Simulation of Critical IC-Fabrication Steps

PETER PICHLER, STUDENT MEMBER, IEEE, WERNER JÜNGLING, STUDENT MEMBER, IEEE, SIEGFRIED SELBERHERR, SENIOR MEMBER, IEEE, EDGAR GUERRERO, AND HANS W. PÖTZL, MEMBER, IEEE

***Abstract*—Due to the advances in device miniaturization it is often necessary to get a better understanding of the physical fabrication processes by applying advanced physical models. Since existing process modeling programs can handle only specific physical quantities, we have developed general purpose solvers for one and two space dimensions which are able to treat an arbitrary number of coupled partial differential equations for physical quantities. In the paper we will show the general formulation of the equations which can be solved. We deal with the user-interface of the programs and the numerical problems one has to face. To demonstrate the capabilities of the programs we will show typical applications.**

I. Introduction

UP-TO-DATE process simulation has to keep pace with the steadily increasing complexity of physical models and the increasing miniaturization of the devices for VLSI-fabrication. In recent years device simulation programs have proven to be important tools to optimize old structures and to design new devices. In the area of increasing miniaturization the simple models for the doping profile incorporated in most of the device modeling programs turn out to be one of the bottlenecks for device simulation. At the same time it becomes more and more clear that commonly used models for the migration of dopants turn out to be inadequate or have to be checked at least.

Since most existing process modeling programs handle only specific physical quantities, the implementation of advanced models often exceeds the capabilities of these programs or requires simplifications which cannot be justified. On the other hand it is necessary to simplify the models where effects are negligible to reduce the required computer resources and the computation time for engineering programs.

Therefore, we have developed general purpose simulation programs in order to master the above-mentioned difficulties and to obtain a powerful tool for the development and verification of physical models. The programs are able to solve an arbitrary number of partial differential equations for physical quantities. The models which describe the physical processes can be fairly general. The goals of our programs are, therefore, scientific applications. We use our codes to check the validity range of existing models and to study physical phenomena which can be described by advanced models.

Manuscript recieved January 25, 1985; revised May 10, 1985. This work was supported by Siemens Research Laboratories, Munich, Germany, and by the Fonds zur Förderung der wissenschaftlichen Forschung under Project S43/10.

The authors are with the Institut für Allgemeine Elektrotechnik und Elektronik, Abteilung für Physikalische Elektronik, 1040-Vienna, Austria.

The physical models are easy to implement and to exchange in order to ease the usage of the programs. To relieve the user from nonphysical problems the mesh in space and time is adapted automatically. Sophisticated mathematical formalisms for the discretization have been adapted from device simulation programs to minimize the truncation errors [6], [17].

In Section II we shall give an outline of the capabilities of our programs and the differential equations which can be solved. In Section III we will treat some of the numerical problems which occur. The design of a mesh and methods for the interpolation between gridlines and for the integration of PDE's are explained in detail. An example of a coupled boron–arsenic diffusion with strongly moving p-n junction shows the advantages of a fully adaptive quasi-uniform mesh. The automatic timestep and order control is demonstrated by examples of coupled boron–arsenic diffusions and the growth and shrinkage of stacking faults.

II. Features

Although our programs can handle systems of coupled partial differential equations with fairly arbitrary structure and general boundary conditions, much emphasis has been laid on satisfying the particular needs of process modeling.

Process modeling deals with the generation and redistribution of dopant concentrations. The redistribution of dopants is usually modeled by parabolic systems of partial differential equations. For those systems initial conditions have to be defined. Additionally, elliptic differential equations such as the Poisson equation have to be solved.

Usually in modern processes the initial doping concentrations are obtained by predeposition and/or ion implantation. Predeposition can be seen as a diffusion process where a high concentration of dopants is supplied at the surface of the wafer. This allows the modeling of the predeposition as a normal diffusion process with Dirichlet conditions at the boundaries. Ion implantation on the other hand is commonly used in VLSI fabrication because of the high efficiency and reproducibility.

Our programs are, therefore, designed to handle all commonly used numerical models for ion-implantation. The parameters for the range statistics, range straggling in vertical and lateral direction, and the higher moments can be derived by using well established theories of the penetration of ions in solids such as the LSS-theory [12] or the theory of Biersack [3], [2]. Well-established fre-

0018-9383/85/1000-1940$01.00 © 1985 IEEE

quency functions are used to describe the impurity concentration profiles. We have implemented Gaussian, joined-half-Gaussian, and Pearson IV distributions [16]. Due to the flexibility of the program it is easy to implement other frequency functions if desired. The lateral shape of the doping profiles is assumed to be an error function for Gaussian distributions in the vertical direction under the assumption of an infinitely steep mask edge according to Furukawa [7]. The profiles for arbitrary mask edges and frequency functions in the vertical direction are obtained by accounting a convolution integral as proposed by Runge [14] and Ryssel [15].

The programs can read and process data from external files to support the use of Monte Carlo programs and the use of measured data. However, data from SIMS, NAA, spreading resistance, or other measurements often have to be smoothed by a mathematical pretreatment to obtain profiles which are three times continuously differentiable. Additionally it is possible to define the values of the quantities by user-defined subroutines. Due to these specifications the programs for the generation of initial doping profiles have shown to be sufficiently flexible for practical use.

The main part of the program package is the code for the integration of the PDE's. The systems of differential equations which can be solved by our codes consist of N partial differential equations where N denotes the number of physical quantities. Each of the differential equations is represented by a very general continuity equation (1) with general current relation (2) and generation-recombination term:

$$\sum_{j=1}^{N} a_{ij} \cdot \frac{\partial C_j}{\partial t} + \text{div } J_i + G_i - R_i = 0 \tag{1}$$

$$J_i = \sum_{j=1}^{N} (d_{ij} \cdot \text{grad } C_j + \mu_{ij} \cdot C_j \cdot \text{grad } \psi). \tag{2}$$

The coefficients a_{ij}, d_{ij}, μ_{ij}, G_i, and R_i can be functions of the simulation time, the simulation temperature, the space variables, and any of the dependent variables C_i. The quantity ψ in the current relation is one of the C_j and denotes the electrical potential. This part of the current relation is mainly used to describe the field-induced current. It could, in principle, be incorporated in the first term of (2) but the above-used notation supports sophisticated discretization schemes for finite difference discretizations [18]. The structure of the programs defining the coefficients of the PDE's is outlined in the appendix.

With the described system of equations we can treat coupled and uncoupled diffusion, dynamic and static cluster models, coupled Poisson equations, etc.

The general formulation of the boundary conditions (3) allows Dirichlet, Neumann, and mixed-boundary conditions which correspond physically to predeposition, inert diffusion, and an oxidizing ambient where, for instance, point defects are emitted at the surface:

$$\sum_{j=1}^{N} \xi_{ij} \cdot (J_j \cdot n) + F_i = 0. \tag{3}$$

The coefficients ξ_{ij} can be functions of the simulation time and the space variables, and F_i can be functions of the quantities C_j, time, and the space variables. The quantity n denotes the unit vector normal to the boundary.

It should be mentioned at this point that due to the very general formulation of the PDE's and the boundary conditions it is possible to specify systems which are improperly posed and, therefore, cannot be solved.

In the present state of our program we restrict ourselves to fixed boundaries. We exclude moving boundaries at present. Nevertheless because of the flexibility regarding the boundary conditions it is possible to simulate some effects of oxidation such as interstitial and vacancy generation at the surface.

III. Examples and Numerical Aspects

The goal of this section is to present the mathematical formalisms which have been installed in our programs. Our codes are written in a modular form so that improvements in the applied numerical mathematics can be carried out without severe problems in the program architecture. In the following subsections we will outline the methods of the numerical methods used for the solution of the PDE's, the grid adaption in the one- as well as in the two-dimensional code, the methods used for the interpolation between gridlines, and the methods used for the transient integration of the PDE's. Examples of coupled boron arsenic diffusions are used to demonstrate the adaption of the spatial grid in the one- and two-dimensional code as well as the design of the transient mesh in the one-dimensional code. An example dealing with stacking fault growth and shrinkage shows the capabilities of the used formalisms.

3.1. Numerical Solution of the PDE's

The main goal of our programs is the solution of nonlinear PDE's. The PDE's are linearized by applying Newton's method and then discretized by using finite difference approaches. The spatial discretization formalisms used for the discretization of the continuity equations are similar to those used in device modeling [18]. The usage of an exponential fitting factor for the continuity equations with field terms improves especially the convergence property of Newton's method. The solution of the linear system is performed using LU-decomposition in the one-dimensional code and point-successive over-relaxation method (SOR) or Gaussian elimination in the two-dimensional code.

The maximum of the electric fields caused by a doping concentration C is assumed to occur in the vicinity of p-n junctions. It can be estimated under the assumption of quasi-neutrality and by applying Boltzmann statistics by

$$\frac{\partial \psi}{\partial x} \sim T^{-1/2} \cdot \exp\left(\frac{E_g}{2kT}\right) \cdot \frac{\partial C}{\partial x} \tag{4}$$

where ψ denotes the electric potential, T the process temperature, E_g the energy gap, and k Boltzmann's constant.

