Fault Model Generation for MEMS

Abhijeet Kolpekwar, Chris Kellen and R. D. (Shawn) Blanton

Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213-3890

ABSTRACT

Most MEMS applications demand extraordinary levels of product reliability. This results in the need to design a comprehensive testing methodology for MEMS. Effectiveness of any testing methodology depends on the accuracy of fault models utilized. Our goal in this work is to systematically generate fault models for microelectromechanical systems. We have developed a tool called CARAMEL (Contamination and Reliability Analysis for Microelectromechanical Layout) that performs inductive fault analysis of MEMS. This facilitates studying a large spectrum of MEMS faulty behavior that leads to the generation of MEMS fault models. Here, we describe CARAMEL and its use in performing contamination analysis of MEMS. The effectiveness of CARAMEL is illustrated by the generation of defective three-dimensional micromechanical structures caused by process contaminations.

Keywords: Fault modeling, MEMS testing, inductive fault analysis, fault diagnosis

INTRODUCTION

Most MEMS applications demand extraordinary levels of product reliability. The intent of this work is to develop a comprehensive MEMS testing methodology to ensure high quality of MEMS-based products. The effectiveness of any testing methodology depends on the accuracy of the fault models applied. So, the first phase of our work focuses on generating accurate fault models for MEMS. Creating accurate fault models requires a complete knowledge of all the possible failure mechanisms in MEMS. The effect of these failure mechanisms is then analyzed by evaluating their impact on the final functionality of the product. Thus, a broad spectrum of faulty MEMS behavior needs to be evaluated. Our industry interaction has shown that most faulty MEMS behaviors result from spot contaminations introduced into the fabrication process. Thus, our approach to MEMS fault model generation involves performing manufacturing process and low-level electromechanical simulations using realistic contamination data. A large number of contamination simulations can be performed to produce a wide spectrum of faulty MEMS behavior which can then be mapped to appropriate fault classes. We believe such fault characterization will lead to effective testing strategies, design for testability techniques and fault diagnostic capabilities for MEMS.

Figure 1 depicts our method for contamination analysis for MEMS. There are four important phases defined in this flow.

1. **Process Simulation:** This phase involves simulating the complete fabrication process with arbitrarily introduced contaminations.

2. **Structure Extraction:** This phase involves extracting a three-dimensional (3D) micromechanical structure that may have been damaged due to contamination effects. This phase produces a netlist for mechanical simulation.

3. **Mechanical Simulation:** This phase performs mechanical simulation to evaluate the mechanical parameters of the extracted micromechanical structure.

4. **Parameter Analysis:** This (manual) phase involves evaluation of faulty MEMS behavior and a systematic fault categorization.

We have developed a tool CARAMEL (Contamination And Reliability Analysis for MicroElectromechanical Layout) that automates the contamination analysis flow described above. CARAMEL accepts fabrication process information, design layout and contamination information and analyzes the impact of contaminations on the operational parameters of MEMS. We have used the surface micromachined microresonator as our research vehicle to carry out contamination simulations using CARAMEL. The examples discussed later in this paper illustrate the effective use of CARAMEL for generation and evaluation of defective 3D micromechanical structure caused by process contaminations.

The rest of this paper is organized as follows. The next section presents a detailed description of CARAMEL and its algorithms. The section on simulation results illustrates the use of CARAMEL on several defective resonators. The last section outlines our conclusion and future work.

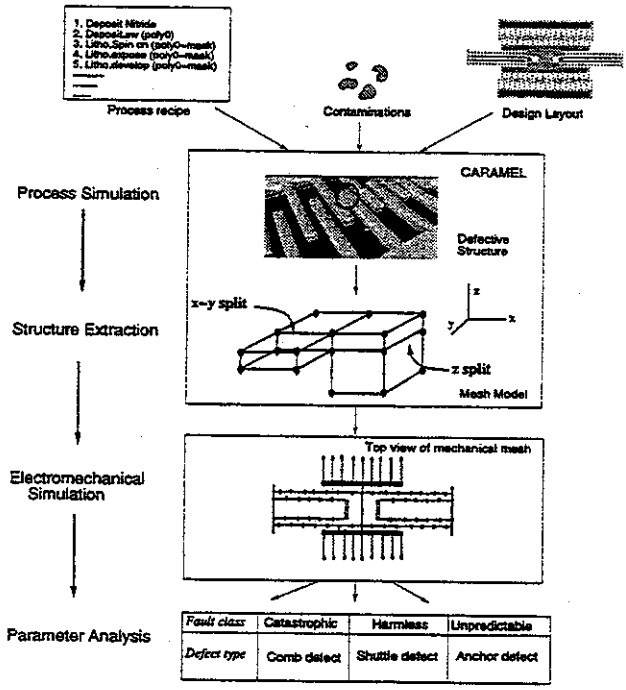

Figure 1: MEMS contamination analysis performed by CARAMEL.

