Copyright © 1992, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

MTCAD: AN INTEGRATED FRAMEWORK OF MANUFACTURING EQUIPMENT MODELS AND *TCAD*

by

Mehdi Hosseini

Memorandum No. UCB/ERL M92/136

17 November 1992

MTCAD: AN INTEGRATED FRAMEWORK OF *M*ANUFACTURING EQUIPMENT MODELS AND *TCAD*

by

Mehdi Hosseini

Memorandum No. UCB/ERL M92/136

17 November 1992

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

MTCAD: AN INTEGRATED FRAMEWORK OF *M*ANUFACTURING EQUIPMENT MODELS AND *TCAD*

by

Mehdi Hosseini

Memorandum No. UCB/ERL M92/136

17 November 1992

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

MTCAD: An Integrated Framework of *M*anufacturing Equipment Models and *TCAD*

Mehdi Hosseini

Electronics Research Laboratory

Department of Electrical Engineering and Computer Sciences

University of California, Berkeley, CA 94720

November 17, 1992

ABSTRACT

Traditional Technology CAD (TCAD) tools lack the ability to describe a specific fabrication line at an adequate level of detail. This is due to the fact that various fabrication steps are simulated at the abstract "process" level, without taking into account the effects of specific manufacturing equipment. However, as process technologies become more complex, the need to predict the manufacturability of an IC design becomes more important. This prediction requires modeling at the lowest possible level of abstraction. To address this problem, we have integrated the latest version of SIMPL-IPX, a TCAD

framework, with the Berkeley Computer Aided Manufacturing (BCAM) system, and have created a prototype unified framework we call Manufacturing TCAD (MTCAD). MTCAD provides the designer with access to TCAD tools, such as process and device simulators, as well as CAM utilities, such as a recipe management system and rigorous, statistically based equipment models. TCAD simulators and equipment models require significantly different parameters in their operation. To address this problem, we have created a family of intermediate models. These intermediate models form a "wrapper" that provides full access to all of the capabilities of the continuously updated standard and adaptive equipment models of the BCAM system. This wrapper enabled us to create modular links between SIMPL-IPX and BCAM.

Acknowledgments

I thank Professor Costas J. Spanos, my research advisor, not only for his valuable research guidance, but also for the encouragement and the light his way of thinking and working shined on my project. I am also indebted to Professor A. R. Neureuther for reading my project report.

Special thanks go to Tom Luan, for his early work in MTCAD. It was his work indeed that paved the path of my project. During the early stages of this work, Tom Luan and John Thomson provided valuable help in installing the first generation of MTCAD and in pointing to the relevant literature.

Also many thanks to Eric Boskin, Sherry Lee, Hao-Cheng Liu, John Thomson, Lauren Massa-Lochridge, Raymond L. Chen, Soheila Bana, Zeina Daoud, Bart Bombay, and Sovarong Leang whose contributions indeed makes BCAM a wonderful group.

In several cases, Alexander Wong provided valuable insight into the SIMPL-IPX framework and helped me install the latest version of SIMPL-IPX on the radon cluster.

During some stages of this work, Lauren Massa-Lochridge's help to facilitate my full access to the Ingres database was very important.

I am especially thankful to Bart Bombay, since I have used the code he had developed for his "controller", to access the Ingres database. I would like to express my special thanks to Shahram M. Shahruz, and Eric Boskin for proofreading this document.

Finally, I would like to thank my wife, Delnaz Khorramabadi, for her love and support that made this project possible.

The funding of this project by the SRC (92-DC-008) is gratefully acknowledged.

Table of Contents

Chapter 1 **Introduction** page 7

1.0 Overview

1.1 History

Chapter 2 **The Development of a Joint CAD/CAM Framework** page11

2.0 Profile Interchange Format (PIF)

2.1 Semiconductor Wafer Representation (SWR), and Semiconductor Process Representation (SPR)

2.2 Idea of a Central Supervisor Unit

2.3 Use of a Relational Database for Short Term and Long Term Data Storage

2.4 Berkeley Process Flow Language (BPFL)

Chapter 3 **The Major Components of MTCAD** page 19

3.0 An Overview of the SIMPL-IPX TCAD Framework

3.1 An Overview of the BCAM system of Equipment Models

3.1.1 LPCVD of Polysilicon

3.1.2 Plasma Etch

3.1.3 Spin-Coat and Bake of Photoresist

3.1.4 Photolithography: Stepper and Developer

Chapter4 **Connecting SIMPL-IPX to BCAM** page 30

4.0 Different Approaches for the Design of an Interface

4.1 Wrapper: a Generic Translator Encapsulating Equipment Models

4.2 Implementation Issues

4.3 Tool Wrapper versus Equipment Model Wrapper

Chapter 5 **An Application Example** page 42

5.0 Process Simulation of a CMOS inverter

Chapter 6 **Conclusion** page 53

6.1 Summary

6.2 Future Work

References page 55

Appendix page 58

Chapter 1

Introduction

1.0 Overview

Technology CAD (TCAD) tools, which include process simulators, device simulators, and support tools such as input parsers and visualizers, have been used to predict process and device behavior. SIMPL [1], SAMPLE [2], SUPREM [3], and FABRICS [4] are examples of such process and device simulators. However, TCAD tools have not been used extensively to predict design manufacturability, mainly because they lack the ability to describe any fabrication equipment in the necessary level of abstraction. In recent years, there has been a growing effort to create integrated frameworks which allow a process designer to simulate different processing steps needed in the fabrication of an Integrated Circuit [5] [6]. A more recent approach enhances a TCAD framework with links to equipment models that allow hard-wired access to the manufacturing environment [7].

Within the general effort to include the effects of fabrication equipment models into process simulation, this project presents an advanced framework that brings together a TCAD framework and a Computer-Aided Manufacturing system. More specifically, we have incorporated the workstation based SIMPL-IPX TCAD framework [8], with the Berkeley Computer Aided Manufacturing (BCAM) system. SIMPL-IPX was chosen since it has internal simulation programs and a modular structure to call external simulators, and

BCAM was selected because it offers up to date, statistically based equipment models and utilities, such as a recipe management system, a graphic response surface visualization, etc. The BCAM system also supports a public, object oriented library of standard and adaptive equipment models.

With the help of a unified framework such as MTCAD, the process developer can switch freely between TCAD and CAM operations. In this way, the process developer can perform process and device simulation at a higher level of abstraction, and then evaluate the manufacturability by shifting the process description at a lower level of abstraction. This design session will yield not only the description of the device, but also equipment-dependent recipes that are ready to be used on the production line.

This report will provide a detailed description of the components, implementation issues, and application of the MTCAD framework. In the rest of this chapter a brief history of the previous work done in this area is provided. In chapter two, we will introduce the SIMPL-IPX a TCAD framework, and we will present the BCAM equipment model structure. Chapter three discusses the implementation issues involved in the integration of SIMPL-IPX and the BCAM equipment model library. In chapter four, through an application example, we will demonstrate how the MTCAD framework may be used to simulate the fabrication of a CMOS inverter. Chapter five presents a discussion of some of the issues related to the development of a unified CAD/CAM framework. A summary, conclusions, and our perspective of future trends will be presented in chapter six, the concluding chapter of this report.

1.1 History

The notion of integrating a manufacturing system of equipment models and a TCAD framework is not new. In fact, as early as 1986, Hughes and Shott suggested supplement-

ing existing TCAD tools with a library of equipment models [9]. Though this suggestion was never implemented, the problem with such a plan was that it would require a continuous effort to supply and update the equipment models needed to describe today's semiconductor factory lines. One way to overcome this shortcoming has been to enhance a TCAD framework with direct links to the manufacturing environment. However, since a manufacturing environment evolves continuously, such a hard-wiring scheme may not be the most appropriate solution to the problem [7].

In 1990, MacDonald *et al* integrated a process simulator into a commercial CAM system [10]. In that implementation, the simulator was used by the manufacturing engineers for on-line process control. Although this system provides physical insight during production it does not use equipment models and therefore it cannot offer information about manufacturing equipment to process designers.

Currently, the CAD Framework Initiative (CFI) is undertaking the definition of standards that will enable the integration of various tools. The CFI TCAD framework organization consists of three working groups addressing programming interfaces for Semiconductor Wafer Representation (SWR), Semiconductor Process flow Representation (SPR), and Information Modeling [11], [12]. The Information Modeling effort is being pursued by two technical subcommittees within the Electronics Design Interchange Format (EDIF) group: the Device Modeling and Verification (DM&V) group and the Process and Device (P&D) group. In August 1991, the CFI demonstrated a prototype implementation of their standards discoursing very basic operations in process and device simulation [13]. Although CFI has not yet addressed the issue of incorporating the TCAD framework with the Computer Integrated Manufacturing, their effort to standardize TCAD framework development will facilitate any future development of joint TCAD and

CAM environments. Unfortunately, standards that will facilitate this development are still in the future. Therefore, in this work we have implemented the prototype MTCAD framework with the hope that future TCAD/CAM standards can benefit from our experience.

Next we describe the general ideas behind a unified CAM/CAD framework such as MTCAD. Some of these issues are common to any framework architecture, therefore we start the discussion by defining the scope of a CAM/CAD framework.

Chapter 2

The Development of a Joint CAD/ CAM Framework

2.0 Profile Interchange Format (PIF)

The Integration of TCAD tools has long been a major problem. Each tool uses a different input and output format, leaving the job of data exchange between the tools up to specialized translators. Naturally, along with the number of tools, the number of the specialized translators increases. For complex TCAD frameworks, therefore, numerous translators might be needed. Probably, the most comprehensive answer to tool integration problem, using the above approach, has been the use of standard data format, such as, for example, the Profile Interchange Format (PIF)[14]. Due to varying needs of different tools, usually, developing a common format for profile specification is not a trivial task. Yet, the PIF specifications are general enough to cover a wide range of profiles. However, it is usually too complex to realize a full implementation of the specifications exactly as described. The PIF concentrated on specification for cross-sectional profile information. Now it is believed that this approach is not effective since (a) it requires agreement on an unreasonably low level of detail; (b) evolving the format requires extensive changes to the tools that use them; (c) developers must create their own data manipulation software since no functionality is presently available in the public domain.

Currently, two modes of PIF are available: APIF, and BPIF. As the names imply, APIF uses ASCII representation for sending data, while BPIF transfers data between tools in binary format. When the simulation tools use a common database, the BPIF becomes a more efficient way of exchanging data.

2.1 Semiconductor Wafer Representation (SWR) and Semiconductor Process Representation (SPR)

Because of the difficulties of fulfilling the requirements introduced by PIF, other alternatives have been under development. A recent and more comprehensive solution has been concentrated on the idea of a Semiconductor Wafer Representation (SWR) and Semiconductor Process Representation (SPR). SWR and SPR use an object-oriented approach to describe wafer data. Also, in order to take advantage of X-window system, CFI has chosen to use the *client/server model* (Figure 1). In this model, the *wafer state* is maintained by a SWR server of sufficient functionality and efficiency, such that individual tools do not need internal representation.

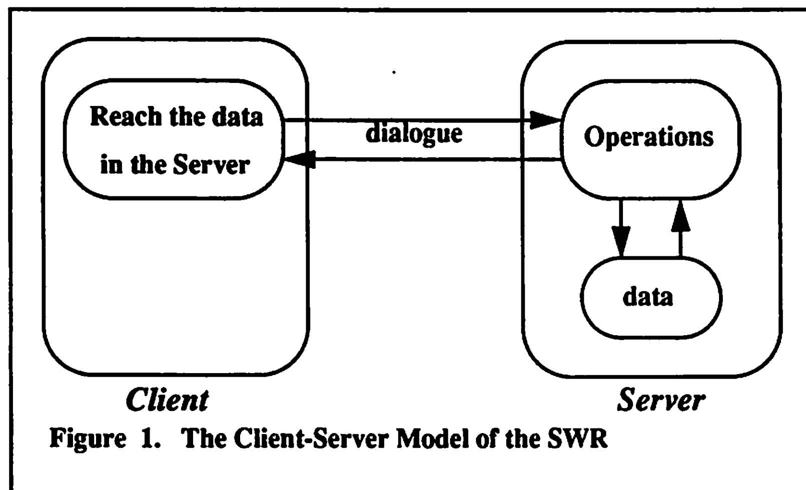

Figure 1. The Client-Server Model of the SWR

The *wafer state* consists of *fields (meshes), geometry,* and *attributes* (Figure 2). When SWR is used, tools do not need their own internal data structures for meshes, geometry, or

attributes. SWR data structure seems to be efficient and general enough to encapsulate various simulation tools, and extract necessary information out of participating tools. Therefore, the problem of data conversion between different formats during tool communication would be eliminated through the use of a common server [15].

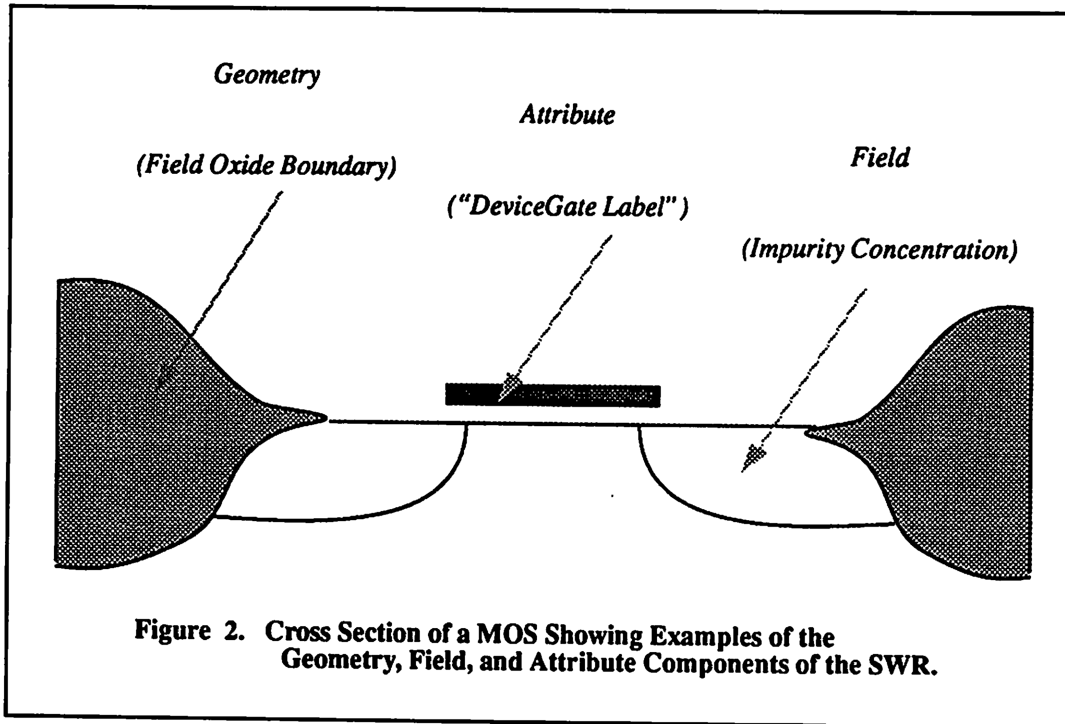

Figure 2. Cross Section of a MOS Showing Examples of the Geometry, Field, and Attribute Components of the SWR.

