Layout-based 3D Solid Modeling for IC

Zhiping Yu, Ken Wang, Tao Chen, Robert W. Dutton, and Jeff T. Watt[*]

Center for Integrated Systems, Stanford University, CA 94305, U.S.A.
[*]Cypress Semiconductor Corp., San Jose, CA 95134, U.S.A.

Abstract --- A software system for 3D solid modeling of IC structures (devices, interconnection, and circuits) based on the layout design and processing information is described. Both the 3D perspective view and 2D cross sections of the structure can be examined using the system. Features of the system include automatic update of the geometry as the layout is changed and dynamic display of the process for constructing a structure layer by layer. Coupled with the process simulator and meshing tool, this modeling system can be used for device simulation. Further analysis of interconnection and circuit performance is made possible with other parameter extraction tools. The system configuration, its software components, and numerical techniques in processing mask data will be discussed. The entire process of building an SRAM cell and generating an array of cells will be demonstrated as an application example of the system.

INTRODUCTION

With the increasing number of IC's being designed and fabricated using the sub half-micron rule, 3D effects such as narrow width and trench isolation on MOSFET's characteristics become ever important. To accurately assess the impact of the layout design on device/circuit performance and manufacturability of the tight process integration, IC designers often demand the capability of examining the detailed structure such as the step coverage of a deposited layer inside an IC component (e.g. a device or a circuit unit like a RAM cell) before it is actually made. In this paper we describe a modeling system which allows users to create a solid object of an IC component based on the layout design and processing specification. The object can be viewed either from a 3D perspective or be dissected into slices or cross sections to reveal the internal structure. The modeling process is layout-based, meaning that the 3D object is updated automatically once the layout is modified. With incorporation of 2D/3D process simulator such as SUPREM IV and etching/deposition modeler such as SPEEDIE, both programs being from Stanford, this system is expected to deliver accurate description of each layer in an IC structure including bird's beak of the field oxide and step coverage of thin layers. Furthermore, we have also developed a 2D/3D mesh generator, SMB (for Stanford Mesh Builder), with which the 3D object or its 2D cross sections can be meshed for 2D/3D device simulation and interconnection analysis.

The paper is organized as follows. The software components of the system are first described. Then the requirement for input data including both the layout and process specification is discussed. The data structure for mask description and the numerical technique for processing polygons in mask representation are introduced. An SRAM cell with four transistors in CMOS technology with feature size of 0.6 μm is then modeled using the system. The layer-by-layer building process is shown. Finally the process of generating an entire array of cells is demonstrated.

SOFTWARE SYSTEM CONFIGURATION

A 3D solid modeling process includes (1) constructing individual components which constitute an object, and for IC part these components are typically layers such as oxide and polysilicon layers; (2) assembling the object by placing each component in a proper position; (3) displaying and manipulating the entire object or part of it in a 3D perspective in order to examine its external structure; (4) the ability to cut cross sections through the object for either internal structure investigation (e.g. step coverage of deposited layers) or further device analysis/simulation; (5) and finally the growth (or expansion) of the structure from a single cell to an entire circuit. The above requirements demand the following tools or procedures:

1. A preprocessor to convert the layout data and process spec (in this case actually the layer thickness and the shape of the cross section such as the bird's beak) to what a geometry modeler takes for building 3D objects.
2. A geometry modeler to describe 3D objects and their relative position using the information provided by the preprocessor.
3. A visualization tool to display 3D objects either individually or in a collective manner. The tool should have all the capabilities of manipulating and dissecting an object in a 3D perspective.
4. Utility programs to place and rotate unit cells to form array of cells and the capability of peeling off particular layers.

The approach in developing this 3D modeling system is to use the combination of commercially available programs and customarily developed software. In particular, ACIS from Spatial technology, Inc. is used for geometry modeler and AVS from Advanced Visual Systems, Inc. is used for visualization. The software developed by ourselves includes a C program which takes layout data and process spec to write an input file for ACIS and a number of script files, both at UNIX shell and AVS levels.

In the following section we discuss the processing of the layout data to generate input information needed by the geometry modeler.

