLST)
C
      INTEGER LUTO,LUTI
      COMMON /LUNIT/ LUTI,LUTO
C
      CHARACTER DSCRIP(46)*11,OUTSTR*80
      INTEGER BASE,COL,NUM,MAX,ROWS,PLST(46),MATSW
      DATA OUTSTR /' '/
      IF (MATSW.EQ.1) THEN
C     *** Silicon ***
           MAX=39
      ELSEIF (MATSW.EQ.2) THEN
C     *** Gallium Arsenide ***
           MAX=46
      ELSE
C ---option--- Third material option, set to be the same
```

```
C           as for Si.
            MAX=39
         ENDIF
         OUTSTR(4:4)='>'
         OUTSTR(19:19)='>'
         OUTSTR(34:34)='>'
         OUTSTR(49:49)='>'
         OUTSTR(64:64)='>'
         WRITE (LUTO,'(//,30X,''Plot Parameter Menu'',///)')
         BASE=2
         ROWS=MAX/5
         IF ((ROWS*5).NE.MAX) ROWS=ROWS+1
         DO 20 I=1,ROWS
              JE=5
              IF ((I*5).GT.MAX) JE=MAX-5*(I-1)
              DO 10 J=1,JE
                   NUM=5*(I-1)+J
                   COL=BASE+15*(J-1)
                   WRITE (OUTSTR(COL:COL+1),'(I2)') NUM
                   WRITE (OUTSTR(COL+3:COL+14),'(1X,A11)')
     *                              DSCRIP(PLST(NUM))
10            CONTINUE
              WRITE (LUTO,*) OUTSTR(:COL+14)
              WRITE (LUTO,*)
20       CONTINUE
         RETURN
         END
C
C Function TEST defines the parameter 'TEST'.  Subroutine ZEROIN
C then finds the value of FERMI which minimizes 'TEST'.
C
         FUNCTION TEST(FERMI)
C
         REAL P(46)
         COMMON /PARMS/ P
         REAL ETA,ZETA
         COMMON /CALC/ ETA,ZETA
         LOGICAL DOPANT(6)
         COMMON /TSTCOM/ DOPANT
         INTEGER PFORM(46),INDVAR,OUTPUT(28),OUTNUM,
     *        PF(4,2),PNUM,LUDEF,MAXOUT,MATSW
         LOGICAL TRMSW,LSTSW,PLTSW
         COMMON /GLOBAL/ PFORM,INDVAR,OUTPUT,OUTNUM,PF,PNUM,
     *                LUDEF,TRMSW,LSTSW,PLTSW,MATSW,MAXOUT
C
         INTEGER CN,CP,DI1,DI2,DI3,DI4,DI5,EG,CAYT,TPOWR,EFME,FNC,FNV,DN1,
     * EN1,DG1,DN2,EN2,DG2,DN3,EN3,DG3,DN4,EN4,DG4,DN5,EN5,DG5,
     * FNCP,FNVP,CNG,CNL,CNX,DELGL,DELGX,DI6,DN6,EN6,DG6
         PARAMETER (CN=4,CP=5,DI1=6,DI2=7,DI3=8,DI4=9,DI5=10,EG=11,
     * CAYT=13,TPOWR=16,EFME=17,FNC=19,FNV=20,DN1=21,EN1=22,DG1=23,
     * DN2=24,EN2=25,DG2=26,DN3=27,EN3=28,DG3=29,DN4=30,EN4=31,DG4=32,
     * DN5=33,EN5=34,DG5=35,FNCP=36,FNVP=37,CNG=38,CNL=39,CNX=40,
     * DELGL=41,DELGX=42,DI6=43,DN6=44,EN6=45,DG6=46)
C
C        *** CONST = 2*(2*pi*k*m0/h**2)**(3/2), where
C        *** k is Boltzmann's constant, 1.3807E-23 J/K,
C        *** m0 is the free electron mass, 9.1095E-31 kg, and
C        *** h is Planck's constant, 6.6262E-34 J-s.  CONST is
C        *** written in units of K**-3/2 cm**-3.