The SWR server architecture is partitioned into a geometry component, a field component, and a utilities library. Some examples of geometry objects are *point, edge, face, cell,* and *cell-complex*. Typical operations of a geometry server include: *create, delete, access,* and *section*. The geometry and field server need to work together to provide client applications with a consistent view of the structure being represented [15]. The prototype organization of a CFI compliant framework is given in Figure 3. Implemented in this prototype is an inter tool communication mechanism. Given a certain set of simulation tools,

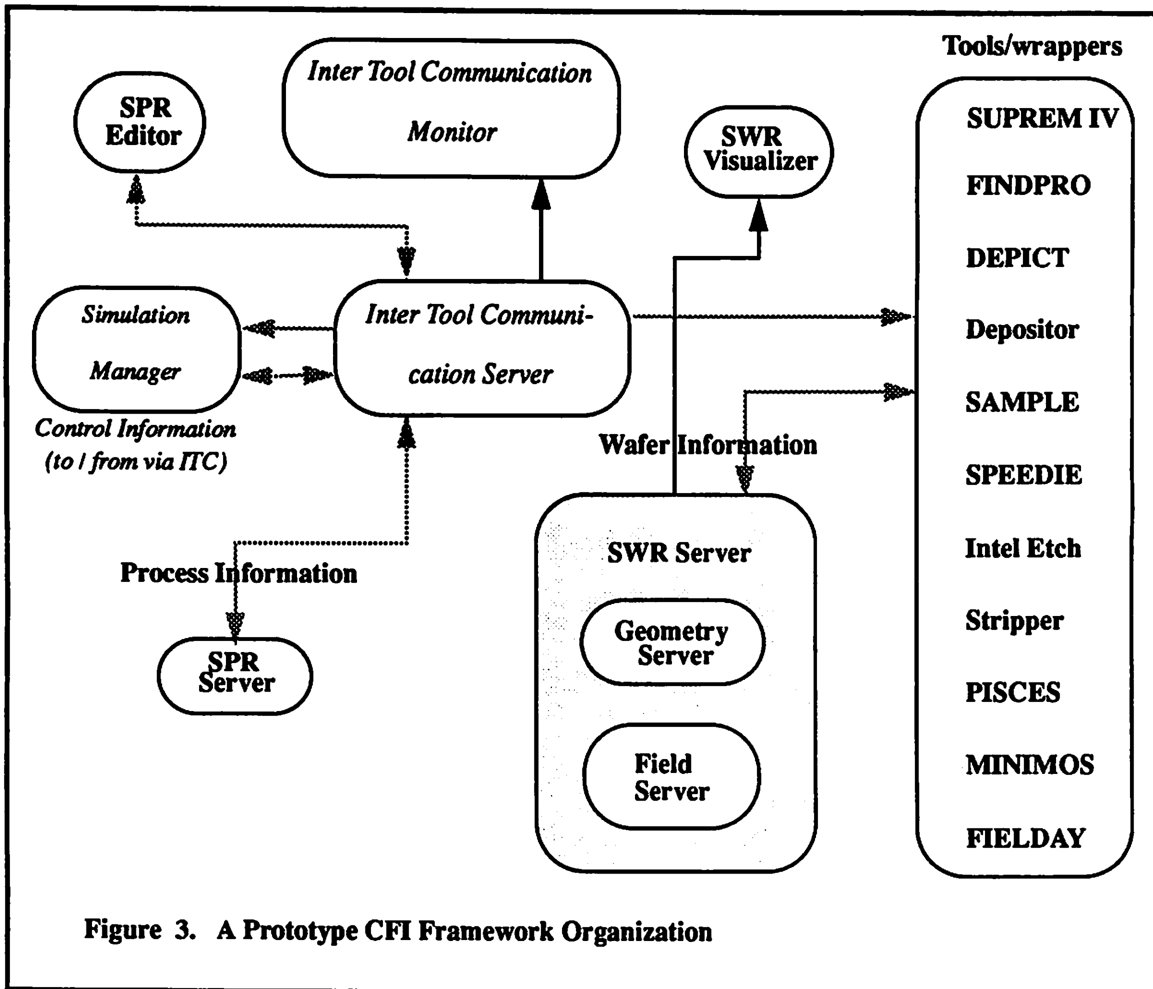

Figure 3. A Prototype CFI Framework Organization

this mechanism connects the tools in a regulated and convenient manner. Also, through an inter tool communication monitor, the user can interactively track the tools that are involved in a sequence of processing steps.

2.2 Idea of a Central Supervisor Unit

Critical to any framework architecture is the ability of the tools to communicate. This could be accomplished by having a central supervisor unit. A central supervisor unit will monitor the data abstraction, and message passing interactions among different tools. Most of the today's frameworks suffer from numerous translators required for the com-

munication of tools. Translation is unavoidable so long as simulation tools are not written in the same programming language and are not supported by identical data structures. This becomes more problematic as the number of simulation tools increases. Implementing a central supervisor unit will eliminate numerous translation interfaces between individual tools.

The existence of a central supervisor unit might not be critical for a framework with only few simulation tools. However, as the number of tools increases, the communication between tools becomes more important, as shown in Figure 4. In fact, the problem arises

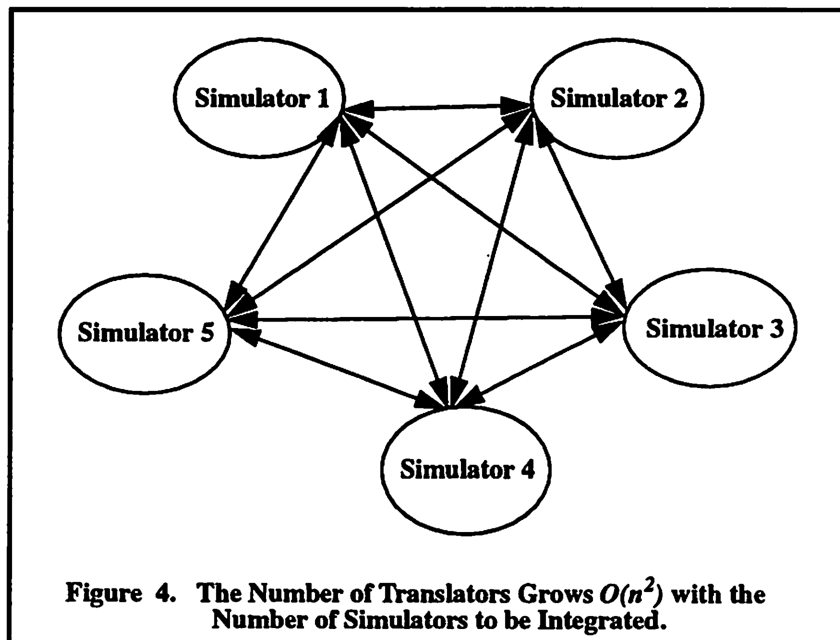

Figure 4. The Number of Translators Grows $O(n^2)$ with the Number of Simulators to be Integrated.

when the tool translators are required to provide tool communication. Common examples of this situation are the cases of the translation required between simulation tools written in different programming languages[1], and/or between the tools with different data structures[2].

1. Simulation tools source codes are typically written in FORTRAN, LISP, C, and C++, etc. with increasing popularity of C and C++.

2. SAMPLE and SUPREM IV have a mesh-based data structure, whereas CREEP has a node and line segment data structure.

In summary, the most compelling reason for using a central supervisor unit is to reduce the number of translators which in turn, will reduce the amount of required translation time between n tools from $O(n^2)$ to $O(n)$. Another important reason is to avoid loss of data consistency due to incomplete representations and/or poor translation between tools. Figure 5 shows the inter-tool communication using a central supervisor unit.

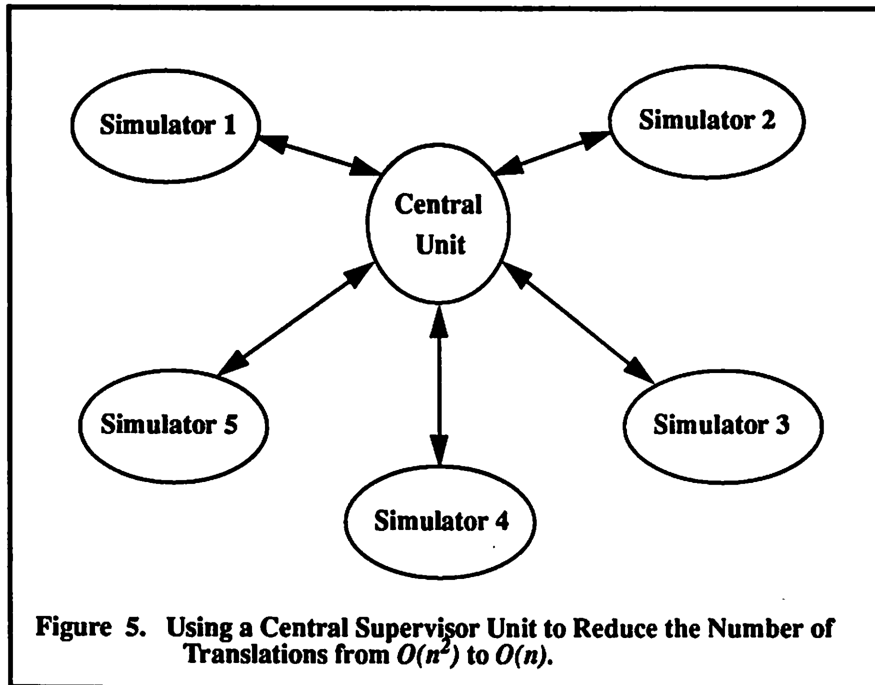

Figure 5. Using a Central Supervisor Unit to Reduce the Number of Translations from $O(n^2)$ to $O(n)$.

It should be mentioned here, that lack of a central supervisor unit, as modeled by Figure 5, is probably the most important problem that SIMPL-IPX is suffering from. Of course, this might not be a big problem, as long as there are not more than a few simulation tools involved in the framework.

2.3 Use of a Relational Database for Short Term and Long Term Data Storage

To help in the ongoing derivation of equipment models and model applications, the BCAM group has developed an object oriented library that uses the Ingres database for

persistent storage of equipment models. Within the Ingres database, there are sub-databases such as "bcam", and "bcamdev". In these databases, the models are stored in relational tables. The table lookup format provides the user with easy access to the desired equipment model and relevant information regarding that model. This information could be equipment settings, equipment input/output relations, etc. For example, the "equipment_index" table provides the name of equipment models available, and the "equipment_io" table provides information about input/output parameters, units, etc.[1]

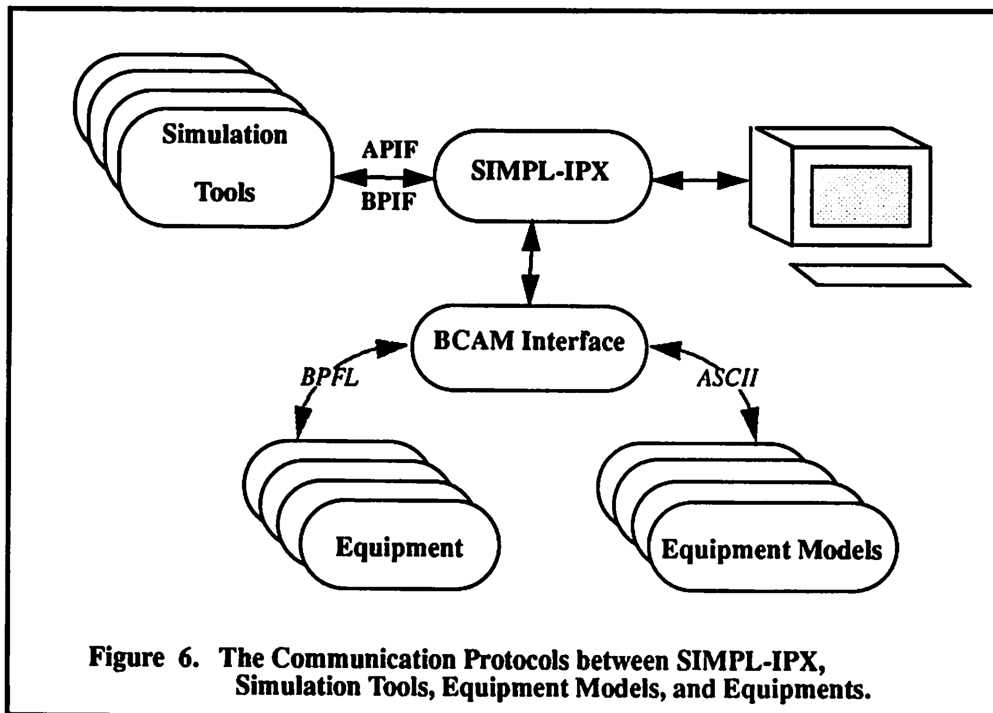

Figure 6. The Communication Protocols between SIMPL-IPX, Simulation Tools, Equipment Models, and Equipments.

2.4 The Berkeley Process Flow Language (BPFL)

Process Flow Representation is another area where TCAD and CAM intersect. It is very desirable to have the same process flow description for both the actual and simulated

1. For a more comprehensive description of Ingres Table Formats, please refer to B. Bombay, "The BCAM Control and Monitoring Environment", pp 52-59 [16].

manufacturing process. The Berkeley Process Flow Language (BPFL) is a formal description medium that can derive both TCAD data, e.g., mask data for pattern transfer steps, as well as BCAM data, e.g., equipment model and wafer tracking. Figure 6 gives an structural picture of BPFL with respect to SIMPL-IPX and Design Manufacturing.

Chapter 3

The Major Components of MTCAD

MTCAD consists of two major components: a Technology CAD (TCAD) framework (SIMPL-IPX), and the Berkeley Computer Aided Manufacturing (BCAM) library of equipment models. In the following, we will briefly highlight the critical aspects of the two constituent components of MTCAD.

3.0 An Overview of the SIMPL-IPX TCAD Framework

SIMPL-IPX (*SIM*ulated *P*rofile from *L*ayout-*I*ntegrated *P*rocess Simulation with *X*-windows) is an integrated process simulation framework. SIMPL-IPX is engineered to help the circuit designer obtain physical insight of the structure being developed from the layout of a circuit. SIMPL-IPX, itself, has two major components: SIMPL-DIX and SIMPL-2 [17]. (Figure 7)

(i) SIMPL-DIX (*SIM*ulated *P*rofile from *L*ayout-*D*esign *I*nterface in *X*-windows), is a workstation based design interface with graphical options to be used with SIMPL-2. SIMPL-DIX allows the user to generate two dimensional cross sectional profiles from the layout. SIMPL-DIX takes a CIF (Caltech Intermediate Format) file as the input. The CIF file ("file.*cif*") contains all the information about a specific mask layout. In addition to a CIF file, SIMPL-DIX will look for a technology pattern file ("file.*pattern*") to get the nec-

essary information about the patterns and the colors that will be used to draw the cross sectional view of the wafer on the display. The pattern file provides all the information needed to draw the color patterns of the mask layout. In order to invoke SIMPL-DIX, the user is expected to specify both the CIF file and the pattern file in order to obtain a colored cross sectional view of the wafer.