CARAMEL

CARAMEL is a process simulator that maps particle contaminations to defective microstructures. It is constructed around the tool CODEF, which is a contamination-to-defect-to-fault mapper for electrical layouts [1]. This section presents the implementation details of CARAMEL and explains its operation.

Process Simulation

This phase of CARAMEL maps spot contaminations to layout defects and is implemented using a modified version of CODEF. CODEF determines the impact of a spot contamination on a portion of an IC layout. It accepts layout information, a process description, and contamination statistics for each processing step of interest. CODEF allows for a characteristics of contamination particle to be simulated including the particle's size, density, conductivity, and fabrication step of introduction. Given the contamination parameters, it simulates all the fabrication steps and creates a 3D structure of the defective device. CODEF, in its unmodified form, is used to analyze the effects of particles on the electrical properties of an IC layout. It utilizes a circuit extractor which traverses the 3D structure to create a SPICE netlist. Defective circuit behavior resulting from contaminations can then be analyzed through SPICE simulations.

To make use of CODEF for MEMS contamination analysis, we have defined a complete MUMPs fabrication process as a sequence of steps in the PREDITOR format [4] [2]. The MUMPs process is used to form micromechanical structures comprised of thin films on the surface of the substrate. These thin-film microstructures are called surface micromachined MEMS. A simplified version of the MUMPs process is described in [5]. In the MUMPs process, a low-stress silicon nitride is deposited first on the silicon substrate to provide electrical isolation between microstructures. An electrical interconnection layer of polysilicon is then deposited and patterned. Next, a 2 μm-thick layer of phosphosilicate glass (PSG) is deposited. PSG acts as sacrificial spacer layer for the microstructures. After contact cuts are made in PSG, a 2 μm-thick layer of polysilicon is deposited and patterned to form a microstructure. A final wet etch in hydrofluoric acid (HF) dissolves the PSG and releases the microstructure so that it is free to move. Contact cuts in the PSG act as anchor points that fix the microstructure to the substrate.

In CARAMEL, we have modified CODEF to handle the MUMPs "release" step (i.e. the MUMPs step that produces free-moving electromechanical structures) as described above. Adding this step, which is not part of any standard CMOS process, allows us to perform MUMPs process simulations using realistic contaminations.

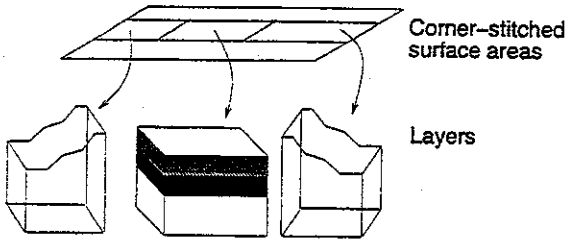

Figure 2: Organization of the Chip DataBase (CDB) of CODEF.

Structure Extraction

The structure extraction phase of CARAMEL produces a 3D mesh representation of the defective microelectromechanical structure generated by process simulation. Mechanical simulation of the mesh (using finite element analysis (FEA) tools like ABAQUS [8]) allows analysis of contamination effects on the mechanical structure.

Process simulation creates a 3D representation of the microstructure consisting of hierarchically connected layers of material represented by the Chip Data Base (CDB) [6]. CDB represents the surface (top-view) of the chip as a set of different regions. Each region is composed of a number of layers. A layer has a material type and a layer model, which determines the shape of its surface. Figure 2 illustrates the organization of CDB. The regions are required to be rectangular when viewed from above but layer surfaces can have arbitrary shapes. This allows one to represent any 2D region accurately. CDB is constructed from the MUMPS process flow and the CIF layout of the MEMS design. In addition, contamination particles may be introduced at arbitrary locations in the process flow, under control of the process simulator. CARAMEL creates a separate "mesh database" from the CDB to represent the MEMS structure as a set of 2D corner-stitched rectangular regions [7]. Each region of the mesh consists of a set of material elements. However, the mesh database differs from a traditional corner-stitched database because elements in the former are allowed to be less than maximal width. This is a necessary feature since the mesh used for mechanical simulation requires that each region either have a single neighbor or no neighbor along each of its edges. Each element also has associated data that describes its material characteristics, its position along the z axis, and a flag that indicates if the base of the element is anchored.

