A 3-dimensional process-simulator based on an open architecture

Tetsunori Wada, Hiroyuki Umimoto, Masato Fujinaga,
Mitsunori Kimura, Tetsuya Uchida, Kaina Suzuki,
Yutaka Akiyama, Masami Hane, Masahiro Takenaka,
Noriyuki Miura, Norihiko Kotani

Semiconductor Leading Edge Technologies, Inc.
Advanced Technology Research Dept.
292 Yoshida-cho, Totsuka-ku, Yokohama Yokohama 244-0817 JAPAN

Abstract

A 3-dimensional process simulator based on an open architecture has been design. A C++ like input language is designed to flexibly control process flow and to easily add user designed function. A self explaining data file format and its input/output C++ libraries are developed to ensure data exchange between user's data file and HySyProS by using C++ class libraries. Several classes for discretizing and assembling drift-diffusion equations are developed for testing diffusion models efficiently.

1.Introduction

As device size shrinks, various phenomena, such as transient enhanced diffusion, dose loss, etc. are getting dominantly affecting transistor performances. TCAD tools are expected to predict semiconductor device performance, especially process-simulator, by modeling these phenomena quickly. This requires process simulator to serve as a basis of implementing new models. It is also required to simulate device performance under many slightly different conditions such as ion-implantation dose, thermal treatment conditions. This requires input language to have a kind of programming language in describing process flow, such as variables, loop, conditional jump, etc. When using simulators in device design, measured and/or simulated data are expected to be assembled quickly, for getting solutions of process integration, failure analysis, etc. This requires a simulator to provide some methods for linking external software, data, and implementing newly proposed model of diffusion, oxidation, etc.

A newly developed 3-dimensional process-simulator HySyProS uses the following technology to solve above problems:

1)C++ like input data language and its interpreter program to control modeling programs

2)simple but versatile data-file format with read/write libraries

3)object oriented programming to access device structure data safely and efficiently

2.Design Concept

As described above, HySyProS is designed not only be used as a process simulator, but also as an environment for developing process models. As a process simulator, so called Fair model[1], point defect model[2] for impurity diffusion, visco-elastic model for oxidation, analytic model (Gaussian, Pearson-IV, dual-Pearson, etc.) with plus-1 model for ion-implantation, idealistic isotropic deposition, directional model for etching are installed.

As an environment for process model developments, it is very important to provide efficient and secure method for handling data, such as grid, elements, region, their relations, and matrix assembling methods. Several C++ classes, including oct-tree based mixed element (tetragonal, pyramid, prism, and rectangler solid) for solving diffusion and oxidation model, have been developed for this purpose. A C++ class for discretizing and solving drift-diffusion equation using CVM (control volume method) is also developed and applied in solving diffusion models mentioned above. Etching and deposition models are implemented by using equi-contour method [3].

Figure 1 shows a schematic of HySyProS. An interpreter program is linked to process-models and I/O functions. One can link external program with HySyProS, like a mask-imaging model in Fig.1, by using a calling function. During the simulation, data representing device structure is accessed and updated mainly through the member function of C++ classes.

2-1. Interpreter program

We have designed a C++ like input language (hereafter we call it "scl") for describing process flow, in which variables (with member function), conditional jump, loop, importing/exporting functions, overloading function-name, external program calling are available. By using scl and restarting function, a complicated process-flow, including branches, conditional sequence, etc. can be expressed. Overloading function-name helps to develop or add, for example, a different diffusion models having the same function name with different arguments. One can also run external programs through his input data, and this helps mixed use with other software. Or, one can use external program as if it is an intrinsic function of HySyProS, by wrapping his program with scl scripts, and register a function name to a function-table.

2-2. Data-file format

Many simulators and tools, which will be frequently modified, are used by transferring and exchanging data files. In this case, when some tool is modified to put additional output data, a system using data file format based on a relative addressing, or a format using implicit rules, cause confusion. We have developed simple, self-explaining format to avoid this confusion. It is assumed that a data file can be divided into several blocks, and our format is applicable to such a file. We designed a script, hereafter we call it a header, attached to the beginning of each data block. The meaning, ordering, types are explained in the header by combinig symbols. A C++ libraries for reading and writing in our format are also developed. This helps one to make a program code which ignore other data blocks automatically and access specified data.

Figure 2 shows a concept of our self-explaining data-file format. Our data format is not designed for realizing automatic data-conversion [4] from some format to this or vice-versa. It is designed to read simple data, like scalar-, array-, structure-type data by inserting a header. Rules of representing these information is designed as simple as possible to make a header easily.