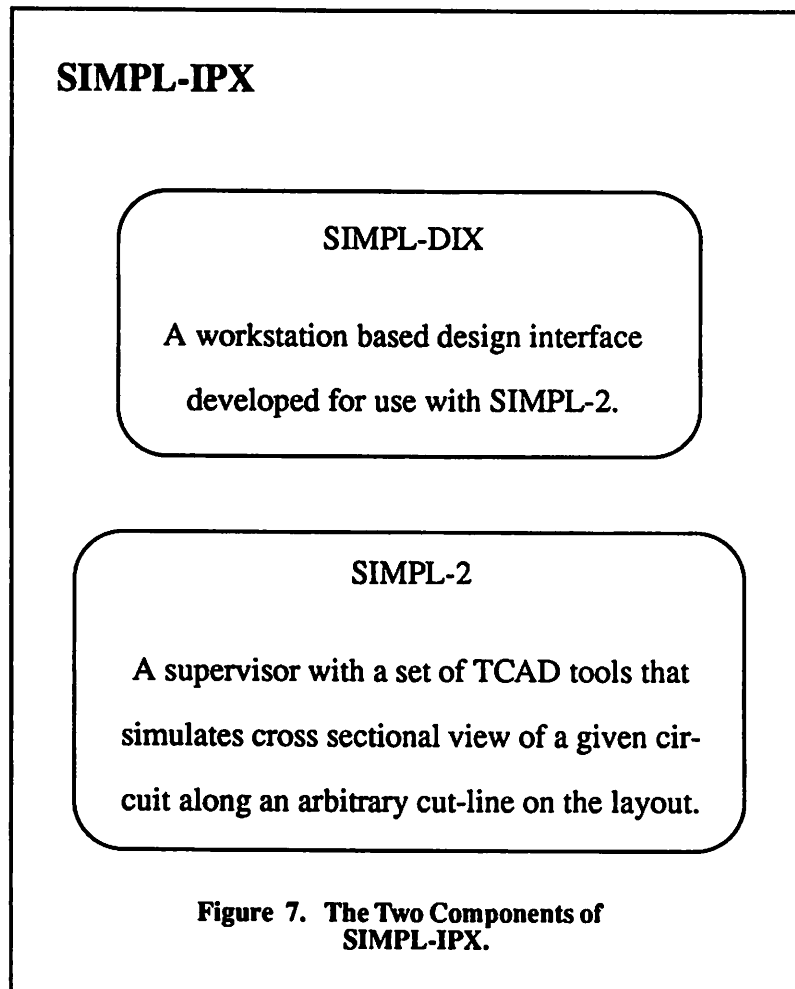

Figure 7. The Two Components of SIMPL-IPX.

(ii) SIMPL-2 (*SIM*ulated *P*rofile from *L*ayout-version 2), the second component of SIMPL-IPX, serves as a supervisor for a set of CAD tools that simulate the processing steps of an IC while displaying a cross sectional view along an arbitrary cut line on the layout of that integrated circuit. When a process simulator is invoked as a result of the

applied processing step, the cross section of the wafer (and the attributes associated with each layer) get modified. In order to accommodate these changes, SIMPL-2 provides several ways to manipulate the displayed cross section.

SIMPL-2 contains first order internal models for certain basic processing steps like ion implantation, deposition, etching, exposure, development, and oxidation. Moreover, through its modular structure, SIMPL-2 has the capability of calling more rigorous external simulators. Figure 8 gives a more detailed picture of SIMPL-2 with internal and external process and device simulators.

A process simulator contains routines required to simulate a certain processing step

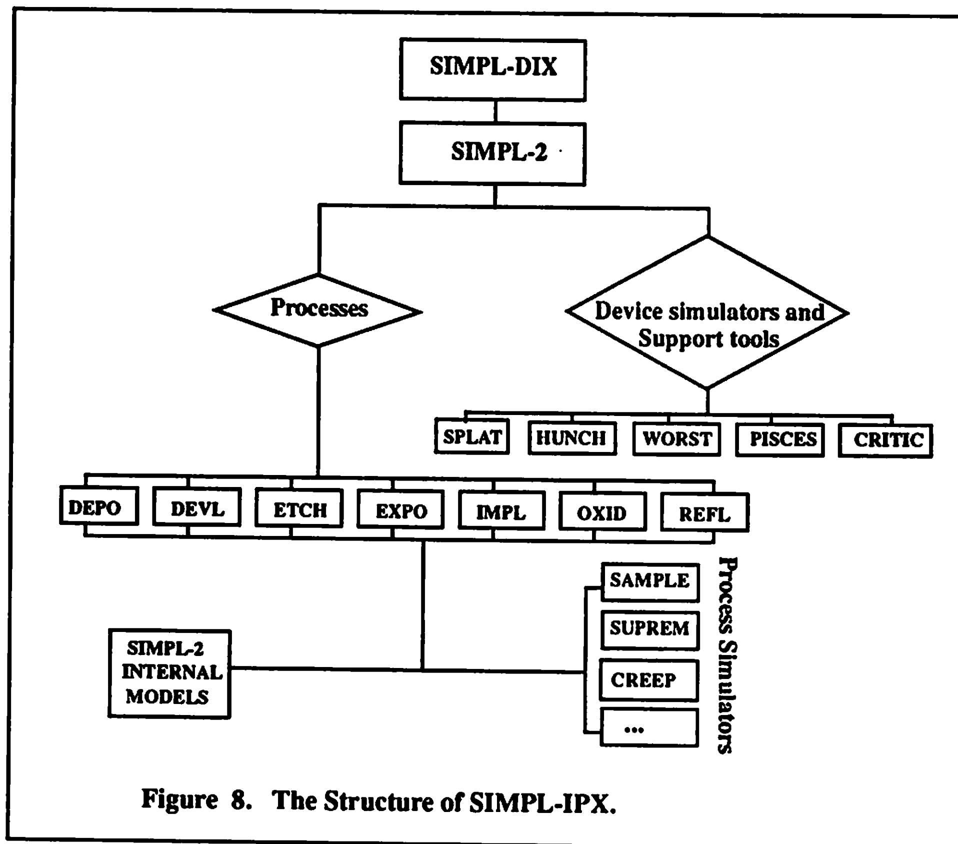

Figure 8. The Structure of SIMPL-IPX.

such as ion implantation, or deposition of gate polysilicon. Figure 9 shows an example on

how multiple simulation tools can be used in order to simulate a complex VLSI structure. In this example SAMPLE is the process simulator used for the deposition of a polysilicon gate. A device simulator, on the other hand, provides the necessary means to obtain an

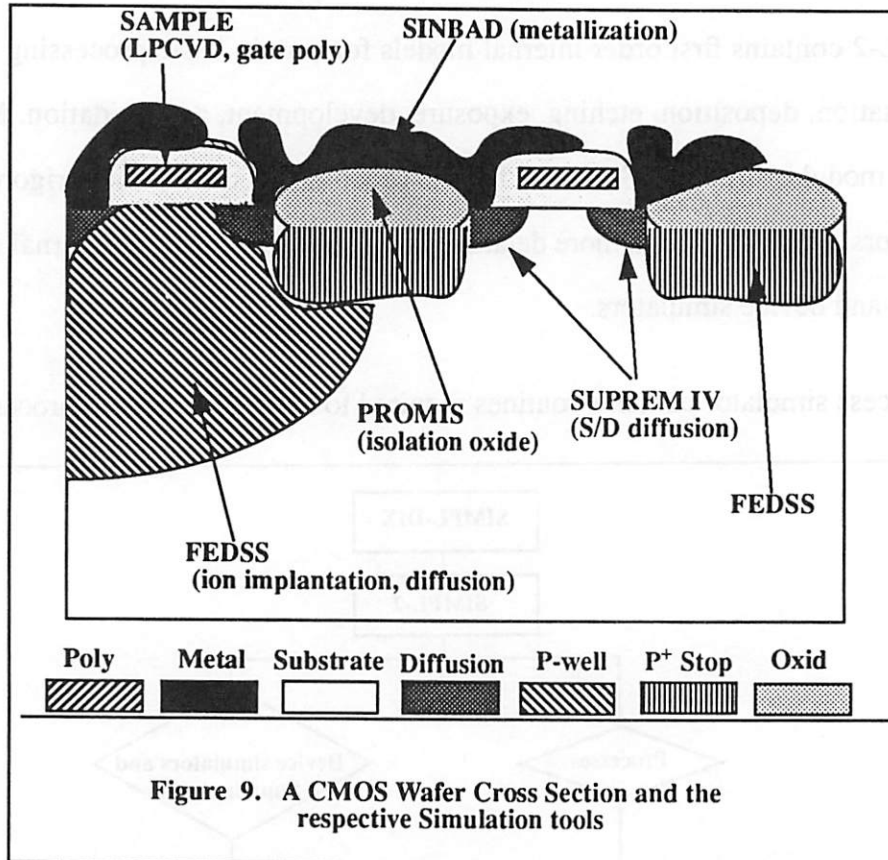

Figure 9. A CMOS Wafer Cross Section and the respective Simulation tools

insight about the device characteristics, such as current-voltage relations and device parasitic elements (Figure 10).

3.1 An Overview of the BCAM Library of Equipment Models

The Berkeley Computer Aided Manufacturing (BCAM) group has developed several statistically based polynomial models that describe the behavior of some key IC processing equipment such as the Tylan Low Pressure Chemical Vapor Deposition (LPCVD) furnace, the Lam Autoetch plasma etcher and the photolithography workcell consisting of an Eaton Spin-Coat and Bake wafer track, a GCA stepper, and an MTI omnichuck post-expo-

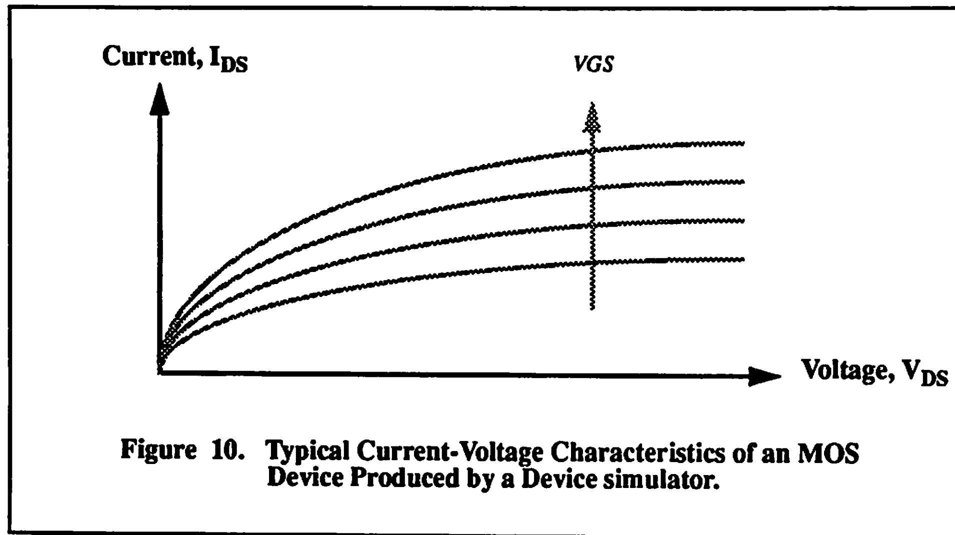

Figure 10. Typical Current-Voltage Characteristics of an MOS Device Produced by a Device simulator.

sure developer. These models are empirical in nature and have been developed with the help of extensive statistically designed experiments. The models are mathematical expressions that, given the settings of each step, can predict its measurable outcome [18] [16].

Furthermore, the BCAM models are *adaptive*. This means that, through automated measurements and use of a feedback control system, the equipment models are continuously modified to reflect the aging effects and other changes in the equipment behavior. Each *adaptive* model has two parts: an *original* model, which represents the original state of the equipment, and a *correction* model which describes the deviation from that original state [16].

Initially, a basic model is derived to represent the general structure of the equipment. This basic model includes coefficients which represent the original state of the model within that general structure. When, over time, the basic model fails to accurately represent the behavior of the equipment, corrections may be applied to the coefficients of this basic model. These corrections make up the correction model and are calculated by means of an update algorithm. The model update algorithm is initiated by a statistical process

control alarm which is generated whenever the machine outputs differ significantly from those predicted by the model [16].

In order to provide multiple users simultaneous access to the equipment models, the BCAM system uses the Ingres database and library functions to store and retrieve information about the models. The BCAM Control and Monitoring Environment [16] also provides a library of routines to manipulate equipment models. Models may be stored in either an Ingres database format or an ASCII file format. Once the BCAM software has loaded some models into memory, it can use them for prediction, simulation, sensitivity analysis, graphical visualization, recipe generation and statistical process control. The BCAM model manipulation software is organized as a library which can be accessed by other modules inside or outside the BCAM environment. This library has been implemented using C++, an object-oriented superset of the C programming language, and X Windows. This implementation has taken advantage of such C++ features as data abstraction, class hierarchy, inheritance, and modularity [19].

The currently set of equipment models on MTCAD includes standard and adaptive models for the following equipment: the Tylan Wafer Furnace for LPCVD, the LAM Autoetch polysilicon plasma etcher, the Eaton photoresist spin-coat and bake track, the GCA photolithography stepper, and the MTI photolithography developer. Figure 11 depicts the five families of equipment models available on MTCAD. Next, we briefly describe these equipment models.

