SIMULATION OF SEMICONDUCTOR DEVICES AND PROCESSES Vol. 4
Edited by W. Fichtner, D. Aemmer - Zurich (Switzerland) September 12-14, 1991 - Hartung-Gorre

Grid Generation for 3-D Nonplanar Semiconductor Device Structures

Nancy Hitschfeld Paolo Conti Wolfgang Fichtner
Integrated Systems Laboratory, ETH-Zentrum, 8092 Zurich, Switzerland

Abstract

We have developed an automatic grid generator, called Ω, for complex nonplanar three-dimensional (3-D) semiconductor device structures. The meshes generated by Ω permit an exact geometry modeling of the rather general domain boundaries of modern semiconductor technologies. Avoiding obtuse angles by construction, Ω is the ideal preprocessor for a 3-D device simulator.

The numerical solution of partial differential equations (PDEs) is invaluable in the design and the optimization of semiconductor devices and integrated circuits. The spatial discretization of the structure to be simulated is the key to the accuracy of the computed solution. A reasonable approximation of the geometry to be modeled and of all internal quantities relevant to the solution of the PDEs, such as the doping profile, is extremely important. Additional constraints arise from the discretization schemes used. As classical finite element schemes seem inappropriate, PDEs are usually solved using the control volume or box method [9]. This necessitates that obtuse angles be avoided, another nontrivial condition.

In two dimensions(2-D), both rectangles and triangles have been used for the initial coverage of the integration domain. Bank and co-workers have proposed covering the integration domain (a closed polygon) with a carefully chosen triangulation, and to recursively subdivide triangles where higher mesh resolution is needed into four similar triangles, through the addition of four new mid-edge points [2, 3]. Yerry and Shephard proposed to use a modified quadtree data structure: the integration domain is encapsulated in a square, and the square's quadrants are recursively subdivided in quadrants until the mesh density is sufficient to model the domain geometry and internal quantities appropriately [4]. Müller et al.[13] have recently extended the quadtree idea in the implementation of a fully automatic 2-D mesh generator.

In 3-D, the recursive refinement of simplices (i.e., tetrahedra) is much harder than in 2-D [5]. Major problems include the difficulty in generating a well-shaped initial tetrahedral grid, the impossibility to regularly subdivide tetrahedra in similar sub-elements, and the problems with tetrahedra between dense and coarse mesh regions. Therefore, the modified quadtree approach has been extended successfully to three dimensions [6, 7, 8]. The 3-D domain is enclosed in a cube, whose octants are repeatedly refined until the boundary and corresponding internal quantities are sufficiently approximated. A slightly different approach for the generation of octree-based Delaunay meshes has been proposed by Schroeder and Shephard [11].

The first version of the grid generator Ω_{oct}(Ω octree), as presented in [10], was based on a conventional modified octree approach. It was shown that the octants of an octree could be completed with additional edges on the surface or in the interior of each octant. Modified octrees suffer from three serious drawbacks. First, all octants are cubic and the point density is locally constant along all three coordinate axes. However, relevant quantities in a semiconductor device may vary strongly along one axis, while remaining constant in the plane perpendicular to the axis. In these cases, isotropic cells, as those resulting from modified octrees, lead to a large number of redundant mesh points. Second, the algorithm used to obtain Delaunay meshes requires that material interfaces intersect an octant in the center of its edges. As a result, several refinements and a huge number of mesh points were required along material interfaces. Third, there may be inaccuracies in computing the volume discretization for interface elements.

In order to overcome these problems, the current version of Ω_{me}(Ω mixed-elements) generalizes the modified octree approach in several aspects. The whole device is no longer encapsulated in a single octree, but partitioned in a set of *macro-elements* consisting of cubes, rectangular prisms and rectangular pyramids. We use this set of elements because they allow the discretization of the device volume using the box method. Furthermore, they are closed under the refinement process, that is, each element obtained after a partition belongs to the same set[1].

