10/3-2-90 08S(1)

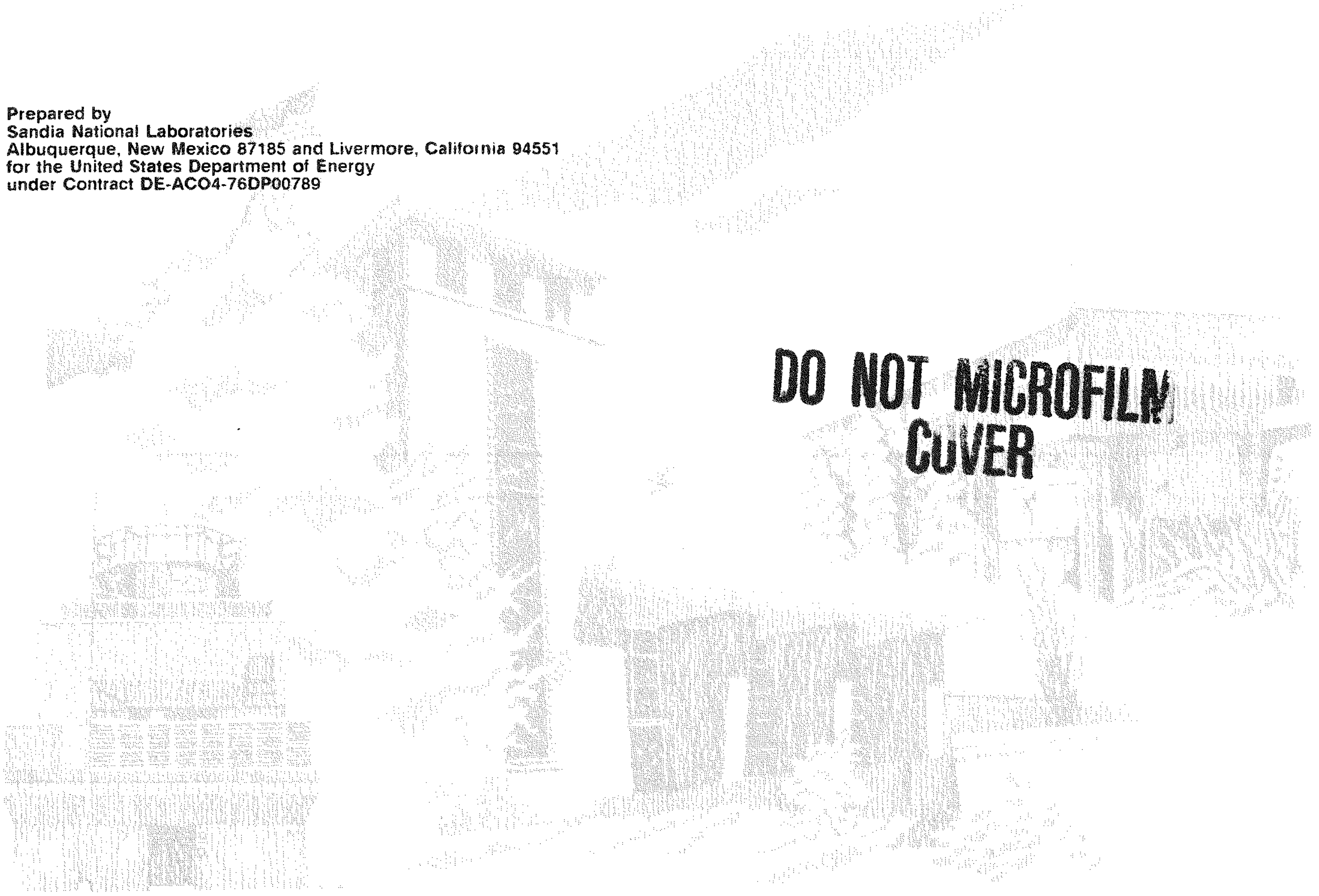
SANDIA REPORT

SAND89-8266 • UC-905
Unlimited Release
Printed January 1990

DANCIR: A Three-Dimensional Steady-State Semiconductor Device Simulator

J. C. Meza, J. F. Grcar

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94551
for the United States Department of Energy
under Contract DE-AC04-76DP00789



DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of the contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof or any of their contractors or subconractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof or any of their contractors or subcontractors.

This report has been reproduced from the best available copy.

Available to DOE and DOE contractors from:

Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge TN 37831

Prices available from (615) 576-8401, FTS 626-8401.

Available to the public from:

National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Rd.
Springfield, VA 22161.

NTIS price codes
Printed copy: A03
Microfiche copy: A01

SAND--89-8266

DE90 006370

UC-905

SAND89-8266
Unlimited Release
Printed January 1990

DANCIR: A THREE-DIMENSIONAL STEADY-STATE SEMICONDUCTOR DEVICE SIMULATOR

J. C. Meza and J.F. Grcar
Scientific Computing and Applied Mathematics Division
Sandia National Laboratories
Livermore, CA 94551-0969

ABSTRACT

This report discusses a program called DANCIR that was developed to model semiconductor devices. DANCIR, which stands for Device and Circuit Simulator, enables the user to compute the steady-state solution to the drift-diffusion equations for a single semiconductor device. The drift-diffusion equations that describe carrier motion in a semiconductor device can be used to compute operating characteristics for a semiconductor device. This allows the engineer to design different devices and predict the behavior of these devices without resorting to the time-consuming process of building many prototypes.

Contents

DANCIR: A THREE-DIMENSIONAL STEADY-STATE SEMICONDUCTOR DEVICE SIMULATOR

1. Introduction

Integrated circuits today are primarily designed by building and testing prototypes. Even though this methodology is a reliable way of assessing a circuit's performance, it is a slow and expensive way to produce a design. As a result, the design and production of semiconductor devices take several years and cost millions of dollars. An alternative is to design a semiconductor device by using mathematical models and computer simulations, coupled with a much reduced verification and testing program. This strategy has the potential of reducing the time needed to design and produce an integrated chip to as little as one year.