3.1.1 LPCVD of Polysilicon

This model represents the operation of a Tylan Low-Pressure Chemical Vapor Deposition Furnace. The model is built using a 2-stage, 24-run D-optimum statistical experiment [20]. The thickness of the polysilicon deposited over the wafer is found to be a

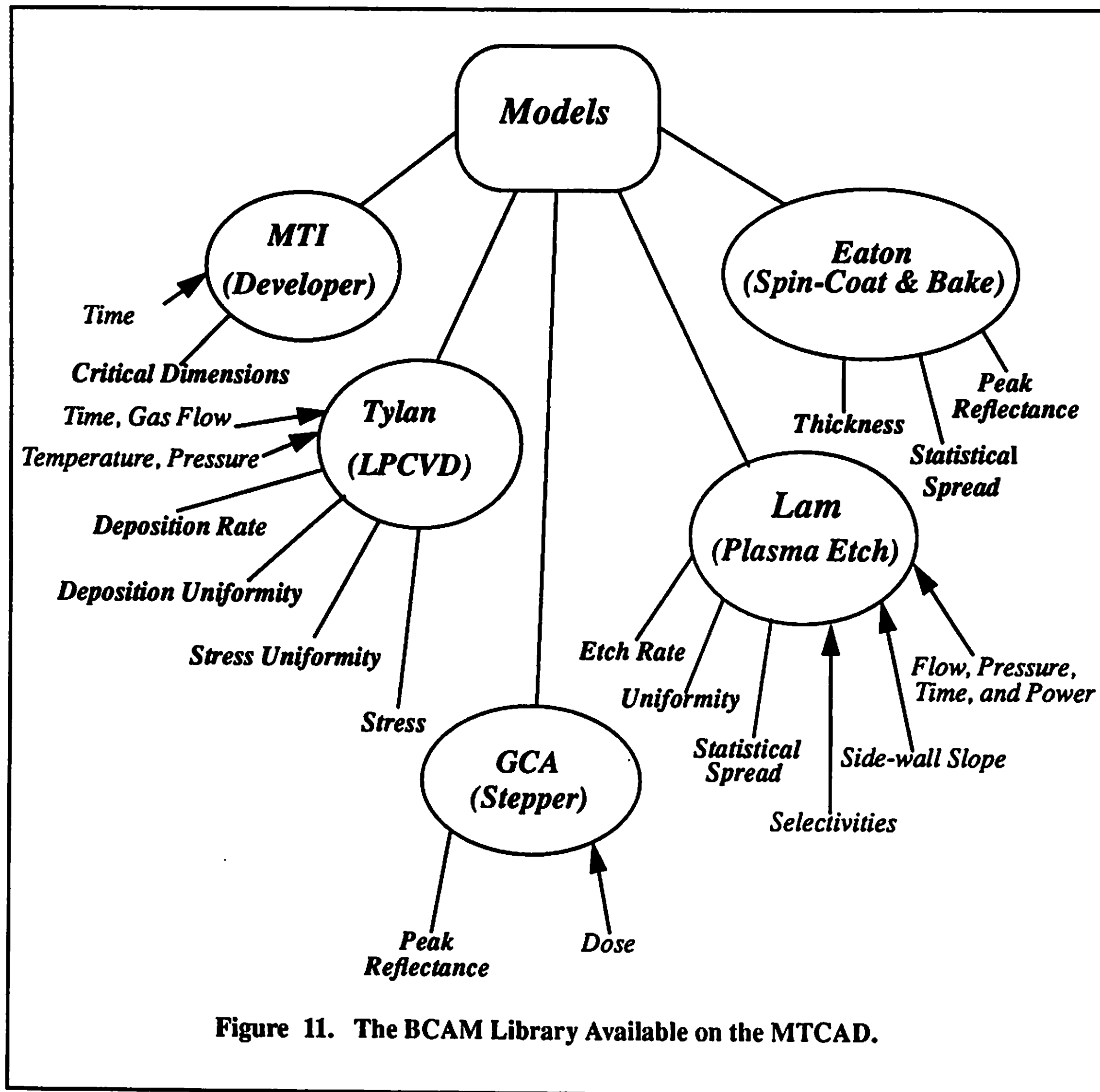

Figure 11. The BCAM Library Available on the MTCAD.

function of pressure, temperature, Silane flow, and time. The user-defined "recipe", includes the pressure, temperature, Silane flow, and deposition time. Given the recipe, the model predicts the *deposition thickness*, the *deposition non-uniformity*, the *built-in stress*, and the *built-in stress non-uniformity*. As a representative example of all the BCAM equipment models, the first stage linearized deposition rate model is given next:

$$\ln R_0 = 21.51 + 0.22\ (\ln P) - 15414.13\ (T^{-1}) - 61.92(Q^{-1}) \text{ in } \ln \text{ Angstroms/min} \quad (1)$$

where P is pressure in *mtorr*, T is temperature in *Kelvin*, Q is Silane flow in *sccm*, and R_0 in *Angstrom/min*. The standard errors of all the four coefficients of the above equation are: 0.92, 0.08, 632.83 and 8.34 for the constant term, pressure, temperature, and Silane flow, respectively. Different equations, predict the uniformity of the deposition, and the stress of the deposited polysilicon layer.

3.1.2 Photolithography

A complete photolithographic sequence is modelled using a set of response surfaces obtained through a number of fractional factorial statistical experiments [21] [22]. The models can be used to predict the thickness ([b] in Figure 12) and the peak reflectance of spin-coated photoresist versus the spin speed, the spin time, and the pre-bake temperature for an Eaton wafer track. Other models predict the post-exposure reflectance versus dose for a GCA stepper, and the Critical Dimension ([c] in Figure 12) versus the development time for an MTI developer. These models allow the complete simulation of all the measurable parameters during the photolithographic sequence in the Berkeley Microfabrication Laboratory.

Applying a modern supervisory system that controls the process on a run-to-run basis, S. Leang has been able to compensate for instabilities created by drifts due to equipment aging, fluctuations in ambient conditions, etc. for a photolithographic process sequence [23], [24].

Although both the equipment settings and the "noise" variables (such as ambient temperature, ambient humidity, and photoresist viscosity) affect the performance of each step, the latter set of variables cannot be easily monitored nor controlled during the process. For example, the photoresist thickness gives little information about the PAC of the

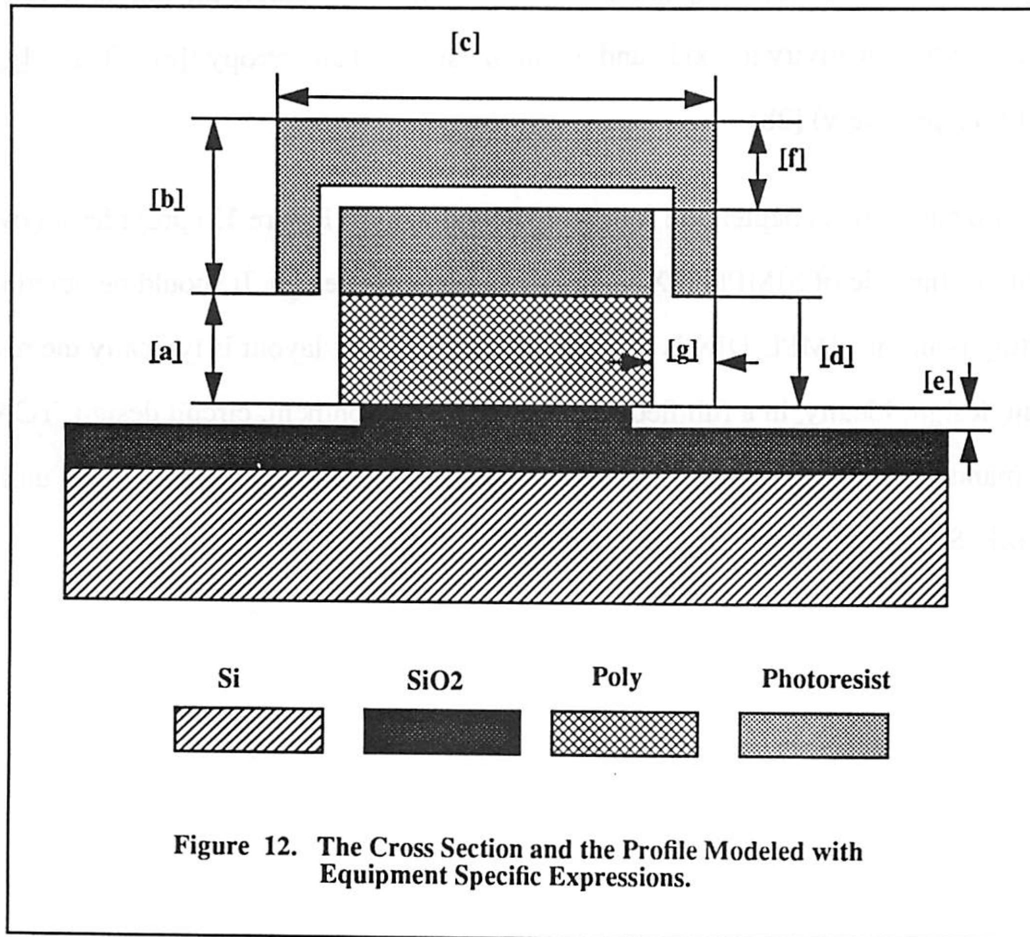

Figure 12. The Cross Section and the Profile Modeled with Equipment Specific Expressions.

photoresist, and the pattern dimension measurement is only available after the completion of the entire pattern transfer process. However, considerable efforts have been directed towards the means of evaluation of the PAC. Watts *et al* suggested measuring the absorptivity of light in the photoresist layer [25]. As a result of the work done by the BCAM group, our equipment model for the GCA stepper has the capability of predicting the post-exposure peak reflectance as a function of dose [26].

3.1.3 Plasma Etching

This model has been extracted through a 2-stage, 53-run Box-Wilson experiment on a Lam Research Autoetch 490 [27]. The recipe includes the RF power, pressure, electrode

spacing, as well as the He, CCl_4 and O_2 flows. The model predicts the polysilicon etch rate, uniformity, selectivity to oxide and to photoresist, and anisotropy ([e], [f], and [g] in Figure 12, respectively) [28].

To summarize this chapter, the following flow diagram (Figure 13) provides an overall picture of the role of SIMPL-IPX in circuit and process design. It should be noted that the starting point of SIMPL-DIX is a circuit layout. A circuit layout is typically the result of circuit design. Ideally, in a full fledged simulation environment, circuit design, TCAD, and the manufacturing system of equipment models, all should be integrated into a unified framework. Such a framework is described next.

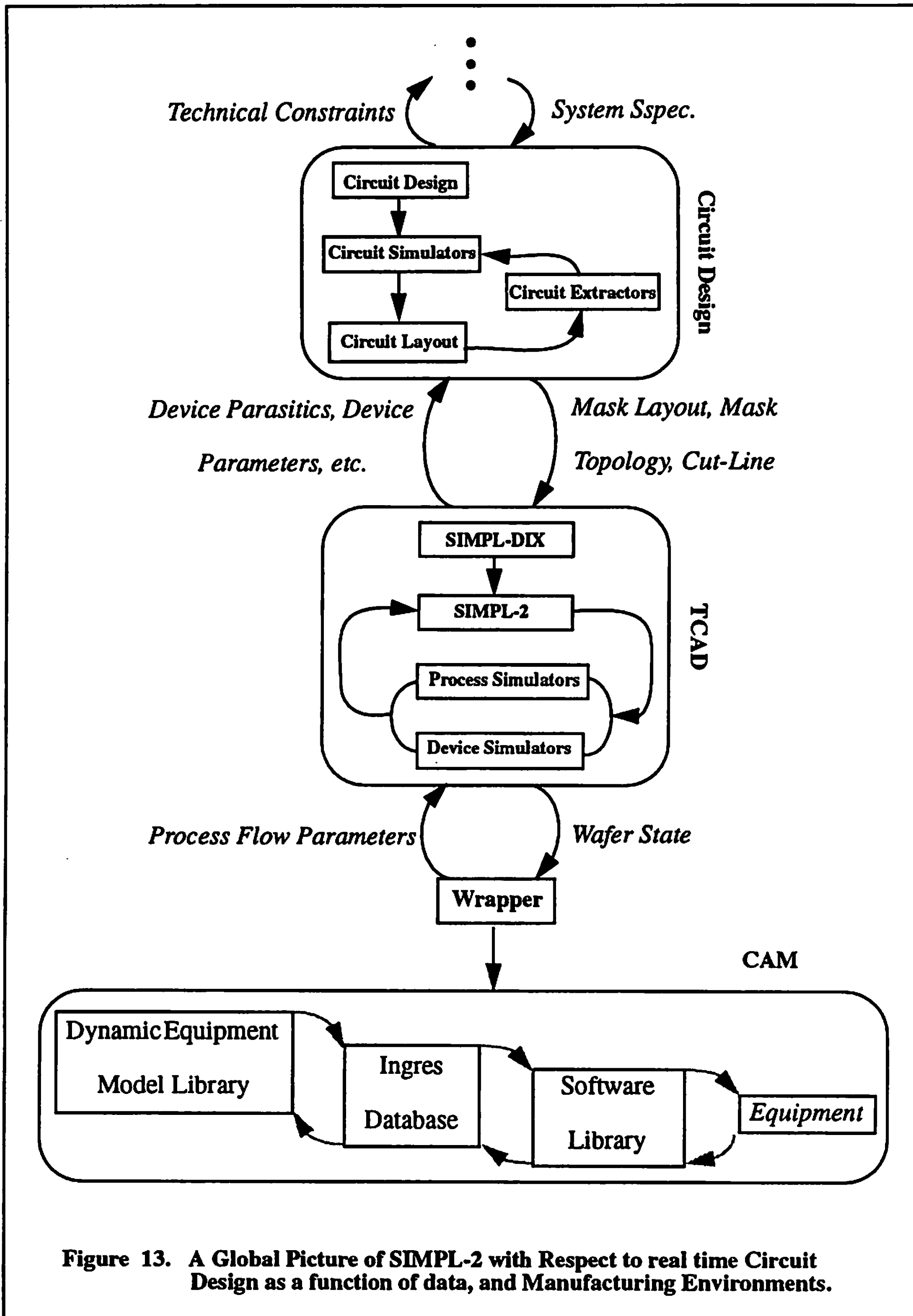

Figure 13. A Global Picture of SIMPL-2 with Respect to real time Circuit Design as a function of data, and Manufacturing Environments.

Chapter 4

Connecting SIMPL-IPX to BCAM

4.0 Different Approaches to the Design of an Interface

In order to introduce the effects of various manufacturing equipment models of the BCAM system into the SIMPL-IPX Technology CAD framework, an interface that provides communication links between the two worlds is required. This interface should be able to provide a link between each TCAD tool and the corresponding equipment model in the BCAM library. Through this link the input settings of an equipment model will be transferred to the model evaluator of the BCAM function library, and after the evaluation process is complete, the results will be made available to the TCAD tool within the SIMPL-IPX framework.

A previous attempt to treat such an interface led to a hard-wired prototype[7]. In that implementation, each TCAD process simulator had a dedicated link to an equipment model. In the hard-wired scheme, the coefficients of the model polynomial equation were not extracted from the dynamic BCAM library, but were manually entered in a file. In addition, a separate recipe editor, and a separate model evaluator were required for each equipment model. As a result, there were as many recipe editors and model evaluators as equipment models. This prototype, though useful, led to a great deal of redundancy, and

instead of relying on dynamic data directly collected from a fabrication line, was based on static data. Figure 14 depicts the structure of the hard-wired prototype.