Using this approach, arbitrary plane-faced geometries can be represented using a reasonable number of mesh points, and no refinement along material interfaces is required to fit the geometry of the device. Subsequently, the desired mesh density in the interior of the device is obtained through the recursive refinement of prisms, pyramids, and cuboids. If a finer mesh is required along one, two, or three coordinate axes, cubes, for example, are subdivided into halves, four quadrants, and eight octants, respectively. Then, the elements with additional mid-edge points are subdivided into tetrahedra, pyramids, prisms, or bricks in order to get a proper finite element mesh.

1 New approach on the mesh generation process

Ω starts by generating a 3-D tensor-product mesh, i.e., an initial grid in a cuboid called the *bounding volume* is drawn, which surrounds the perimeter of the device geometry with the minimum volume. The bounding volume looks like a set of cuboids that may be crossed by boundary or internal interfaces. Uncut cuboids are already valid as *macro-elements*. Cut cuboids are tessellated in valid *macro-elements* using a set of predefined patterns that fit a particular cut cuboid. For example, two simple cut cuboids and their tessellations are shown in Figure 1. The left cuboid is tessellated in two rectangular prisms and the right one in three rectangular pyramids.

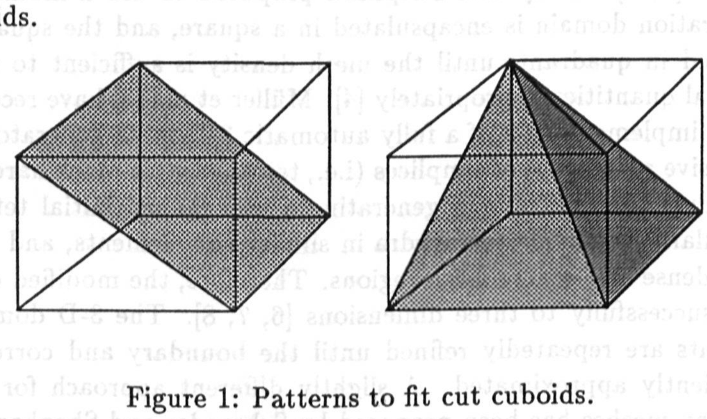

Figure 1: Patterns to fit cut cuboids.

Finally, the material of each *macro-element* is set. Figure 2 shows the macro-grid for the a model of the ECL bipolar transistor. At the left side, the whole device was separated into its

[1]A very simple element, the rectangular tetrahedron, cannot be used as *macro-element* because it is neither possible to discretize its volume using the box discretization method nor to find a partition including only elements belonging to the same set.

two different material regions in order to show how the interface was fit, and at the right side, a zoomed view shows clearly the type of *macro-elements* used.