         REAL CONST,DEL,H,MBL,MH,MX,PI
         PARAMETER (CONST=4.8293E15,DEL=0.341,H=6.6262E-34,MBL=.52,
     * MH=.5,MX=.85,PI=3.1416)
         REAL A,AH,AL,B,BH,BL,C,CH,CL,D,DH,DL
         PARAMETER (AH=0.400892,BH=0.263693,CH=.0590925,DH=-.314104)
         PARAMETER (AL=.0339615,BL=-.543866,CL=5.202820, DL=-6.48801)
         PARAMETER (A=AH+AL,B=(BH+BL)*1.5,C=(CH+CL)*3.75,D=(DH+DL)*13.125)
```

```
C
      REAL ALPHA,BETA,FERMI,MLL
      INTEGER EN(6),DN(6),DI(6),DG(6)
      DATA EN /EN1,EN2,EN3,EN4,EN5,EN6/
      DATA DN /DN1,DN2,DN3,DN4,DN5,DN6/
      DATA DI /DI1,DI2,DI3,DI4,DI5,DI6/
      DATA DG /DG1,DG2,DG3,DG4,DG5,DG6/
C
      MLL=P(EG)/(20.0-P(EG))
C
      ETA=(FERMI-P(EG))/P(CAYT)
C     *** Computation of P(CN) ***
      IF (MATSW.EQ.1) THEN
C     *** Silicon ***
           P(CN)=P(FNC)*FD(0.5,ETA)
      ELSEIF (MATSW.EQ.2) THEN
C     *** Gallium Arsenide ***
           ALPHA=-(1.-P(EFME))**2.*(3.*P(EG)**2.+4.*P(EG)*DEL+
     *          2.*DEL**2.)/(P(EG)+DEL)/(3.*P(EG)+2.*DEL)
           IF (ETA.GE.-80.) THEN
                P(FNCP)=P(FNC)*(1.-15.*ALPHA*P(CAYT)*FD(1.5,ETA)/
     *               (4.*P(EG)*FD(0.5,ETA)))
           ELSE
                P(FNCP)=P(FNC)*(1.-15.*ALPHA*P(CAYT)/(4.*P(EG)))
           ENDIF
           P(CNG)=P(FNCP)*FD(0.5,ETA)
           IF ((ETA-P(DELGL)/P(CAYT)).LE.-85) THEN
             P(CNL)=0.
           ELSE
             P(CNL)=CONST*P(TPOWR)*MBL**(1.5)*EXP(ETA-P(DELGL)/P(CAYT))
           ENDIF
           IF ((ETA-P(DELGX)/P(CAYT)).LE.-85) THEN
             P(CNX)=0.
           ELSE
             P(CNX)=CONST*P(TPOWR)*MX**(1.5)*EXP(ETA-P(DELGX)/P(CAYT))
           ENDIF
           P(CN)=P(CNG)+P(CNL)+P(CNX)
      ELSE
C ---option--- Third material formulation for the electron density.
C          Here it is taken to be the same as for Si.
           P(CN)=P(FNC)*FD(0.5,ETA)
      ENDIF
      ZETA=-FERMI/P(CAYT)
C     *** Computation of P(CP) ***
      IF (MATSW.EQ.1) THEN
C     *** Silicon ***
           P(CP)=P(FNV)*FD(0.5,ZETA)
      ELSEIF (MATSW.EQ.2) THEN
C     *** Gallium Arsenide ***
           BETA=-(1.+P(EG)/2./DEL)/(1-P(EG)/20.0)**2.
           IF (ZETA.GT.-80) THEN
C This uses the valence band density of states for GaAs described in
C J. S. Blakemore, J. Appl. Phys. 53, R123-R181 (1982)
C               P(FNVP)=CONST*P(TPOWR)*((MH**1.5)+(MLL**1.5)*(1.-
C     *          15.*BETA*P(CAYT)*FD(1.5,ZETA)/(4.*P(EG)*FD(0.5,ZETA))))
C This uses the valence band density of states for GaAs described in
C J. R. Lowney and A. H. Kahn, J. Appl. Phys. 64, 447-450 (1988).
               P(FNVP)=CONST*P(TPOWR)*(A+(
     *                   B*P(CAYT)*FD(1.5,ZETA)
     *                   +C*P(CAYT)**2*FD(2.5,ZETA)
     *                   +D*P(CAYT)**3*FD(3.5,ZETA)
     *                   )/FD(.5,ZETA))
           ELSE
C Following Blakemore:
C               P(FNVP)=CONST*P(TPOWR)*((MH**1.5)+(MLL**1.5)*(1.-
C     *          15.*BETA*P(CAYT)/(4.*P(EG))))
```

```
C Following Lowney and Kahn:
                  P(FNVP)=CONST*P(TPOWR)*
     *                    (A+B*P(CAYT)+C*P(CAYT)**2+D*P(CAYT)**3)
            ENDIF
            P(CP)=P(FNVP)*FD(0.5,ZETA)
       ELSE
C ---option--- Third material formulation for the hole density.
C           Here it is taken to be the same as for Si.
            P(CP)=P(FNV)*FD(0.5,ZETA)
       ENDIF
C      *** Computation of P(DI'S) ***
       DO 35 L=1,6
       IF (DOPANT(L)) THEN
            IF (L.LE.3) THEN
                 POWER=(FERMI-P(EG)+P(EN(L)))/P(CAYT)
            ELSE
                 POWER=(P(EN(L))-FERMI)/P(CAYT)
            ENDIF
            IF (POWER.LE.-80.) THEN
                 P(DI(L))=P(DN(L))
            ELSE IF (POWER.LT.80.) THEN
                 P(DI(L))=P(DN(L))/(1.+(EXP(POWER)*P(DG(L))))
            ELSE
                 P(DI(L))=0.
            ENDIF
       ELSE
            P(DI(L))=0.