LAYOUT DATA PROCESSING

The layout design is described by mask specification in commonly acceptable formats such as GDS-II or CIF. Currently GDS-II ASCII files are used for input of layout data in this system. Each mask layer consists of a number of polygons with unique ID number to identify the mask level to which they belong. So the first step in processing the layout data is to store the polygons (and hence masks). In our customarily written C program, a structure (`struct`) is defined for each polygons in the mask layer. The following is a piece of C code for describing a mask.

```
typedef struct wire {
    int layerID;
    int noVerts;
    COORDINATE *cords;
    struct wire *nextWire;
} WIRE;
```

where `layerID` is the ID number of the mask layer and polygon is specified through the chained structure `COORDINATE`. Next polygon can be introduced by the pointer `nextWire`. Using this structure, the entire layout data can be accessed by a single pointer (to `struct WIRE`). The polygons are used in solid modeling mainly for two purposes: to create blocks by sweeping along some line segment and to allow certain cross section (also a polygon) to be swept along themselves to form rings. After all these simple 3D blocks and structures are created, logical geometric operations such as subtract, intersect, and unite on them are used to obtain desired layers for constructing the final IC structure. To perform the above two basic functions, some polygons need to be processed before being used. We now discuss the numerical techniques for processing the polygon.

A polygon is specified by a series of x-y coordinates (i.e. vertices) with closing vertex the same as the starting one. To form a ring structure using the cross section along one polygon, one must know, when walking down the path from the starting vertex, on which side of the path the interior (or equivalently exterior) of the polygon lies in order to determine the orientation of the cross section. This distinction can be made by realizing the fact that all polygons without void can be considered as formed by a band with finite width which may vary. By heuristic reasoning, the following rule can be found to determine the interior of the polygon when processed along the path. At any internal point on an edge of the polygon, draw a perpendicular line to the edge. Then count the number of intersections of this line with the edges of the polygon on each side of the present edge. Counting zero intersection as an even number, one can conclude that the odd number of the intersections indicates the interior of the polygon. For example, see

Figure 1.

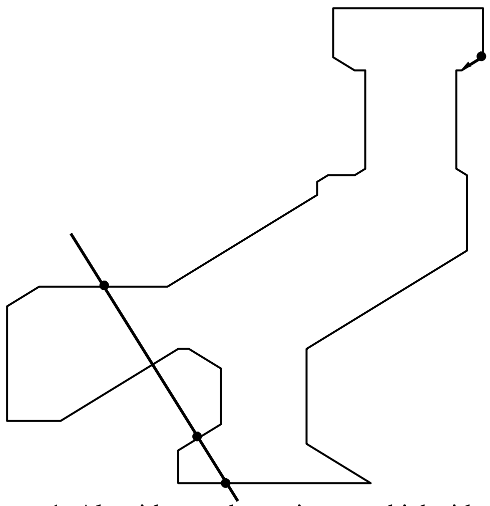

Figure 1: Algorithm to determine on which side along the path the polygon lies. The arrow shows where the series of coordinates start and the direction of the path. In the line drawn, there are one intersection (odd) on the right hand side along the path and two (even) on the left. So the polygon lies on the right hand side of the path.

A second case where a polygon representation needs processing is for polygons with a void such as the one with doughnut shape. In GDS-II format, such a polygon is still represented by a continuous series of coordinates but with certain edges overlapped. This is called the coincidental facial structure and can not be handled by the geometry modeler ACIS readily. So we need to partition the one piece representation of a polygon into two disjoint polygons. The numerical approach is simple and we call it as the look-ahead technique: when read in the new vertex after reading the starting one into the data structure, check forward to see if there is any identical vertex down the series. If there is a one it means that this polygon has a void and the dissection of the polygon can be proceeded by skipping the vertices between two identical ones to form an outer polygon and use the vertices in between to form an inner polygon. For example, see Figure 2.

With these two techniques incorporated in the program, the GDS-II data can be readily used for creating layers in IC structure.

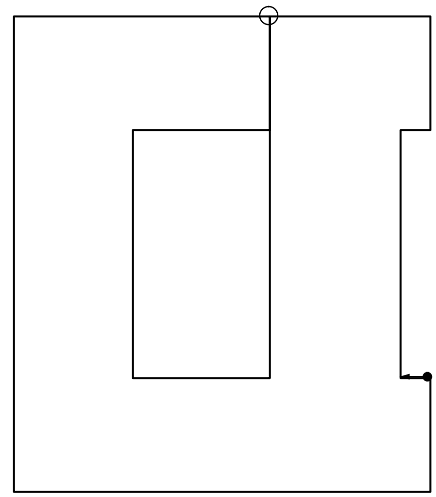

Figure 2: Polygon with void in the interior, i.e. doughnut shape. It must be broken into two disjoint polygons before ACIS can process. The open circle indicates two identical vertices.