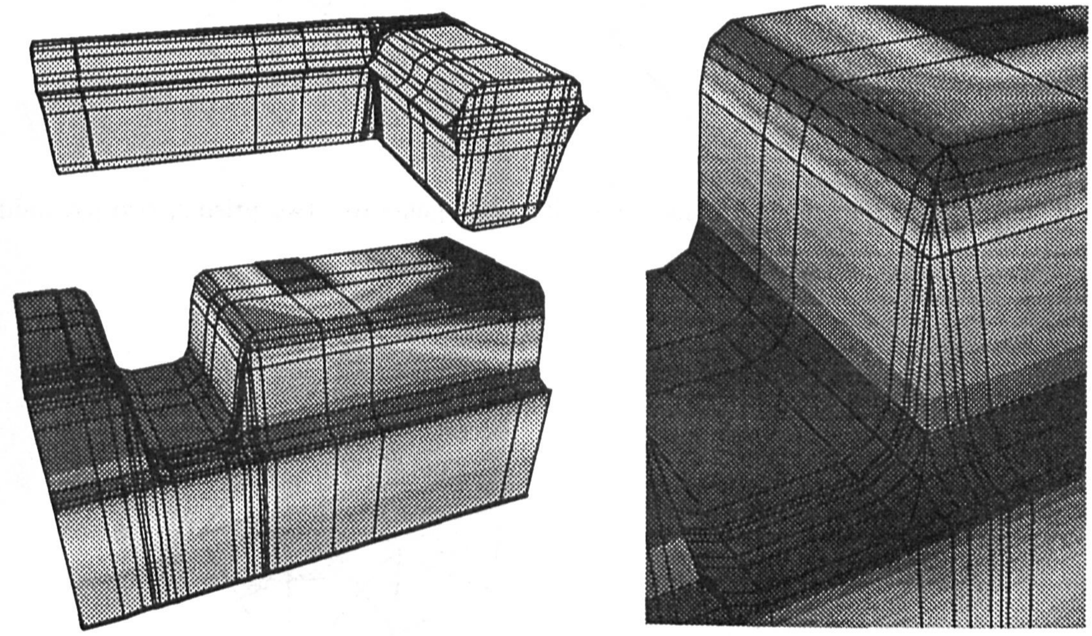

Figure 2: Macro-grid for the ECL bipolar transistor. The number of *macro-elements* is 2,069: 28 pyramids, 369 prisms and 1,672 cuboids.

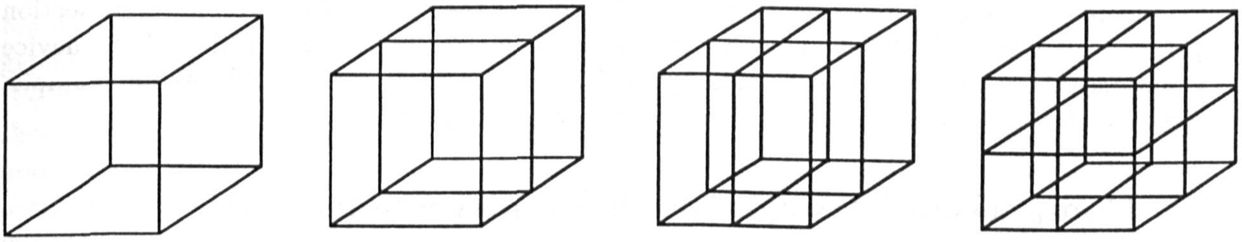

Figure 3: Cuboids refined in one, two, or three directions generate two, four, or eight cuboids, respectively.

The *macro-elements* must be recursively refined to resolve geometry and doping until an adequate mesh density is obtained. In Figures 3 and 4, the allowed direction refinement for a cubic and a pyramid *macro-element* are shown. Elements considered too coarse are repeatedly refined until the final grid is appropriately dense in all regions of the device.

After the adequate mesh density is obtained, all the trees are made 1-irregular[2]. If possible, Ω tessellates 1-irregular leaves into tetrahedra, pyramids, prism, and cuboids based on precomputed information. If, in some cases, this information is not available or the leaf fails to fulfill eccentricity conditions, Ω has several heuristic strategies that either refine perpendicular to the longest edge or add some additional mid-edge points to render the current leaf splittable. In order to guarantee convergence, no element is refined finest beyond the level of refinement attained before the 1-irregular step.

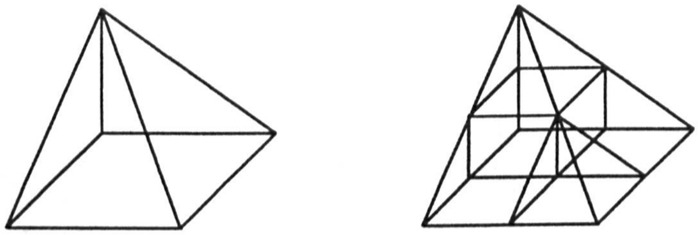

Figure 4: Pyramid refinement in the three directions generates two prisms, two pyramids and one cuboid.