In the past, semiconductor device simulation has been used only as a secondary design tool because it was always easier to build and test a prototype than to build and validate a simulator. Integrated circuits based on bipolar device technology were commercially available in the late 1950's, but it was not until 1964 that Gummel obtained the first numerical solution of van Roosbroeck's three nonlinear partial differential equations describing a one-dimensional bipolar transistor [8]. Shortly after the first one-dimensional calculations had been perfected in the early 1970's, advances in process technology made high density MOS circuits feasible. These devices were soon modeled with two-dimensional device simulators, but limitations in computing power made realistic simulations impractical [7]. Today time-dependent, two-dimensional simulations are routine. One such example, is the two-dimensional simulator PISCES II which is still widely used in the United States [11,12]. As devices became smaller and more complex it became evident that three-dimensional effects needed to be incorporated in a device simulator. The first three-dimensional simulator appears to be the FIELDAY program announced by IBM in 1981 [1]. Toshiba Corporation announced its own three-dimensional simulator, TOPMOST, in 1982 [15]. In marked contrast to the FIELDAY program, a stream of research reports followed, indicating that significant engineering analyses were being undertaken with the aid of the TOPMOST program [16,17,18,19]. Hitachi Corporation announced its three-dimensional simulator, CADDETH, in 1985 [9,21] which is reportedly used hundreds of times monthly on dedicated Hitachi supercomputers [22].

Although much of the early device simulation work was performed in the United States, today most of the engineering applications of three-dimensional modeling are done in Japan. Starting in the early 1980's, the Japanese computer manufacturers undertook three-dimensional modeling efforts and by the mid 1980's had achieved dominance in the crucial semiconductor memory market. Because of this intense Japanese effort, the United States quickly lost its edge in the semiconductor industry and in related crucial technologies. Our position has been recognized by several governmental studies, including ones by the FCCSET (Federal Coordinating Council on Science, Engineering, and Technology) committee [4], and the Department of Defense Critical Technologies Plan [2]. Many reasons have been given for this phenomenon, one of which is that existing science and technology programs do not recognize the new market-driven model of innovation. Even though the United States clearly has the technical expertise to generate innovative ideas, the Japanese have dominated the crucial markets by quickly bringing to market new products based on ideas originated in the United States. The use of device simulators by the Japanese in their production cycle is clearly an important factor in the speed of their design process.

This report discusses the program DANCIR that was developed to model semiconductor devices. DANCIR, which stands for Device and Circuit Simulator, is a full three-dimensional simulator capable of computing the steady-state solution of the drift-diffusion equations for a single semiconductor device. Section 2 discusses the drift-diffusion equations, that are used to model a semiconductor device. Section 3 briefly discusses the numerical algorithms used to solve the drift-diffusion equations. Section 4 describes the input data required to run a typical simulation and includes a simple example.

2. The Drift-Diffusion Equations

The drift-diffusion model consists of a set of three, coupled, nonlinear partial differential equations: the potential equation plus two continuity equations, one each for the electron and hole current densities.

The potential or Poisson equation is given by

$$\epsilon \nabla \cdot E = -\epsilon \nabla^2 \psi = \rho, \tag{2.1}$$

where ϵ is the scalar permittivity of the semiconductor, ψ is the electric potential, and $E = -\nabla \psi$ is the electric field. The total electric charge density, ρ, is given by

$$\rho = q(p - n + N_D - N_A), \tag{2.2}$$

where q is the elementary charge, n is the density of free electrons, p is the density of holes, N_D is the density of donor impurities, and N_A is the density of acceptors.

The continuity equations for the electron and hole currents can be stated as

$$\frac{\partial n}{\partial t} - \frac{1}{q} \nabla \cdot J_n - R = 0 \tag{2.3}$$

$$\frac{\partial p}{\partial t} + \frac{1}{q} \nabla \cdot J_p - R = 0, \tag{2.4}$$

where J_n is the electron current density, J_p is the hole current density, and R is a term that accounts for the recombination and generation of electrons and holes. The recombination-generation term is a nonlinear function of the electric field and the bulk material temperature T.

In the drift-diffusion model, the movement of carriers is modeled by a drift term due to the acceleration of carriers by an external electric field and by a diffusion term due to a concentration gradient:

$$\begin{aligned} J_n &= q\mu_n nE + qD_n\nabla n, && (2.5)\\ J_p &= q\mu_p pE - qD_p\nabla p, && (2.6) \end{aligned}$$

where μ_n and μ_p are the electron and hole mobilities respectively, and D_n and D_p are the diffusion coefficients.

At low electric fields, the mobilities may be thought of as simply a constant of proportionality between a carrier's drift velocity and the electric field, that is,

$$v_d = \mu E. \tag{2.7}$$

The diffusion coefficients can in turn be related to the mobilities by the Einstein relation,

$$D_n = \mu_n \frac{kT}{q}, \qquad D_p = \mu_p \frac{kT}{q}, \tag{2.8}$$

where k is the Boltzmann constant.

Substitution of equations (2.5–2.6) into (2.3–2.4) and rearranging leads to

$$\frac{\partial n}{\partial t} - \nabla\cdot(\mu_n nE + D_n\nabla n) - R = 0, \tag{2.9}$$

$$\frac{\partial p}{\partial t} + \nabla\cdot(\mu_p pE - D_p\nabla p) - R = 0. \tag{2.10}$$

Equations (2.1) and (2.9–2.10), along with suitable boundary conditions, constitute the drift-diffusion equations. This set of equations can also be rigorously derived from the Boltzmann transport equation; however this section is only meant as a brief overview. For a more complete derivation of the equations the reader can consult a variety of references, for example [10,14,20].

3. Numerical Methods

The numerical solution of the drift-diffusion equations involves many issues. In this section, we briefly outline the numerical methods used in the DANCIR code for the solution of the steady-state, drift-diffusion equations. The numerical methods employed in the DANCIR code can be divided into three areas: 1) the spatial discretization of the nonlinear partial-differential equations, 2) the solution of the resulting nonlinear equations, and 3) the solution of the linear equations arising in the solution of the nonlinear system of equations. An added issue which must be addressed is the question of scaling. The particular scaling used in a simulator will impact the numerical methods used; therefore we will discuss the scaling issues involved in the solution of the drift-diffusion equations first.