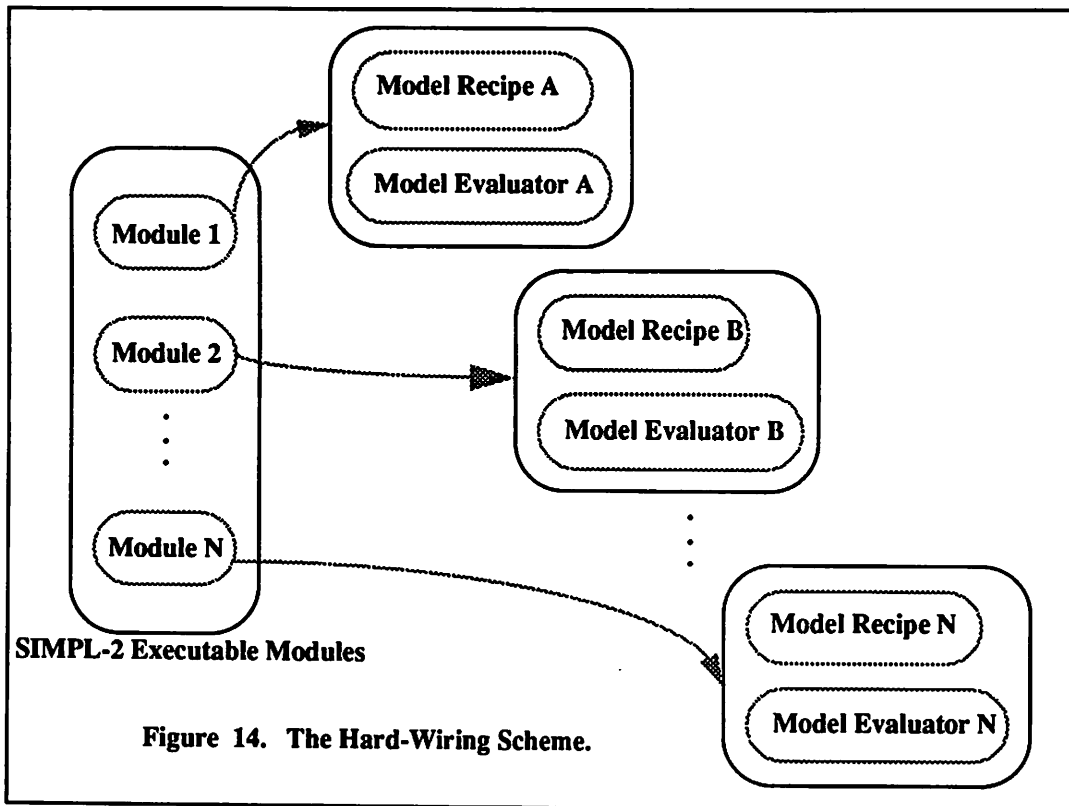

Figure 14. The Hard-Wiring Scheme.

The new implementation, on the other hand, has been focusing on the idea of developing a generic interface that would encapsulate all the existing *(standard*, and *adaptive)* equipment models. Without a need for further extensions, the wrapper interface has the capability to automatically accommodate any future additions to the equipment model library. Furthermore, since the equipment models within the library are dynamically updated by the BCAM controller, the wrapper has eliminated the usage of any static data. Figure 15 depicts the new scheme, in which a generic recipe editor and a generic model evaluator have replaced all the individual recipe editors and model evaluators of the hard-

wired approach. In fact, it was the extensive usage of data abstraction in the BCAM repre-

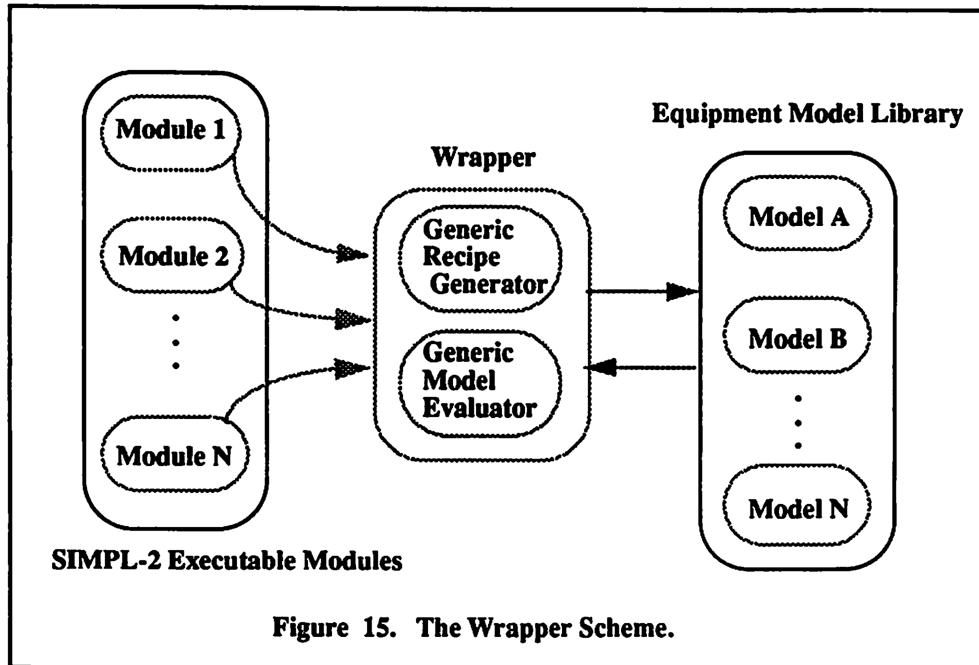

Figure 15. The Wrapper Scheme.

sentation of the dynamic equipment models, and its provisions for access to a public database that facilitated the implementation of the new interface.

4.1 The Wrapper: Generic Encapsulation of Manufacturing-based Equipment Models

Upon a request from a process simulator from inside the SIMPL-IPX TCAD framework, the wrapper, our generic interface, will invoke the desired equipment model in the BCAM environment, performing the following four major tasks:

(a) it accesses the BCAM environment in order to obtain information about a specific equipment model,

(b) using an X-window interface, it interacts with the user through recipe editors, and dialog boxes to obtain the input parameter values for the equipment model,

(c) it calls the BCAM model library in order to evaluate the outputs of the equipment models,

(d) and finally it returns the results back to the process simulator in the TCAD environment.

To be able to perform these four major tasks, the wrapper requires the process simulator to provide the name of the equipment model, and to select the name(s) of the desired output(s).

In the object-oriented implementation of the BCAM model library, each model is treated as an *object* with fields for *number of inputs, input parameter names, number of outputs, output parameter names, and output units*. Given the object properties of the equipment models, it is easy for an interfacing mechanism such as our wrapper to encapsulate any of the equipment models of the BCAM library.

The first task of the wrapper is to load information about the equipment model from the BCAM library. The wrapper opens a connection to the Ingres database and uses the equipment name provided by the TCAD tool in order to load a certain version (*standard* or *adaptive*) of the model data structure.[1] The model data structure contains detailed information about input settings, outputs and also provides an evaluator. The wrapper uses the information about the *input settings* of the models (*number of the inputs, name of the inputs*, *units of the inputs*), and the *output results* of the models (*number of the outputs, name of the outputs*, and *the units of the outputs*) and creates a recipe editor with default settings for the specific model. The process designer is allowed to edit the recipe parame-

1. It is not necessary for the user to know the names the equipment models available in the Ingres database to be able to call the appropriate model. By clicking on the "Equipment Model" Button, an "Equipment Activation" menu will provide a list of all the available equipment models.

ters within their respective ranges. As an example of equipment model object and its corresponding fields, consider the case of Tylan16 Wafer Furnace equipment model object:.

Objects	Number of Inputs Field	Input Names Field	Input Units Field	Number of Outputs Field	Output Names Field	Output Units Field
Tylan16	4	Pressure	mtorr	2	Thickness	Micrometer
		Temperature	kelvin		Stress Uniformity	%
		Silan Flow	sccm			
		Time	minutes			

Once the input parameters in the recipe are set by the user, the wrapper uses the model evaluator to calculate the output results of the equipment model. After the evaluation, the requested outputs will be returned to the TCAD process simulator, that initiated the call.

This completes the cycle and once again inside SIMPL-IPX the process designer can continue with the following step. Figure 10 depicts the events associated with evaluation of an equipment model.

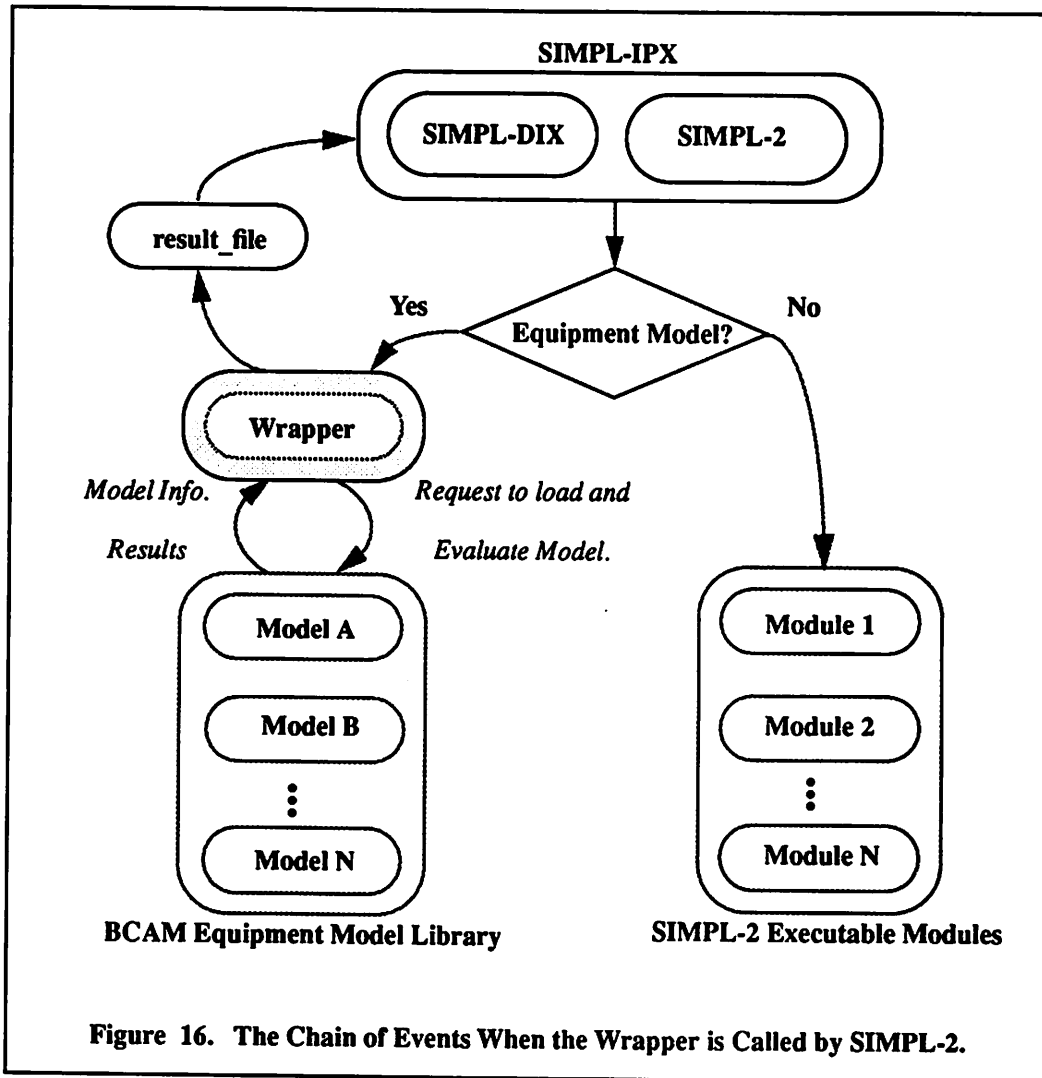

Figure 16. The Chain of Events When the Wrapper is Called by SIMPL-2.

The wrapper is totally transparent to the TCAD user. The generic character of the wrapper allows great flexibility in the equipment model library. In fact, without any further modification of the wrapper, any new model stored in the BCAM database can automatically be integrated into the framework. The four tasks of the wrapper are identical for

all TCAD processes, regardless of the type of the process. The data translation between the two systems is simple and general.

In summary, from the TCAD system a request is sent to the BCAM environment. In that request, all that has to be provided by any TCAD process is simply an equipment name along with its output names. In response to the TCAD request, the BCAM environment provides the calculated values for these desired outputs.

4.2 Implementation Issues

The integration of the TCAD process simulators and the BCAM equipment models into the unified framework of MTCAD involved the design of an interface between the two, and some modifications in the SIMPL-IPX code.

The interface (wrapper) was implemented as an independent module that is invoked through a system call by any of several TCAD process simulators. The wrapper is written in C++ and uses many of the BCAM routines and data structures. The most important data structure used in the wrapper is the *machineClass* structure of the BCAM environment. This data structure includes many member classes and functions providing all the information necessary to interact with a specific piece of equipment [16].

When the wrapper is invoked with a machine name, it creates an instance of the *machineClass* structure. In order to load the machine into this structure the wrapper first establishes a connection with the Ingres database. The routine *ConnectIngres(NULL)* is used for this purpose. Subsequently, the member function *GetMachine(equipmentName)* of the *machineClass* structure is called with the equipment name in order to load the specific machine. The data loaded for the machine contains information about the number of inputs, their units, names and default values. This information is used to create a simple

recipe editor with an X Window interface.[1] The recipe initially uses the default values supplied by the model. The user can edit these values, and those values that are outside the acceptable range for an input are rejected.

Given the input setting, the wrapper uses the member function *Predict(outputNum)* to evaluate the respective model and to calculate each output requested by the TCAD process simulator. The outputs are written to a result-file ("EQUIPFILE") and control is returned back to the TCAD process simulator.

As an example of how the wrapper works, let us consider the case when we choose to call upon the Tylan16 equipment model in order to deposit a certain thickness of polysilicon over a given wafer. The user accesses the recipe editor and a recipe will be generated for the Tylan16 model. Once the recipe is accepted, the *Predict(outputNum)* evaluator function of the model member class will solve the polynomial given by equation (1) section 2.1.1. Note that the coefficients of the polynomial are provided by the equipment model already stored in the database, and the parameters of the polynomial (temperature, pressure, etc.) are obtained from the recipe. The routine *Predict()* is a generic model evaluator, which, given an object-model, will extract the number and values of coefficients, the constant term of the polynomial, and all other relevant information needed to solve the polynomial from the fields of the object-model.

SIMPL-DIX and SIMPL-2, the two modules of SIMPL-IPX, have been modified in certain junctures. The modifications in SIMPL-DIX involved adding menu options and the corresponding call back routines to the existing menus. For each process in TCAD that could use an equipment model in BCAM (deposition, etching, etc.) two menu items

---

1. As an important part of the wrapper interface we have used "The Equipment Recipe Menu", and "The Equipment Model Menu" of the BCAM software library. For a detailed description of the above, please refer to B. Bombay, "The BCAM Control and Monitoring Environment", pp 25-27 [16].

were created to let the user choose between the SIMPL simulator and the appropriate BCAM equipment model[1].

