# Application of Mechanical-Technology CAD to Microelectronic Device Design and Manufacturing

Fariborz Maseeh*, Robert M. Harris, Duane S. Boning,
Michael L. Heytens, Sean A. Gelston, and Stephen D. Senturia
Microsystems Technology Laboratories
Massachusetts Institute of Technology, Cambridge, MA 02139

## 1 Introduction

The semiconductor industry has a continuing need for computer tools for many different tasks, such as device and process design and simulation, scheduling, computer supervised machine operation, and lot tracking. In addition, increasing attention is being paid to the *mechanical* structure of microelectronic devices, in part because device reliability depends directly on the mechanical integrity of the device and its package, both during manufacture and in use. This paper reports on the development of a micro-electro-mechanical CAD system (MEMCAD), and describes its application both to device design and manufacturing. The MEMCAD system provides for the modeling of the 3-D structure created by a given process sequence and mask layout, both for electrical analysis (such as determination of capacitances between conductors and line inductances), and mechanical analysis (such as the mechanical stress distribution in a structure). It can also be used, for example, to evaluate the risk of fracture in as-fabricated structures, the effects of temperature changes on mechanical behavior, and the mechanical interaction of the device with its package.

This system is being developed as part of the MIT Computer-Aided-Fabrication (CAF) Project. A central idea of that project is the creation of a formalized process description, called the Process Flow Representation, or PFR[1], which is a single process description that can be used for all the tasks mentioned above: process design and simulation, scheduling, equipment operation, lot tracking, and with the addition of the MEMCAD system, mechanical visualization and analysis of device structure.

There are two critical problems facing the MEMCAD system: (1) creation of a 3-D solid model of a device structure for its process flow and mask set; and (2) the incorporation of the correct process dependence of the mechanical properties of the constituent materials. Koppelman[2] has developed a highly-successful program called OYSTER which creates polyhedral 3-D solid models from process and mask information. OYSTER requires its own process description syntax, is restricted to polyhedra, and uses a boundary representation of the elements of the solid model. This approach is quite satisfactory for visualization of the shapes but does not provide a solid-model representation which permits an easy interface with finite-element method (FEM) mechanical analysis programs. Further, the use of polyhedra will create stress-concentration artifacts at corners. Finally, OYSTER has no provision at present for dealing with the mechanical properties of the constituent materials.

The present MEMCAD system is being developed with the specific goals of (1) use of the PFR for the process description; (2) creation of the 3-D solid model in a representation and format which permits the use of curved surfaces and is directly usable by finite-element codes; and (3) capturing of the process dependence of the material properties for use in the mechanical analysis. The architecture of the this system has been reported previously[3, 4] and is shown in Fig. 1. It consists of three blocks. The Microelectronic CAD block includes process description with PFR, mask layout, and a new function called the *Structure Simulator* (STS) which creates a 3-D solid model from the process and mask information in a specific format which can be read by the Mechanical CAD system. The Mechanical CAD