Scaling of Variables.

The wide range in magnitude of both the dependent and independent variables creates difficulties in the numerical solution process. Independent variables such as the concentrations of impurity dopings N_D and N_A range from 10^{13} to 10^{19} carriers per cubic centimeter. Dependent variables such as the carrier densities n and p range from virtually 0 to 10^{19} carriers per cubic centimeter. The difficulties associated with the wide range in the magnitude of the dependent variables can be circumvented somewhat by employing different variables. The most popular formulations recast the drift-diffusion equations by expressing the carrier concentrations n and p in terms of either the quasi-Fermi potentials, ϕ_n and ϕ_p,

$$n = n_{ie}\exp\left(\frac{q(\psi-\phi_n)}{kT}\right), \tag{3.11}$$

$$p = n_{ie}\exp\left(\frac{q(\phi_p-\psi)}{kT}\right), \tag{3.12}$$

or the Slotboom variables, u and v,

$$n = n_{ie}\exp\left(\frac{q\psi}{kT}\right)u, \tag{3.13}$$

$$p = n_{ie}\exp\left(\frac{-q\psi}{kT}\right)v, \tag{3.14}$$

where n_{ie} is the effective intrinsic carrier concentration.

Polak [13] notes that changing variables amounts to trading high variability in the dependent variables for increased nonlinearity in the equations. For example, the quasi-Fermi potentials have a small range of magnitude, but the current densities are exponential functions of them, which makes the solution of the resulting nonlinear system more difficult. The Slotboom variables have an enormous range of magnitude, but the current continuity equations assume the form of Poisson or potential equations facilitating both the mathematical analysis and the numerical solution process.

In the DANCIR code, we chose to scale the variables by using the Slotboom variables. Using this scaling the current densities can be written as

$$J_n = kT\mu_n n_{ie}\exp(q\psi/kT)\nabla u, \tag{3.15}$$

$$J_p = -kT\mu_p n_{ie}\exp(-q\psi/kT)\nabla v. \tag{3.16}$$

The resulting steady-state, drift-diffusion equations can then be expressed as

$$f_1(\psi,u,v) = \epsilon\nabla^2\psi + q\left[n_{ie}e^{\frac{-q\psi}{kT}}v - n_{ie}e^{\frac{q\psi}{kT}}u + N_D - N_A\right] = 0, \tag{3.17}$$

$$f_2(\psi,u,v) = \nabla\cdot\left[kT\mu_n n_{ie}e^{\frac{q\psi}{kT}}\nabla u\right] + R = 0, \tag{3.18}$$

$$f_3(\psi,u,v) = \nabla\cdot\left[kT\mu_p n_{ie}e^{\frac{-q\psi}{kT}}\nabla v\right] + R = 0, \tag{3.19}$$

Spatial Discretization.

DANCIR uses a control volume discretization in the spatial domain. This technique is widely used in fluid dynamics whenever conservation properties are important. In the special case of a uniform, rectangular grid this method can be shown to be equivalent to a centered difference approach [5].

Nonlinear Equations.

DANCIR uses Gummel's method to solve the nonlinear equations arising from the discretization of the drift-diffusion equations. If f_1, f_2, and f_3 are defined by equations (3.17–3.19) then the nonlinear system of equations arising from the spatial discretization of the drift-diffusion equations can be written as

$$F(\psi, u, v) = \begin{pmatrix} f_1 \\ f_3 \\ f_3 \end{pmatrix} = 0. \tag{3.20}$$

There are many techniques available in the literature for solving systems of nonlinear equations. Gummel [8] proposed a solution method where each one of the equations f_1, f_2, and f_3 is linearized by holding two of the three variables fixed. This method is known in the numerical analysis literature as the nonlinear Gauss-Seidel method. Specifically the method consists of the following algorithm:

```
Algorithm: Gummel Iteration.
Compute Initial Estimates $\psi^0, u^0, v^0$
For $k = 0, 1, 2, \ldots$
    Solve $f_1(\psi^{k+1}, u^k, v^k) = 0$ for $\psi^{k+1}$.
    Solve $f_2(\psi^{k+1}, u^{k+1}, v^k) = 0$ for $u^{k+1}$.
    Solve $f_3(\psi^{k+1}, u^{k+1}, v^{k+1}) = 0$ for $v^{k+1}$.
    Check for convergence.
```

Gummel's method has the advantage of only having to solve three linear systems at each iteration. The disadvantage is that convergence can be quite slow in certain circumstances, for example in high voltage situations.

In practice, the solution of the steady-state, drift-diffusion equations is accomplished by solving a series of continuation steps where each continuation step is in turn a steady-state problem. The first steady-state problem solved is that of the device with no external voltages applied. The potential at the contacts is then incremented until the desired voltage is reached at the contacts. Future work will concentrate on developing more efficient techniques utilizing quasi-Newton methods (see for example [3]).

The initial estimate for the potential is computed by solving a nonlinear potential equation. DANCIR uses a Newton method for this calculation because the Jacobian is symmetric in this case thereby not incurring any extra expense for storage over the Gummel iteration.

Linear Equations.

The final step in the solution of the drift-diffusion equations consists of solving the linear systems arising in the Gummel iteration. As we mentioned above, the use of the Slotboom variables has the effect of transforming the continuity equations for the electron and hole currents into a set of self-adjoint partial-differential equations. In practice this means that the linear systems resulting from the discretization are symmetric so that any of a number of techniques for solving symmetric linear systems could be used in the solution process. The particular method used in DANCIR is a preconditioned conjugate gradient method with an incomplete Cholesky factorization used as the preconditioner [6].

4. User Input Data Sets

There are two data sets the user will need to supply to run a simulation. The first data set consists of a device definition file which specifies the geometry of the device and certain operating parameters. The second data set the user must provide is a subroutine which computes the doping profile for the device of interest.