The SIMPL-2 code pertaining to processes that can use a BCAM equipment model were modified to handle the two options of using a SIMPL-2 (internal or external) simulator versus a BCAM equipment model.[2] These processes will invoke the wrapper with the name of the model to be used and the expected output names. The output values are returned by the wrapper in a file containing the results. This chain of events is depicted in Figure 16. The path name of the wrapper and the results file are defined in a source file (.simplrc) along with other executables and file path names of SIMPL-IPX. As a summa-

1. The modified files in SIMPL-DIX are listed in appendix A.
2. The modified files in SIMPL-2 are listed in appendix A.

tion, Figure 17 provides the overall picture of the structure of the wrapper with respect to the SIMPL-PIX and BCAM environments.

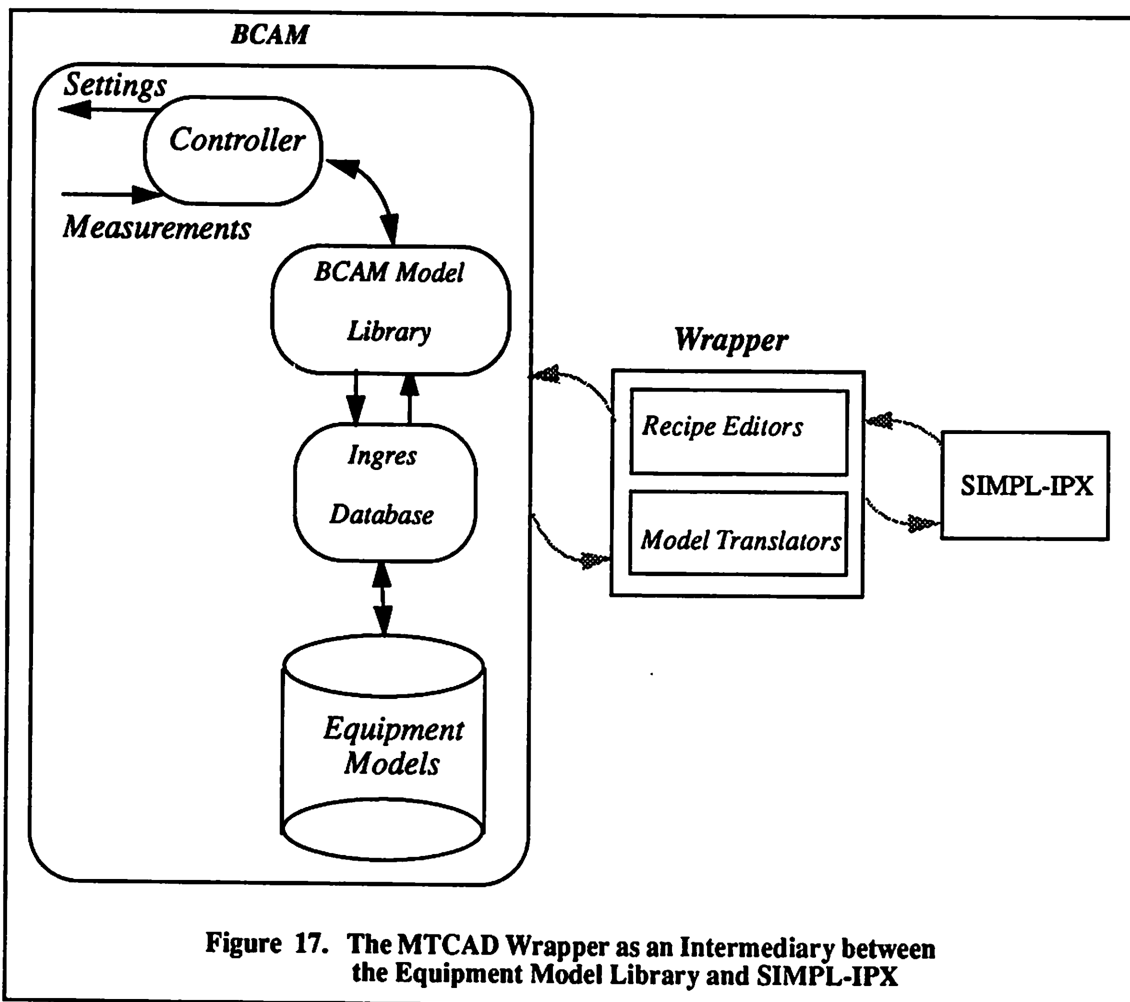

Figure 17. The MTCAD Wrapper as an Intermediary between the Equipment Model Library and SIMPL-IPX

4.3 Simulation Tool Wrapper versus Equipment Model Wrapper

It is worth mentioning that the idea of a "wrapper" has been used in conjunction with the CAD Framework Initiative (CFI) before. In a CFI context, process or device simulation tools are considered integrated by virtue of their ability to communicate with each other in terms of data abstractions. This resource abstraction is provided through tool encapsulation. In the case of a CFI environment,

(a) the wrapper encapsulates various unmodified process and device simulators,

(b) the wrapper accesses Semiconductor Wafer Representation (SWR) compliant servers to set up input files,

(c) the tool is run, creating output files,

(d) the output files are used by the wrapper to update the server (Figure 18).

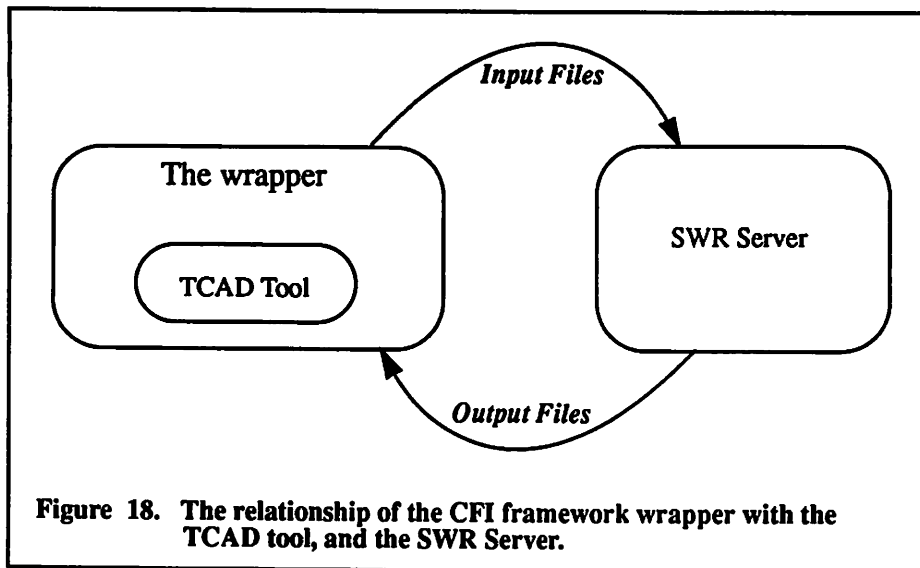

Figure 18. The relationship of the CFI framework wrapper with the TCAD tool, and the SWR Server.

In the case of MTCAD, however,

(a) the wrapper encapsulates the BCAM equipment models,

(b) it creates the equipment recipe,

(c) the wrapper runs the equipment model and sets up the result-file,

(d) the wrapper makes the result-file available to the TCAD environment. (See Figure 16).

The two wrappers, thus complement each other, and both of them could be implemented at the front-end of an integrated framework to provide appropriate avenues to simulation tools on the one hand, and equipment models on the other. Figure 19 depicts the

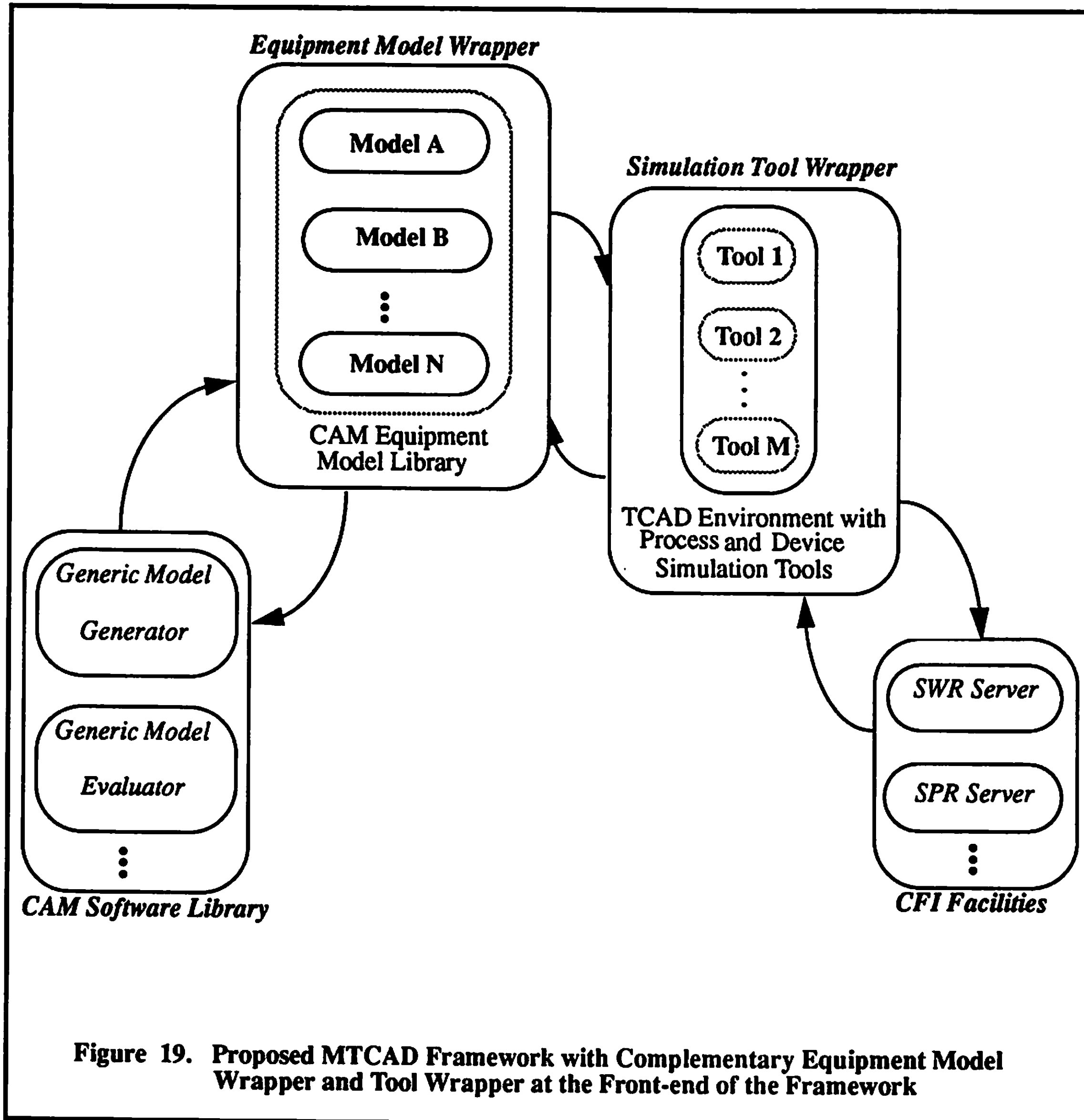

Figure 19. Proposed MTCAD Framework with Complementary Equipment Model Wrapper and Tool Wrapper at the Front-end of the Framework

proposed integration of the two wrappers in a unified CAD/CAM framework.

Chapter 5

Application Example

5.0 Process Simulation of a CMOS Inverter

In this application example a sequence of steps will be applied to a p-type silicon wafer in order to produce a CMOS inverter. The process designer will switch between TCAD tools and BCAM equipment models wherever appropriate.

In MTCAD, by selecting the RUN SIMPL-2 command from the root menu, SIMPL-DIX will invoke the program SIMPL-2 for cross-section view generation. As we mentioned before, we start with the circuit layout. SIMPL requires the layout mask of a circuit to be in Caltech Intermediate Format (CIF). Moreover, SIMPL requires a technology pattern file in order to get the necessary information about the patterns and the colors that will be used to display the cross-section of the wafer during the simulation. Once the layout file of a CMOS inverter (e.g. a cmos.cif), and a pattern file (e.g. cmos.pattern) are provided, SIMPL-DIX will ask the designer to specify a cut-line on the layout in order to determine the plane of the cross-sectional view during the simulation.

After drawing the cut-line, the designer will be asked to specify the doping type and doping concentration for the substrate. In interactive mode ("INPUT MODE"), the

designer can select a command to continue with one of the processes available in the menu displayed at the bottom of Figure 20. The steps are outlined next.

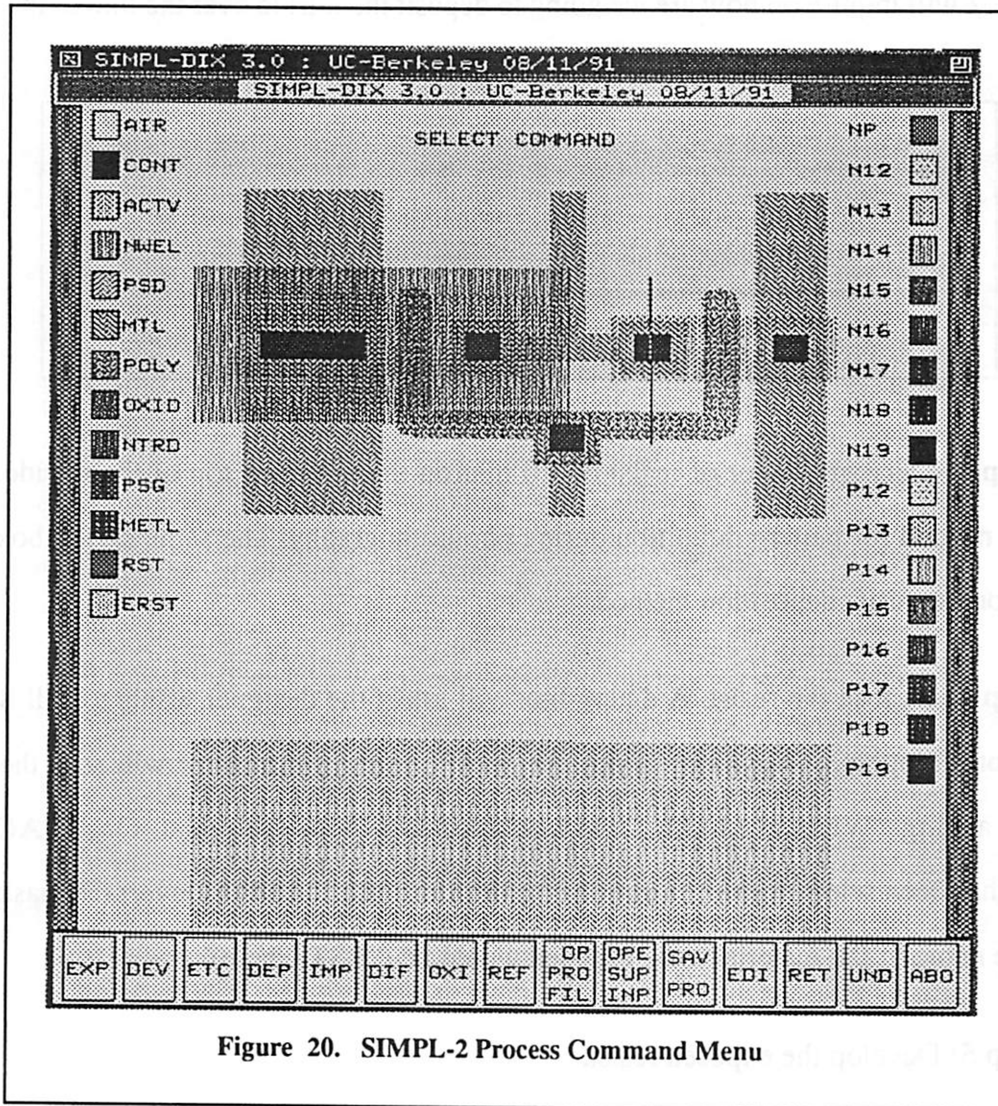

Figure 20. SIMPL-2 Process Command Menu

Step 1: Grow a thin oxide layer of about 0.1 micron (this thin layer works as a stress buffer between silicon and nitride) on a slice of wafer. For this purpose, the OXID process is selected.

