Solid-State Electronics, 1975, Vol. 18, pp. 799–804. Pergamon Press. Printed in Great Britain

EQUATION FORMULATION CONSIDERATIONS FOR EFFICIENT NUMERICAL MODELING OF SEMICONDUCTOR PHENOMENA*

LEO C. MCAFEE, JR.

Electron Physics Laboratory, Department of Electrical Computer Engineering, The University of Michigan, Ann Arbor, Michigan 48104, U.S.A.

(*Received* 2 *November* 1973; *in revised form* 9 *November* 1974)

Abstract—This paper highlights the importance of equation formulation and associated programming efficiency with respect to modeling of semiconductor phenomena via numerical methods.

Two numerical modeling efforts are developed in this paper for one-space dimension device modeling. It is shown on a per iteration basis that the ratio of the computational effort between the two methods is a factor of sixteen. The reduction in the computational effort between the two methods was realized by reformulating the mathematical equations and by reconsidering the effect of programming efficiency. A by-product of the reformulation was a factor of two to three improvement in the convergence rate of the nonlinear iteration. With all considerations, the overall improvement in the solution times for one-space dimension device numerical modeling was determined to be a factor of 30–50.

When considering equation formulation alone, the per iteration improvement is a factor of 1·43. Coupled with the two to three convergence rate improvement the overall improvement due to equation formulation was approximately 3·67.

NOTATION

c	impurity doping concentration
$D_p (D_n)$	hole (electron) diffusion constant
FDE's	finite difference equations
h_j	grid size between grid points j and $j+1$
j	used as subscript of variable to denote that variable is in discrete model
$J_p (J_n)$	hole (electron) current density in PDE's
$JP_j (JN_j)$	hole (electron) current density in FDE's
J_T	total current density
N	number of grids in discrete model
n	mobile electron carrier concentration
$N_v (N_c)$	density of states for holes (electrons)
ops	operations count for number of equivalent multiplications in numerical algorithm
p	mobile hole carrier concentration
PDE's	partial differential equations
R	recombination–generation rate
$\mu_p (\mu_n)$	hole (electron) mobility
v	voltage distribution
x	space dimension independent variable
$\xi_p (\xi_n)$	hole (electron) electrochemical potential in PDE's
$\xi P (\xi N)$	hole (electron) electrochemical potential in FDE's

INTRODUCTION

Perspective of the problem

In the last few years several researchers have reported on techniques for solving general semiconductor modeling problems via numerical methods [1–7] and/or network models [8–11]. In today's trend of computer-aided analysis and design, either of these approaches leads to consideration of the optimal formulation of the model such that efficient computer-aided algorithms will result with the usage of these models. However, there has been limited discussion of the trade-offs and choices one makes in deciding which of the available numerical techniques to use for a specific application, and the associated programming considerations.

As in most numerical solution efforts, the semiconductor modeling problem can be separated into two phases: (1) the approximation of the physical phenomena by mathematical equation formulation and (2) the development of a numerical algorithm that is computationally efficient and produces accurate results in predicting the performance of devices. Previous research has demonstrated the importance of the second phase in the success of any numerical solution effort [1–7], and this consideration is widely and generally applied.

A more important and unemphasized consideration is the strong relationship between the success of the numerical solution phase of the problem and the formulation of the mathematical equations which are to be solved. This article highlights the importance and effect of equation formulations in the development of a successful algorithm for numerical modeling of semiconductor phenomena and devices. Proper formulation is especially important for the inclusion of general phenomena such as Fermi–Dirac statistics, the generalized Einstein relations, nonconstant mobilities, etc.