The device definition file format is designed to be user-friendly. The input for DANCIR has a structure similar to a high-level language and can be broken down into a hierarchy of nested blocks. Each block is started by a particular keyword and terminated by the same keyword prefixed by the string **end**. Within each block the input consists of a sequence of alphanumeric keywords along with a numerical value specifying a particular parameter. For example, the top-most block is started with a line containing the keywords *steady version 2.0* and should be terminated by the keywords *end steady.* Within the main block the input can be divided into 5 sub-blocks:

1. Operating environment characteristics.
2. Definition of a device.
3. Writing of a device for restart purposes.
4. Reading of a device for restart purposes.
5. Parameters for the solution of the steady state.

In the following sections, the characters a, i, x in an input specification stand for alphanumeric, integer, and real values required from the user. Any line which starts with the character ! is considered a comment. A sample input data set is given in Appendix A for the purposes of illustration.

Specify the operating environment.

This block is started with the keywords: **specify the operating environment.** Valid keywords within this block include:

print level $= i$

Specifies the amount of print output. A print level = 0 will produce the mininal amount of output. Large numbers will increase the amount of output.

specify the scratch file

Specifies parameters for the scratch file used on the Cray version. Allowed parameters include:

maximum number of blocks $= x$

maximum number of words $= x$

physical device number $= i$

Specifies the device number for the solid state device. This keyword is only used on the Cray version.

Define a device.

This block is started with the keywords: **define a device.** Valid keywords within this block include:

name of device $= a$

Name of the device enclosed in quotes.

doping routine $= a$

Name of the doping routine that DANCIR will call to determine the doping profile of the device.

number of contacts $= i$

Number of device contacts.

name of contact $i = a$

The name of contact i. There must be one name for each contact specified above.

name of axis $i = a$

A name for axis i (for example x, y, z).

dielectric constant of $a = x$

The dielectric constant of either the insulator or the semiconductor is set to x. The character string a must be either "insulator" or "semiconductor".

Construct a grid.

DANCIR has the capability to easily construct a rectangular grid based on some user-supplied data. Either a uniform grid or a nonuniform grid may be constructed. If the user does not specify a grid for a particular axis then DANCIR will generate a unicell grid for that axis. A uniform grid is fairly easy to specify, requiring a minimum of input, but for any realistic device a nonuniform grid will be required to generate accurate results. The generation of a nonuniform grid can also be accomplished through the specification of a few simple criteria. DANCIR will read the user input and attempt to generate an optimal nonuniform grid through the use of a rather complicated algorithm. To generate either a nonuniform or a uniform grid the user should have a line of the form: **construct a [non]uniform grid for axis i**, where the axis is either 1, 2, or 3. Valid keywords for a uniform grid include:

boundary at x
A boundary point is set at the point x.

step from x_1 **to** $x_2 = x_3$
The step size used in the interval $[x_1, x_2]$

Valid keywords for a nonuniform grid include:

default maximum step $= x$
The default maximum step allowed in the generation of a nonuniform grid.

default maximum step ratio $= x$
The maximum step ratio allowed between adjoining cells in the generation of a nonuniform grid.

default step at boundaries $= x$
The step used at the boundaries in the generation of a nonuniform grid.

default maximum step $= x$
The maximum step allowed in the generation of a nonuniform grid.

maximum step from x_1 **to** $x_2 = x_3$
The maximum step allowed in the interval $[x_1, x_2]$ in the generation of a nonuniform grid.

maximum step ratio from x_1 **to** $x_2 = x_3$
The maximum step ratio allowed between adjoining points in the interval specified by x_1 and x_2

boundary at x **with default step**
A boundary point is set at x. A cell is then centered about that point with a step give by the default step size.

boundary at x_1 with step = x_2

A boundary point is set at x_1. A cell is then centered about that point with a step give by step x_2.

Write the device.

This block is started with the keywords: **write the device**. Valid keywords within this block include:

name of file = a

Name of the file to write. This file will be created if it doesn't exist.

number of block in file = i

File number to write.

append block to file = a

Append current block to end of file a.

remark = a

Any identifying comments (must be enclosed in quotes).

Read a device.

This block is started with the keywords: **read a device**. Valid keywords within this block include:

name of file = a

Name of the file to read. This file must already exist.

number of block in file = i

File number to read.

Determine steady state.

This block is started with the keywords: **determine steady state**. Valid keywords within this block include:

potential of contact $a = x$

The potential of contact a is set to the value of x. There should be one line for each contact specified in the define device block.

maximum change to contact potentials = x

The maximum change to the contact potentials between continuation steps.

maximum number of gummel steps = i

The maximum number of Gummel steps to take before
stopping execution.

accuracy of potential from gummel method = x

Convergence tolerance for the potential in the gummel method.

maximum number of newton steps = i

The maximum number of Newton steps to take before
stopping execution. Newton's method is only used in the computation
of the initial estimate of the potential for the Gummel iteration.

accuracy of potential from newton method = x

Convergence tolerance for the potential in Newton's method.

level of fill in iccg factorization = i

The level of fill-in in the Incomplete Cholesky factorization
used as the preconditioner for the conjugate gradient method.
Larger numbers will speed up the convergence rate of the
conjugate gradient method at the expense of greater storage
requirements. Value must be ≥ 0.

maximum number of iccg steps = i

The maximum number of ICCG steps to take before
stopping execution.

sufficient relative residual from iccg = x

Convergence tolerance for the conjugate gradient method.

write the contact data

Flag specifying that DANCIR should write out the contact data.

Doping Routine.

