AMIGOS: Analytical Model Interface & General Object-Oriented Solver

M. Radi, E. Leitner, E. Hollensteiner, and S. Selberherr
Institute for Microelectronics, TU Vienna
Gusshausstrasse 27–29, A-1040 Vienna, Austria
http://www.iue.tuwien.ac.at/

***Abstract*—To accurately simulate modern semiconductor process steps, a simulation tool must include a variety of physical models and numerical methods. Increasingly complex physical formulations are required to account for effects that were not important in simulating previous generations of technology. Thus flexibility in definition of models as well as numerical solving methods is highly desirable. An object-oriented approach has been applied to implementing a dimension independent solver which uses an analytical input interface in the manner of Math-Cad, Mathematica, Matlab, etc., but highly optimized for high performance semiconductor modeling. To demonstrate the abilities of AMIGOS a new approach to the local oxidation in three dimensions is presented, based on a parameter dependent smooth transition zone between silicon and silicon-dioxide. The resulting two phase problem is solved by calculating a free diffusive oxygen concentration and its chemical reaction with pure silicon to silicon-dioxide. This effect causes a volume dilatation which leads to mechanical stress concerning the surrounding boundary conditions. By a suitable set of parameters this kind of approach is equivalent to the standard sharp interface model based on the fundamental work of Deal and Grove.**

I. Introduction

During recent years it became increasingly difficult to meet all requirements in semiconductor simulation since a lot of new more complex physical models were developed. The transposition to computer science turned out to be very time consuming and resulted in a profusion of different simulators which can handle more or less a special but very restricted field of physical behavior especially in three dimensions. At the same time the demand for single process simulation decreased rapidly and instead of simple singular process steps the industry demands more and more a complete simulation flow control which can handle all necessary simulation steps to describe the behavior of any semiconductor device and its manufacturing process. However, the development of new complex physical models and its practical realization requires a large expenditure of time. To reduce this amount of developing time we analyzed the requirements for a general solver carefully and separated the common features such as modeling for physical definitions, parameter and model library, grid-adaptation, numerical solver, simple front-end controller as well as geometry and boundary definition. As a result we developed AMIGOS that translates a mathematical formulation of any nonlinear discretized coupled partial differential equation system into a highly optimized model which will then be passed to a nonlinear numerical solver that can handle different physical models on various grids as well as interfaces and boundaries.

II. AMIGOS

AMIGOS (Fig. 1) is a problem independent simulation system which can handle any nonlinear partial differential equation system in time and space in either one, two or three dimension(s). It is designed to automatically generate optimized numerical models from a simple mathematical input language, so that no significant speed loss in comparison to 'hand coded' standard simulation tools occurs. In difference to similar algorithms working with the so called 'operator on demand' concept [4], AMIGOS is completely independent of the kind of discretization since the model developer can formulate any discretization of choice. There are no restrictions whether using scalar, field or even tensor quantities within a model, and, if desired, any derived field quantity can be calculated, too. Furthermore, the user can influence the numerical behavior of the differential equation system by complete control of the residual vector and its derivative (e.g. punishing terms, damping terms, etc.). Even interpolation and grid-adaptation formulations can be used within a developed model and can thus be very well fitted to the special problem.

AMIGOS also provides several layers of access to the variety of users. For example, a process engineer (user-mode) may need only to select a model appropriate for the process step to calculate (e.g. oxidation), modify the process parameters (e.g. duration, temperature, material characteristics, etc.) and define the geometries and boundaries the process should be calculated on. On the

0-7803-3775-1/97/$10.00 © 1997 IEEE.