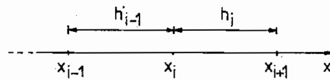

Fig. 1. The adopted nomenclature for finite differences.

The carrier concentration and potential profiles in process modeling are therefore smoother than the profiles of potential, electron, and hole concentrations in device modeling due to the much higher temperatures. This means in practice that Newton's method converges qualitatively better and nearly without overshoots for most of the diffusion problems. Nevertheless we have implemented a Deufrlhard-like damping [5] which guarantees that the norm of the residual in Newton's method is decreased in each iteration step.

3.2. Design of a Spatial Grid

One of the main problems in numerical simulation is the design of a good mesh. The heuristic mesh design rules used in most of the existing programs prevent many authors from giving explicit information.

3.2.1. One-Dimensional Mesh Adaption: In the one-dimensional code we have implemented a fully adaptive quasi-uniform mesh. The characteristic of this mesh type is that the ratio between two adjacent grid distances h_i and h_{i-1} is close to unity. The mathematical formulation is given by (5):

$$h_{i-1} = h_i \cdot (1 + O(h_i)). \tag{5}$$

A reason for the use of a quasi-uniform mesh is its property to reduce the truncation errors caused by the discretization of the differential operators with the square of the mesh spacing. This property is shown in the following. For these considerations let us assume a one-dimensional finite difference discretization scheme as shown in Fig. 1. For the discretization of an arbitrary differential equation div U = div A grad C we replace the differential operators by difference operators, cf. [16]. Under the usual assumption that U is three times continuously differentiable we obtain

$$\left.\frac{\partial U}{\partial x}\right|_i = \frac{U_{i+1/2} - U_{i-1/2}}{\frac{h_i + h_{i-1}}{2}} + \frac{h_i - h_{i-1}}{2} \left.\frac{\partial^2 U}{\partial x^2}\right|_i + O\left(\frac{h_i^3 + h_{i-1}^3}{h_i + h_{i-1}}\right). \tag{6}$$

We assume that U is constant within each interval. Therefore, we obtain

$$U_{i+1/2} = A_{i+1/2} \cdot \left(\frac{C_{i+1} - C_i}{h_i} + O(h^2) \cdot \left.\frac{\partial^3 C}{\partial x^3}\right|_{i+1/2}\right). \tag{7}$$

Since (7) holds for a quasi-uniform mesh we can see that the local truncation error is on the order of the square of the mesh spacing if the ratio between two adjacent mesh spacings minus unity is small compared to unity.

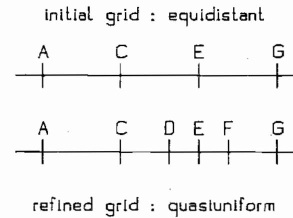

Fig. 2. Strategy for the quasi-uniform refinement of a grid.

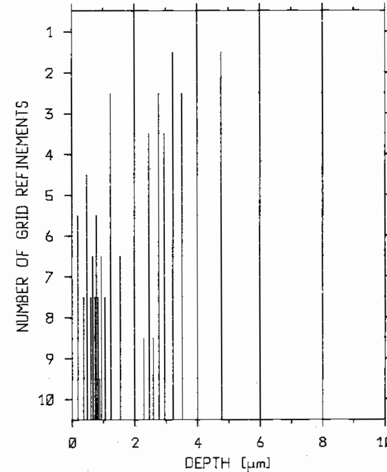

Fig. 3. Construction of a quasi-uniform mesh.

Theoretical investigations suggest some ideas as to how to create a spatial grid and how to reduce the numerical errors during the simulation [13]. The implemented ideas which follow represent an amount of numerical effort which leads to satisfying spatial grids and acceptable computation time. The proposed mathematical formalisms are independent of the physical model under consideration and work without user interaction.

Considerations of error minimization lead automatically to modifications of the spatial grid and suggest the use of a dynamic instead of a rigid grid. Quasi-uniformity seems to be a good criterion for a spatial grid. A simple consideration leads to the following grid strategy which is also demonstrated in Fig. 2. Imagine an equidistant mesh $A - C - E - G$. If we intersect the interval $C - E$ by the midpoint D, our new mesh is not quasiuniform since the ratio of distance $C - D$ to $A - C$ is 1:2. Therefore, we have to move D towards E. Now we have three distances $A - C$, $C - D$, and $D - E$, with the relation $A - C = C - D + D - E$. The optimal quasi-uniformity is obtained when $A - C : C - D = C - D : D - E$. This leads to a reduction factor of $0.5 + \text{sqrt}(1.25) = 1.618 \cdots$ — well known as *sectio aurea*. Fig. 3 shows the first ten steps of a grid refinement in the half of the simulation domain. The quantity to be discretized is an implantation profile with its maximum at about 1.2 μm. The initial grid is an equidistant mesh consisting of 11 points within 0–20 μm. Generally larger grid domains have to be refined to obtain the quasi-uniformity of the mesh if a certain interval shall be divided. In the first example two points (D and F) have to be inserted which lie symmetrically to E. A grid which is built up this way turns out to have many practical properties.

- Quasi-uniformity guarantees a quadratic decrease of the trunction errors with the local mesh spacing.
- It permits easily the construction of meshes which include domains of fine resolution and very coarse mesh regions close together.
- The strict recipe how to introduce new grid lines guarantees that two distributions which need a fine resolution at the same depth will have many gridlines in common. During grid updates this property assures that interpolation is necessary only at a few gridlines whereas the majority of values for the new grid can be taken unchanged from the old grid. Since it minimizes the number of interpolations at introduced meshlines during grid updates it increases the accuracy of the simulation. (Simple and therefore fast interpolation always introduces errors into the simulation.)
- The organization of the refinement of a quasi-uniform mesh is simple. (On the other hand it is difficult to coarsen a quasi-uniform grid and maintain quasi-uniformity, cf. [10].)

The way the spatial mesh is created, refined or updated depends only on the distribution of the quantities and the way the discretization error is computed. Our main concern is the minimization of the discretization error. In the one-dimensional program we derive the estimate of the error from the fact that our discretization scheme is exact if the distribution of the quantities can be described by a polynomial of not more than second order. If the distribution is described by a second-order polynomial then the values of the quantities are identical to the values obtained by a second-order polynomial as a function of depth. The deviations of the values of the quantities from an optimally fitted second-order polynomial is, therefore, a measure for the discretization error. An additional criterion, which may be turned on or off by the user, controls the maximum difference or ratio between adjacent quantities and inserts points until the difference/ratio is below a user-specified value. The grid refinement is performed in such a way that the largest discretization error is attacked. This simple strategy guarantees that the grid is optimal even if there are too few gridpoints available to obtain the accurcy specified by the user. The user may set a minimal grid distance to avoid too fine a spatial resolution. This option is important if the quantities are discontinuous (simulations of predepositions, measured values as initial solutions, etc.).

Example 1 is the simulation of a coupled boron–arsenic diffusion.

A coupled B–As diffusion can be described by (8)–(10):

$$\text{div grad } \psi = \frac{q}{\epsilon} \cdot \left(2 \cdot n_i \cdot \sinh\left(\frac{\psi}{U_t}\right) + C_B - C_{\text{As}}\right) \tag{8}$$

$$\frac{\partial C_B}{\partial t} = \text{div}\left(D_B\left(\text{grad } C_B - \frac{C_B}{U_t} \cdot \text{grad } \psi\right)\right) \tag{9}$$

$$\frac{\partial C_{\text{As}}}{\partial t} = \text{div}\left(D_{\text{As}}\left(\text{grad } C_{\text{As}} + \frac{C_{\text{As}}}{U_t} \cdot \text{grad } \psi\right)\right). \tag{10}$$

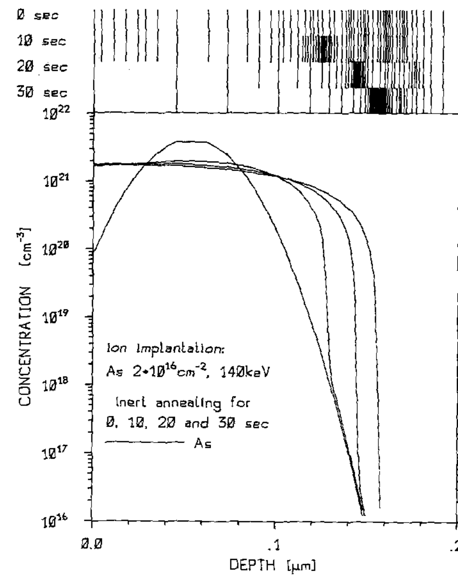

Fig. 4. Spreading of the arsenic profile at the very beginning of the diffusion.

This system of equations consists of a Poisson equation (8) and two continuity equations for the dopants (9) and (10). In order to improve the transparency of the results and to obtain fairly high electric fields we neglect clustering, precipitation, and degenerate statistics.