CONFORMING DEPOSITION AND ANISOTROPIC ETCHING

To model the IC process, a major challenge is to implement the conforming deposition process. By conforming, we mean at present (if not using programs such as SPEEDIE) an isotropic deposition on top of the current topology. This can be done again with the so-called look-ahead scheme. When the non-planar structure first appears, not only the current layer but also the subsequent (blanket) layers are deposited simultaneously. For example, to deposit a gate oxide on top of the field oxide is accomplished by depositing two layers (one for field and the other for gate oxide) consecutively before patterns are etched in the field oxide. The same principle can be applied to more complicated cases. Currently, in the program a strict isotropic deposition (i.e. uniform thickness in all directions) scheme is adopted and is of course not realistic. But this assumption can be easily overridden if a more detailed cross section is known from simulation. The etching operation is relatively easy to implement and it essentially involves only the subtract logical operation. In the present system, an anisotropic etching, i.e. etching only in the vertical direction, is implemented.

GENERATION OF A UNIT CELL

In order to demonstrate the capability of the system, we have successfully modeled the SRAM design with both six and four transistors in a unit cell. The feature sizes are 0.5 and 0.6 μm, respectively. In the four-transistor cell, there are eleven layers including the silicon substrate. From the substrate up, these layers are: field and gate oxides, thin and first polysilicon (poly) layers (for ground and word lines), inter-poly dielectric layer and second poly layer as the load resistance and power line, a planarized BPSG layer, first metal layer for bit lines, and inter-metal dielectric and second metal layers. The finishing SRAM cell and an intermediate structure after the formation of the field oxide are shown in Figure 3 and Figure 6, respectively. For further device simulation, either a slab (Figure 4) or cross section (Figure 5) can be taken from the 3D object. In Figure 6, the bird's beak of the field oxide can clearly be seen. The cutplane in Figure 3 is through one of the pass transistors.

GENERATION OF A CELL ARRAY AND INTERCONNECTION

An important aspect of modern IC design is the need of analyzing parasitics and interconnection. The system described here can be applied to generation of an entire circuitry based on the unit cell rendering. To generate a cell array, a basic block, which may consist of several unit cells with different orientation, has to be constructed first. This can be done by flipping and flopping the unit cells. For example, in four-transistor SRAM design, two cells have a ground lines abutted and together they form a basic block. The entire cell array can then simply be created by repetitive displacement of the blocks. A 4X4 cell array with first metal layer (for bit lines) exposed is shown in Figure 7. In order to reveal the word line interconnection, the first metal layer and contacts are stripped in the front-right cell.

Once the interconnection is constructed together with the ground and interleaving dielectric layers, effects of parasitics and interconnect on circuit performance can be analyzed through such tools as FASTCAP and FASTHENRY from MIT.

The entire process of building a cell layer by layer and creating a cell array can be shown dynamically using the system. The output files for hard copy of the image can be in either .pnm (for Portable aNy iMage), .gif, or PostScript format.

MESHING AND FUTURE WORK

A companion software tool for the 3D solid modeling system is the mesh generation. Based on an improvement to quadtree scheme, a 2D and 3D mesh generation program has been developed at Stanford [1]. This meshing tool can be used for either 2D cross sections or an entire 3D object for further analysis. An example is shown in Figure 8 which is a mesh on a 3D vertical, sidewall flash EPROM cell proposed by H. Pein [2].

Further development of this modeling system includes the incorporation of 2D and 3D numerical and semi-analytical simulation for etching and deposition and process simulation to have more accurate description of layers and their cross sections. The system is being extended for computational prototyping of atomic level solid state structures projected for use in the 21st century.

ACKNOWLEDGEMENT

Support from ARPA through contracts DAAL 03-91-C-0043 and DABT 63-93-C-0053 is gratefully acknowledged. This work has been performed as a joint effort between Stanford and Cypress. The encouragement from Dr. A. R. Alvarez of Cypress is mostly appreciated.

REFERENCES

[1] D. Yang, K. Law, and R. W. Dutton, "δ-zone triangulation: a boundary refinement scheme for quadtree based mesh," *Proc. Int'l Workshop on VLSI Process and Device Modeling*, pp. 211-213, 1993.

[2] H. B. Pein and J. D. Plummer, "Performance of the 3-D sidewall flash EPROM cell," *Digest IEDM '93*, p. 11, Washington D.C., 1993.