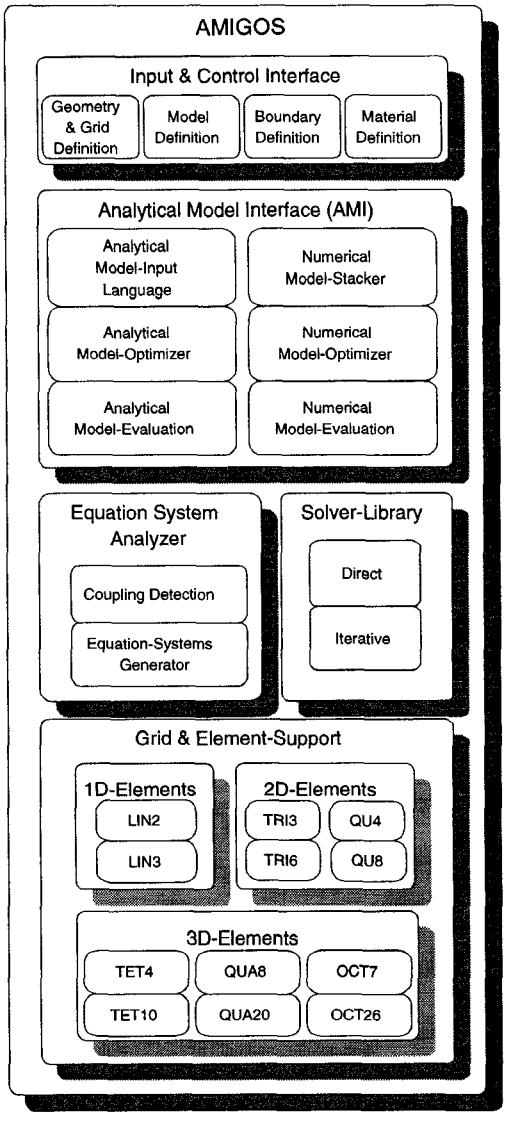

Fig. 1. Block structure of AMIGOS.

other hand, at a lower level (developer-mode) a model developer may need to modify existing equations by adding parameters, mathematical terms or equations, or even develop a complete new model. In contrast with previous generations of software none of the described modes requires access to and modification of the source code. Even during model development the analytical user input will be interpreted, optimized, transformed and solved on any complex simulation domain at once without the necessity of time consuming recompilations (one-pass concept) supporting a variety of several testing and debugging features. After finishing the test and calibration phase the user can switch to the two-pass concept where all modifications are translated to C-code and are linked to a model library for high performance calculations on large simulation domains in the standard user-mode.

The Input & Control Interface is mainly for describing the complete simulation flow process. The user has to define which physical models and boundary conditions should be used, where to read the geometry from and which sort of grid to use for discretization. Furthermore the kind of interpolation as well as the numerical solving methods are chosen within the Input & Control Interface. This information represents the basic input needed to solve a differential equation with AMIGOS. All of the remaining tasks are done automatically by the simulation system without any further interactions by the user. It detects whether to use moving grids in case of e.g. mechanical deformations as well as the dimension of the problem. It prepares all boundaries and interface grids and checks for existing interconnections between several areas. It initializes all models, its quantities and derivatives and allocates all necessary memory. After analyzing the partial differential equations to be solved it prepares the global system matrices and residual vectors and starts solving. If necessary it starts a Newton iteration for nonlinear problems and adapts the grid where the error-level is greater than a given epsilon area. After finishing it writes the results on a file either for later reuse or just for visualization tools (Fig. 2).

The Analytical Model Interface (AMI) is an interpreter for translating mathematical expressions describing a physical model into a numerical one, highly optimized for numerical simulation. Within this tool the models are defined and therefore it is just used in developer mode. The model description language (Fig. 3) is similar to well known systems like Math-CAD, Mathematica or Matlab, on the one hand, because of the physical and mathematical background, and on the other hand to reduce the break-in period for scientists who are not experienced in high level programming languages like C, C++ or Fortran. The user has to set up the quantities to be solved for, can select any kind of discretization (finite elements, finite differences or finite boxes) and has just to define the discretized residual vector and its derivative. To simplify the interactive process in user mode, several parameters and derivative quantities can be defined, too. This special quantities and parameters are detected by AMIGOS and are treated in a special manner so that the performance of the system is as high as possible. Therefore AMI is also equipped with an analytical as well as numerical optimizer which is tailor-made for matrix operations and standard mathematical expressions (e.g. +, -, /, *, sin, cos, atanh, ...) to minimize the number of operations for evaluation of a defined model. Every calculated mathematical term is stored and is reused whenever a similar term is detected (taking commutative law, associative law, etc. into account). Operations like multiplying by zero or one etc. are detected and immediately eliminated which reduces the amount of operations drastically, especially when us-