The initial profile has been obtained by the simulation of two ion implantations of arsenic with a dose of 2×10^{16} cm^{-2} and an energy of 140 keV and boron with a dose of 3×10^{15} cm^{-2} and an energy of 400 keV. The background impurities have been assumed to be 10^{13} cm^{-3} for both dopants. The diffusivities are modeled by using superimposed Arrhenius expressions assuming diffusion via neutral and singularly charged vacancies. The simulation domain is 0–20 μm in space and 0–2 h in time.

Fig. 4 shows the spreading of the arsenic profiles at the very beginning and the corresponding spatial grids. Fig. 5 shows four snapshots of the distributions of boron and arsenic and the related spatial grids. This figure shows that up to about 1000 s the diffusion is uncoupled (except the field enhancement by the intrinsic charge of boron and arsenic). After that period a strong interaction between boron and arsenic via the electric field takes place. In Example 1 the accuracy has been checked down to 10^{10} cm^{-3}. Therefore the transition from areas with steep arsenic and boron gradients to the constant background concentration is resolved, too. Figs. 6 and 7 show the complete spatial and transient grids for the simulation of Example 1 in a linear and logarithmic time scale. Continuous lines represent existing gridlines; beginning or terminating lines indicate grid modifications. Three domains of fine resolution can be seen. The most impressive one represents the steep gradient of the arsenic profile which moves into the bulk. As soon as an interaction with the boron atoms takes place (after approximately 1000 s), this dark domain is identical to the moving p-n junction and becomes broader. The other two domains are related to the spreading of the boron profile. In the beginning the

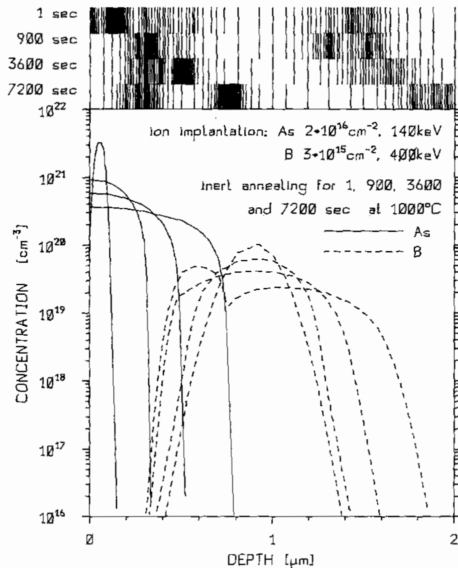

Fig. 5. Boron and arsenic concentrations at 4 different times during a 2-h annealing step at 1000°C.

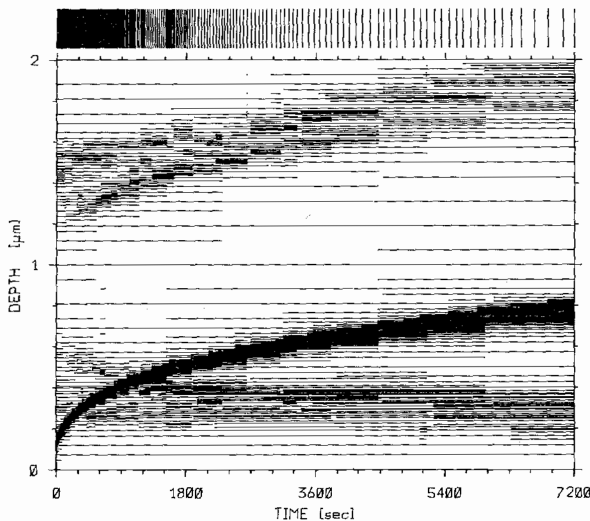

Fig. 6. Grid modifications during the simulation of Example 1 (linear timescale).

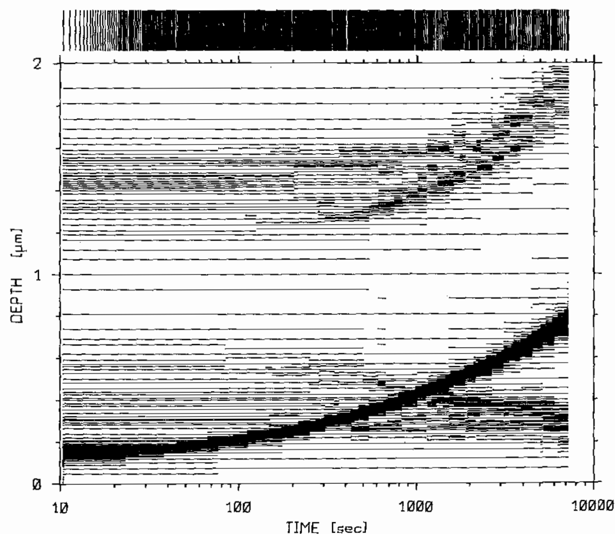

Fig. 7. Grid modifications during the simulation of Example 1 (logarithmic timescale).

spreading is symmetrical. After the beginning of the interaction with arsenic, the diffusivity of boron in the n-doped regime is strongly reduced (suppressed V^+ concentration) compared to the diffusivity in the other domain. Therefore, the spreading of boron towards the surface is slower than the spreading towards the bulk. This effect can be clearly seen by inspection of the dynamic grid. The grid lines which resolve the steep arsenic gradient show an interesting behavior. Towards the bulk we find a continuous motion of gridlines whereas there are steps towards the surface. This phenomenon is related to the way the dynamic grid is established and will be discussed further.

As mentioned earlier, a dynamic grid is easily refined but it is difficult to coarsen. Furthermore, the deletion of a mesh line requires the computation of the discretization error before and after the deletion of the meshline. If the discretization error after the removal is too large the grid line has to be inserted again and the gridline with the second smallest discretization error has to be removed. The difficulty of describing these activities may give the reader a slight impression of how difficult, time consuming, and laborious it is to put it into a programming language. Therefore, we decided to create a complete new mesh after a certain number of additional gridpoints have been inserted or after a certain number of time steps have passed. The problem of interpolation is reduced since the new grid contains many points from the old mesh. In practice the old and the new mesh have 85–95 percent of the points in common. The remaining 5–15 percent of the new grid points require interpolation as well as the grid lines inserted during the refinement of the grid.

3.2.2. Two-Dimensional Grid Adaption: The adaption of a spatial grid in two dimensions has to satisfy numerous demands which are mostly qualitative. The test of new models requires that the computational accuracy is sufficiently high. This is usually checked by examining the conservation of the dosis during time integration. Other important goals are the determination of the position of the p-n junction in the bulk and the fine resolution of model-immanent kinks and spikes in the impurity profiles. On the other hand no special information such as space charge in device modeling can be used to get information about necessary insertions and deletions. Since the maximum number of gridlines is usually restricted to a relatively small number of lines we have to restrict ourselves to the equidistribution of characteristic functions computed by the grid criteria.

The first criterion which is used in the two-dimensional code is the equidistribution of the second derivatives of the profiles weighted with the discretization area. This criterion inserts gridlines mainly in the vicinity of the maxima of the profiles and is, therefore, indirectly responsible for the conservation of the dose. A second criterion checks the maximum ratio of the quantity values at two adjacent meshlines and inserts lines mainly at the slopes of the profiles to resolve p-n junctions properly.

The example which we present to demonstrate the adaption of the spatial grid is an extrinsic boron–arsenic dif-

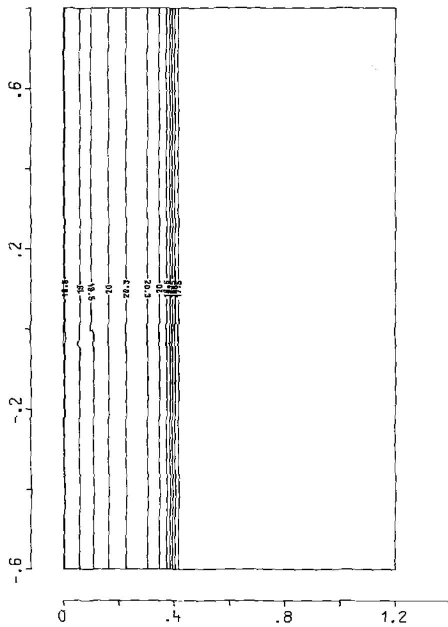

Fig. 8. Isoconcentration plot of the boron profile at the beginning of the coupled diffusion.

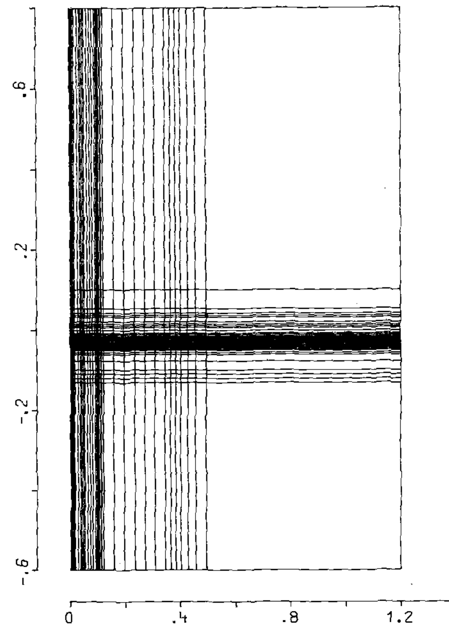

Fig. 10. Two-dimensional mesh at the beginning of the simulation.