       ENDIF
35     CONTINUE
C      *** Testing to see how close to neutrality we have come,
C      the parameter test is the net density of residual charges
C      when the FERMI level is at its present position  ***
140    TEST=(P(CN)+P(DI4)+P(DI5)+P(DI6))-(P(CP)+P(DI1)+P(DI2)+P(DI3))
       RETURN
       END
C
C Subroutine TSEL allows selection of temperature or 1000/T.
C
       SUBROUTINE TSEL(MATSW)
C
       REAL P(46)
       COMMON /PARMS/ P
       CHARACTER NAMES(6)*13,DSCRIP(46)*11,FORMS(6)*15
       COMMON /STRING/ NAMES,DSCRIP,FORMS
C
       REAL LOWLIM,UPLIM
       INTEGER T,RTEMP,MATSW
       PARAMETER (T=1,RTEMP=2)
       INTEGER LUTI,LUTO
       COMMON /LUNIT/ LUTI,LUTO
C
       INTEGER RSPON
       CHARACTER CH
C
       IF (MATSW.EQ.1) THEN
            UPLIM=1688.0
       ELSEIF (MATSW.EQ.2) THEN
            UPLIM=1511.0
       ELSE
C ---option--- Third material melting point, set here to be the
C           melting point of InP, but it could be changed to
C           that appropriate for any semiconductor.
            UPLIM=1343.
       ENDIF
       LOWLIM=2.5
5      WRITE (LUTO,'(////,30X,''Temperature Selection'',///)')
```

```
      WRITE (LUTO,'(30X,''1   Temperature (K)'',/)')
      WRITE (LUTO,'(30X,''2   1000/T (K^-1)'',/)')
      CALL PROMPT('     Enter number > ',2,25)
      READ (LUTI,'(I2)',ERR=9505) RSPON
10    IF (RSPON.EQ.1) THEN
           CALL PROMPT(' Enter new temperature (K) > ',0,0)
           READ (LUTI,*,ERR=9500) P(T)
           P(RTEMP)=1000./P(T)
      ELSE
           CALL PROMPT(' Enter new value for 1000/T > ',0,0)
           READ (LUTI,*,ERR=9500) P(RTEMP)
           P(T)=1000./P(RTEMP)
      ENDIF
      IF (P(T).LT.LOWLIM) THEN
           WRITE (LUTO,'(1X,''Temperature must always be greater'',
     *     /,1X,''than 2.5 degrees (absolute).   Start over.'')')
           GOTO 10
      ENDIF
      IF (P(T).GT.UPLIM) THEN
           WRITE (LUTO,'(1X,''Calculation attempted past the '',
     *     ''melting point of the material.'',/,1x,
     *     ''('',E12.5,'' degrees)     Start over.'')') UPLIM
           GOTO 10
      ENDIF
      RETURN
9500  WRITE (LUTO,*)' Error: response must be in F, D, or E format'
      GOTO 10
9505  WRITE (LUTO,*)' Error: response must be an integer'
      WRITE (LUTO,*)' Hit <RETURN> to continue.'
      READ (LUTI,'(A)') CH
      GOTO 5
      END
C
C Subroutine TVARS calculates temperature dependent constants.
C
      SUBROUTINE TVARS(INDVAR,MATSW)
C
      REAL P(46)
      COMMON /PARMS/ P
C
      INTEGER T,RTEMP,EG,CAYT,RDIEL,TPOWR,EFME,
     *        EFMH,FNC,FNV,DELGL,DELGX
      PARAMETER (T=1,RTEMP=2,EG=11,CAYT=13,ECMEF=14,RDIEL=15,
     * TPOWR=16,EFME=17,EFMH=18,FNC=19,FNV=20,DELGL=41,DELGX=42)
      REAL CONST,MH
      PARAMETER (CONST=4.8293E15,MH=0.5)
C
      INTEGER INDVAR,MATSW
      REAL MLL
C
      IF (INDVAR.EQ.T) THEN
           P(RTEMP)=1000./P(T)
      ELSE
           P(T)=1000./P(RTEMP)
      ENDIF
      P(CAYT)=8.6173E-5*P(T)
      P(TPOWR)=P(T)**1.5
      IF (MATSW.EQ.1) THEN
C        *** Computed for silicon ***
         P(RDIEL)=11.4294*EXP(7.8E-5*P(T))
C        This formulation for the relative dielectric constant
C        was taken from R. D. Larrabee, W. R. Thurber, and W. M. Bullis
C        NBS Spec. Pub. 400-63 (1980).
C        *** P(EFME) and P(EFMH) are the effective masses of electrons
C        and holes.  P(EFME) and P(EFMH) eqs. from least squares fit
C        of data in BARBER, SSE10, 1039 (1967). P(EFME) and P(EFMH)
```

```
C         eqs. fitted over temp range 0 to 600 K.  ***
          P(EFME)=(((((4.54649E-17*P(T)-9.66067E-14)*P(T)+8.04032E-11)*
     *    P(T)-3.320130E-8)*P(T)+6.83008E-6)*P(T)-.000161708)*P(T)+
     *    1.0627
          P(EFMH)=(((((1.11997E-16*P(T)-2.596730E-13)*P(T)+2.30049E-10)*
     *    P(T)-9.67212E-8)*P(T)+1.85678E-5)*P(T)-.000523548)*P(T)+
     *    .590525
C         *** P(EG) from fit of data from MACFARLANE,
C                          PHYS. REV. 111, 1245 (1958) ***
          P(EG)=(((-3.80977E-13*P(T)+9.95E-10)*P(T)-8.70110E-7)*
     *    P(T)+.0000323741)*P(T)+1.15556
        ELSEIF (MATSW.EQ.2) THEN
C         *** Computed for gallium arsenide ***
          P(RDIEL)=12.4*(1.+1.2E-04*P(T))
C         This formulation for the relative dielectric constant
C         was taken from Blakemore, see reference [3] of
C         EPROP manuscript.