---

[*] present address: Teknekron Sensor Development Corp, Menlo Park, CA 94025

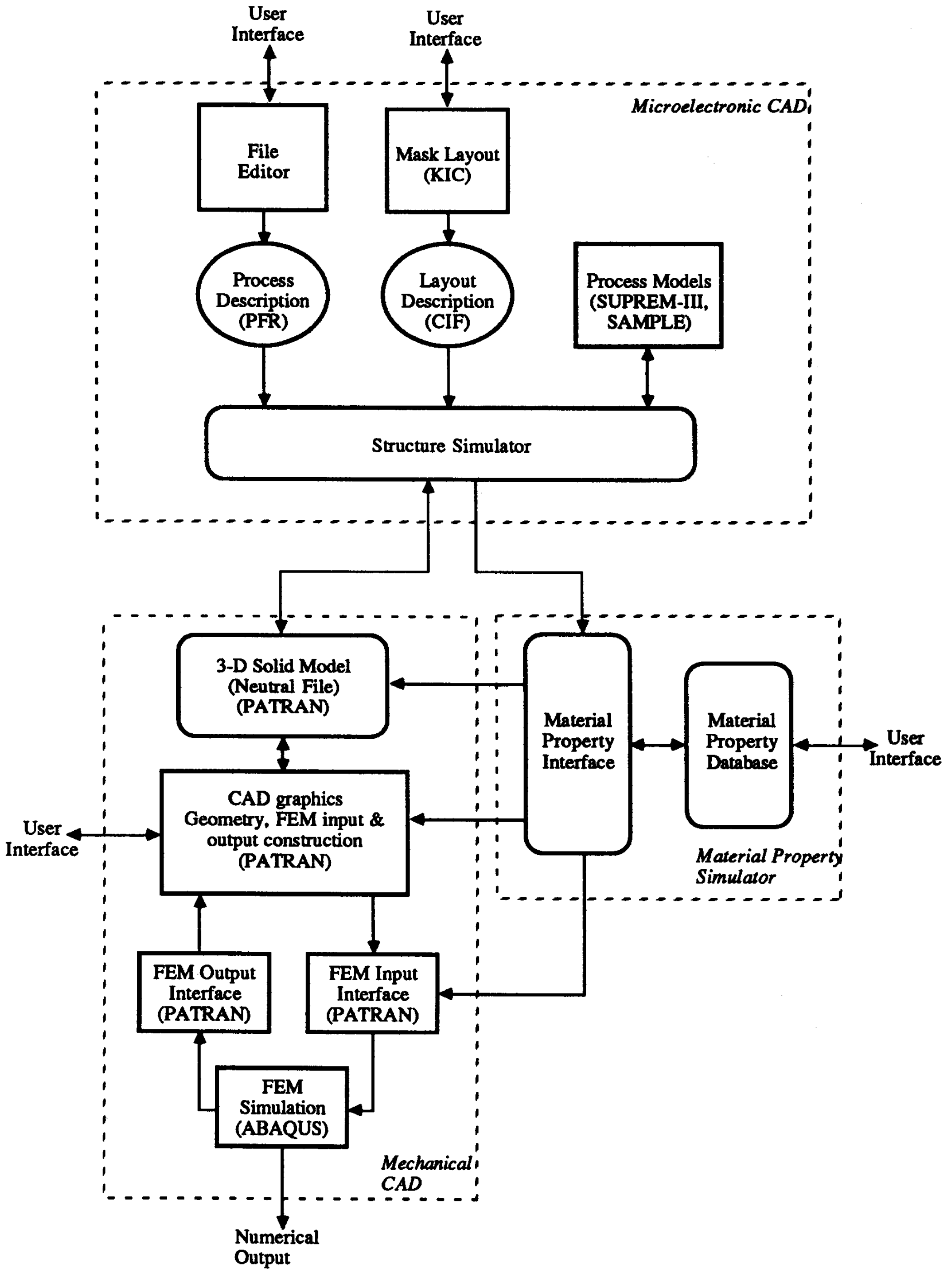


Figure 1: MEMCAD system architecture

block is based entirely on commercially available modules: PATRAN (for 3-D modeling, visualization, meshing, and insertion of loads and boundary conditions for finite-element modeling), and ABAQUS (for finite-element analysis). The third block is the *Material Property Simulator* (MPS), and is being developed to extract the process-dependent material properties from an object-oriented database using the PFR-derived process history, and to insert these material properties into the 3-D solid model prior to finite-element analysis.

In our previous reports[3, 4], we demonstrated the automatic generation of a 3-D solid model and subsequent mechanical analysis (stress distribution and wafer bending) of a chip which had been oxidized and then, in a lithographic step, had a portion of the oxide etched off. Material properties in this case were derived from PATRAN-readable lookup tables. This paper reports two new results: the 3-D modeling of the initial portions of the MIT Baseline CMOS process; and the first implementation of the object-oriented material property database described in Ref. [3].

## 2 Structure Simulator

The Structure Simulator (STS) works on a process-step-by-process-step basis, modifying the solid model after each process step. This mimics the physical fabrication sequence, in which each process step causes a change in the wafer. The final structure is therefore the result of a sequence of such changes. The step-by-step operation of the STS also permits simulation of intermediate results of the process sequence. This can be used, for example, to simulate whether the structure will maintain mechanical integrity throughout the entire process sequence.

The operation of the STS is summarized in Table 1. For each process step, the Structure Simulator begins by reading the current step from the PFR, and the effect of this step is determined. (For example, the process ***step*** "oxidize in dry $O_2$ for 100 min. at 950°C" is converted into the process ***effect*** "grow 420Å silicon dioxide on bare ⟨100⟩ silicon".) When necessary, process information is passed to process simulators and the simulation results are used to determine the process effect.