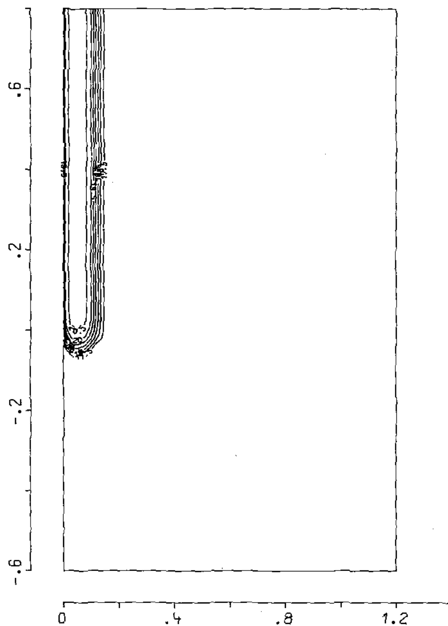

Fig. 9. Isoconcentration plot of the arsenic profile at the beginning of the coupled diffusion.

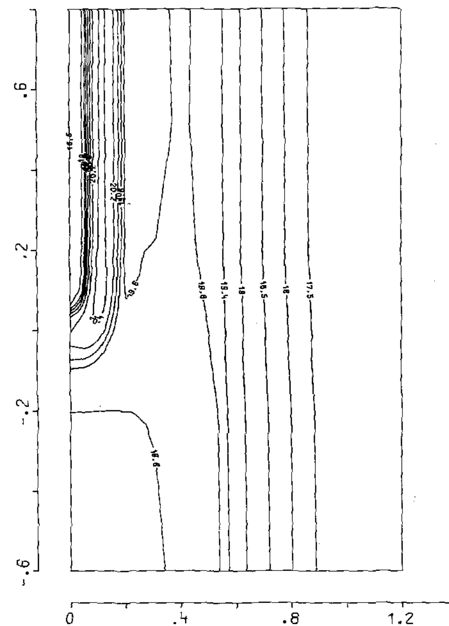

Fig. 11. Isoconcentration plot of the boron profile after 60-min coupled diffusion at 950°C.

fusion. The differential equations are given by (8)–(10). The initial doping conditions have been obtained by ion implantation of arsenic and boron with 4×10^{15} cm^{-2} at 90 and 80 keV, respectively. The arsenic has been implanted through a mask with an infinitely steep edge to simulate a MOS transistor process. Figs. 8 and 9 show isoline plots of the profiles of boron and arsenic at the very beginning of the diffusion process. Fig. 10 shows the spatial mesh at this snapshot. The lines are, as can be seen, extremely tight in the regions with high gradients of the arsenic profile and relatively coarse at the flanks of the boron profile since the arsenic profile is much steeper. Figs. 11 and 12 show the boron and arsenic concentrations after a 3600-s inert diffusion at 950°C.

The arsenic profile shows a distinct, J-shaped maximum. This maximum is forced by the divergence of the electric field in the vicinity of the p-n junction. Analogously the boron profile forms a maximum of the same shape and value at the same position. The migration of boron at the surface in regions of high arsenic concentrations is strongly retarded due to the reduced diffusion coefficient whereas the boron profile spreads unaffectedly elsewhere near the surface.

The mesh (Fig. 13) shows accumulations of gridlines in the vicinity of the maximum and in the regions where the boron profile crosses from the J-shaped maximum to the region in the bulk where the arsenic accumulation has no direct influence.

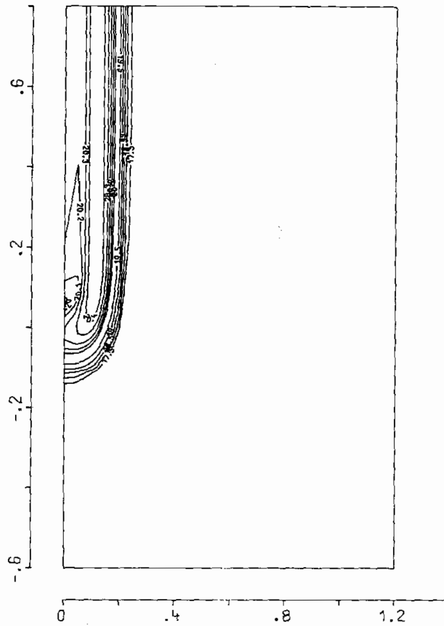

Fig. 12. Isoconcentration plot of the arsenic profile after 60-min coupled diffusion at 950°C.

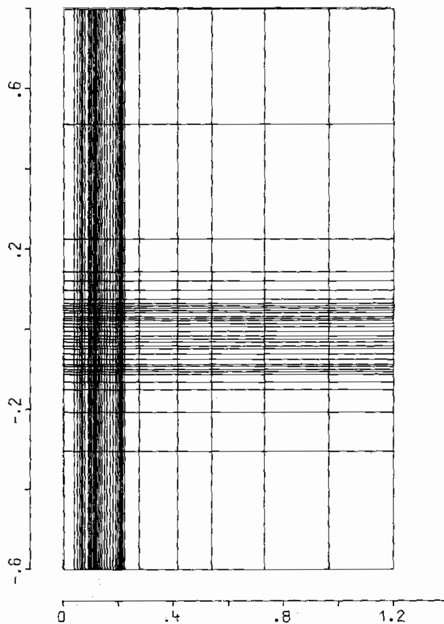

Fig. 13. Two-dimensional mesh after 60-min coupled diffusion.

3.3. Interpolation between Gridpoints

Interpolation becomes necessary for grid refinements and grid updates. It can be a source of unpredictable errors during the simulation and must therefore be treated carefully.

The simplest approach is to interpolate linear quantities bilinearly and logarithmic quantities bilogarithmically. A coarse definition of linear quantities is: they have a dynamic range which is less than the dynamic range of the computer accuracy; they may contain zero and change sign. Typical examples of linear quantities are the electrostatic potential, stacking fault length, or radii of precipitates. Logarithmic quantities have a large dynamic range of interest, they must not be zero or change sign. Typical examples are carrier and dopant densities, point defect concentrations, etc. This suffices for the quasi-uniform mesh in the one-dimensional code if the mesh is fine enough. An obvious advantage of this method is the fast performance which is important when backward information has to be interpolated too. In the two-dimensional code, where the grid is coarser a better interpolation must be used such as spline interpolation or Akima's interpolation method [1]. Since these methods tend to overshoot it is useful to improve the above-mentioned methods by monotonicity considerations [9]. For logarithmic quantities it is useful to interpolate the logarithm of the values of the profiles.

The best results for the interpolation in meshes of consecutive timesteps have been obtained by additionally interpolating with the differential equations themselves. For this method all mesh points of the unrefined mesh are treated like Dirichlet points. The PDE's are solved for all inserted points and the recent timesteps which are stored for backward difference formulas used for the integration of the PDE's.

This method ensures that the local truncation error of the interpolation is of the same order as the truncation error of the values on the unrefined mesh.

3.4. Transient Integration

For the transient integration of the PDE's backward differentiation formulas are used. In the two-dimensional program, backward Euler discretization with time-step prediction and correction is used.

From the mathematical point of view backward Euler is only correct if the transient behavior of the quantity is a function of first order. Backward differentiation formulas (= BDF) of higher order have already been studied extensively and have been applied to circuit simulations. Reference [4] gives a nice presentation of the basic ideas and proposes some suggestions for error, step width, and order control. Certainly some changes are necessary since process simulation reveals problems which are unknown in circuit simulation, e.g., high dynamic range of quantities, interpolation, spatial operators, moving grids, etc. The use of BDF of up to sixth order has many advantages.

- BDF guarantee an exact integration of functions of up to sixth order.
- The integration method shows "$A(\alpha)$-stability," cf. [8].
- It enables a predictor–corrector scheme which speeds up the Newton iterations by better initial guesses and gives a realistic method of error control.

Drastic changes have been necessary concerning the error control and the determination of the step width. We restrict ourselves to the control of the relative error and have avoided all attempts to obtain an absolute error control (as is proposed in [4]). The deviation between predicted and corrected values is a measure of the error. It determines the increase or reduction of the next time step

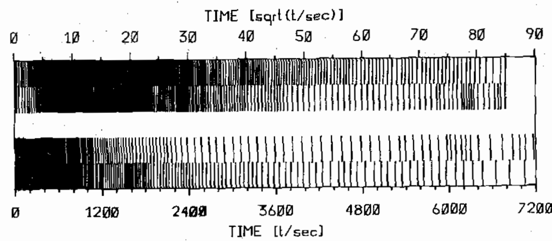

Fig. 14. Transient grids for Example 1 and 2 in different timescales.