          P(EG)=1.519-(5.405E-04*P(T)*P(T)/(P(T)+204.))
          MLL=P(EG)/(20.0-P(EG))
          P(EFME)=1./(1.+7.51*(2./P(EG)+1./(P(EG)+.341)))
          P(EFMH)=(MH**1.5+MLL**1.5)**(2./3.)
          P(DELGL)=.296-6.45E-5*P(T)**2/(P(T)+204.)
          P(DELGX)=.462+8.05E-5*P(T)**2/(P(T)+204.)
        ELSE
C ---option--- Temperature-dependent constants for third material;
C         here constants for InP have been used.  These do not
C         have any temperature dependence and are not necessarily
C         the best available.
          P(RDIEL)=10.6
          P(EFME)=0.078
          P(EFMH)=0.8
          P(EG)=1.35
        ENDIF
C       *** P(FNC) and P(FNV) are the density of states in the
C           conduction and valence bands respectively ***
        P(FNC)=CONST*(P(EFME)**1.5)*P(TPOWR)
        P(FNV)=CONST*(P(EFMH)**1.5)*P(TPOWR)
        RETURN
        END
C
C Subroutine UNITS defines the logical unit numbers for terminal I/O.
C
SUBROUTINE UNITS(LUTI,LUTO)
        INTEGER LUTI,LUTO
C VAX/VMS assignments:
        LUTI=5
        LUTO=6
C
C       MacFORTRAN assignments:
C       LUTI=9
C       LUTO=9
C
        RETURN
        END
C
C Subroutine USPEC allows the user to create a custom output set.
C
        SUBROUTINE USPEC (DSCRIP,INDVAR,HYBNUM,HYBSET,MAXOUT)
C
        INTEGER T,RTEMP,EF,CN,CP,DI1,DI2,DI3,DI4,DI5,EG,SUM,CAYT,ECMEF,
       * RDIEL,TPOWR,EFME,EFMH,FNC,FNV,DN1,EN1,DG1,DN2,EN2,DG2,DN3,EN3,
       * DG3,DN4,EN4,DG4,DN5,EN5,DG5,FNCP,FNVP,CNG,CNL,CNX,DELGL,DELGX
       * DI6,DN6,EN6,DG6
        PARAMETER (T=1,RTEMP=2,EF=3,CN=4,CP=5,DI1=6,DI2=7,DI3=8,DI4=9,
       * DI5=10,EG=11,SUM=12,CAYT=13,ECMEF=14,RDIEL=15,TPOWR=16,EFME=17,
       * EFMH=18,FNC=19,FNV=20,DN1=21,EN1=22,DG1=23,DN2=24,EN2=25,DG2=26,
```

```
     * DN3=27,EN3=28,DG3=29,DN4=30,EN4=31,DG4=32,DN5=33,EN5=34,DG5=35,
     * FNCP=36,FNVP=37,CNG=38,CNL=39,CNX=40,DELGL=41,DELGX=42,DI6=43,
     * DN6=44,EN6=45,DG6=46)
      INTEGER LUTI,LUTO
      COMMON /LUNIT/ LUTI,LUTO
C
      INTEGER INDVAR,INDEX1,INDEX2,HYBNUM,HYBSET(28),MAXOUT
      INTEGER BASE,COL,SP,ACOUNT,ADDLST(27),REMLST(28),ROWS
      INTEGER RSPON,OPTION(28)
      CHARACTER*80 OUTSTR*80,INSTR*80,DSCRIP(46)*11,CH*1
      LOGICAL REM,R,ADD,NEW,SKIP,DIGIT,INRNGE
      DATA OPTION /T,RTEMP,EF,CN,CP,DI1,DI2,DI3,DI4,
     *         DI5,DI6,CAYT,EG,ECMEF,SUM,TPOWR,RDIEL,
     *         EFME,EFMH,FNC,FNV,FNCP,FNVP,CNG,CNL,
     *         CNX,DELGL,DELGX/
      DATA OUTSTR /' '/
      HYBSET(1)=INDVAR
      WRITE (LUTO,'(/////)')
1     REM=.FALSE.
      R=.FALSE.
      ADD=.FALSE.
      NEW=.FALSE.
      SKIP=.FALSE.