Modification of the solid model based on the process effect is done in two steps. First, the process effect is decomposed into a combination of primitive construction operators. These primitive operators are then used to modify the solid model from the previous process step, both the geometry (in the PATRAN Neutral File) and the material-type information (in the Process History File). The updated model is output for use with the next process step and this sequence is then repeated for the next process step.

Table 1: Summary of Structure Simulator Operation

1. Determine effect of process step.
   Consult layout information and process modelers, as necessary.
2. Decompose process effect into primitive construction operators.
3. Modify solid model using primitive operators.
4. Output results
5. Repeat steps 1–4 for next process step.

For microelectromechanical design, the following primitive operators constitute a useful minimal set: film deposition and growth, film etching, impurity introduction and movement, and wafer joining. These primitives are combined using selection operators which restrict the operation of primitive operator to a certain region of the solid model. Selection may be done on the basis of layout (mask information) or material type.

Implementation of the Structure Simulator is in the initial stages. Currently, only planar film growth and film etching are supported. Selection can only be done on the basis of a single mask feature. Process information is not read directly from the PFR. Instead, the program queries the user for the process information needed. The STS is interfaced to SUPREM-III and is able to use it to model oxidation. The program also outputs the geometry information in PATRAN Neutral File Format which is directly usable by the Mechanical CAD block.

## 3 Material Property Simulator

The Material Property Simulator reads the process sequence for each component of the solid model from

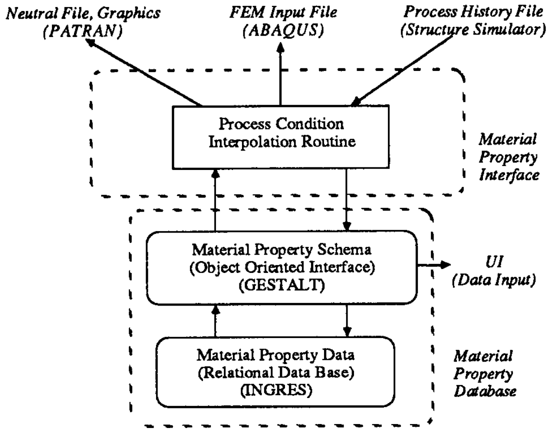
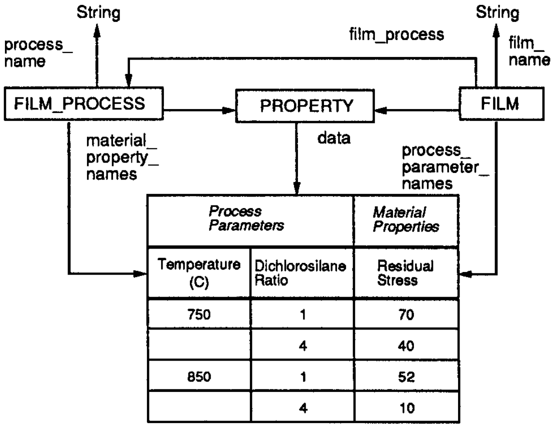


Figure 2: Material property simulator architecture



| Process Parameters | | Material Properties |
|---|---|---|
| Temperature (C) | Dichlorosilane Ratio | Residual Stress |
| 750 | 1 | 70 |
| | 4 | 40 |
| 850 | 1 | 52 |
| | 4 | 10 |

Figure 3: Schema for Material Property Database

the Process History File and generates a set of material property data. The material properties are passed either to PATRAN or are input directly to the FEM program by modify the FEM input file (see Fig. 1). Because of the detailed organization of PATRAN, *residual* stress (as opposed to *thermal*) cannot be passed to PATRAN, but must instead be input directly to the FEM program. In addition, this direct input of material properties allows the use of FEM simulator independently of PATRAN.

The Material Property Simulator consists of a *Material Property Interface* and a *Material Property Database* (see Fig. 2). The Material Property Database contains three basic types of interacting objects: "FILM", "FILM PROCESS", and "PROPERTY". The schema for the database is shown in Fig. 3. A FILM object describes the type of film (`film_name`), how it was made (`film_process`), and the material properties for which there is data (`material_property_names`). A FILM PROCESS object describes a process used to make a film. It consists of name which identifies the process (`process_name`) and the process parameters that may be varied (`process_parameters`). A PROPERTY object contains the actual data for a material property (`material_name`) versus process parameters (`process_parameter_name`) for a particular film type (`film_name`) and process (`process_name`). The data is stored as a table of `material_property_values` versus `process_parameter_values` in the PROPERTY object.