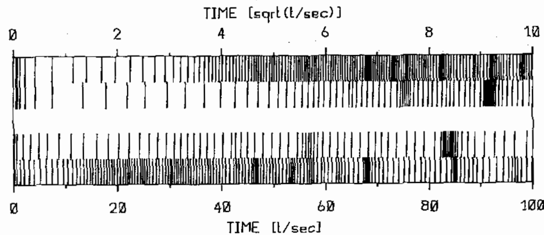

Fig. 15. Transient grids for Example 1 and 2 in different timescales for the begin of the simulation.

compared to the current one as well as the choice of the order for the transient integration. The step and order control is a worst-case estimation since the most critical component determines the step width and the order. In our implementation every quantity is checked at every depth to fulfil the error criterion. The step width and the order of integration is updated after every time step. Interpolation causes no program controlled change of the order and step width, nevertheless we observe the phenomenon that grid updates which are accompanied by the insertion of many new gridlines lead to decreased step width. This phenomenon can be clearly seen in the transient grids of this example.

In order to investigate the influence of field coupling on the transient integration we have simulated Example 2, which is identical to Example 1 in Section 3.2.1 except that all field induced currents have been set equal zero. Figs. 14 and 15 show transient grids for Examples 1 and 2 in a linear scale and a scale proportional to sqrt(t). The outer parts are the transient grids for Example 1 the inner ones show the grid for Example 2. The first time step is made very small (1 ms in our examples). For both examples the step width turns out to be too small and is steadily increased. After about 15 s a physical effect appears which needs finer transient discretization. This effect can be clearly seen in Fig. 4 and turns out to be the diffusion of arsenic with a very steep slope in its concentration. Observing a constant depth we see that the arsenic concentrations increase during a very short time from 10^{13} cm^{-3} to 10^{21} cm^{-3}. Since Gear's method tries to predict this behavior by polynomials it takes several steps to pass through this large dynamic range. The diffusivity of arsenic is strongly concentration dependent, enhanced in domains with high arsenic concentrations and decreases, therefore, with time in Examples 1 and 2. This explains the fine transient grid within 25–900 s and coarser time steps afterwards. Since sqrt$(D \cdot t)$ is a measure for the speed of a diffusion, the transient grid should be an equidistant one if plotted over sqrt(t). This is approximately the case as seen in the upper two grids in Fig. 14 for the time from 1600 to 7200 s. It is interesting that the transition from the n-i-p to the n-p structure is not reflected in the transient grid but only in the spatial grid. (The black strip in Figs. 6 and 7 becomes significantly broader after 1000 s of simulation time.) This indicates that the coupled diffusion is no problem for a transient integration using BDF and a proper spatial grid. To confirm this hypothesis we have simulated Example 2 and plotted the transient mesh (inner grids in Figs. 14 and 15). The figures reveal that there is no qualitative difference between the grids of Ex. 1 and Ex. 2. The step width in Example 2 is larger and effects are delayed. The delay and the wider grid is affected by a factor between 1 and 2 and can easily be explained by the lack of field enhancement in Example 2. Since there is no difference in the grid when arsenic starts to interact with boron we find a confirmation of our assumption.

The accuracy criteria in Examples 1 and 2 have been made very small, i.e., they are far more restrictive than usual. Some quantitative data for Examples 1 and 2 are as follows.

Example 1 took 496 time steps, Example 2 only 311. This ratio reflects the enhanced diffusion by the field in Example 1 which should be between 1 and 2. The average number of grid points during the simulation in Example 1 is 161, the minimum number after a grid update has been 154, the maximum number before a grid update has been 186. The total number of different grid points which occurred during the whole simulation is about 1550 points. This number of gridlines would have been necessary to perform the same simulation using a rigid quasi-uniform mesh (the user would have to know in advance where to place the gridlines). The comparison with a rigid equidistant grid would lead to some ten of thousands of gridlines. The ratio between the largest and smallest spatial grid is more than 5000. The initial grid for Examples 1 and 2 consists of 41 points equidistantly distributed in the simulation domain from 0 to 20 μm.

These numbers show that the dynamic grid speeds up the simulation as well as it increases the accuracy. The savings of using only a tenth of the gridlines of a rigid grid is far larger than the time invested in the computation of the discretization errors and the grid modifications.

The last example deals with the simulation of point defects and the stacking fault growth. The simplest models which describe the kinetics of these phenomena are given by (11)–(13), cf. [11]:

$$\frac{\partial C_I}{\partial t} = \frac{\partial}{\partial x}\left(D_I \frac{\partial C_I}{\partial x}\right) - k_B(C_I C_V - C_I^{eq} C_V^{eq}) - k_1 r \frac{\partial r}{\partial t} \tag{11}$$

$$\frac{\partial C_V}{\partial t} = \frac{\partial}{\partial x}\left(D_V \frac{\partial C_V}{\partial x}\right) - k_B(C_I C_V - C_I^{eq} C_V^{eq}) \tag{12}$$

$$\frac{\partial r}{\partial t} = k_2(C_I - C_I^{eq}) \cdot \exp\left(\frac{\gamma}{kT}\right). \tag{13}$$

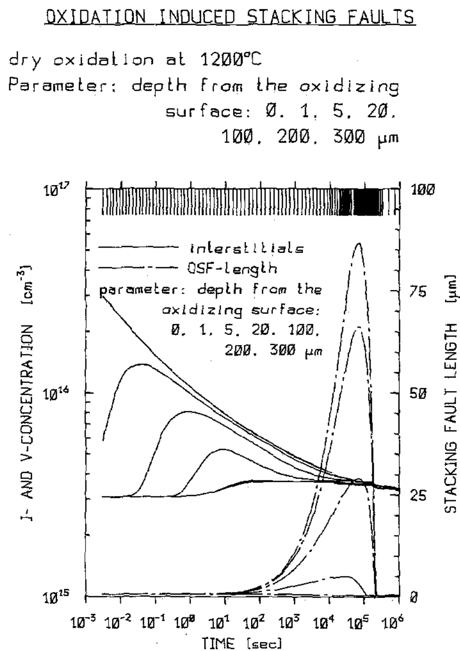

Fig. 16. Transient behavior of the interstitial concentration and the stacking fault length.

"I" denotes interstitials, "V" vacancies, and "r" the radius of the stacking faults (OSF). For OSF shrinkage this approach is only valid when the length of the OSF is larger than the change δr between two timesteps. If the shrinkage of OSF's is simulated, the models compute "negative stacking fault lengths." We have implemented a radius dependent growth/shrinkage of the OSF's when the radius is smaller than a critical value. $\eta(r)$ is defined as

$$\eta(r) = \begin{cases} 0, & \text{for } r < 0 \\ 3 \cdot (r/r_c)^2 - 2 \cdot (r/r_c)^3, & \text{for } 0 < r < r_c \\ 1, & \text{for } r > r_c. \end{cases} \quad (13)$$

This improved model states that OSF's do not shrink beyond a critical length and remain stable as long as there is no supersaturation beyond the critical value for the OSF's. This improved model avoids "negative stacking fault lengths" but has become a challenging example for the transient step and error control. The time step control must decrease the step width whenever a radius is close to the critical radius to accurately simulate the transition between the radius dependent and the radius independent domains. The step δt must be small enough so that δr is smaller than r_c. To check this performance we have simulated the point defect kinetics of a dry oxidation at 1200°C for a sufficiently long time to observe growth as well as shrinkage of the stacking faults. Fig. 16 shows the time dependence of the interstitial concentration and the stacking fault length. The figure shows that supersaturation of interstitials occurs at the beginning of the oxidation and close to the surface. (Bulk recombination reduces the effective range of the surface generation of interstitials.) Therefore, the OSF growth is also constrained to the surface. As long as the interstitial concentration is beyond the critical value, we find growth of the OSF. After about 10^5 s, OSF's are sources of interstitials which tend to keep the interstitial concentration at the critical value and become smaller. Since the rate of shrinkage is independent of the radius of the OSF's, the unmodified model would lead straightforwardly to "negative stacking fault length." This would not only cause incorrect results in the length of the OSF but would also falsify the point defect kinetics since the negative OSF's would work as an ideal source of interstials. On the top of Fig. 16 we find the transient grid for the simulation of the improved model. It reveals that our grid control has detected the change in the shrinkage rate and reduced the step width. Fig. 17 shows the critical period (the disappearance of the stacking faults close to the surface) and the corresponding transient grid. We see that the step width gets smaller the larger the shrinkage rate of the OSF becomes. After all OSF's have disappeared the grid control speeds up automatically to a step width which is some orders of magnitude larger than the smallest one during the simulation. Simulations with different critical radii r_c reveal that the time steps get smaller as the critical radius goes down. In all our simulations the time steps have been small enough to avoid the "negative stacking fault radii."