DANCIR is setup to call one of 10 subroutines named dope00 – dope09 for the calculation of the doping profile. The user must specify in the input data set the name of one of these doping routines. DANCIR calls the specified doping routine once for each set of coordinates. The doping routine should return the doping level for both acceptors and donors at that coordinate point. In addition, the subroutine should also return values indicating whether that point is part of a contact and what type of material it is. The calling sequence for the doping routine is:

```
      subroutine dope00(error, text, contac, coord1, coord2,
     +                  coord3, nd, na, mtrl)
```

The parameters are:

error	Output	Logical. Error flag
text	Input	Integer. File number for error messages

contac	Output	Integer. Contact number
coord1	Input	Real*8. Coordinate number 1 (usually x)
coord2	Input	Real*8. Coordinate number 2 (usually y)
coord3	Input	Real*8. Coordinate number 3 (usually z)
nd	Output	Real*8. Doping level of donors in units of carriers/cm^3
na	Output	Real*8. Doping level of acceptors in units of carriers/cm^3

mtrl	Output	Real*8. Material number
		semiconductor = 0.0
		insulator = 1.0

A sample doping routine for a 2D model of an abrupt $p-n$ junction diode is given in Appendix B as a guideline to developers.

Using the sample input data set given in Appendix A along with the doping routine in Appendix B we computed the steady state solution for a $p-n$ junction diode at 1 volt. Figures 1-2 display the electron and hole concentrations computed by DANCIR. Figure 3 is a plot of the potential at 1 volt. Figure 4 is a plot of the current through the diode as a function of the contact voltage applied.

Figure 1. Electron Concentration For P-N Junction Diode

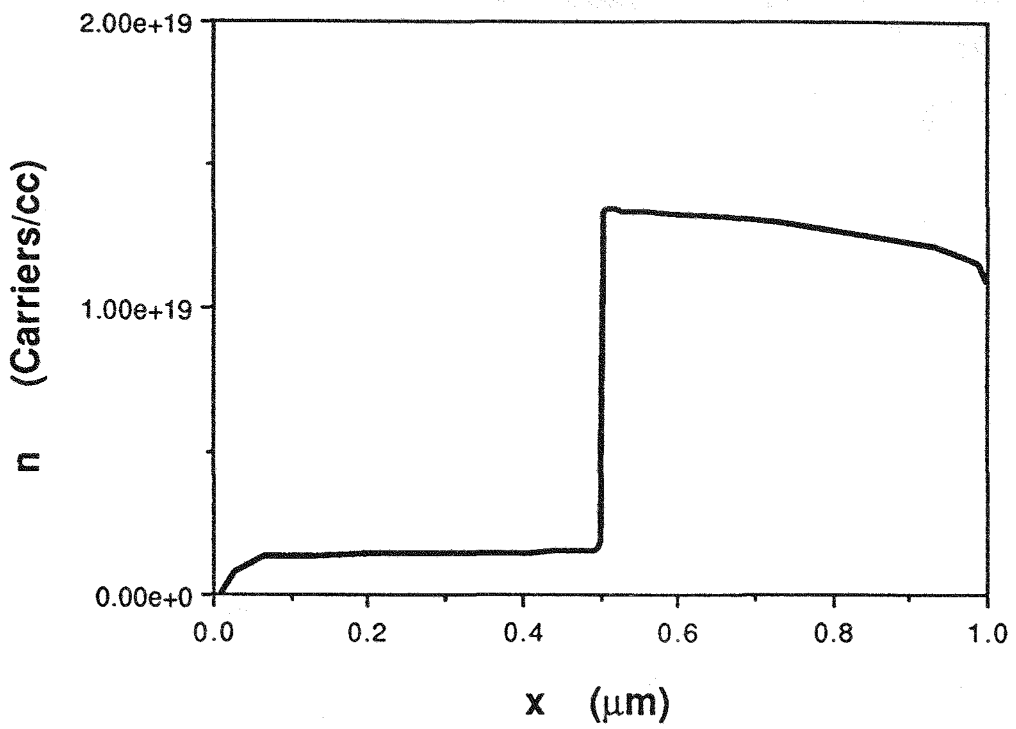


Figure 2. Hole Concentration For P-N Junction Diode

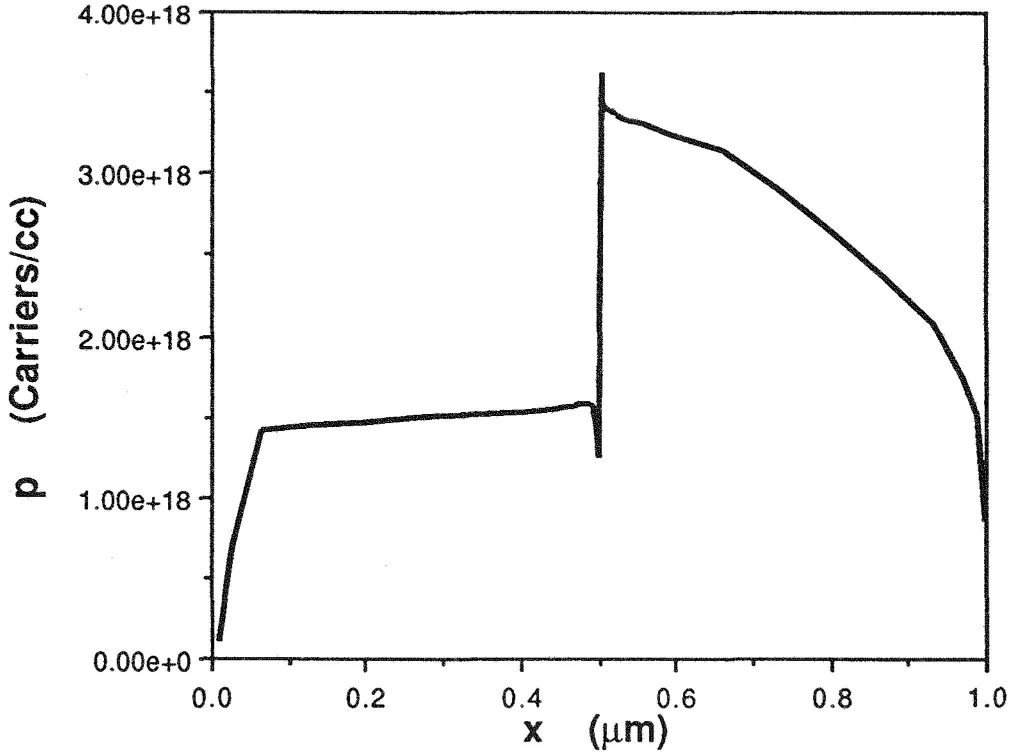


Figure 3. Potential For P-N Juntion Diode At 1 V.