The Material Property Database has been implemented using the object-oriented database environment, GESTALT[5] which has several advantages. First, it shields application programs from the many details of the underlying database (currently INGRES, a commercial relational database system[6]). GESTALT, however, is flexible enough that the underlying database could be replaced or upgraded without affecting existing application programs. It also provides for an environment in which application programs can be written in different higher level languages (currently C and LISP).

The Material Property Interface does three functions. First, it reads the information from the Process History File generated by the Structure Simulator and passes this to an interpolation routine. This routine accesses the database to retrieve the PROPERTY objects for the given film and process type. The routine then takes the retrieved data table entries of material property value versus different parameters values and interpolates them to a single value appropriate for the given process conditions. Finally, the Material Property Interface returns the calculated material properties to PATRAN by storing them in the Neutral File or modifies the FEM input file directly.

The first version of the Material Property Simulator has been written in C, and initial data for the process dependence of some of the properties of silicon dioxide and silicon nitride have been entered into the database. Retrieval and interpolation (one-dimensional cubic spline) have been demonstrated, but the interfaces to the rest of the system have not yet been implemented.

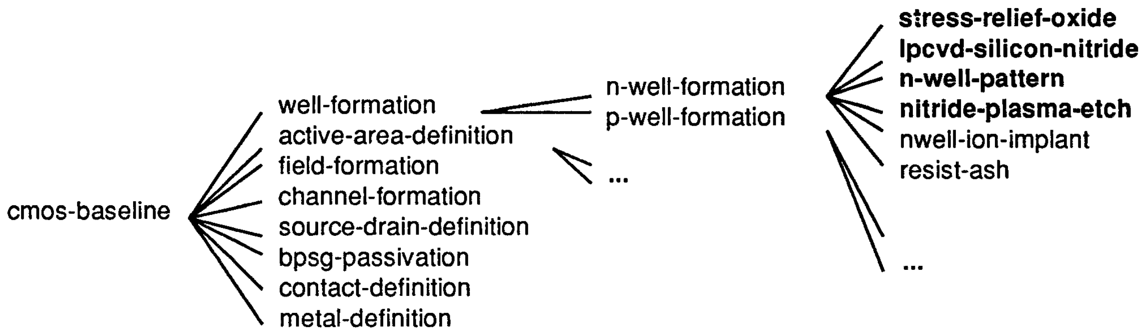


Figure 4: Hierachical structure of the CMOS Baseline process in the MIT Process Flow Representation.

## 4 Example Process

We illustrate the use of our MEMCAD architecture with an analysis of the initial steps of the MIT baseline twin-well CMOS process. The overall fabrication process involves 60 process steps (excluding cleans). The final structure consists of five distinct materials: silicon, polysilicon, BPSG (borophosphoro-silicate glass), silicon dioxide, and aluminum. A sixth material, silicon nitride, is used as a temporary masking layer. The silicon is used as the substrate, and as the active p- and n-channels when heavily doped with boron and phosphorus. The oxide is grown thermally under dry (gate oxide), and wet (field oxide) conditions. The fabrication process involves oxidation, conformal material deposition (such as LPCVD nitride, and polysilicon), photolithography, etching (dry and wet), and diffusion (ion-implantation, and drive-in). These processes are consistent with the planned set of construction operators for the Structure Simulator.

A complete description of the MIT baseline CMOS has been generated using PFR. The hierarchical structure of the CMOS baseline process, which requires about 1800 lines of PFR text, is shown in Fig. 4. Shown in bold are the steps modeled by the structure simulator in the example below. The STS uses two types of information from the PFR. The "`treatment`" describes the process step and is used for simulation and for the Process History File. The corresponding "process effect" is described in the `change-wafer-state` entry and is used in forming the 3-D geometry. A PFR fragment for the `stress-relief-oxide` operation, shown in Fig. 5, illustrates these details.

The general CAD architecture is exercised for the first four process steps. The result of these process steps is the n-well region ready for ion implantation. The specific process steps are: RCA clean and stress relief oxidation (420Å), LPCVD nitride deposition (1500Å), photolithography (for well definition) and plasma etching of the exposed nitride. The next step, phosphorus implantation, is beyond the scope of our current Structure Simulator implementation.

The Structure Simulator was used to generate solid models after each process step in PATRAN Neutral File Format. The material name, process name, and process parameters are attached to each layer and are written in the Process History File. The Process History File is interfaced with the MPS and a set of material properties are extracted from the object oriented database. The material properties are written into a PATRAN-readable file but, at present, the properties must be manually associated with individual geometries.