3.Examples

Figure 3 shows a simulation of stacked DRAM-cell structure. About 80*40*40 cell are used to simulate complicated structure formed after 33 process steps, and required CPU time is about 40 minutes (HP C-240). Figure 4(a) shows a trench corner after oxidation process (980C 5 min). As is shown in it, oct-tree based mixed element grid is successfully generated. At the corner and edges, grid is automatically refined during oxidation step, to represent accurate structure. Figure 4(b) demonstrates the merging of two oxide regions after oxidation of poly-silicon layer on an silicon di-oxide film. An oct-tree base mixed elements with the combination of moving boundary is successfully representing device structure.

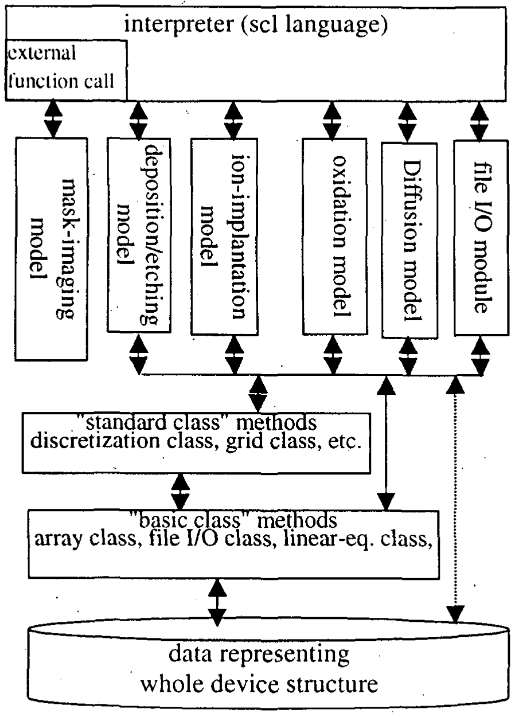
We have also been developing 3-dimensional device simulator, based on a same concept with those of the present process-simulator. Figure 5 shows simulated I-V curves of nMOSFET's using device structure data calculated by the process-simulator.

Acknowledgement

Authors thank to Dr. T. Yamauchi in Mathematical System Inc. for excellent effort of realizing our concepts. We also thank to the member of our device-simulator team.

REFERENCES

[1] R.B.Fair."Silicon Integrated Circuits", Part B, pp.5-20, Academic Press, 1981.
[2] K. Ghaderi and G. Hobler, J. Electrochem, Soc., vol. 142, no. 5, pp. 1654-1658, 1995.
[3] M. Fujinaga and N. Kotani, IEEE Trans. Electron Devices, Vol.44, No.2, p.226, 1997.
[4] S. G. Duvall, IEEE Trans. on CAD, vol.7, No. 7, pp.741-754, 1988.



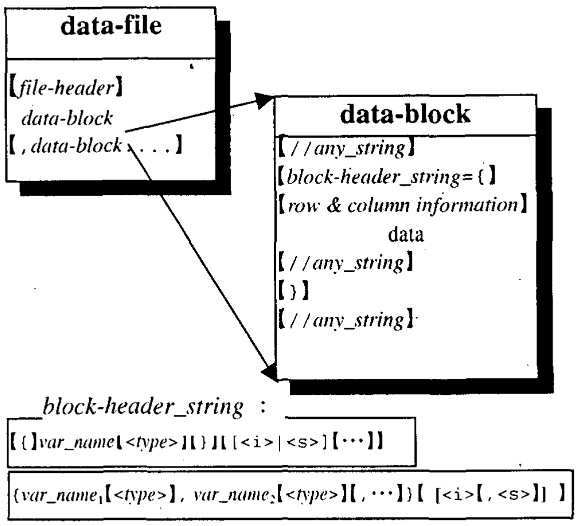


Fig.2 Concept of self-explaining data-file format. Name, type, configuration file format. Name, type, configurationof data is written in a block-header.

Fig.1 System diagram. Interpreter controls process models such as diffusion, etc. Device structure data is accessed through class-methods. Arrows shows data flow.