Step 2: Deposit a thin layer of silicon nitride patterned after the oxidation mask. In the dialog box, we specify 0.1 micron as the thickness of the silicon nitride layer. A further dialog box will inquire on how are we going to deposit the nitride over the thin oxide (Figure 15):

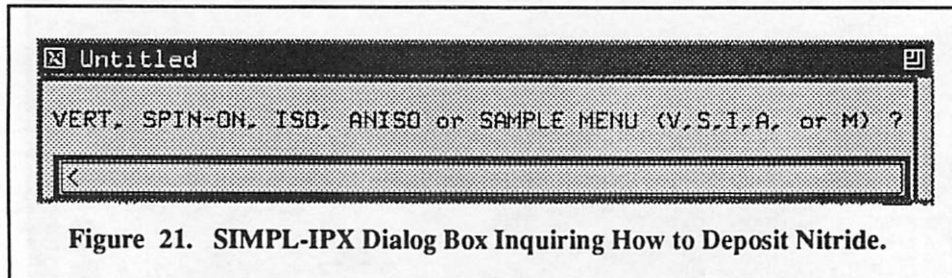

Figure 21. SIMPL-IPX Dialog Box Inquiring How to Deposit Nitride.

Step 3: Deposit a layer of resist of 1.0 micron thickness on top of the nitride. The resist is needed for pattern definition during photolithography. The same dialog box will inquire on the resist deposition method.

Step 4: Expose the resist. A dialog box will ask if the designer wants to call SAMPLE photolithography simulator. If not, the SIMPL internal simulator will start the process by asking "WHICH MASK?" will be used to expose the resist. Once "ACTV" (active) has been selected, the designer has to decide if s/he intends to invert the mask, and what the name of the material to be exposed would be ("RST", here).

Step 5: Develop the exposed resist.

Step 6: In order to etch, either the Sample Nonplanar Etch or the SIMPL-2 internal modules may be called to etch 0.1 microns of nitride, and all of the photoresist layer. The remaining nitride will help to prevent the further oxidation of the active region during

field oxidation. The cross-section of the wafer after these steps should look like Figure 22:

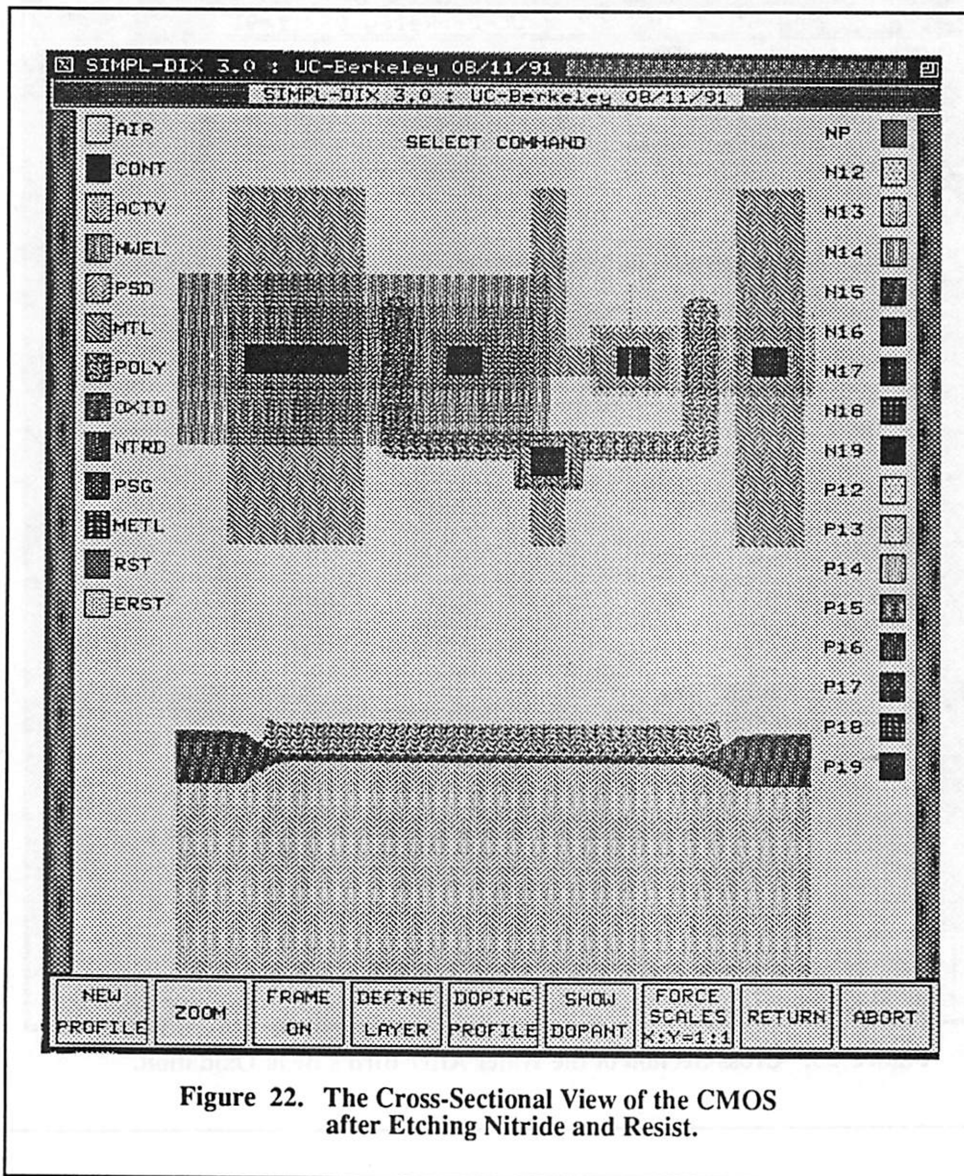

Figure 22. The Cross-Sectional View of the CMOS after Etching Nitride and Resist.

Step 7: Grow field oxide to define the isolation region. Once the "iso" is defined, we can remove the nitride layer.

Step 8: After etching all of the nitride, the cross-section of the wafer will look like

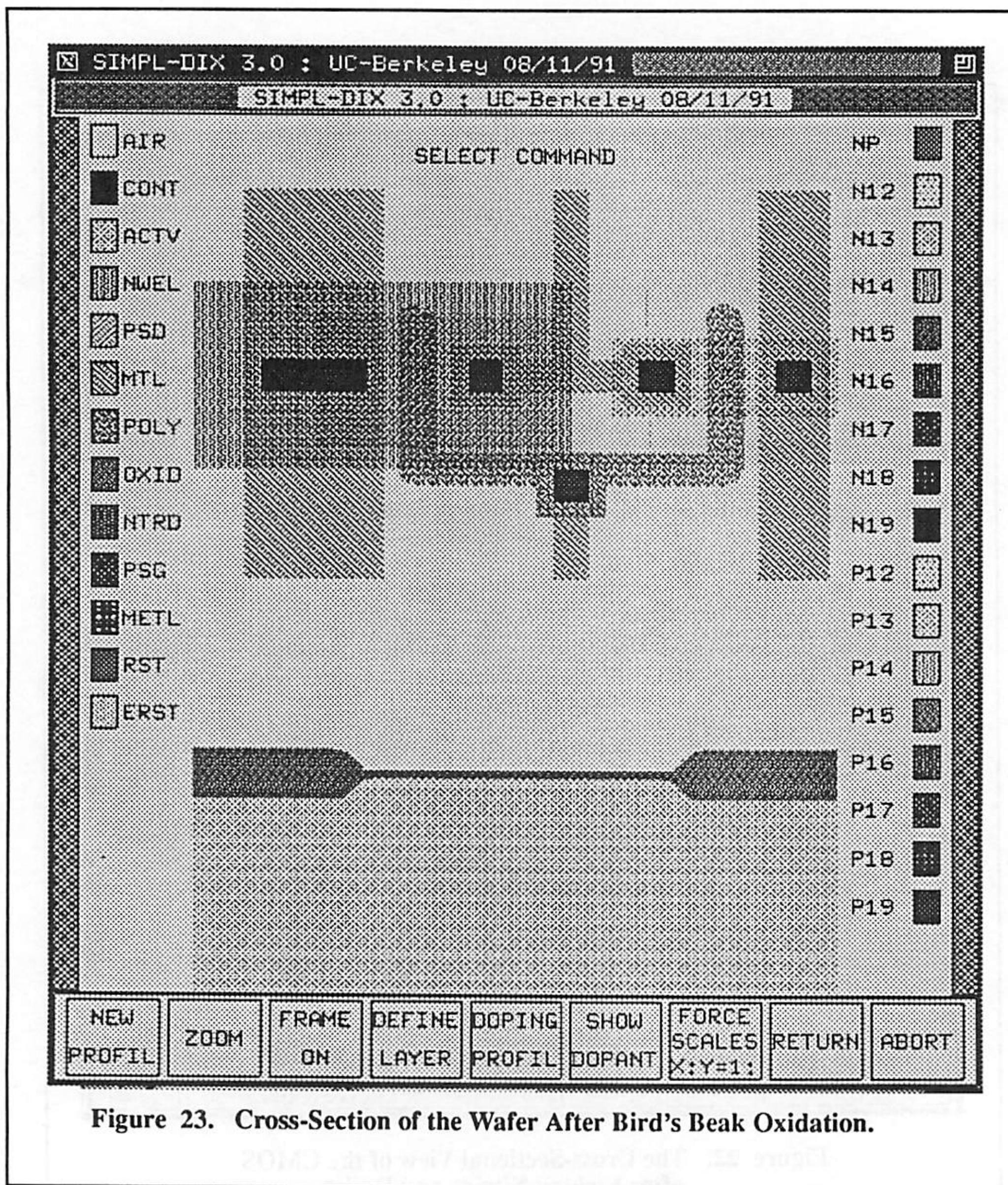

Figure 23. Cross-Section of the Wafer After Bird's Beak Oxidation.

Figure 23.

Step 9: Threshold voltage adjustment implantation to create the source/drain regions. Threshold voltage adjustment implantation can be simulated by either SUPREM IV, or by

the SIMPL-2 internal simulators. After implantation, the cross-section of the wafer will look like Figure 24.

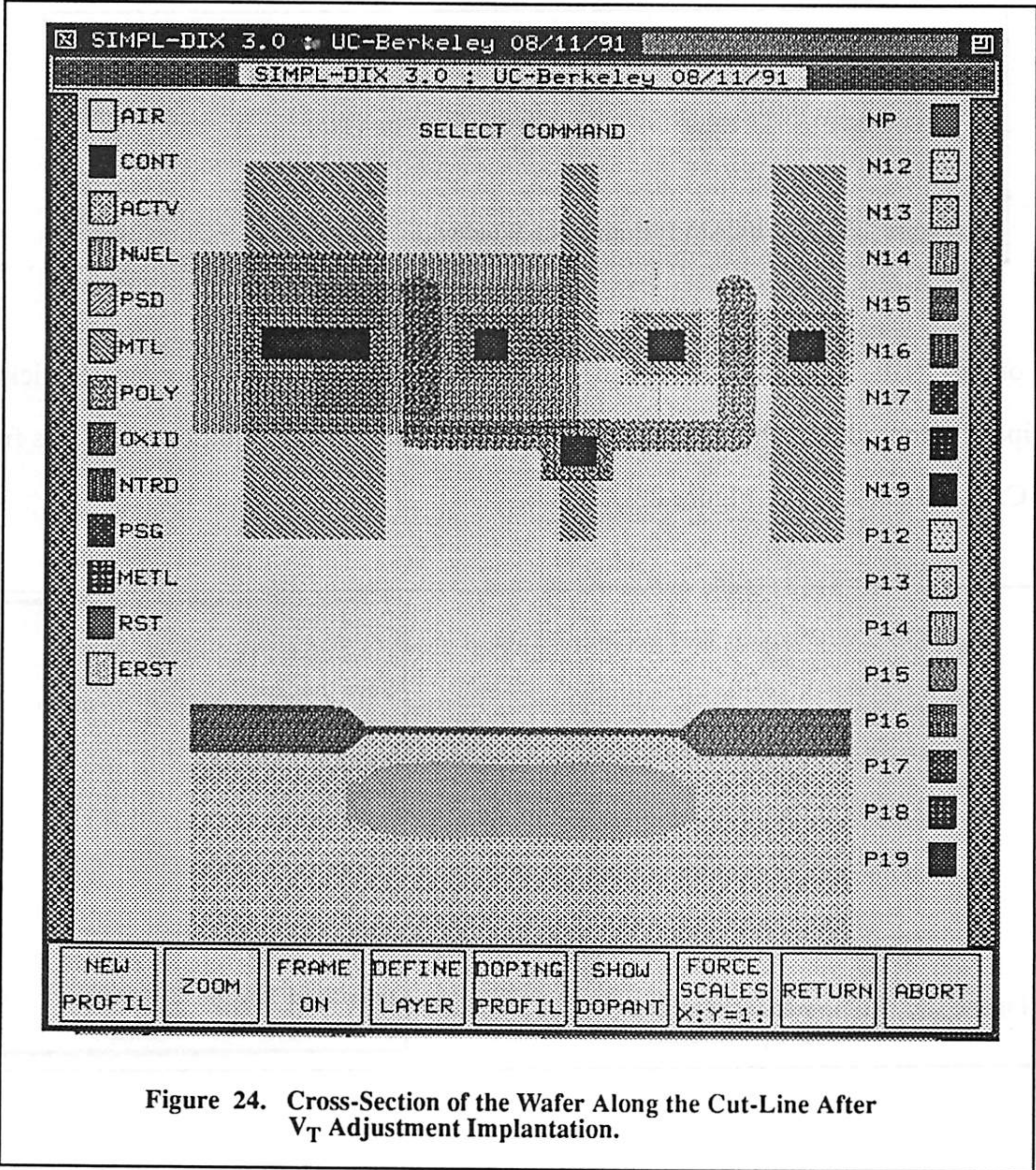

Figure 24. Cross-Section of the Wafer Along the Cut-Line After V_T Adjustment Implantation.