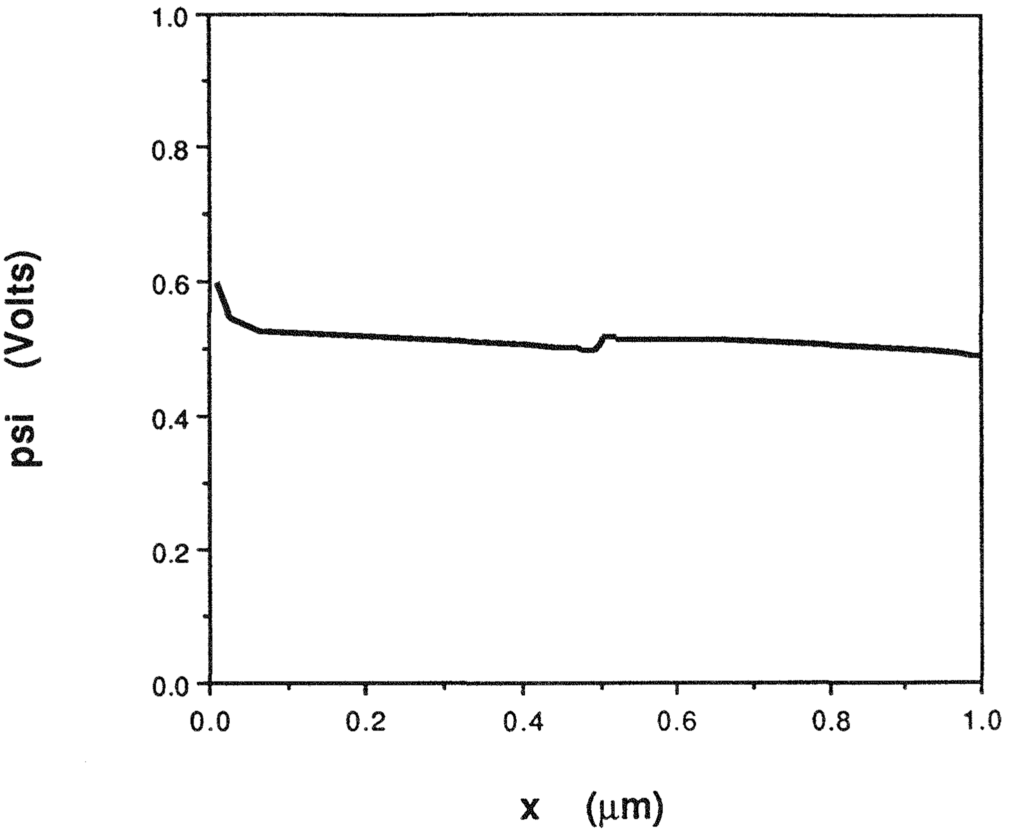
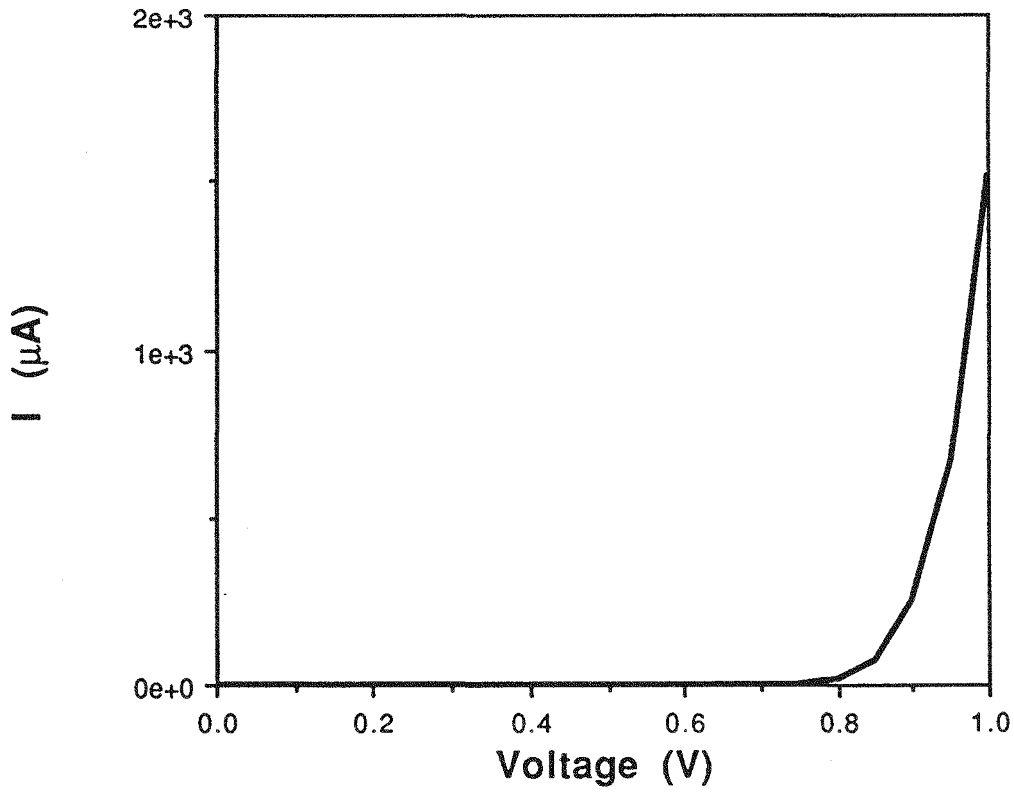


Figure 4. Current For A P-N Junction Diode



REFERENCES

[1] E. M. Buturla, P. E. Cottrell, B. M. Grossman and K. A. Salsburg, *Finite-Element Analysis of Semiconductor Devices: the FIELDAY Program*, IBM Journal of Research and Development, vol. 25 no. 4 (July 1981), pp. 218- 231.