      ACOUNT=0
      INSTR=' '
      DO 5 I=1,28
           REMLST(I)=0
5     CONTINUE
C     *** Create user specified output set menu ***
      WRITE (LUTO,'(25X,''User-Specified Output Set'',/)')
      WRITE (LUTO,'(2X,''Current Set :'')')
      OUTSTR(1:1)=' '
      OUTSTR(4:4)='>'
      OUTSTR(19:19)='>'
      OUTSTR(34:34)='>'
      OUTSTR(49:49)='>'
      OUTSTR(64:64)='>'
      BASE=2
      ROWS=HYBNUM/5
      IF (HYBNUM.NE.ROWS*5) ROWS=ROWS+1
      DO 25 I=1,ROWS
           COL=0
           INDEX1 = (I-1)*5+1
           INDEX2 = INDEX1+4
           IF (INDEX2.GT.HYBNUM) INDEX2=HYBNUM
           DO 20 J=INDEX1,INDEX2
                SP=BASE+15*COL
                WRITE (OUTSTR(SP:SP+1),'(I2)') J
                WRITE (OUTSTR(SP+3:SP+14),'(1X,A11,1X)')
     *               DSCRIP(HYBSET(J))
                COL=COL+1
20         CONTINUE
      WRITE (LUTO,*) OUTSTR(:SP+14)
25    CONTINUE
      WRITE (LUTO,'(/,2X,''Optional Output Parameters :'')')
      ROWS=MAXOUT/5
      IF (MAXOUT.NE.ROWS*5) ROWS=ROWS+1
      DO 40 I=1,ROWS
           COL=0
           INDEX1 = (I-1)*5+1
           INDEX2 = INDEX1+4
           IF (INDEX2.GE.MAXOUT) INDEX2=MAXOUT
           DO 30 J=INDEX1,INDEX2
                SP=BASE+15*COL
                WRITE (OUTSTR(SP:SP+1),'(I2)') J
                WRITE (OUTSTR(SP+3:SP+14),'(1X,A11,1X)')
```

```
     *              DSCRIP(OPTION(J))
                COL=COL+1
30         CONTINUE
           WRITE (LUTO,*) OUTSTR(:SP+14)
40    CONTINUE
      WRITE (LUTO,3000)
3000  FORMAT(/,10X,'Enter R[number(s)] to remove from the current',
     *       ' output set,',/,10X,'       A[number(s)] to add to the',
     *       ' current output set, or',/,10X,'       C[number(s)]',
     *       ' to create a new set.',/,10X,'Separate numbers by',
     *       ' commas or spaces, press return to exit.',/,10x,
     *       ' (Note: Use numbers from the Current set for a removal)',/)
      CALL PROMPT(' Input > ',0,2)
      READ (LUTI,'(A)') INSTR
      IF (INSTR(:20).EQ.'                    ') GOTO 100
      DO 50 I=1,80
           IF ((INSTR(I:I).EQ.' ').OR.(INSTR(I:I).EQ.',').OR.SKIP) THEN
                SKIP =.FALSE.
           ELSEIF ((INSTR(I:I).EQ.'R').OR.(INSTR(I:I).EQ.'r')) THEN
                IF (ADD.OR.NEW) THEN
                     WRITE (LUTO,*)' Error: Remove operation',
     *                    ' must be done first.'
                     WRITE (LUTO,*)'        Hit <RETURN> to continue.'
                     READ (LUTI,'(A)') CH
                     GOTO 1
                ELSE
                     REM=.TRUE.
                     R=.TRUE.
                ENDIF
           ELSEIF ((INSTR(I:I).EQ.'A').OR.(INSTR(I:I).EQ.'a')) THEN
                IF (NEW) THEN
                     WRITE (LUTO,*)' Error: Once a create command',
     *                    ' is given, no others are accepted.'
                     WRITE (LUTO,*)'        Hit <RETURN> to continue.'
                     READ (LUTI,'(A)') CH
                     GOTO 1
                ELSE
                     ADD=.TRUE.
                     REM=.FALSE.
                ENDIF
           ELSEIF ((INSTR(I:I).EQ.'C').OR.(INSTR(I:I).EQ.'c')) THEN
                ADD=.FALSE.
                REM=.FALSE.
                NEW=.TRUE.
                ACOUNT=1
           ELSEIF (DIGIT(INSTR(I:I))) THEN
                IF (DIGIT(INSTR(I+1:I+1))) THEN
                     SKIP=.TRUE.