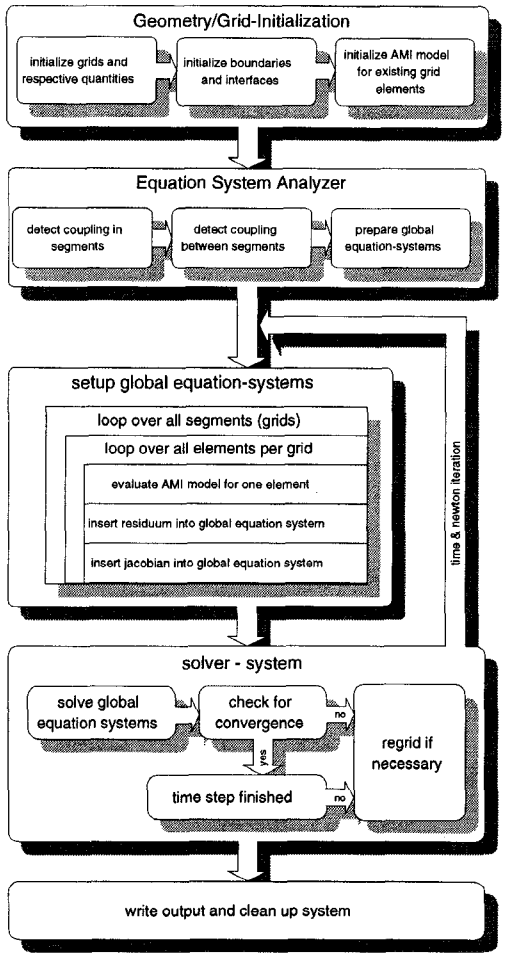

Fig. 2. AMIGOS's flow diagram of a complete simulation process.

```
MODEL ModelName = [x1,x2,...,xn];

   # MODEL is a predefined keyword
   # ModelName: the name the model should get
   # X = [x1,x2,x3,...] the solution-vector
{
   # Begin Model Description

# any user defined mathematical expression
# including matrices and tensors
var1 = x1+5*x2;
var2 = var1+2*x3;
var3(var1,var2) = var1 - var2;
...
# running index-variable from 1 to n
i = 1..n-1;
f[i] = var3(X[i],X[i+1]);

   # Definition of residual and its derivative to
   # get the numerical form:
   #       [jacobian] * [X] = [residual]

residual = [[f1(x1,x2,...,xn)]
            [f2(x1,x2,...,xn)]
            [          ...    ]
            [fn(x1,x2,...,xn)]];
jacobian = [[df1/dx1] [df1/dx2] ... [df1/dxn]
            [df2/dx1] [df2/dx2] ... [df2/dxn]
                                ...
            [dfn/dx1] [dfn/dx2] ... [dfn/dxn]];
}
```

Fig. 3. General model description language

ing matrices. Therefore the number of mathematical operations of a complete model can internally be reduced considerably (about two third) which leads to very high performance despite to high flexibility.

The Equation System Analyzer detects all couplings among different quantities within a volume model as well as connections between several different models caused by boundary conditions. As a result, the Equation System Analyzer distributes independent quantities to separate global matrices so that the amount of memory and evaluation time can be kept to a minimum. Furthermore, it prepares all necessary global stiffness matrices for the integrated numerical solver which can either be used as an iterative solver or, if necessary, as a direct Gaussian solver.

The Grid- & Element-Manager [1] which automatically detects the kind of elements defined on a given grid as well as its dimension. It is equipped with an adaptive-hierarchical grid algorithm thus the local discretization can be well fitted to the given physical problem.

III. Example: A Novel Diffusion Coupled Oxidation Model

To solve the local oxidation, a model with a smooth transition zone between silicon and silicon dioxide is assumed. To calculate the growth-rate of silicon-dioxide a generation/recombination term of SiO_2 and O_2

$$R_O = k_r \ (1 - \eta(x,t)) \ C_O \tag{1}$$

is defined, where $\eta = f\left(\frac{C_{SiO_2}(x,t)}{C_{Si_0}}\right)$ is a function of a normalized silicon dioxide concentration related to the C_{Si_0} concentration of silicon in pure crystal. η varies between one (pure silicon dioxide) and zero (pure silicon). The generation of silicon dioxide itself is handled by the formulation

$$\frac{\partial C_{SiO_2}}{\partial t} = R_O. \tag{2}$$

The free oxidant diffusion is described by a nonlinear partial diffusion equation which determines the growth of the concentration of generated silicon dioxide and therefore the growth-rate of the oxide at the material interface

$$\frac{\partial C_O}{\partial t} = div\left[D\left(\eta(x,t)\right) \cdot grad\left(C_O\right)\right] - 2 \cdot R_O. \qquad (3)$$