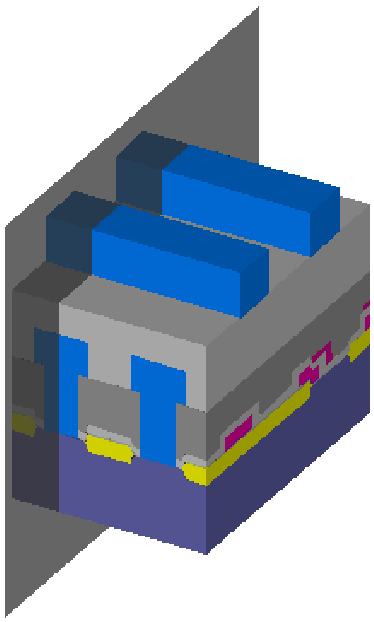

Figure 3: The 3D solid model of a 4-transistor SRAM cell with the cut-plane through one of the pass transistors.

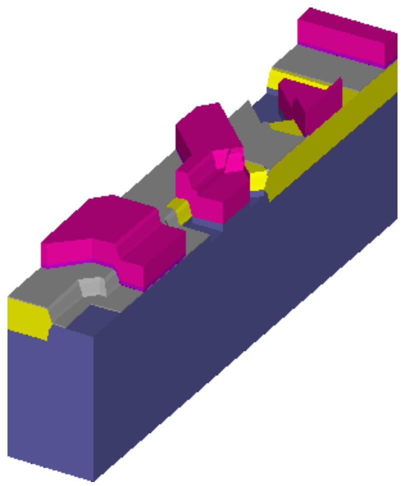

Figure 4: Slice of the above cell as viewed from the cut-plane towards the left.

Figure 5: 2D cross section from Figure 3. Only silicon substrate, field/gate oxides, and first poly layer are shown. The pass transistor is located on the top left and the field oxide isolation is on the top right.

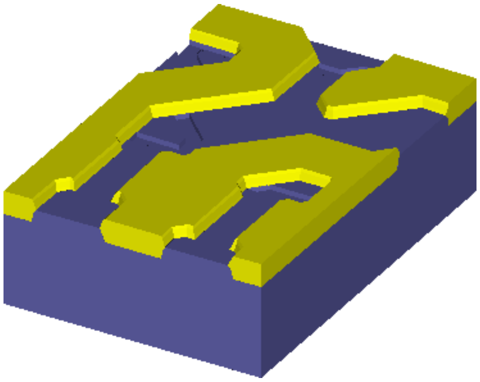

Figure 6: A 3D view after the growth and patterning of the field oxide. The active region is the exposed silicon and bird's beak can be seen.

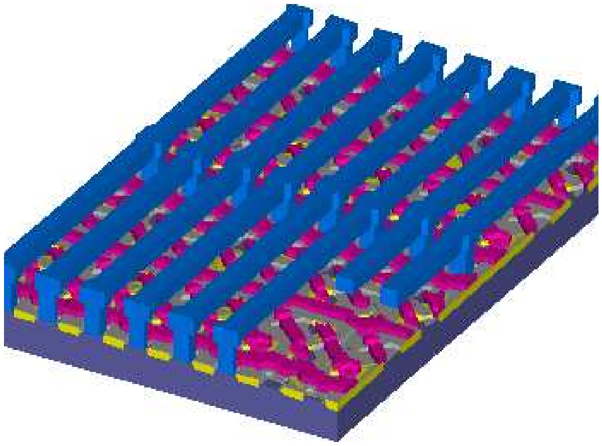

Figure 7: An array of 4 x 4 SRAM cells. Bit (blue) and word (red) line interconnects are shown.

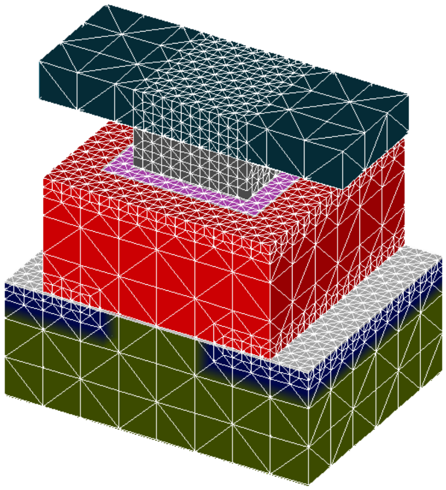

Figure 8: 3D mesh generated by Stanford mesh builder for a vertical, sidewall flash EPROM cell.