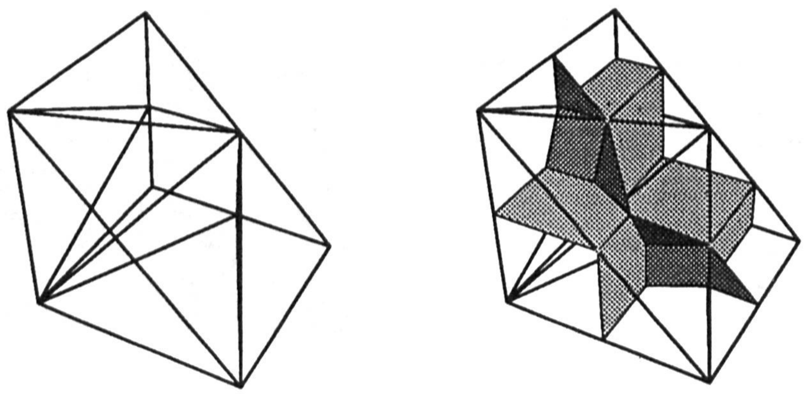

Figure 5: Voronoi cross sections for a 1-irregular prism with three split edges

In addition to the finite element grid itself, Ω provides the surface of the Voronoi cross section perpendicular to each Delaunay edge of the grid. Necessary for the integration of the device equations, these surfaces are evaluated symbolically using *Mathematica* [12]. As an example, Figure 5 shows the Voronoi cross sections for a 1-irregular prism.

2 Comparison between the different versions of Ω

The current version of Ω is slower than the old one but generates significantly fewer points, thus allowing the simulation of more complex semiconductor structures. Table 1 shows the mesh sizes and CPU and memory requirements for differents device structures. run on a SUN SPARCstation 1+ with 20Mb main memory. The last three examples could only be generated using Ω_{me}due to their nonplanarity.

Device	CMOS		ECL	LOCOS	MCT
ΩVersion	Ω_{oct}	Ω_{me}	Ω_{me}	Ω_{me}	Ω_{me}
# points	68820	13926	17678	42131	29355
# elements	100033	13008	22904	66524	41938
CPU [s]	284	165	2330	9867	1655
memory (mb)	13.7	10.8	11,3	18,54	14,2

Table 1: CPU and memory requirements of Ω_{oct} and Ω_{me}

There is no obvious relation between the number of points and CPU time. The CPU time depends strongly on the geometrical complexity of the modeled device. Ω_{me} takes more time to generate a mesh if pyramids were used to fit the geometry. The reason is the pyramid's lack of flexibility in refining and handling the 1-irregular condition in a pyramid.

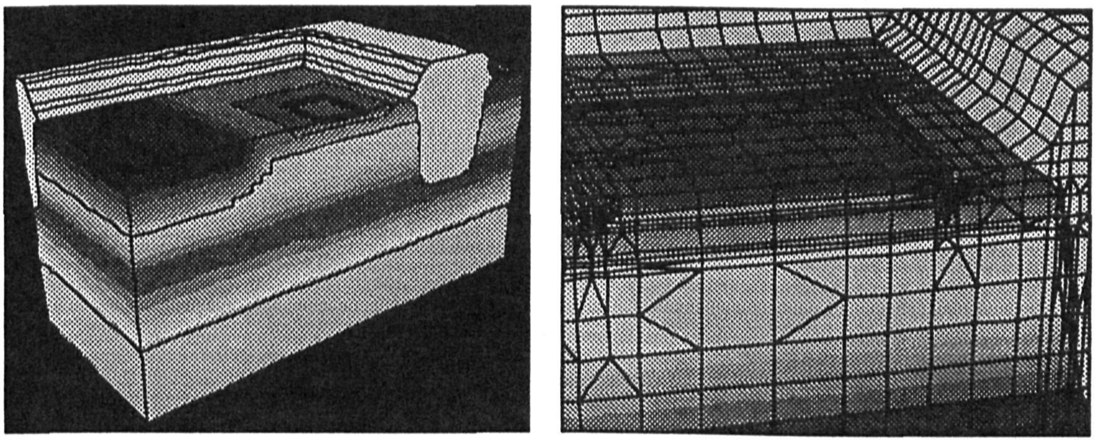

Figure 6: Trench-isolated bipolar transistor. The curved oxide trench was approximated using pyramids and prisms. The whole device has 17,313 points and 21,978 3-D elements.