For the volumetric expansion we solve the equilibrium condition

$$\left(\int_V \mathcal{L}^T \cdot \mathcal{D}\left(\eta(x,t)\right) \cdot \mathcal{L} \cdot \, dV\right) \cdot \{u\} \;\; = \;\; = \int_V \mathcal{L}^T \cdot \mathcal{D}\left(\eta(x,t)\right) \cdot \{\epsilon_o\} \cdot \, dV \qquad (4)$$

where u, ϵ_o, $\mathcal{L}$ and $\mathcal{D}$ represents the displacement vector, the strain caused by silicon dioxide generation, the mechanical operator defined as $\{\epsilon\} = \mathcal{L} \cdot \{u\}$ and the elasticity matrix, respectively. The right hand side of (4) can be interpreted as an energy term caused by the chemical reaction between silicon and silicon-dioxide. Finally, the volume dilatation within the oxide is calculated by a hydrostatic pressure term

$$p = -\chi \cdot (\epsilon_{xx} + \epsilon_{yy} + \epsilon_{zz}) \quad \Longrightarrow \quad \Delta V \qquad (5)$$

As an example two typical three-dimensional effects arising in the corners of the nitride mask have been calculated (Fig. 4 and Fig. 5). Starting from a pure silicon block the results show, that the introduced model can even handle structures without pad oxide below the nitride mask, which leads to the effect, that the interface meets the nitride layer vertically, which can hardly be solved by algorithms based on a sharp interface formulation.

IV. Conclusion

The consequent progress in semiconductor device fabrication makes it necessary that simulation tools have to become increasingly flexible to keep up the pace of advance. With AMIGOS we have developed a powerful model development tool which supports the major interests of numerical engineering. The new approach for the simulation of local oxidation of silicon in three dimensions, considering the interface between silicon and silicon-dioxide as a smooth transition layer, demonstrates the abilities of AMIGOS in a convincing manner. According to theoretical investigations it can be shown that by a suitable set of parameters identical results to the standard Deal & Grove model can be obtained [2]. The advantage of this model against sharp interface descriptions is, that the mesh remains topologically invariant during the progress of oxidation and therefore no remeshing is necessary. Furthermore, this extension offers the possibility to handle much more complex physical effects than can be done with remeshing or grid-merging algorithms, since especially topology changing oxidation is hardly solvable with interface tracking algorithms.

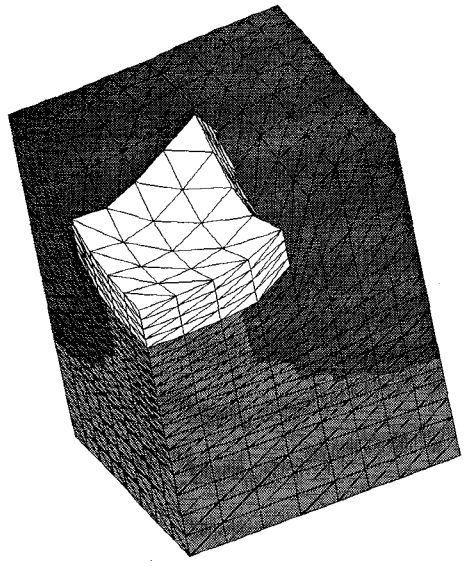

Fig. 4. Oxidation of a pure silicon block with a nitride-mask defined at the corner

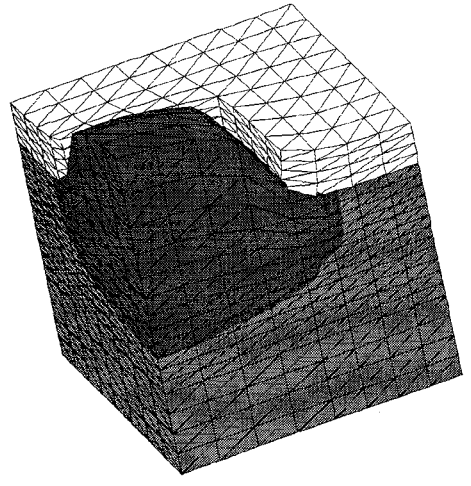

Fig. 5. Oxidation of a pure silicon block with the inverse nitride-mask shown at Fig. 4

Acknowledgment

This work is partly supported by ESPRIT Project 24038 PROMPT II.

References

[1] Leitner, E., and Selberherr, S. Three-Dimensional Grid Adaptation Using a Mixed-Element Decomposition Method. In: Ryssel and Pichler [3], pp. 464–467.

[2] Rank, E., and Weinert, U. A Simulation System for Diffusive Oxidation of Silicon: A Two-Dimensional Finite Element Approach. *IEEE Trans.Computer-Aided Design 9*, 5 (May 1990), 543–550.

[3] Ryssel, H., and Pichler, P., Eds. *Simulation of Semiconductor Devices and Processes* (Wien, 1995), vol. 6, Springer.

[4] Yergeau, D.W., Kan, E.C., Gander, M.J., and Dutton, R.W. ALAMODE: A Layered Model Developement Environment. In: Ryssel and Pichler [3], pp. 66–69.