The semiconductor equations

Semiconductor phenomena are described mathematically by the basic semiconductor transport partial differential equations (PDE's) [12]. The one-space dimension mathematical model is conventionally adopted for the study of macroscopic physical phenomena occurring in semiconductor devices. For computer analysis, these equations normally are scaled using the scale factors for the variables as indicated by DeMari [1, 2]. The scaled semiconductor transport PDE's for one-space dimension modeling are:

Poisson's equation

*This research was supported by National Science Foundation Research Initiation Grant No. GK-27886, and The University of Michigan Horace H. Rackham School of Graduate Studies Faculty Research Grant No. FRR-976.

$$\frac{\partial^2 v}{\partial x^2} = -(c - n + p). \tag{1}$$

Hole continuity equation

$$\frac{\partial p}{\partial t} = -\frac{\partial J_p}{\partial x} - R. \tag{2}$$

Electron continuity equation

$$\frac{\partial n}{\partial t} = \frac{\partial J_n}{\partial x} - R. \tag{3}$$

Hole current density

$$J_p = -\mu_p p \frac{\partial v}{\partial x} - D_p \frac{\partial p}{\partial x}. \tag{4}$$

Electron current density

$$J_n = -\mu_n n \frac{\partial v}{\partial x} + D_n \frac{\partial n}{\partial x}. \tag{5}$$

Total current density

$$J_T = J_p + J_n - \frac{\partial^2 v}{\partial t \partial x}. \tag{6}$$

The variables in the preceding equations are defined in the notation section. Equations (1), (2) and (3) are the three main equations to be solved, with the other three expressions serving as auxiliary relations. There is no loss of generality in using the one-space dimension model.

The general semiconductor problem requires use of Fermi–Dirac statistics. To include this general capability, the generalized Einstein relations[13] are used to obtain the diffusion constants as

$$D_p = \mu_p p \frac{\mathrm{d}\xi_p}{\mathrm{d}p} \tag{7}$$

and

$$D_n = \mu_n n \frac{\mathrm{d}\xi_n}{\mathrm{d}n}, \tag{8}$$

where ξ_p and ξ_n are the hole and electron electrochemical potentials, respectively. The expressions for the scaled current densities then become the following:

$$J_p = -\mu_p p \left(\frac{\partial v}{\partial x} + \frac{\mathrm{d}\xi_p}{\mathrm{d}p}\frac{\partial p}{\partial x}\right) \tag{9}$$

and

$$J_n = -\mu_n n \left(\frac{\partial v}{\partial x} - \frac{\mathrm{d}\xi_n}{\mathrm{d}n}\frac{\partial n}{\partial x}\right). \tag{10}$$

The general semiconductor modeling problem also requires including variations in the mobilities as nonlinear functions of electric field, doping level, and temperature.

The expression for either electrochemical potential depends on whether the magnitude of the carrier concentration is in the Boltzmann statistics region, in the Fermi–Dirac degenerate statistics region, or in the transition region in between those two regions[13]. For Boltzmann statistics the scaled electrochemical potential expressions are given by

$$\xi_p = \ln \frac{p}{N_v} \tag{11}$$

and

$$\xi_n = \ln \frac{n}{N_c}. \tag{12}$$

For degenerate statistics the scaled electrochemical potential expressions are given by

$$\xi_p = \left(\frac{3\sqrt{(\pi)p}}{4N_v}\right)^{2/3} \tag{13}$$

and

$$\xi_n = \left(\frac{3\sqrt{(\pi)n}}{4N_c}\right)^{2/3}. \tag{14}$$

In the transition region between the Boltzmann statistics region and the Fermi–Dirac degenerate statistics region, an expression is used that is a weighted function of the expressions in (a) equations (11) and (13) for the hole electrochemical potential and (b) equations (12) and (14) for the electron electrochemical potential.

FORMULATIONS AND ALGORITHMS

This investigation has been concerned with the development of efficient and accurate numerical algorithms for semiconductor modeling. Two such algorithms are presented. The analysis method suggested from this investigation is called Method 1. A second method, called Method 2, is presented for comparison purposes and because its basic equation formulation has been used in several other studies[1, 2, 4–7].

Both Method 1 and Method 2 are developed from one-space dimension semiconductor equations. At various places throughout the discussion, comparisons are made between Method 1 and Method 2 to illustrate the importance of equation formulation and its contribution to the success of analysis algorithms via numerical methods. A brief discussion on programming considerations is included to illustrate the importance of and some items for consideration of programming efficiency whenever developing any numerical algorithm.