                     READ (INSTR(I:I+1),'(I2)') RSPON
                ELSE
                     READ (INSTR(I:I),'(I1)') RSPON
                ENDIF
                IF (INRNGE(1,MAXOUT,RSPON)) THEN
                     IF (REM) THEN
                          IF (.NOT.INRNGE(1,HYBNUM,RSPON)) THEN
                               WRITE (LUTO,'(//)')
                               WRITE (LUTO,*)' Warning: ',
     *                              'attempt to remove entry #',
     *                              RSPON,' ignored'
                          ELSEIF (RSPON.NE.1) THEN
                               REMLST(RSPON)=-1
                          ELSE
                               WRITE (LUTO,'(//)')
                               WRITE (LUTO,*)' Warning:  your',
     *                              ' attempt to remove the indepen',
     *                              'dent variable from the first',
```

```
     *                              ' column was ignored'
                              ENDIF
                         ELSEIF (ADD.OR.NEW) THEN
                              IF (ACOUNT.LE.(MAXOUT-1)) THEN
                                   ACOUNT=ACOUNT+1
                                   ADDLST(ACOUNT)=RSPON
                              ELSE
                                   WRITE (LUTO,*)' Warning:',
     *                                 '  attempt to add more than',
     *                                 MAXOUT,
     *                                 ' variables ignored'
                                   GOTO 51
                              ENDIF
                         ELSE
                              WRITE (LUTO,*)' Error: input ',
     *                              'line must start with a command'
                              WRITE (LUTO,*)'        Hit <RETURN> ',
     *                                    'to continue.'
                              READ (LUTI,'(A)') CH
                              GOTO 1
                         ENDIF
                    ELSE
                         WRITE (LUTO,*)' Warning: entry # ',RSPON,
     *                        ' is out of range and has been ignored'
                    ENDIF
               ELSE
                    WRITE (LUTO,*)' Error: illegal character ',
     *                   'in input line'
                    WRITE (LUTO,*)'        Hit <RETURN> to continue.'
                    READ (LUTI,'(A)') CH
                    GOTO 1
               ENDIF
50        CONTINUE
51        IF (NEW) THEN
               IF (ACOUNT.GT.MAXOUT) THEN
                    WRITE (LUTO,*)' Warning: too many parameters.',
     *                '-- list truncated.'
                    ACOUNT=MAXOUT
               ENDIF
               DO 55 I=2,ACOUNT
                    HYBSET(I)=OPTION(ADDLST(I))
55             CONTINUE
               HYBNUM=ACOUNT
          ELSE
               IF (R) THEN
                    I=1
60                  IF (REMLST(I).EQ.-1) THEN
                         DO 65 J=I,HYBNUM-1
                              REMLST(J)=REMLST(J+1)
                              HYBSET(J)=HYBSET(J+1)
65                       CONTINUE
                         HYBNUM=HYBNUM-1
                    ELSE
                         I=I+1
                    ENDIF
                    IF (I.LT.HYBNUM) GOTO 60
                    IF (REMLST(HYBNUM).EQ.-1) HYBNUM=HYBNUM-1
               ENDIF
               IF (ADD) THEN
                    L=ACOUNT+HYBNUM
                    IF (L.GT.MAXOUT) THEN
                         WRITE (LUTO,*)' Warning: too many',
     *                      ' parameters.-- list truncated.'
                         ACOUNT=MAXOUT-HYBNUM
                    ENDIF
                    DO 70 I=1,ACOUNT
```

```
                    HYBNUM=HYBNUM+1
                    HYBSET(HYBNUM)=OPTION(ADDLST(I))
70              CONTINUE
           ENDIF
      ENDIF
      GOTO 1
100   RETURN
      END
C
C Subroutine WRAPUP closes all opened files.
C
      SUBROUTINE WRAPUP(LUDEF,LSTSW,NUMER,PLTSW,PNUM)
C
      INTEGER LUIN,LULST,LUPL1,LUPL2,
     * LUPL3,LUPL4,LUTMP,LUER
      PARAMETER (LUIN=20,LULST=21,LUPL1=22,LUPL2=23,
     * LUPL3=24,LUPL4=25,LUTMP=26,LUER=27)
C
      INTEGER LUTI,LUTO
      COMMON /LUNIT/ LUTI,LUTO
      INTEGER LUDEF,LUPL(4),PNUM
      LOGICAL TRMSW,LSTSW,PLTSW
      CHARACTER*80 OUTSTR
      DATA LUPL /LUPL1,LUPL2,LUPL3,LUPL4/
C
      IF (NUMER.NE.0) THEN
           IF (LSTSW) THEN
                WRITE (LULST,5) NUMER
5               FORMAT(16X,I3,' errors in computation. ',
     *          'See ERROR.LIS')
           ELSE
                REWIND (UNIT=LUER)
7               READ (LUER,'(A)',ERR=8,END=8) OUTSTR
                WRITE (LUTO,'(A)') OUTSTR
                GOTO 7
           ENDIF
8          WRITE (LUTO,5) NUMER
           CLOSE (UNIT=LUER)
      ELSE
           CLOSE (UNIT=LUER,STATUS='DELETE')
      ENDIF
      IF (LSTSW) CLOSE (UNIT=LULST)
      IF (LUDEF.EQ.LUTMP) CLOSE (UNIT=LUTMP,STATUS='DELETE')
      IF (PLTSW) THEN
           DO 10 I=1,PNUM
                CLOSE (UNIT=LUPL(I))
10         CONTINUE
      ENDIF
      WRITE (LUTO,'(''           --------------  Computation '',
     *           ''Complete  --------------'')')
      RETURN
      END
C
C Subroutine ZEROIN
C
      SUBROUTINE ZEROIN(F,B,C,RE,AE,IFLAG)
C
C     Sandia Mathematical Program Library
C     Mathematical Computing Services Division 5422
C     Sandia Laboratories
C     P. O. Box 5800
C     Albuquerque, New Mexico  87115
C     Control Data 6600 Version 4.5, 1 November 1971
C
C     Modified to run at NBS by D. Kahaner, Division 713
C
```

```
C     Abstract
C        ZEROIN searches for a zero of a function F(X) between
C        the given values B and C until the width of the interval
C        (B,C) has collapsed to within a tolerance specified by
C        the stopping criterion, ABS(B-C) .LE. 2.*(RW*ABS(B)+AE).