[2] Department of Defense, *Critical Technologies Plan*, 15 March 1989.

[3] J.E. Dennis, Jr. and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Inc., Englewood Cliffs, 1983.

[4] Federal Coordinating Council on Science and Engineering Technology, *Grand Challenges in Science and Technology and Their Implications for High Performance Computing*, Office of Science and Technology Policy, September 8, 1989.

[5] C.A.J. Fletcher, Computational Techniques for Fluid Dynamics, Volume 1, Springer-Verlag, 1988.

[6] G. H. Golub and C. F. van Loan, Matrix Computations, The Johns Hopkins University Press, Baltimore, 1983.

[7] J. A. Greenfield and R. W. Dutton, *Nonplanar VLSI Device Analysis Using the Solution of Poisson's Equation*, IEEE Transactions on Electron Devices, vol. 27 (1980), pp. 1520+.

[8] H.K. Gummel. A Self-Consistent Iterative Scheme for One-Dimensional Steady State Transistor calculations. *IEEE Transactions on Electron Devices*, ED-11:455–465, 1964.

[9] H. Masuda, T. Toyabe, H. Shukuri, K. Kazuyoshi and K. Itoh, *A Full Three-Dimensional Simulation on Alpha-Particle Induced DRAM Soft-Errors*, Proceedings IEDM, December 1985, pp. 496-499.

[10] M. S. Mock, Analysis of Mathematical Models of Semiconductor Devices, Boole Press, Dublin, 1983.

[11] M. R. Pinto, C. S. Rafferty and R. W. Dutton, *PISCES II: Poisson and Continuity Equation Solver (PISCES-II Users Manual)*, Report, Stanford Electronics Laboratories, Stanford University, September 1984.

[12] M. R. Pinto, C. S. Rafferty, H. R. Yeager and R. W. Dutton, *PISCES-IIB Supplementary Report*, Report, Stanford Electronics Laboratories, Stanford University, September 1984.

[13] S. J. Polak, C. Den Heijer, W. H. A. Schilders and P. Markowich, *Semiconductor device modelling from the numerical point of view*, International Journal for Numerical Methods in Engineering, vol. 24 (1987), pp. 763-838.

[14] Siegfried Selberherr. *Analysis and Simulation of Semiconductor Devices.* Springer-Verlag, New York, 1984.

[15] N. Shigyo, M. Konaka and R. Dang, *Three-dimensional simulation of inverse narrow-channel effect*, Electron. Lett., vol. 18 (1982), pp. 274-275.

[16] N. Shigyo, S. Onga and R. Dang, *A three-dimensional MOS device simulator*, in New Problems and New Solutions for Device and Process Modeling, J. J. H. Miller, editor, Boole Press, Dublin, 1985, pp. 138- 149..

[17] N. Shigyo and R. Dang, *Analysis of an anomalous subthreshold current in a fully recessed oxide MOSFET using a three- dimensional device simulation*, IEEE Transactions on Electron Devices, vol. 32 (1985), pp. 441-445.

[18] N. Shigyo and R. Dang, *Three-dimensional device simulation using a mixed process/device simulator*, in Process and Device Modeling, W. L. Engl, editor, North Holland, Amsterdam, 1986, pp. 301-327.

[19] N. Shigyo, S. Fukuda, T. Wada, K. Hieda, T. Hamamoto, H. Watanabe, K. Sunouchi, and H. Tango, *Three-Dimensional Analysis of Subthreshold Swing and Transconductance for Fully Recessed Oxide (Trench) Isolated 1/4-mm-Width MOSFET's*, IEEE Transactions on Electron Devices, vol. 35 no. 7 (July 1988), pp. 945-951.

[20] S.M. Sze, *Physics of Semiconductor Devices* (Wiley, New York, 1981).

[21] T. Toyabe, H. Masuda, Y. Aoki, H. Shukuri and T. Hagiwara, *Three-dimensional device simulator CADDETH with highly convergent matrix solution algorithms*, IEEE Transactions on Electron Devices, vol. 32 no. 10 (October 1985), pp. 2038-2043.

[22] T. Toyabe, Central Research Laboratory, Hitachi Ltd., private communication.

Appendix A. Sample Input Data Set

```
!
!    Input data set for 2D pn junction diode
!
steady version 2.0
   specify the operating environment
      print level = 2
   end specify environment

   define a device
!
!   Supply the name of the doping routine
!
      name of device = "abruptpn"
      doping routine = dope00
      number of contacts = 2
      name of contact 1 = left
      name of contact 2 = right
      name of axis 1 = x
      name of axis 2 = y
!
!   Define the mesh
!
      construct a nonuniform grid for axis 1
         default maximum step       = 0.1
         default maximum step ratio = 2.0
         boundary at 0.00   with step 0.01
         boundary at 0.475  with step 0.0025
         boundary at 0.525  with step 0.0025
         boundary at 1.00   with step 0.01
         maximum step from 0.475 to 0.525 = .0025
      end construct grid

      construct a uniform grid for axis 2
         boundary at 0.00
         boundary at 1.00
         step from 0.0 to 1.00 = 0.25
      end construct grid
   end define device
!
!  Compute the steady state solution at equilibrium
!
   determine the steady state
      accuracy of potential from gummel method = 1.0e-9
      level of fill in iccg factorizations     = 1
      maximum change to contact potentials     = 1.00
      maximum number of gummel steps           = 200
      maximum number of iccg steps             = 500
      sufficient relative residual from iccg   = 1.0e-9
      potential of left  = 0.00
      potential of right = 0.00
   end determine
!
!  Write out a restart file
!
   write the device
      name of file = pndiode
      number of block in file = 1
      remark = "diode example"
      remark = "in equilibrium"
   end write
!
!  Compute the steady state solution at V = 1.0
!
   determine the steady state
      accuracy of potential from gummel method = 1.0e-9
      level of fill in iccg factorizations = 1
      maximum change to contact potentials = 0.05
      maximum number of gummel steps = 500
      maximum number of iccg steps   = 1000
      sufficient relative residual from iccg = 1.0e-9
      potential of left = 1.0
   end determine
!
!  Write out a restart file
!
```

```
   write the device
      name of file = pndiode
      number of block in file = 2
      remark = "left contact at 1.0"
   end write
end steady
```