The MEMS meshing phase:

- Creates a corner-stitched database containing the regions to be simulated.
- Determines region connectivity.
- "Splits" the mesh database to ensure that each region has only one or zero neighbors in the x, y, and z directions.
- Identifies the elements and generates node numbers for all vertices of each region. (This is required by the FEA tool for mechanical simulation.)
- Generates the final mesh input model for the FEA tool.

Mesh Database Creation

The mesh database for the micromechanical structure consists only of polysilicon (POLY1) and defect material, i.e., the material of the particle contamination. The mesh is created by examining every region of CODEF's CDB. Each element of POLY1 or defect material is added to the mesh database along with a flag indicating if the element is fixed to some other material or contains free space underneath. The flag information is used later to determine which elements are free to move.

Determining Element Connectivity

Understanding the behavior of the mechanical structure requires mechanical simulation of its moving parts. For example, the moving parts of the resonator includes the mass shuttle and everything connected to the shuttle. Because a particle contamination can alter the topography of the resonator, the resulting defective structure must be derived from the CDB of elements. In this phase, CARAMEL determines all the connected elements from a user-specified element on the resonator's layout. A simple algorithm is employed that marks the user-specified element and continues to mark neighboring elements until all connected regions are marked.

Element Splitting

The mechanical mesh required by FEA tools must be constructed so that adjacent regions meet only at region vertices (nodes). CARAMEL must meet this adjacency constraint not only in the x and y directions but also in the z direction.

- **x-y splitting:** In a standard corner-stitched database, a region edge can have multiple neighboring regions or a partial neighbor as shown in Figure 3a. Such a situation violates the constraint described above for the mechanical mesh. The x-y splitting operation modifies an element so that every region edge has a single neighboring element or no neighbor at all. Splitting is performed by analyzing the edge neighbors of each region in the CDB. For each edge that violates the constraint, the current region is split at the neighbor's edge. (See Figure 3b.)

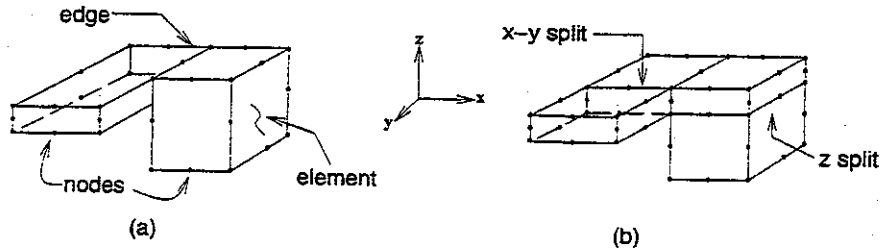

Figure 3: Illustration of element splitting for mechanical mesh construction: (a) A set of two regions before split and (b) the resulting four regions after x-y and z splits.

- **z splitting:** In an analogous way, elements may need to be split in the z direction as well. In this case, adjacent regions that have elements of differing heights must be split in the z direction so that the resulting elements share nodes. This is accomplished using a simple algorithm that compares the top and bottom coordinates of an element with all its neighboring elements. If an adjacent element is contained in the current element, the current element is split. (See Figure 3.) There is one exception to the z-split operation when the element of concern is fixed. Typically, split elements inherit identical properties (material, conductivity, etc.) from the original element. In the case of fixed elements, only the newly-created bottom-half element is fixed.

The mesh database thus created can now be used to generate a netlist for FEA tools.

Bookkeeping of Nodes

Every region in the mesh database is visited and each element which is a part of moving structure is recorded in an ABAQUS input file. For each element in the mesh database, twenty nodes are identified. Nodes are located at the eight corners and the midpoints of twelve edges. Figure 3 shows such elements with twenty nodes each. Each node is a point in three-dimensional space with a unique number assigned to it. To simplify the labeling and bookkeeping of nodes, the NodeSet data structure was created. This data structure ensures an assignment of a unique node number for the given x, y, and z coordinates. It may be noted that a single node can be shared by more than one element. The routines associated with node handling are : *Node_GetNumber*(), *Node_GetInfo*() and *Node_Free*().