C
C     Description of Arguments
C        F     - Name of the real valued external function.  This name
C                must be in an external statement in the calling
C                program.  F must be a function of one real argument.
C        B     - One end of the interval (B,C).  The value returned for
C                B usually is the better approximation to a zero of F.
C        C     - The other end of the interval (B,C)
C        RE    - Relative error used for RW in the stopping criterion.
C                If the requested RE is less than machine precision,
C                then RW is set to approximately machine precision.q
C        AE    - Absolute error used in the stopping criterion.  If the
C                given interval (B,C) contains the origin, then a
C                nonzero value should be chosen for AE.
C        IFLAG - Returns a status of the results indicating which
C                of the following hold.
C                    A - ABS(B-C) .LE. 2.*(RW*ABS(B)+AE)
C                    B - F(B) * F(C) .LT. 0.
C                    C - ABS(F(B)) .LE. ABS(F(C))
C                    D - ABS(F(B   )) .LE. MAX(ABS(F(B  )),ABS(F(C  )))
C                                  OUT                IN          IN
C                    E - Number of evaluations of F(X) .LE. 500
C                  =1 Indicates normal case.  All conditions above hold.
C                  =2 Indicates F(B) = 0.  Condition A may not hold.
C                  =3 Indicates conditions A, B, C, and E hold but D does
C                     not.  (B,C) probably contains a singular point of F.
C                  =4 Indicates conditions A and E hold but B does not.
C                     A local minimum of F(X) in (B,C) may have been found.
C                  =5 Indicates search was aborted when condition E failed.
C
C     References
C       1.  L F Shampine and H A Watts, ZEROIN, A Root-Solving Code,
C           SC-TM-70-631, Sept 1970.
C       2.  T J Dekker, Finding a Zero by means of Successive Linear
C           Interpolation, *Constructive Aspects of the Fundamental
C           Theorem of Algebra*, Edited by B Dejon and P Henrici, 1969.
C       3.  L F Shampine and R C Allen, Numerical Computing--
C           an Introduction, 1973.
C
C     Initialize for UNIVAC 1108 or VAX 11/780
      DATA ER/6.0E-8/
C     Initialize in a portable manner
C     DATA ER/0.0/
C     IF (ER .EQ. 0.0) ER = 4. * (R1MACH (4) )
C
      RW=AMAX1(RE,ER)
      IC=0
      ACBS=ABS(B-C)
      A=C
      FA=F(A)
      FB=F(B)
      FC=FA
      KOUNT=2
      FX=AMAX1(ABS(FB),ABS(FC))
C
    1 IF (ABS(FC) .GE. ABS(FB)) GO TO 2
C     Perform interchange
      A=B
      FA=FB
      B=C
      FB=FC
```

```
      C=A
      FC=FA
C
    2 CMB=0.5*(C-B)
      ACMB=ABS(CMB)
      TOL=RW*ABS(B)+AE
C
C     Test stopping criterion
      IF (ACMB .LE. TOL) GO TO 10
C
C     Calculate new iterate implicitly as B+P/Q,
C     where we arrange P.GE.0.
C     The implicit form is used to prevent overflow.
      P=(B-A)*FB
      Q=FA-FB
      IF (P .GE. 0.) GO TO 3
      P=-P
      Q=-Q
C
C     Update A and check for satisfactory reduction
C     in the size of our bounding interval.
    3 A=B
      FA=FB
      IC=IC+1
      IF (IC .LT. 4) GO TO 4
      IF (8.*ACMB .GE. ACBS) GO TO 6
      IC=0
      ACBS=ACMB
C
C     Test for too small a change
    4 IF (P .GT. ABS(Q)*TOL) GO TO 5
C
C     Increment by tolerance
      B=B+SIGN(TOL,CMB)
      GO TO 7
C
C     Root ought to be between B and (C+B)/2.