Standard numerical techniques were used throughout the two algorithms. The PDE's were approximated by finite difference equations (FDE's) using centered differencing; the nonlinear FDE's were solved using Newton's iteration with a convergence acceleration procedure and the matrix equations were solved efficiently by taking advantage of the block-tridiagonal (sparse matrix) structure of the Jacobian.

Equation formulations

Method 2 formulation. The Method 2 formulation is directly an extension of equations (9) and (10). The formulation is shown for the hole current density, with the understanding that the electron current density can be similarly formulated. Using the relation

$$\frac{d\xi_p}{dx} = \frac{d\xi_p}{dp}\frac{\partial p}{\partial x}, \tag{15}$$

the hole current term for Method 2 from equation (9) becomes

$$J_p = -\mu_p p\left(\frac{\partial v}{\partial x} + \frac{\partial \xi_p}{\partial x}\right). \tag{16}$$

With this formulation the finite difference approximation for the current density is given by

$$JP_{j+1/2} = -\mu_p p_{j+1/2}\left(\frac{v_{j+1} - v_j}{h_j} + \frac{\xi P_{j+1} - \xi P_j}{h_j}\right). \tag{17}$$

From this equation it is readily seen that the electrochemical potential must be calculated for each grid point. Thus depending on whether Boltzmann, transition, or degenerate approximations are valid, equations (11) and/or (12) must be used to calculate ξP.

The calculation of ξP involves a natural logarithm and/or raising a real number to a real power. The execution times for these operations with double precision arithmetic on the IBM 360/67 are given in Table 1. The analysis of the computational effort is made assuming that Boltzmann statistics are being used; it is expected that most problems will require numerical modeling efforts using mostly Boltzmann statistics.

For the program written to implement Method 2, a Fortran external function called ZPN was used to compute either ξP or $d\xi P/dp$. In Newton's Method the value of ξP is used in forming the residual equation vector, and the derivative $d\xi P/dp$ is used in forming the Jacobian. At each internal grid point ZPN was called to compute ξP, ξN, $d\xi P/dp$ and $d\xi N/dn$ four times each for a total of sixteen calls to ZPN.

Inside ZPN, the computation of either ξP or $d\xi P/dp$ used the following operations:

$$\begin{aligned}\text{ops} &= 1 \text{ logarithm} + 5 \text{ multiplications} \\ &\quad + 3 \text{ powers} \\ &\approx 153 \text{ multiplications.}\end{aligned}$$

Because ZPN was called sixteen times per grid point the equivalent multiplications introduced by ZPN alone were 16(153) = 2448.

A count of the multiplication operations indicated that 59 operations were otherwise required. Thus the total operations to set up the Jacobian and residual vector per grid point were

$$\begin{aligned}\text{ops} &= 59 + 2448 \\ &= 2507\end{aligned} \tag{18}$$

and the operation count to set up the Jacobian and residual vector for Boltzmann statistics is given by

$$\text{ops} = 2507(N-1) \tag{19}$$

for a numerical model with N grids (which implies $N-1$ internal grid points).

Table 1. Execution times on IBM 360/67

Operation	Execution time (μsec)	Equivalent multiplications
Multiply ($a \cdot b$)	7·29	1
Logarithm ($\ln x$)	161	22
Exponential (e^x)	138	19
Power (a^b)	306	42

The IBM 360/67 specifies that a multiplication takes 7·29 μsec and an addition 2·5 μsec. An observation for most numerical algorithms, and specifically the semiconductor modeling program, indicates that for each multiplication an addition is also performed. A rule-of-thumb figure that has been used to get an estimate of the computation time is to multiply by 11 μsec for every multiplication operation. The resulting estimate of CPU time is reasonably close to that actually required.