Node_GetNumber takes the x, y, and z coordinates, as well as the fixed flag, and returns the node number for that coordinate. It works by looking for a node with the given coordinates in the list of current nodes. If it does not exist, a new node is created with the given coordinates and fixed flag status. If it does exist, the fixed flag is updated and the node number is returned. Updating the fixed flag consists of marking the node as fixed if any of the elements sharing it is fixed. *Node_GetInfo* is used to get the coordinate and fixed flag status about a given node. This is used to write the node information in a file. *Node_Free*() frees all of the memory used by the NodeSet data structure.

Mechanical Simulation

By using the *Node_GetInfo* routine, the information for each node is used to generate a netlist for FEA analysis. Mechanical simulation is performed using the commercial FEA tool ABAQUS [8]. The generated netlist can then be used to evaluate various parameters like resonant frequency, stiffness, etc. For example, to compute resonator stiffness, a center node of the resonator is displaced by a few μms. ABAQUS reports the restoring force produced at the center node due to this displacement. The ratio of restoring force and the displacement can be computed to give the stiffness of the structure.

The result of CARAMEL's extraction phase is a mechanical mesh that is directly compatible with the FEA tool ABAQUS. The resulting mechanical mesh captures the impact of the particle contamination on the mechanical properties of the resonator. It should be noted here that CARAMEL is a general tool and is therefore not restricted to the resonator but is applicable to any generic MEMS layout. It is therefore possible to analyze the impact of particle contaminations on a wide variety of MEMS topologies. This feature makes CARAMEL quite useful for a variety of electromechanical structures.

SIMULATION RESULTS

We have conducted contamination simulations using the surface micromachined microresonator using CARAMEL. The microresonator is a single-polysilicon structure and utilizes only a subset of the complete three polysilicon MUMPs process defined in [2]. Figure 4a shows the top view of the microresonator. It is anchored at the four points shown and is free to move in the y direction. The comb drive is used to create a differential capacitance arrangement for sensing acceleration in the

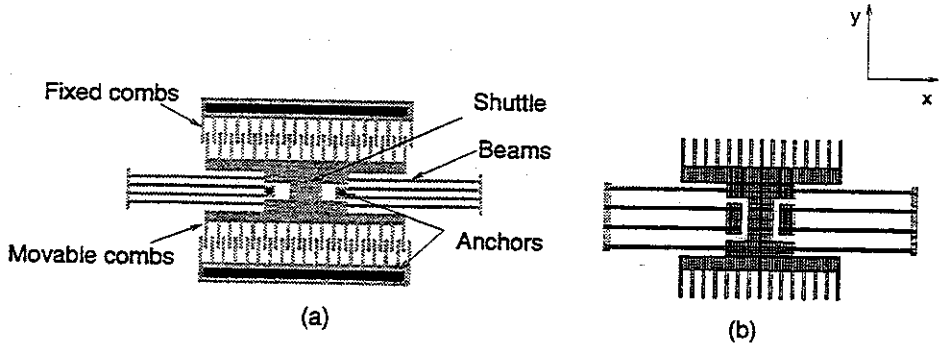

Figure 4: Top view of Comb-drive microresonator and corresponding defect-free mesh.

y direction. The folded flexure (consisting of the long beams on either side of the shuttle) provides a restoring force for shuttle movement. Figure 4b shows a mesh model for a defect-free resonator. Mechanical simulation of a defect-free resonator requires only the moving parts, that is, the shuttle and flexure.

We now describe some interesting results generated by CARAMEL. The MUMPs process steps mentioned here are described in [2].

- **Comb Defect:** A comb defect occurs when a contamination is located between adjacent comb fingers during POLY1 deposition. The process simulation phase of CARAMEL reveals that the contamination welds together the two normally-moving fingers. The 3D illustration of a comb defect generated by CARAMEL is shown in Figure 5a. Such a connection transforms the two comb drives into a single structure thereby changing the mesh model required for mechanical simulation. Figure 5b shows a mesh model for the comb defect. Notice the fixed combs are now part of the mesh model. The result obtained from mechanical simulation shows a 33% increase in resonant frequency, an indication that this defect has caused a catastrophic failure.

- **Beam Defect:** If a contamination lodges on a flexure beam after the PSG deposition, the result is a slightly heavier beam. The increased mass of the beam affects the resonant frequency. Meshing complexity of the area surrounding the defect's location increases. The increased "meshing" is a reflection of the number of splits required to accurately represent the region containing the defect. However, mechanical simulation results do not show any significant change in resonant frequency.