Appendix B. Sample Doping Routine

```
      subroutine dope00(error, text, contac, coord1, coord2,
     +                  coord3, nd, na, mtrl)
c----------------------------------------------------------------
c     dope00
c
c     Purpose
c       Supply doping profiles for DANCIR
c
c     Parameters
c       error   <----   Logical error flag
c       text     ---->  Integer file number for error messages
c       contac  <----   Integer contact number
c       coord1   ---->  Real*8 coordinate number 1 (usually x)
c       coord2   ---->  Real*8 coordinate number 2 (usually y)
c       coord3   ---->  Real*8 coordinate number 3 (usually z)
c       nd      <----   Real*8 doping level of donors in
c                       units of carrier/cc
c       na      <----   Real*8 doping level of acceptors in
c                       units of carriers/cc
c       mtrl    <----   Real*8 material number
c                       semiconductor = 0.0
c                       insulator     = 1.0
c     Notes
c       For flexibility the doping profiles are read in
c       from a user supplied file
c----------------------------------------------------------------

      character id*9
      integer contac, text
      logical error
      real*8  coord1, coord2, coord3, nd, na, mtrl,

      parameter (id = 'dope00:  ')
      parameter (zinsul = 1.0, zsemic = 0.0)
      parameter (iuser = 88)

      integer i, iregion
      logical first
      data    first /.true./

      common /dopegrd/ x0, xn, xl(5), xr(5), ndu(5), nau(5), nmtrl
      real*8           x0, xn, xl(5), xr(5), ndu(5), nau(5)
      integer          nmtrl

      save first
      save /dopegrd/

      x = coord1
      y = coord2

      if (first) then
          open(unit=iuser,file='user.dat',status='old')
          read(iuser,*) x0, xn
          read(iuser,*) nmtrl
          write(text,8801) x0, xn, nmtrl
          write(text,8802)
          do 5 i=1,nmtrl
             read(iuser,*) xl(i), xr(i), ndu(i), nau(i)
             write(text,8803) xl(i), xr(i), ndu(i), nau(i)
 5        continue
          first = .false.
      endif

      if (x .lt. x0  .or. x .gt. xn) then
          write(text,*) 'dope00:  coordinates out of bounds'
          write(text,*) 'x0, xn, x = ', x0, xn, x
          go to 99999
      endif
c
c     FIND THE REGION THAT (X,Y) IS IN AND SET THE DOPING.
c
      mtrl = zsemic
      nd = 0.0
      na = 0.0

      do 100 i=1,nmtrl
```

```
         if (x .ge. xl(i)  .and. x .le. xr(i)) then
            iregion = i
            goto 101
         endif
 100  continue

      iregion = 0
      write(text,*) 'dope00: x not in any region.'
      write(text,*) '        doping set to zero.'
      goto 99999

 101  continue

      nd = ndu(iregion)
      na = nau(iregion)
c
c     SET THE CONTACTS.
c     1 => left contact
c     2 => right contact
c     0 => no contact
c
      if (x .eq. x0) then
         contac = 1
      else if (x .eq. xn) then
         contac = 2
      else
         contac = 0
      end if

 8801 format(/,' dope00: Doping Profile for 2D Test Problem',/,
     &         ' x0 = ', 1pe12.4,/,
     &         ' xn = ', 1pe12.4,/,
     &         ' number of regions = ', i2)
 8802 format(8x, 'xleft', t20, 'xright', t32, 'Nd', t44, 'Na')
 8803 format(1x, 4(1pe12.4))
99999 continue
      return
      end
```

UNLIMITED RELEASE
INITIAL DISTRIBUTION

Karl Hess
Coordinated Science Laboratory
University of Illinois
1101 West Springfield
Urbana, IL 61801

Jack Higman
Coordinated Science Laboratory
University of Illinois
1101 West Springfield
Urbana, IL 61801

Joe Jerome
Northwestern University
2033 Sheridan Rd.
Evanston, IL 60208

Yannick LeCoz
Electrical Engineering Dept.
Rensselaer Polytechnic Inst.
110 8th St.
Troy, NY 12180-3590

Linda R. Petzold
Lawrence Livermore National Laboratory
Mail Stop, L -316
P.O. Box 808
Livermore, CA 94550

Martin Schultz
Scientific Computing Associates
246 Church St. Suite 307
New Haven, CT 06510

Andrew Sherman
Scientific Computing Associates
246 Church St. Suite 307
New Haven, CT 06510

Homer Walker
Dept. of Mathematics & Statistics
Utah State University
Logan, UTAH 84322-3900

A.B. White
Los Alamos National Laboratory
Mail Stop B-265
Los Alamos, NM 87545

William Williamson, Jr.
Dept. of Physics & Astronomy
University of Toledo
2801 W. Bancroft Street
Toledo, OH 43606

2140	J.B. Woodard
2147	F.W. Sexton
2147	P.S. Winokur
8000	J.C. Crawford Attn: E.E. Ives, 8100 P.L. Mattern, 8300 R.C. Wayne, 8400 P.E. Brewer, 8500
8130	L.A. Hiles
8200	R.J. Detry
8230	W.D. Wilson
8233	R.E. Cline
8233	J.M. Harris
8233	J.F. Grcar (20)
8233	J.C. Meza (20)
8233	S. Keeton
8233	J.F. Lathrop
8233	D.B. McGarrah
8233	R.A. Whiteside
8234	M.H. Pendley
8235	D.L. Crawford
8236	P.W. Dean
8237	W.E. Mason
8240	C.W. Robinson
8245	R.J. Kee
8535	Publications for OSTI (10)
8535	Publication Division/3141 Technical Library Processes Division
3141	Technical Library Processes Division (3)
8524-2	Central Technical Files (3)