    5 IF (P .GE. CMB*Q) GO TO 6
C
C     Interpolate
      B=B+P/Q
      GO TO 7
C
    6 B=0.5*(C+B)
C     Bisect
C
C     Have completed computation for new iterate B
    7 FB=F(B)
      IF (FB .EQ. 0.) GO TO 11
      KOUNT=KOUNT+1
      IF (KOUNT .GT. 500) GO TO 15
C
C     Decide whether next step is interpolation or extrapolation
      IF (SIGN(1.0,FB) .NE. SIGN(1.0,FC)) GO TO 1
      C=A
      FC=FA
      GO TO 1
C
C     Finished.  Process results for proper setting of IFLAG
C
   10 IF (FB*FC .GT. 0.) GO TO 13
      IF (ABS(FB) .GT. FX) GO TO 12
      IFLAG = 1
      RETURN
   11 IFLAG = 2
      RETURN
```

```
12 IFLAG = 3
   RETURN
13 IFLAG = 4
   RETURN
15 IFLAG = 5
   RETURN
   END
```

☆ U.S. GOVERNMENT PRINTING OFFICE: 1990— 261-913/20600

NIST-114A
(REV. 3-89)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

BIBLIOGRAPHIC DATA SHEET

1. PUBLICATION OR REPORT NUMBER
NIST/SP-400/85

2. PERFORMING ORGANIZATION REPORT NUMBER

3. PUBLICATION DATE
May 1990

4. TITLE AND SUBTITLE
Semiconductor Measurement Technology: EPROP: An Interactive FORTRAN Program for Computing Selected Electronic Properties of Gallium Arsenide and Silicon

5. AUTHOR(S)
A. C. Seabaugh, J. J. Mathias, and M. I. Bell

6. PERFORMING ORGANIZATION (IF JOINT OR OTHER THAN NIST, SEE INSTRUCTIONS)
U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
GAITHERSBURG, MD 20899

7. CONTRACT/GRANT NUMBER

8. TYPE OF REPORT AND PERIOD COVERED
Final

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (STREET, CITY, STATE, ZIP)
Same as item #6

10. SUPPLEMENTARY NOTES

[ ] DOCUMENT DESCRIBES A COMPUTER PROGRAM; SF-185, FIPS SOFTWARE SUMMARY, IS ATTACHED.

11. ABSTRACT (A 200-WORD OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, MENTION IT HERE.)

A new computer program, EPROP (an acronym for Electronic PROPerties) is presented for use in interpreting measurements and experiments on gallium arsenide and silicon. EPROP computes a solution of the charge balance equation in thermodynamic equilibrium for up to six different impurities. The user supplies the density, energy level, and degeneracy for each impurity, and in response the program returns as many as 28 output parameters, such as the Fermi level, carrier density, and ionized impurity densities. These can be computed as functions of the temperature (or reciprocal temperature) or the density, energy, or degeneracy of any of the six possible impurities. Listings can also be obtained of various temperature-dependent parameters, such as the bandgap, densities of states, and effective masses. The interactive features of the program allow the user to send the output data to any combination of destinations: a terminal, a listing file, and/or up to four graphic output files, all at the user's direction. The user is also given freedom and ability to customize the data output to these destinations through menu-driven controls. The program is written in ANSI standard FORTRAN 77 and has been successfully compiled and run on both mainframe and microcomputers. Documentation is provided to assist the interested user in customizing the program for special applications, extracting portions for use elsewhere, or modifying the code to treat semiconductors other than silicon and gallium arsenide.

12. KEY WORDS (6 TO 12 ENTRIES; ALPHABETICAL ORDER; CAPITALIZE ONLY PROPER NAMES; AND SEPARATE KEY WORDS BY SEMICOLONS)
electronic properties; Fermi level; FORTRAN; GaAs; gallium arsenide; Si; silicon

13. AVAILABILITY

[X] UNLIMITED

[ ] FOR OFFICIAL DISTRIBUTION. DO NOT RELEASE TO NATIONAL TECHNICAL INFORMATION SERVICE (NTIS).

[X] ORDER FROM SUPERINTENDENT OF DOCUMENTS, U.S. GOVERNMENT PRINTING OFFICE, WASHINGTON, DC 20402.

[X] ORDER FROM NATIONAL TECHNICAL INFORMATION SERVICE (NTIS), SPRINGFIELD, VA 22161.

14. NUMBER OF PRINTED PAGES
121

15. PRICE

ELECTRONIC FORM

NIST Technical Publications

Periodical

Journal of Research of the National Institute of Standards and Technology—Reports NIST research and development in those disciplines of the physical and engineering sciences in which the Institute is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Institute's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Institute's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NIST, NIST annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

Applied Mathematics Series—Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NIST under the authority of the National Standard Data Act (Public Law 90-396). NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published quarterly for NIST by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements are available from ACS, 1155 Sixteenth St., NW., Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Institute on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NIST under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NIST administers this program as a supplement to the activities of the private sector standardizing organizations.

Consumer Information Series—Practical information, based on NIST research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.

Order the **above** *NIST publications from: Superintendent of Documents, Government Printing Office, Washington, DC 20402.*

Order the **following** *NIST publications—FIPS and NISTIRs—from the National Technical Information Service, Springfield, VA 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NIST pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NIST Interagency Reports (NISTIR)—A special series of interim or final reports on work performed by NIST for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service, Springfield, VA 22161, in paper copy or microfiche form.

U.S. Department of Commerce
National Institute of Standards and Technology
(formerly National Bureau of Standards)
Gaithersburg, MD 20899

Official Business
Penalty for Private Use $300