Method 1 formulation. For this method the current density relation in equation (9) is used, that is, the expression used for the current density is

$$J_p = -\mu_p p\left(\frac{\partial v}{\partial x} + \frac{d\xi_p}{dp}\frac{\partial p}{\partial x}\right). \tag{20}$$

The finite difference approximation for this expression is

$$JP_{j+1/2} = -\mu_p p_{j+1/2}\left(\frac{v_{j+1} - v_j}{h_j} + \left.\frac{d\xi_p}{dp}\right|_{p_{j+1/2}} \frac{p_{j+1} - p_j}{h_j}\right). \tag{21}$$

Again using Boltzmann statistics for the comparison, the derivative of the electrochemical potential with respect to the carrier concentration translates into a multiplication in computational effort,

$$\frac{d\xi_p}{dp} = \frac{1}{p}. \tag{22}$$

Through further investigation it was determined that with the Method 1 formulation the formation of the Jacobian required only a further division (or equivalently a multiplication).

With the spirit of efficient computation as motivation it was determined that some programming steps should be undertaken to reduce the computation time. One step was to save all of the intermediate values that would be used more than once in the setup of the Jacobian and the equation vector. With these observations it turned out that the total number of equivalent multiplication operations was reduced to 58 per internal grid point. Thus the Boltzmann statistics computational effort is given by

$$\text{ops} = 58(N-1). \tag{23}$$

Some time was consumed in data initialization, subroutine linkage, and the retrieval and storage of data in memory. It turned out that an estimate of 11·8 μsec for each multiplication operation in Method 1 gave almost exact computation times. For Fermi–Dirac degenerate statistics the computational effort was found to be

$$\text{ops} = 146(N-1). \tag{24}$$

Current density J_p has been used to show the difference of formulation between Methods 1 and 2. Identical operations for formulation and differencing are used with current density J_n.

Programming considerations

Other changes that affected the efficiency of computation were strictly programming changes. When the importance of formulation was realized, a decision was made to conduct a more in-depth study of the effect that programming efficiency would have on the analysis costs. Programming items that were found to make a difference were (1) structure of arrays for storage, (2) usage of subprograms, and (3) storage of recurring values.

Array storage. At each internal jth grid point, there are three equations with the three unknowns p, v and n at the j-1, j and $j+1$ grid points. The Jacobian matrix (for Newton's iteration) that results is a block-tridiagonal matrix with each block submatrix being a 3×3 submatrix.

In the Method 2 algorithm the Jacobian was directly stored as a block–tridiagonal matrix made up of 3×3 submatrices. The arrays were stored as three-subscripted Fortran arrays, with one subscript indexing the submatrix number and the other two subscripts indexing the positions within the 3×3 submatrix.

In the Method 1 algorithm a less direct, but more efficient, method for storing the Jacobian was used. One-subscripted arrays were used and each 3×3 submatrix was stored in nine positions of the one-subscripted arrays. This method requires more programming effort but the trade-off is improved computational efficiency.

Subprograms. Another programming consideration was the number of calls to subprograms. The complete algorithm of Method 2 was written in a functional-block subprogram format. For example, a subprogram was used for the computation of ξ_p and ξ_n, a subprogram was used for the matrix multiplication, etc. The trade-off for this ease of programming was reduced efficiency of computation. In Method 1, subprograms such as the previously mentioned ZPN were eliminated and incorporated in the instruction stream wherever needed. This change along with the one-subscripted array change was measured as reducing the setup time for the residual equation vector and Jacobian matrix by a factor of 7·5.

A more costly effect of functional blocks was in the matrix solution routine. In Method 2 there were subprograms for matrix addition, matrix multiplication, and matrix equation solution. For each interior grid point a call was made to each of the subprograms when performing the overall LU factorization of the Jacobian matrix, and also when performing the backward substitution. The alignment of variable storage and linking between the calling program and called subprograms was repetitively being done and therefore was a cost item. In Method 1, subprograms such as these were removed and incorporated in the instruction stream by explicit coding of the required matrix operations. Also, changing to one-subscripted arrays led to some reduction. Overall, the Method 1: Method 2 measured factor of improvement was 24 for the matrix solution routines.

Recurring values. After incorporating the electrochemical potential expressions in the instruction stream, it became obvious that numerous values that are used in several different computations could be stored and used when needed. This programming consideration was pointed out earlier when discussing the Method 1 formulation.