At this point, PATRAN can be used to visualize the solid model. The model can then be meshed, material properties attached to geometries, and loads and boundary conditions applied to create a complete FEM model. The model is then translated into an suitable input file for FEM analysis. After analysis the result can be translated into a PATRAN-readable form and PATRAN can be used to visualize the results.

## 5 Conclusion

A CAD architecture for microelectromechanical systems is presented in which conventional mask layout and process simulation tools are linked to three-dimensional mechanical CAD and finite-element tools for analysis and simulation. The architecture is exercised at an elementary level by modeling the first steps of the MIT baseline CMOS process. An architecture for an object-oriented material property

```
(define stress-relief-oxide
  (flow
   (:doc "Stress Relief Oxide.  Purpose is to
        minimize the stress effects of nitride
        deposition and processing. Operation
        consists of a clean, a furnace step, and
         an inspection.")
   (:body rca-clean
    (flow
     (:doc "SRO furnace processing")
     (:change-wafer-state
      (:oxidation :thickness
                 (:angstroms (:mean 430 :range 20))))
     (:treatment
      (furnace-rampup-treatment
       :final-temperature (:mean 950 :range 10))
      (furnace-dryox-treatment
       :temperature 950 :time (:minutes 100))
      (furnace-rampdown-treatment
       :start-temperature 950))
     (:time-required (:hours 5 :minutes 0))
     (:machine GateOxTube)
     (:settings :recipe 210))
    (inspect-thickness :where "Center Wafer"
                       :film-type "oxide"
                       :machine "ellipsometer"))))

(define (furnace-rampup-treatment final-temperature)
  (sequence
   "Includes push-in, stabilization, thermal ramp
    from 800 to the peak temperature (the parameter),
    and a stabilization at that peak temperature."
   ;;Push-In
   (:hightemp :temperature 800
              :time (:minutes 20) :ambient :N2)
   ;;Stabilization
   (:hightemp :temperature 800
              :time (:minutes 10) :ambient :N2)
   ;;Ramp-Up
   (:hightemp :temperature 800 :ambient :N2
              :time (:minutes (/ (- final-temperature
                                    800) 10.0))
              :temp-rate 10)
   ;;Stabilization
   (:hightemp :temperature final-temperature
              :time (:minutes 10) :ambient :N2)))

(define (furnace-dryox-treatment temperature time)
  (:hightemp :temperature temperature
             :time time :ambient :O2))
```

Figure 5: PFR definitions for the stress-relief-oxide operation.

simulator is shown in which material properties and their process dependence are stored and are accessed based on the specific process conditions.

It is anticipated that this architecture, which links the process dependence of material properties with a solid modeling capability, will find broad applications in conventional microelectronic device design, in device packaging, and related fields.

## Acknowledgments

This work is supported in part by the Department of Justice under contract J-FBI-88-067 and DARPA under contract MDA972-88-K-0008 and is carried out in the Microsytems Technology Laboratories of the Massachusetts Institute of Technology.

## References

[1] M. B. McIlrath and D. S. Boning, "Integrating semiconductor process design and manufacture using a unified process flow representation," in *Proc. 1990 Intl. Conf. Computer Integrated Manufacturing*, May 1990.

[2] G. M. Koppelman, "OYSTER, a three-dimensional structural simulator for microelectromechanical design," *Sensors and Actuators*, vol. 20, pp. 179–185, Nov. 1989.

[3] F. Maseeh, R. M. Harris, and S. D. Senturia, "A CAD architecture for microelectromechanical systems," in *Proc. IEEE Micro Electro Mechanical Systems*, (Napa Valley, CA), pp. 44–49, 1990.

[4] R. M. Harris, F. Maseeh, and S. D. Senturia, "Automatic generation of a 3-D solid model of a microfabricated structure," in *Tech. Dig. IEEE Solid-State Sensor and Actuator Worshop*, (Hilton Head, SC), pp. 36–41, 1990.

[5] M. L. Heytens and R. S. Nikhil, "GESTALT: An expressive database programming system," *ACM SIGMOD Record*, vol. 18, pp. 54–67, Mar. 1989.

[6] M. Stonebraker, G. Wong, P. Kreps, and G. Held, "The design and implementation of INGRES," *ACM Transaction on Database Systems*, vol. 1, no. 3, pp. 189–222, 1976.