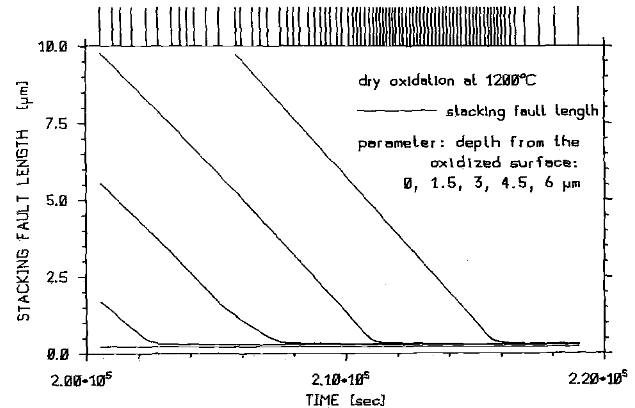

Fig. 17. Shrinkage of the stacking faults during the critical time domain.

The difficulty in this example is the fact that the model is similar to a "switch" in circuit simulation if the critical radius becomes very small. Our example is not an abrupt switch since the critical OSF length is obtained earlier in the bulk than at the surface and the shrinkage rate in the bulk is less critical than at the surface. Nevertheless too careless error criteria lead to a "negative OSF length."

We accept time steps if the error is smaller than a maximum error which is larger than the typical error. The bandwidth between these error boundaries should be large. If a computed error is larger than the maximum error, the time step is rejected. By rejecting time steps we destroy the harmony of the transient grid which may cause unexpected small time steps or strongly varying step widths. This phenomenon occurs especially if the order of integration is high (5 or 6).

Other criteria to reject a time step may be:

- the insertion of too many new grid lines,

- the divergence of a Newton iteration, or
- any of the physical quantities is out of range.

The first criterion is not very critical. Nevertheless some investigations will be necessary to find out how to treat the problem. The other cases are essential. Too large time steps may lead to an initial guess which is out of the convergence radius of the Newton iteration, or lead to a nonphysical prediction (negative OSF length, or negative concentrations). In this case a drastic reduction of the time step (factor 0.5) is essential for the continuation of the simulation. The grid control aborts with a fatal error if the Newton iteration diverges with the minimum time step width (in this case an erroneous Jacobian is mostly the reason for the fatal termination of the simulation).

The third effect is often caused by the linearization of the differential equations. A strict truncation of the overshooting is necessary to avoid unphysical calculations and illegal numerical operations (e.g., negative real to the real power).

IV. Conclusion

In this paper we have presented general purpose programs for the simulation of the migration of dopants in one- and two-space dimensions. The programs support easy implementations of advanced physical models and relieve the user of unnecessary nonphysical tasks. The structure of the differential equations to be solved is explained and the implementation of physical models is shown. The numerical questions which one has to face have been outlined. The advantages of a quasi-uniform mesh and of BD formulas for the integration of the PDE's have been demonstrated. Some critical examples have demonstrated the capabilities of the numerical processors and the automatic grid adaption in one- and two-space dimensions.

Appendix
User Interface

This section is to explain the structure of our codes. The programs consist of modules for the generation of initial profiles and for the integration of PDE's. We shall show which subroutines, parameters, and external files have to be provided by the user.

Initial profiles can be obtained either by reading a profile from an external file, by modeling ion implantation or by executing a user-defined subroutine. For ion implantation the user has to provide a program defining the profile of the mask assumed in this process step in the form:

```
      SUBROUTINE /NAME/ (VDOX,POSY,NCTOT,NCTM,
     +                   YCOR,STOP, INIT)
      DIMENSION YCOR(NCTM),STOP (2)
      LOGICAL INIT
```

where /NAME/ is any unique user-defined subroutine name. VDOX is the mask-thickness weighted with the stopping power estimation STOP(1) for SiO_2 masks and STOP(2) for Si_3N_4 masks. An initialization pass indicated by the logical variable INIT is performed to initialize all internally used variables and the parameters NCTOT and YCOR. NCTOT defines the number of lateral subinterval boundaries and the array YCOR defines the lateral position of the subinterval boundaries. NCTM is used as dimension of YCOR.

Different frequency functions can be used in the vertical direction. The program supports Gaussian, joined-half-Gaussian, and Pearson IV distribution functions. User-defined frequency functions have to be specified in the form:

```
      SUBROUTINE    /NAME/    (VALUE,X,Y,RPV,DRPV,SKEW,CURT,
     +                         DRPL,FMULT,INIT)
      LOGICAL INIT
```

where /NAME/ is any unique user-specified subroutine name, value denotes the function value of the frequency function at the position given by the coordinates X and Y. X specifies the vertical position and Y the lateral distance from the mean value. The parameters RPV, DRPV, SKEW, CURT, and DRPL denote the projected range, vertical standard deviation, third and fourth moment ratio, and the lateral standard deviation. An initialization pass is performed prior to computations of frequency function values where internal variables and the variable FMULT have to be defined. This initialization pass is indicated by the logical variable INIT. FMULT is a multiplication factor so that FMULT times the integral of the frequency function is unity.

The moments of the frequency function are either defined directly or via the implantation energy by applying well-established theories of the penetration of dopants in solids. The program supports moments found by using the LSS theory [12]. Other parameter sets can be defined by a user-defined function in the form:

```
      SUBROUTINE /NAME/ (NELEM,ENERGY,RPV,DRPV,SKEW,
     +                   CURT,DRPL,STOP,NSMOD)
```

where /NAME/ is any unique user-defined subroutine name. NELEM is an integer number which defines the dopant. ENERGY is the specified implantation energy. RPV, DRPV, SKEW, CURT, and DRPL are the projected range, the standard deviation in vertical direction skewness, kurtosis, and the lateral standard deviation. STOP(1) specifies the stopping power of a SiO_2 mask, STOP(2) the stopping power of a Si_3N_4 mask. NSMOD is an auxiliary argument in the program-supported range parameter definition subroutine which defines the way in which the stopping power is estimated. Either the ratio of the vertical standard deviations (NSMOD = 1) or the ratio of the proj-

ected ranges is used (NSMOD=2) as estimation for the stopping power.

The third-mentioned method to define initial profiles is the execution of a user-defined subroutine which has to be given in the form

```
      SUBROUTINE/NAME/(NEQU,NPTX,NPTY, POSX,POSY,W,TIME,
                       NPARM,USPAR,EIPI,NMSCM,NMISW,AMSC,
     +                 NECOM)
      DIMENSION POSX(0:NPTX+1),POSY (0:NPTY+1)
      DIMENSION W(NEQU,NPTY,NPTX),USPAR (NPARM)
      DIMENSION AMSC(NMSCM,NPTY,NPTX)
```

where /NAME/ is any unique user-defined subroutine name. NEQU denotes the number of equations, NPTX and NPTY the number of lines in vertical and lateral direction, respectively. POSX denotes the position vector in vertical direction and POSY the position vector in lateral direction. W is the solution vector whose values have to be defined by this subroutine. TIME and TEMP are auxiliary parameters representing the simulation time and the temperature, USPAR is an array with the length NPARM containing parameters which can be defined explicitly by the user. AMSC is an array which contains the values of the "internal functions" described later. NELEM is an array with the element numbers.

The next pass is usually to initialize all C_j in (1) for which the entries a_{ij} of the tensor A are zero for all i by the above mentioned user-defined initialization subroutines. The values of these C_j are used as starting guess for the iterative solution of the nonlinear system by Newton's method. This is for instance easily done in the case of the coupled boron–arsenic diffusion in Section 5.1 where the potential can be estimated by using the quasi-neutral approximation.

The third step is usually the computation of a steady-state solution or the time integration of the PDE's. The structure and the coefficients of the differential equations have to be defined by subroutines.

For the specification of the differential equations (1) they are assumed to consist of three terms. The first term describes the time dependence of the equations, the second term specifies the elliptic operator of the equation and the third one is a universal generation-recombination term. The equations for the physical quantities are treated individually for the purpose of specification. One of the equations reads

$$\sum_j a_{ij} \cdot \frac{\partial C_j}{\partial t} + \operatorname{div}\left(\sum_j (d_{ij} \cdot \operatorname{grad} C_j + \mu_{ij} \cdot C_j \cdot \operatorname{grad} \Psi)\right) + G_i - R_i = 0. \quad (14)$$

It is possible to distinguish several cases for each term. The time-dependence is specified in the form

```
      SUBROUTINE /NAME/ (NEQU,IAA,AA,DAA,C,POSX,POSY,TEMP,TIME,
     +                   NMSCM,AMSC,XSCALE,WSCALE,NMISM,
     +                   ARMAX,NAUX,SCAUX, NPARM,USPAR)
      DIMENSION  IAA(NEQU),AA(NEQU,NEQU),DAA(NEQU, NEQU,NEQU)
      DIMENSION  C(NEQU),AMSC(NMSCM),WSCALE(NEQU)
      DIMENSION  ARMAX(NMISM),SCAUX (NAUX),USPAR(NPARM).
```

The subroutine name /NAME/ of this and the following subroutine has to be unique in the program and has to be specified in the input deck. IAA defines the structure of the array defining the time dependence, AA defines the coefficients, and DAA the derivatives of AA with respect to the quantities C_j.