3 New examples

As representative examples of the current Ω functionality, Figure 6 shows a trench-isolated bipolar transistor as used in state-of-the-art high-speed ECL designs. The left side shows the geometrical characteristics and impurity distribution after the proper grid was generated, and the right side shows a zoomed view of the grid generated at the p-channel. Figure 7 shows a short channel MOS transistor with surrounding LOCOS isolation. Again the device geometry and the impurity distribution are shown on the left side and a zoomed view on a part of the p-region under the gate is shown in the right. Figure 8 shows a MOS-controlled Thyristor with integrated MOS controlled n^+ emitter shorts and a bipolar turn-on gate. The left figure shows the impurity concentration and the mesh in the top part of the device, and the figure on the right one shows a zoomed view of the very thin oxide layer.

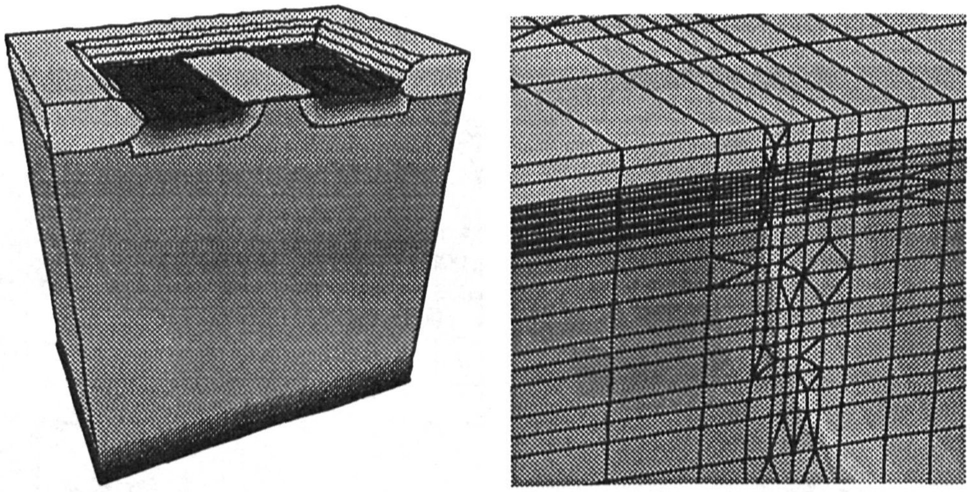

Figure 7: The whole grid for the LOCOS has 42,567 points and 62,442 3-D elements.