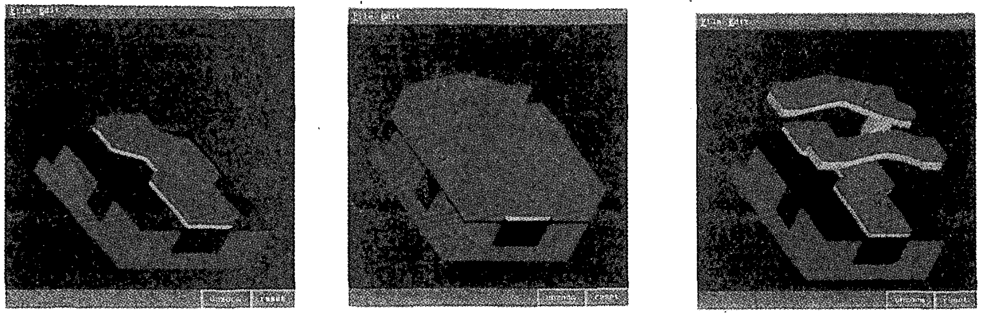
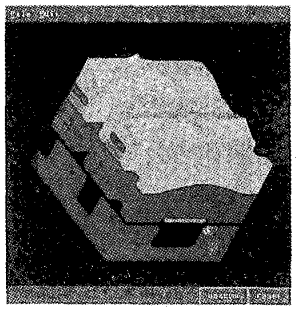
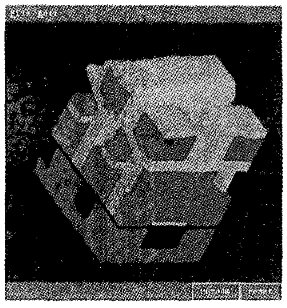


Fig.3 Simulation of DRAM-cell structure. (1) trench-isolation and bit-line patterning, (2), deposition of passivation-layer, (3) word-line patterning (to be continued to next page)


(4)


(5)

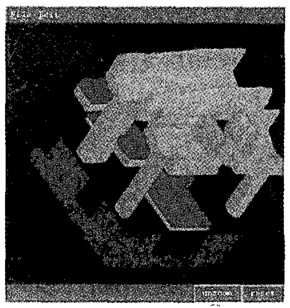
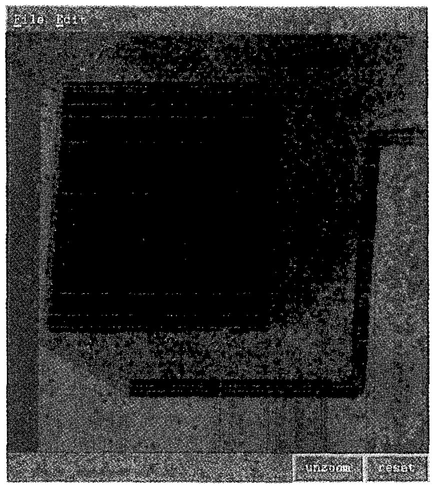

(6)

Fig.3 (4) deposition of passivation-layer, (5),(6)formation of stacked cell


(a)

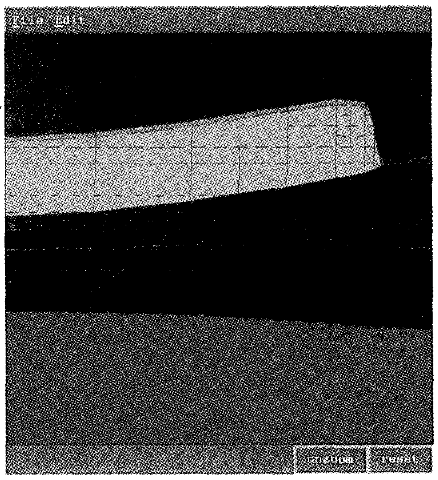

(b)

Fig. 4 simulation examples of oxidation. (a)trench structure, (b)merged oxide layer

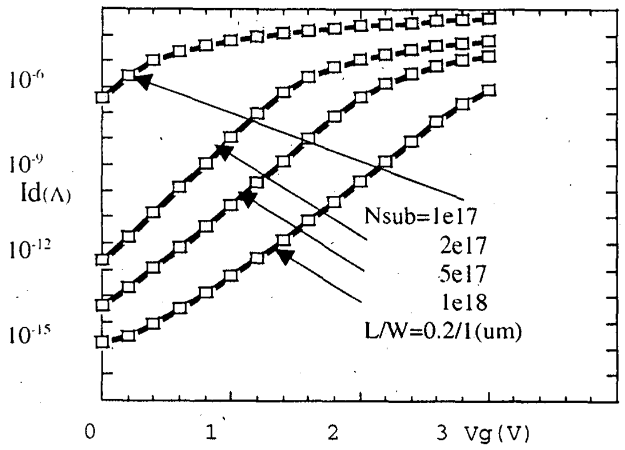


Fig.5 Simulated I-V curves calculated for different channel dose, by using 3D device simulator