This realization resulted in a reduction of three multiplicative operations per grid point for p and n each when using Boltzmann statistics. Without saving these recurring values for Boltzmann statistics, the increase would be from 58 to 64 operations, an increase of approximately 10 per cent. When using Fermi–Dirac statistics, the savings of equivalent multiplicative operations is more significant because values are raised to floating-point powers. For either hole or electron concentrations, a savings of approximately 44 multiplicative operations is realized per grid point; thus a savings of 88 operations is realized per grid point when considering both holes and electrons. Therefore, the operations count for Fermi–Dirac statistics would have been 234 operations instead of 146 operations, an increase of approximately 60 per cent.

Per iteration efficiency of computation

As indicated in the discussion on subprograms, the factors of improvement for the matrix setup and matrix solution were 7·5 and 24, respectively. On a per iteration basis, the execution time for Method 2 was measured to be approximately 16-times larger than that for Method 1.

This factor of 16 improvement per iteration is a result of both equation reformulation and programming considerations. It will be shown later that the factor due to equation reformulation is approximately 1·43 and the factor due to programming considerations is approximately 11·2.

Convergence rate

Not only has the speed of computation per iteration been significantly improved, but the convergence rate per excitation condition has been improved. The improvement in the convergence rate certainly depends on the problem being solved; however in all test cases, Method 1 converged in fewer iterations that did Method 2 when both methods used the same initial guess. Method 1 has been found to converge two to three times faster than Method 2. For example, on a problem with a linearly graded impurity distribution with gradient of 10^{20} in the junction region and constant distribution in the bulk region, Method 1 required four iterations to converge; Method 2 required 11 iterations to converge, a convergence rate improvement ratio of 2·75. The model had twenty grids and the initial guess for each method was the same. For other impurity distributions, mainly when the dominant mechanism for conduction approached a diffusion model and when Boltzmann statistics are valid, the convergence rate improvement ratio approached three. Under transient conditions the ratio has remained in the range of two to three. The techniques used for acceleration of convergence were the same for Method 1 as for Method 2. Thus it appears that the formulation of Method 1 has inherent advantages relative to the convergence rate.

Thus without any effort, the convergence rate was

improved by a factor of two to three. Coupling this result with the per iteration improvement factor of 16, the total improvement factor of Method 1 over Method 2 is between 30–50!!!

The convergence rate improvement due solely to the equation formulation can be explained as follows. Looking back to equations (16) and (20) with auxiliary relations in equations (11) and (22), a multiplication (division) operation in Method 1 has replaced a logarithm operation in Method 2 for Boltzmann statistics. Therefore, the degree of nonlinearity with respect to the unknown is less in Method 1 than it is in Method 2. This change is reflected directly in the residual equation vector in Newton's iteration. Since Method 1 has a "nicer" nonlinearity, it should converge faster than Method 2.

The difference in accuracy of the two methods is indeterminate. For cases in which exact solutions are known, such as thermal equilibrium or bulk resistive materials, the results from the two methods have differed a negligibly small amount. Thus comparisons of accuracy have produced no significant conclusions.

Effect of equation formulation

At this point the question arises as to what would be the effect if both Method 2 and Method 1 were programmed identically; that is, what is the factor of improvement due solely to equation formulation. To answer this question, the operation counts for Method 2 are recomputed for the setup of the Jacobian matrix and residual equation vector.