- **Other Defects:** Other contamination experiments performed using CARAMEL include: shuttle defects where contaminations are encapsulated by the shuttle mass; harmless defects where a contamination does not affect any of the moving parts of the resonator; anchor defects where a contamination becomes sandwiched between the substrate and the resonator thereby providing an undesired anchor.

The above results demonstrate CARAMEL's ability to model and simulate a variety of contamination effects on the MEMS microstructure. The tool can be effectively used to generate several thousand contamination simulations under a Monte-Carlo mode of operation. In this mode, contaminations are introduced into the process at random steps, with random sizes, and at random locations in the layout. Such an analysis can produce a full spectrum of MEMS failure modes. The observed deviations in behavior can then be systematically categorized under various fault classes.

CONCLUSION & FUTURE WORK

We have automated the contamination analysis process for microelectromechanical layout using our tool CARAMEL. CARAMEL can be effectively used to directly investigate the faulty behavior of defective MEMS structures. Here, we have illustrated CARAMEL's use on a folded-flexure resonator. Simulation of different defects indicates that spot contaminations can have a significant impact on the behavior of the resonator. Moreover, the results show that the impact of a particular defect is highly dependent upon *where* it is located in the resonator's layout and *when* it is introduced into the manufacturing process. Currently we are addressing the following issues to improve CARAMEL for MEMS fault model generation:

- **Mesh Optimization:** Presently, the mechanical mesh generated by CARAMEL is not optimized in anyway. It contains many artifacts from the original process simulation that add to the size

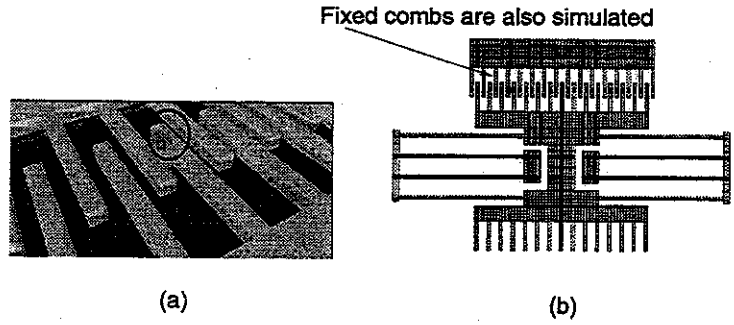

Figure 5: 3D representation of a resonator with a comb defect and its corresponding mesh.

and complexity of the mesh. We plan to remove these artifacts from the mesh in order to accelerate the mechanical simulation phase.

- **Automatic Fault Categorization:** We plan to have CARAMEL sort through the large amount of simulated data and extract the unique defects. This will reduce the amount of mechanical simulation required for fault categorization.
- **Simulation Complexity:** We are currently simulating static structures that do not change during the course of simulation. Static structures do not capture a number of interesting and possibly important defects. For example, a particle located between but not touching the comb finger may limit the motion of the resonator. This and other types of interactions will require more knowledge of the material properties of the contaminants but can be modeled using our approach.

REFERENCES

[1] J. Khare and W. Maly, "From Contamination to Defects, Faults and Yield Loss," Kluwer Academic Publishers, Boston, 1996.

[2] A. Kolpekwar, R. D. Blanton, "Development of MEMS Testing Methodology," Proc. of International Test Conference, pp. 923-931, Nov. 1997.

[3] D. A. Koester, R. Mahadevan and K. W. Markus, "Multi-user MEMS Processes (MUMPS) Introduction and Design Rules," http://mems.mcnc.org/mumps.html, Oct. 1994.

[4] D. M. H. Walker, C. S. Kellen and A. J. Strojwas, "The PREDITOR Process Editor and Statistical Simulator," International Workshop on VLSI Process and Device Modeling, pp. 120-123, May 1991.

[5] G. K. Fedder and T. Mukherjee, "Physical Design for Surface-Micromachined MEMS," In Proc. of the 5th ACM/SIGDA Physical Design Workshop, pp. 53-60, April 1996.

[6] D. M. H. Walker, C. S. Kellen, D. M. Svoboda, and A. J. Strojwas, "The CDB/HCDB Semiconductor Wafer Representation Server," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol.12, No.2, pp. 283-95, Feb. 1993.

[7] J. K. Ousterhout, "Corner Stitching: A Data-Structuring Technique for VLSI," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 3, No. 1, pp. 87-100, Jan. 1984.

[8] Hibbit, Karlsson & Sorensen, Inc., "ABAQUS User Manual," Vol. 2, Pawtucket, RI, 1995.