C(NEQU) denotes in this and in the following subroutines the real values of the dependent variables C_j, NEQU represents the number of equations to be solved. POSX and POSY are the positions in vertical and lateral direction respectively, TIME denotes the simulation time and TEMP the simulation temperature. The vector AMSC with the length NMSCM has been interpolated by using so-called "internal functions" whose specification will be explained later. "Internal functions" are tablefunctions which can be defined as superpositions of previous values of physical quantities and are used for instance to pass the doping profile in device modeling tasks to the PDE's. XSCALE denotes the maximal diameter of the simulation area and the vector WSCALE contains the maxima of the C_j. The vector ARMAX contains the maxima of the internal functions given by AMSC. The vector SCAUX with the length NAUX is used to pass auxiliary parameters to the subroutines defining the PDE's. The vector USPAR with the length NPARM contains parameters which have been explicitly defined in the input deck.

Three cases for the definition of the coefficients are possible and can be defined by parameters.

- The equation is time-invariant, all a_{ij} for equation i are zero. IAA(i) = 0, no other parameters have to be specified.
- a_{ij} is nonzero in equation i if $i = j$ and all other a_{ij} of the equation are zero. IAA(i) = ± 1, a_{ij} has to be specified for $i = j$. The derivatives of a_{ij} with respect to the C_j have to be defined if IAA is negative: AA(i, i) = a_{ij}, DAA(k, i, i) = $(\partial a_{ij})/(\partial C_k)$, $i = j$.
- One of the a_{ij} with $i \neq j$ is nonzero. In this case all a_{ij} have to be specifed. The derivatives with respect to the C_j have to be defined if IAA is negative: IAA(i) = ± 2, AA(j, i) = a_{ij}, DAA(k, j, i) = $\partial a_{ij}/\partial C_k$.

The second term denotes in principle the sum of the divergences of the currents related to the C_j. The parameters which specify the form of the currents in lateral and vertical direction and their coefficients are defined in the form

```
SUBROUTINE /NAME/ (NEQU,IBB,BB,DBB,IPSI,C1,C2, · · · , AMSC1,AMSC2, · · ·)
DIMENSION IBB(NEQU,NEQU),BB(NEQU,2, NEQU),IPIS(NEQU)
DIMENSION DBB(NEQU,NEQU,2,2,NEQU),C1(NEQU),C2(NEQU).
DIMENSION AMSC1(NMSCM),AMSC2(NMSCM)
```

IBB(j, i) specifies the structure of the current relations, BB(j, 1, i) and DBB(m, j, l, 1, i) denote the entries of the tensor D and its derivatives, BB (j, 2, i) and DBB (m, j, 2, i) the entries of the tensor μ and the derivatives of the entries. The cofficients are assumed to be defined at the point in the middle of two meshlines. $C1$ denotes the values of the quantities north or west of the midinterval point and $C2$ the values of the quanities south or east of the point for the current relations in vertical and lateral direction, respectively. In the same way the values of the "internal functions" AMSC1 and AMSC2 are defined. DBB specifies the derivatives of the entries of D and μ, IPSI specifies the Poisson equation. We can distinguish several possibilities for each subcurrent.

- The coefficients d_{ij} and μ_{ij} of the subcurrent of the quantity j in equation i are zero. The element IBB(j, i) is zero. No other coefficients have to be defined for this subcurrent.
- The quantity μ_{ij} of the subcurrent of the quantity j in equation i is zero. IBB(j, i) = ± 1, BB(j, 1, i) = d_{ij}. The derivatives of BB with respect to the quantities $C1_j$ and $C2_j$ have to be defined if IBB(j, i) is negative: DBB(m, j, l, 1, i) = $\partial d_{ij}/\partial C1_m$, where C is $C1$ for $l = 1$ and $C2$ for $l = 2$.
- The quantity d_{ij} of the subcurrent of the quantity j in equation i is zero. IBB(j, i) = $\pm$ 2, BB(j, 2, i) = μ_{ij}. The derivatives of BB with respect to the quantities $C1_j$ and $C2_j$ have to be defined if IBB(j, i) is negative: DBB(m, j, 2, i) = $\partial \mu_{ij}/\partial C_m$, where C is C1 for l = 1 and C2 for l = 2.
- The quantities d_{ij} and μ_{ij} of the subcurrent of the quantity j in equation i are nonzero. IBB(j, i) = ± 3, BB(j, 1, i) = d_{ij}, BB(j, 2, i) = μ_{ij}. The derivatives of BB with respect to the quantities $C1_j$ and $C2_j$ have to be defined if IBB(j, i) is negative: DBB(m, j, l, 1, i) = $\partial d_{ij}/\partial C_m$, DBB($m$, j, 2, i) = $\partial \mu_{ij}/\partial C_m$, where $C1$ is $C1$ for $l = 1$ and $C2$ for $l = 2$.

The generation-recombination term is defined in the form:

```
SUBROUTINE /NAME/ (NEQU,IDD,DD,DDD,C,· · ·)
DIMENSION IDD(NEQU),DD(NEQU),DDD (NEQU,NEQU)
DIMENSION C(NEQU).
```

The array IDD specifies the structure of the generation-recombination term. If IDD is 0 it is assumed that DD(i) is zero, too. If IDD is positive DD(i) has to be specified and is assumed to be constant with respect to the C_j. If IDD is negative the derivatives of the generation-recombination term have to be specified,

$$\text{DDD}(j, i) = \frac{\partial (G_i - R_i)}{\partial C_j}.$$

Similarly the boundary conditions have to be given in the form

```
SUBROUTINE /NAME/ (NEQU,IBD,BD,BDR,DBDR,C,CV,CL,DV,DL,BN,BS,
+ BE,BW,· · · )
DIMENSION IBD(2,2,NEQU),BD(NEQU,2, NEQU),BDR(2,NEQU)(N(NEQU),CL(NEQU)
DIMENSION DBDR(NEQU,2,NEQU),C (NEQU)
LOGICAL BN,BS,BE,BW.
```

For the specification of the boundary conditions, (3) is separated into individual equations for each quantity:

$$\sum_j \xi_{ij} \cdot JN_j + F_i = 0. \tag{15}$$

JN denotes the component of the current normal to the surface. IBD(1, k, i) specifies the summation mode of the quantity currents, IBD(2, k, i) the structure of the term F_i. BD denotes the summation coefficients for the quantity currents, BDR specifies the function F_i, and DBDR the derivatives of F_i with respect to the quantities C_j. CV and CL denote the values of the physical quantitities at the first inner line in vertical and lateral direction. DV and DL denote the distances between the boundary and the first inner lines. In two-dimensional simulation the boundary conditions have to be specified for the north and south boundaries of the simulation area (k = 1) and for the west and east boundaries (k = 2). The boundary condition of the north and the south boundary overwrites the boundary conditions of the east and west boundaries in the corners if a boundary definition conflict occurs. One exception to this rule is a Dirichlet boundary condition at the east and/ or west boundary which overwrites any boundary conditions in north and south direction for which IBD(1, 1, i) is nonzero. The logical parameters BN, BS, BE, BW specify whether the boundary conditions have to be defined for the north, south, east, or west boundary, respectively.

The structure for the summation of the subcurrents of the different quantities and for the functions F_i can be defined individually:

- IBD(1, k, i) = 0 means that all ξ_{ij} for this equation are 0. The BD(j, k, i) does not need to be defined.

- IBD(1, k, i) = 1 means that ξ_{ij} is nonzero for $j = i$ and zero for all other j. BD(j, k, i) = ξ_{ij}
- IBD(1, k, i) = 2 means that one of the ξ_{ij} is nonzero for $j \neq i$. BD(j, k, i) has to be defined for all j.

where TFV is the interpolated tablefunction value of the table function LFU(j, i).

To decrease the probabilities that the PDE's will be solved with wrong or mispositioned physical quantities the user has to provide a subroutine in the form:

```
SUBROUTINE /NAME/ (MODEL,NEQL,NEQUM,NELEL)
DIMENSION NELEL (NEQUM)
CHARACTER *80 MODEL
```

The same declaration form is used to specify the F_i:

- IBD(2, k, i) = 0 means that F is assumed to be zero. BDR and DBDR need not to be defined
- IBD(2, k, i) = 1 means that F is nonzero and invariant with respect to the C_j. BDR(k, i) = F_i has to be defined.
- IBD(2, k, i) = −1 means that F is nonzero and contains functions of the dependent variables C_j. BDR(k, i) = F_i and DBDR(j, 1, k, i) = $\partial F_i/\partial C_j$ have to be defined.
- IBD(2, k, i) = −2 means that F is assumed to contain additional functions of the dependent values at the first inner line. Additionally to IBD(2, k, i) = −1 DBDR(j, 2, k, i) = $2F_i/2Ck_j$ has to be defined, where Ck = CL for $k = 1$ and CV for $k = 2$.