Using the most efficient calculation for Method 2 in the Boltzmann region, the computation of ξP used the following operations

$$\text{ops}|_{\xi P} = 1 \text{ logarithm} + 1 \text{ mult}$$

$$\approx 23 \text{ mults.}$$

The computation of $\mathrm{d}\xi P/\mathrm{d}p$ required

$$\text{ops}|_{\mathrm{d}\xi P/\mathrm{d}p} = 1 \text{ mult.}$$

Combining both the hole and electron calculations the operations count per internal grid point is

$$\text{ops} = 2(23 + 1) = 48 \text{ mults.}$$

With the additional 59 operations per grid point, the total operations count for Method 2 with Boltzmann statistics is

$$\text{ops} = (48 + 59)(N - 1)$$

$$= 107(N - 1). \tag{25}$$

For Fermi–Dirac degenerate statistics

$$\text{ops}|_{\xi P} = 1 \text{ power} + 1 \text{ mult}$$

$$\approx 43 \text{ mults}$$

and

$$\text{ops}|_{\mathrm{d}\xi P/\mathrm{d}\xi} = 1 \text{ power} + 3 \text{ mults}$$

$$\approx 45 \text{ mults.}$$

Therefore the total ops for Fermi–Dirac statistics is

$$\text{ops} = [2(43 + 45) + 59](N - 1)$$

$$= 235(N - 1). \tag{26}$$

Thus the setup time for Method 2 compared to Method 1 is now approximately 87 per cent greater with Boltzmann statistics and approximately 61 per cent greater with Fermi–Dirac degenerate statistics.

Table 2 gives a summary of the two equation formulations and a ratio of improvement factor. The matrix setup and solution times are approximately equal for Method 1 when using Boltzmann statistics. That time has been designated unity; all other times are normalized with respect to that unity time. The per iteration time is the sum of the setup and solution times. Note that the per iteration time ratio of improvement factor is 1·43 for either Boltzmann or Fermi–Dirac statistics. Since the convergence rate improvement has always been in the range of 2–3 for all example problems tested, an average value of 2·5 is used in Table 2. Coupling the average convergence rate improvement with the per iteration improvement, it is clear that the overall improvement, due to the equation formulation of Method 1 as compared to Method 2, is 1·43 (2·5) = 3·67.

CONCLUSIONS

The analysis times for one-space dimension numerical modeling have been improved by a factor of 30–50. This

Table 2. Improvements due to equation reformulation

	Method 2	Method 1	Ratio of improvement factor
Setup time			
Boltzmann	1·85	1	1·85
Fermi–Dirac	4·05	2·52	1·61
Solution time	1	1	1·0
Per iteration time			
Boltzmann	2·85	2·0	1·43
Fermi–Dirac	5·05	3·52	1·43
Convergence rate improvement	2·5	1	2·5
Overall improvement due to equation formulation			3·67 = (2·5)(1·43)

improvement factor was a result of a reformulation of the semiconductor analytic equations and a thorough consideration of programming effects plus a by-product of an improvement in the convergence rate. The improvement due to equation reformulation alone averaged approximately 3·67, a significant result on its own. In this continuing study, similar type improvements resulting from equation reformulation have been realized for two-space dimension modeling.

REFERENCES

1. A. DeMari, *Solid-St. Electron.* **11**, 33 (1968).
2. A. DeMari, *Solid-St. Electron.* **11**, 1021 (1968).
3. D. L. Scharfetter and H. K. Gummel, *IEEE Trans. Electron Devices* **ED-16**, 64 (1969).
4. L. C. McAfee, Ph.D. Thesis, The University of Michigan (1970).
5. G. D. Hachtel, R. C. Joy and J. W. Cooley, *Proc. IEEE* **60**, 86 (1972).
6. J. W. Slotboom, *IEEE Trans. Electron Devices.* **ED-20**, 669 (1973).
7. H. H. Heimeier, *IEEE Trans. Electron Devices.* **ED-20**, 708 (1973).
8. J. G. Linvill, *Models of Transistors and Diodes*, McGraw-Hill, New York (1963).
9. H. N. Ghosh, F. H. DeLaMoneda and N. R. Dono, *Proc. IEEE* **55**, 1897 (1967).
10. K. Kani and T. Ohtsuki, *IEEE Trans. Circuit Theory*, **CT-17**, 26 (1970).
11. L. C. McAfee, *Proc. Tenth Allerton Conf.*, p. 383 (1972).
12. W. Van Roosbroeck, *Bell Syst. Tech. J.* **29**, 560 (1950).
13. E. Spenke, *Electronic Semiconductors*, McGraw-Hill, New York (1958).