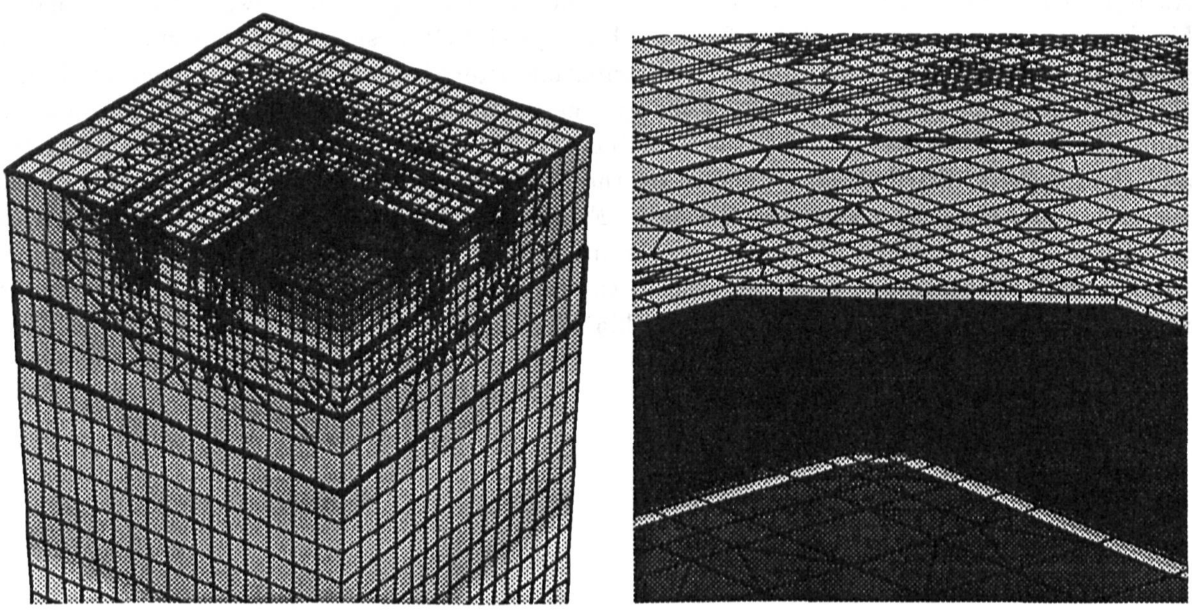

Figure 8: The whole grid for the MCT has 29355 points and 41938 3-D elements.

References

[1] P. Conti. *Grid Generation for Three-dimensional Device Simulation.* PhD thesis, ETH Zürich, 1991. published by Hartung-Gorre Verlag, Konstanz, Germany.

[2] R. E. Bank, A. H. Sherman, and A. Weiser. Refinement algorithms and data structures for regular mesh refinement. In R. Stepleman et al., editors, *Scientific Computing*, pages 3–17. North-Holland, 1983.

[3] R. E. Bank. *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations* Users' Guide 6.0. Society for Industrial and Applied Mathematics, 1990.

[4] M. A. Yerry and M. S. Shephard. A modified quadtree approach to finite element mesh generation. *IEEE Computer Graphics and Applications*, pages 39 – 46, 1983.

[5] P. Conti and W. Fichtner. Automatic grid generation for 3d device simulation. In *Proc. of SISDEP 3 Conf.*, pages 497–505, 1988.

[6] M. A. Yerry and S. Shephard. Automatic three-dimensional mesh generation by the modified-octree technique. *Int. J. Numer. Methods Eng.*, 20:1965–1990, 1984.

[7] M. S. Shephard, M. A. Yerry, and P. L. Baehmann. Automatic mesh generation allowing for efficient a priori and a posteriori mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 55:161–180, 1986.

[8] E. K. Buratynski. A fully automatic three-dimensional mesh generator for complex geometries. *Int. J. Numer. Methods Eng.*, 30:931–952, 1990.

[9] R. E. Bank, D. J. Rose, and W. Fichtner. Numerical methods for semiconductor device simulation. *IEEE Trans. on El. Dev.*, ED-30(9):1031–1041, 1983.

[10] P. Conti, N. Hitschfeld, and W. Fichtner. Ω – an octree-based mixed element grid allocator for adaptive 3d device simulation. *IEEE Trans. on CAD/ICAS*, september 1991. in press.

[11] W. J. Schroeder and M. S. Shephard. A combined octree/Delaunay method for fully automatic 3-d mesh generation. *Int. J. Numer. Methods Eng.*, 29:37–55, 1990.

[12] Stephen Wolfram. *Mathematica: A System for Doing Mathematics by Computer.* Addison-Wesley, 1988.

[13] S. Müller, K. Kells, and W. Fichtner. Automatic rectangle-based adaptive mesh generation without obtuse angles. Submitted to IEEE Trans. on CAD/ICAS, 1990.