Moreover, the scaling of the equations has to be performed by the user. All above mentioned quantities have to be specified in a scaled form. For the purpose of scaling it is assumed that all physical quantities are scaled by their maximum, lengths by the maximal length within the simulation area. These scaling factors and other auxiliary parameters can be computed once for each timestep and stored to the auxiliary array SCAUX. The computation of these auxiliary variables can be performed by a user-defined subroutine in the form:

```
SUBROUTINE /NAME/ (NAUX,SCAUX,TEMP,TIME, XSCALE,
+                 NEQUM,WSCALE,NMISM,ARMAX,INIT)
LOGICAL INIT
```

with the previously defined parameters NAUX . . . ARMAX. INIT is a logical variable which indicates the first timestep.

The definition of the "internal functions" is performed by a subroutine in the form:

```
SUBROUTINE /NAME/ (NMSC,NMISM,NFUNM,NFU,LFU,WFU)
DIMENSION WFU(NMISM),LFU(NFUNM, NMISM),WFU(NFUNM,NMISM).
```

The values of the internal functions are determined by the superposition of interpolated values of user-defined tablefunctions. These tablefunctions can be defined as superposition of quantity values of previous process steps. NFUNM and NMISM are the maximal number of internal functions and the maximal number of tablefunctions. NFUNM specifies the number of tablefunctions to be superposed in order to define the specific tablefunction. The internal function table values AMSC are calculated by

$$AMSC(i) = \sum_{j=1}^{NFU(i)} WFU(LFU(j, i), i) \cdot TFV(LFU(j, i))$$

where MODEL specifies a string which will be printed in the control-output-file, NEQL specifies the number of equations required for the model, and NELEL contains the element numbers of the physical quantities used in the subroutines. NEQL and the element numbers in NELEL will be compared to the actual number of physical quantities and to the element numbers of the actual physical quantities.

Additionally, a diffusion temperature can be specified. To simulate temperature profiles during the simulated heat treatment a subroutine has to be provided by the user in the form

```
SUBROUTINE /NAME/ (TEMPER, TIME).
```

The results can be printed, written to mass storage, or processed beginning with Step 1.

References

[1] H. Akima, "A new method of interpolation and smooth curve fitting based on local procedures," *J. Ass. Comput. Mach.* vol. 17, no. 4, pp. 589–602, 1970.

[2] J. P. Biersack, "Calculation of projected ranges—analytical solutions and a simple general algorithm," *nuclear instruments and methods*, no. 182/183, pp. 199–206, 1981.

[3] J. P. Biersack, "New projected range algorithm as derived from transport equations," *Z. Phys. A*, no. 305, pp. 95–101, 1982.

[4] R. K. Brayton, F. G. Gustavson, and G. D. Hachtel, "A new effective algorithm for solving differential-algebraic systems using implicit backward differentiation formulas," *Proc. IEEE*, vol. 60, pp. 98–108, 1972.

[5] P. Deufrlhard, "A modified newton method for the solution of ill-conditioned systems of nonlinear equations with applications to multiple shooting," *Numer. Math.*, vol. 22, pp. 289–315, 1974.

[6] A. Franz, G. Franz, S. Selberherr, C. Ringhofer, and P. Markowich, "Finite boxes—A generalization of the finite-difference method suitable for semiconductor device simulation," *IEEE Trans. Electron Devices*, vol. ED-30, pp. 1070–1082, 1983.

[7] S. Furukawa, H. Matsumura, and H. Ishiwara, "Theoretical considerations on lateral spread of implanted ions," *Jap. J. Appl. Phys.*, vol. 11, no. 2, pp. 134–142, 1972.

[8] C. W. Gear, "The automatic integration of ordinary differential equations," in *Information Processing*, A. J. H. Morrel, Ed., vol. 68. Amsterdam, The Netherlands, 1968, pp. 187–193.

[9] J. M. Hyman, "Accurate monotonicity preserving cubic interpolation," *SIAM J. Sci. Stat. Comput.*, vol. 4, no. 4, pp. 645–654, 1983.

[10] Knuth, *Fundamental Algorithms*. Reading, MA: Addison-Wesley.

[11] A. M. R. Lin, R. W. Dutton, D. A. Antoniadis, and W. A. Tiller, "The growth of oxidation stacking faults and the point defect generation at Si-SiO interface during thermal oxidation of silicon," *J. Electrochem. Soc.*, vol. 128, no. 5, pp. 1121–1130, 1981.

[12] J. Lindhard, M. Scharff, and H. E. Schiott, "Range concepts and heavy ion ranges," *Mater. Fys. Medd. Vid. Selsk.*, vol. 33, no. 14, pp. 1-42, 1963.
[13] W. C. Rheinboldt, "On a theory of mesh-refinement processes," *SIAM J. Numer. Anal.*, vol. 17, no. 6, pp. 766-778, 1980.
[14] H. Runge, "Distribution of implanted ions under arbitrarily shaped mask edges," *Phys. Status Solidi*, vol. (a) 39, pp. 595-599, 1977.
[15] H. Ryssel, "Implantation and diffusion models for process simulation," in *Proc. VLSI Process and Device Modeling*, Katholieke Universiteit Leuven, Belgium, 1983, pp. 1-41.
[16] S. Selberherr, "Analysis and simulation of semiconductor devices," New York and Vienna, Austria: Springer, 1984.
[17] S. Selberherr, W. Fichtner, and H. W. Pötzl, "MINIMOS—A program package to facilitate MOS device design and analysis," in *Proc. NASECODE I Conf.*, 1979, pp. 275-279.
[18] S. Selberherr, and C. Ringhofer, "Implications of analytical investigations about the semiconductor equations on device modeling programs," *IEEE Trans. Computer Aided Design*, vol. CAD-3, pp. 52-64, 1984.

*

Peter Pichler was born in Wr. Neustadt, Austria, on June 20, 1958. He studied at the Technical University of Vienna, Austria, where he received the Dipl.Ing. degree in industrial electronics and control theory in 1982. His master's thesis dealt with the simulation of operational amplifiers. In December 1982 he joined the Institut für Allgemeine Elektrotechnik und Elektronik at the Technical University of Vienna where he is working on his doctoral thesis on process modeling and process simulation.

*

Werner Jüngling was born in Berlin, Germany, on September 2, 1959. From 1977 to 1983 he studied at the Technical University of Vienna, Austria, and received the Dipl.Ing. degree in communications in 1983. His master's thesis dealt with the modeling of the intrinsic number for process and device simulation.

In June 1983 he joined the Institut für Allgemeine Elektrotechnik und Elektronik at the Technical University of Vienna where he is working on his doctor's thesis on process modeling and process simulation.

Siegfried Selberherr (M'79-SM'84) was born in Klosterneuburg, Austria, on August 3, 1955. He received the Dipl.Ing. degree in control theory and industrial electronics from the Technical University of Vienna Austria, in 1978.

He subsequently joined the Institut für Allgemeine Elektrotechnik und Elektronik, previously called the Institut für Physikalische Elektronik, at the Technical University of Vienna as an Assistant Professor. He finished his thesis, "Two-Dimensional MOS-Transistor Modeling" in 1981.

In 1983 Dr. Selberherr received the Dr. Ernst Fehrer award. He holds the *venia docendi* on computer-aided design since 1984. His current topics are modeling and simulation of devices and circuits for application in electronic systems. He has authored and coauthored more than 80 publications in journals and conference proceedings. He also wrote a book, *Analysis and Simulation of Semiconductor Devices*. He is a member of the Association for Computing Machinery (1979), the Society of Industrial and Applied Mathematics (1980), and the Verband deutscher Elektrotechniker (1984). He is editor of *The Transactions of the Society for Computer Simulation*.

*

Edgar Guerrero was born in Bogota, Colombia, in 1952. He received the Dipl.Ing. degree in technical physics from the Technical University of Vienna, Austria, in 1978. His dissertation involved the measurement and study of the redistribution of dopants in silicon. He received the Ph.D. degree from the Technical University of Vienna in 1984.

*

Hans W. Pötzl (M'62) was born in Vienna, Austria, in 1930. He received the Ph.D. degree in mathematics from the University of Vienna, Austria, in 1952 and the Dipl.Ing. degree from the Technical University of Vienna, Austria, in 1956.

In 1957 he joined the Institue for High Frequency Techniques, and worked under the supervision of Prof. König in the fields of noise and propagation modes in electron beams. In 1964 he became "Dozent" and stayed at the Technical University of Vienna as Full Professor in Physical Electronics. Since 1966 he has worked in the semiconductor field investigating hot electrons, heliconwaves, the acoustoelectric effect, cyclotron radiation from semiconductors in the far infrared region, and diffusion in impatt diodes. Since 1977 he has worked on silicon device and process modeling. He is coauthor of a book and author or coauthor or over 50 journal articles.

Since 1970 he has been member of the Austrian Academy of Science. In 1975 he received the award "Technikpreis der Wiener Wirtschaft" and in 1983 the "Österreichisches Ehrenkreuz für Wissenschaft und Kunst, I.Klasse."