Step 10: At this stage, the process designer decides to deposit gate polysilicon using the Tylan16 equipment model. Therefore, after clicking on "DEPO" button the designer will be asked to choose between either the SIMPL-2 internal simulator, SAMPLE, or the

"Equipment Model" to perform the deposition. Once the designer decides to use the equipment model, the BCAM introductory window will appear as shown in Figure 259. A

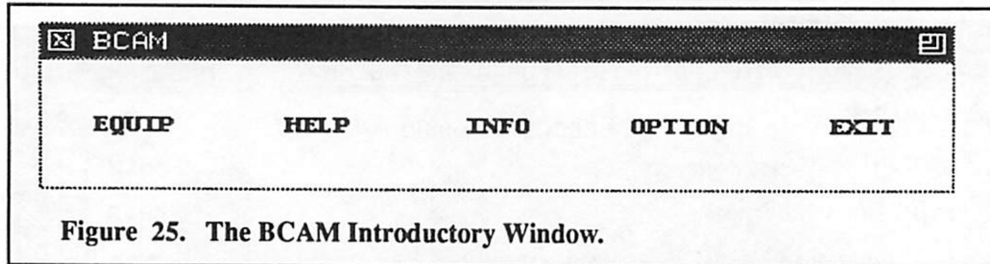

Figure 25. The BCAM Introductory Window.

click on "EQUIP" will result in "Tylan16 Wafer Furnace" window to pop up. A click on "Recipe" button, allows the user to load, edit, store, and retrieve equipment recipes from the BCAM database. (See Figure 26).

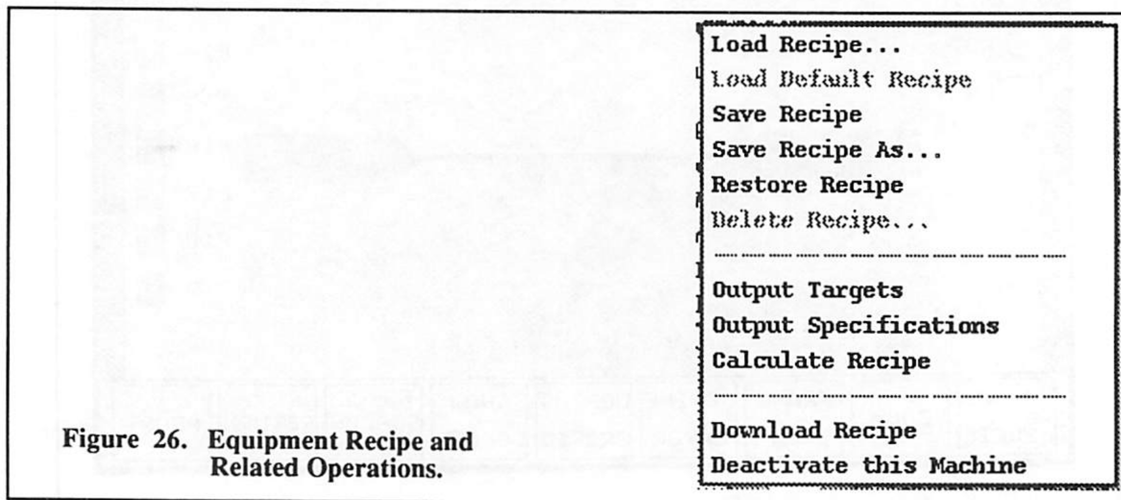

Figure 26. Equipment Recipe and Related Operations.

The designer is required to specify the output(s) of deposition process to be either thickness or stress. After loading the default recipe, or any other previously saved recipe, the designer can edit it and click on the "Calculate Recipe". Once the desired recipe has been set, the designer can save it under an optional name for future reference. The result of the optimal equipment recipe will be passed back to the SIMPL-IPX tool, and the gate

polysilicon with desired thickness and stress uniformity will be deposited over the wafer. The cross section of the CMOS wafer after deposition of gate polysilicon, along with a default Tylan16 furnace recipe is shown in Figure 27.

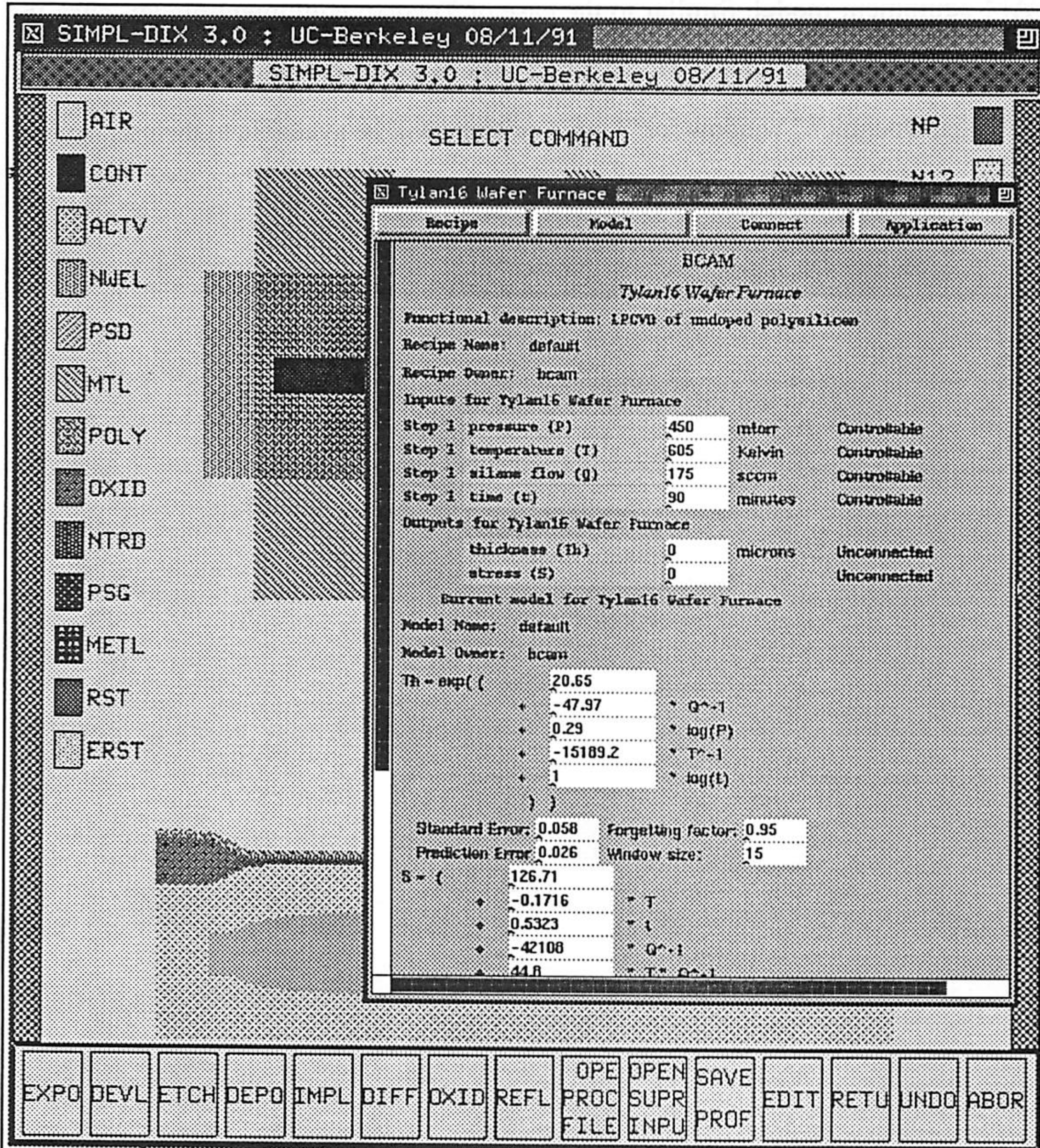

Figure 27. The Cross Section of the CMOS Wafer after Deposition of 0.4 Micron Polysilicon over the Wafer, along with a Tylan16 Equipment Model.

Step 11: After going through Steps 4 and 5 again, the designer will be able to etch the polysilicon layer from over the field area. Etching may be done by either calling SIMPL-2 internal simulators (as well as the SAMPLE Nonplanar Etch Simulator) or by a BCAM equipment model. Similar to the case of Tylan16 equipment model, once the designer clicks on "Equipment Model", first "The BCAM Introductory Window", then the "Recipe Operation" window for "Lam Autoetch" will be available to the user. As was

the case for Tylan16, here the designer can modify the recipes to obtain the desirable result.

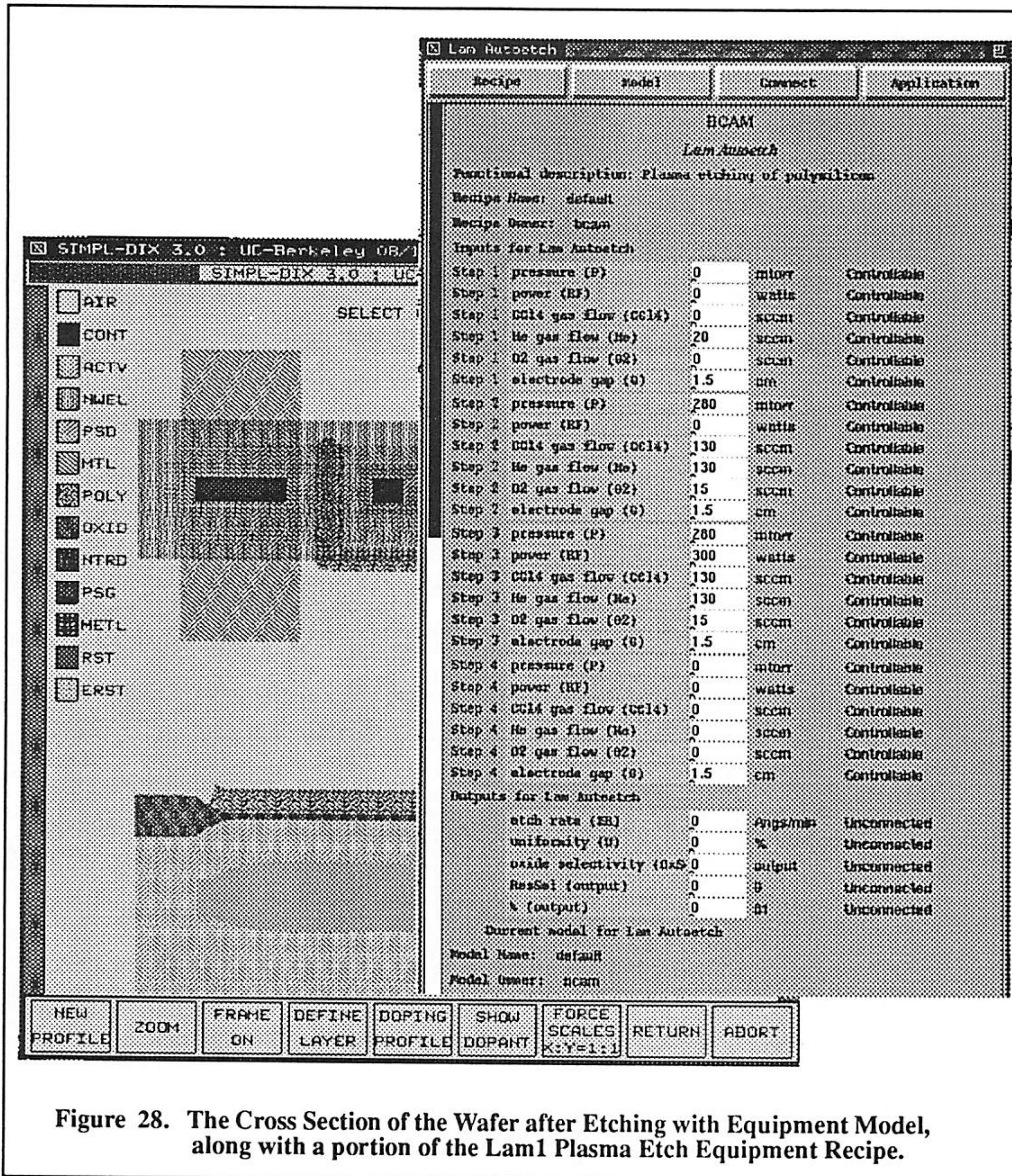

Figure 28. The Cross Section of the Wafer after Etching with Equipment Model, along with a portion of the Lam1 Plasma Etch Equipment Recipe.

Saving, retrieving and restoring any of the generated recipes allows the designer to obtain optimal results. The cross section of the wafer after etching with the equipment

model, along with a sample equipment recipe for Lam1 Plasma Etcher is depicted in Figure 28.[1]

1. Since the full Lam Autoetch recipe is too big to fit in a page this report, we only show a portion of it the previous page.

Chapter 6

Conclusions

6.0 Summary

In this project we focused on the incorporation of SIMPL-IPX, a Technology CAD framework, with the manufacturing based equipment models of the BCAM system. In order to facilitate this incorporation, an equipment model wrapper with generic model encapsulation and evaluation was developed. Through use of the wrapper, several of the BCAM utilities such as dynamic equipment model library, recipe management system, and the controller were included in the resulting unified framework. With MTCAD a designer can take into account the effects of manufacturing equipment, and hence predict the manufacturability of the process and the device under study. As a result of simulating the process with MTCAD, the process designer can directly download the optimized recipe to the corresponding equipment.

Major limitations of the MTCAD prototype include lack of a robust central supervisor unit with tool abstraction capability. Also, at the time of this report SIMPL-DIX had not been updated with the latest versions of X-window tool-kits, and hence is not able to create widgets and pop up menus.

6.1 Future work

A short term goal is to modify the SIMPL-IPX code to allow the activation of photolithography equipment models (already available as members of the BCAM family of equipment models) in the MTCAD environment. In the long run, however, one key issue that should be addressed is an efficient way of representing process and wafer state, along the lines of the CAD Framework Initiative (CFI) [11] [12], which in turn will address the problem of a CAD environment with a central supervisor unit to provide data consistency, and shorter inter-tool translation time [11].

Once these problems are resolved, many more process and device simulators will be accommodated into the MTCAD environment. As a result of such a sophisticated TCAD/CAM environment, circuit/process designers will more adequately be able to predict the manufacturability of complex microstructures.

References

[1] K. Lee and A. R. Neureuther, "SIMPL-2 (SIMulated Profiles from the Layout - Version 2)", Proceedings of the IEEE Symposium on VLSI Technology, Kobe, Japan, Digest of Technical Papers, pp. 64-65, May 1985.

[2] W.G. Oldham, A. R. Neureuther, C. Sung, J. L. Reynolds, and S. N. Nandgaonkar, "A General Simulator for VLSI Lithography and Etching Process: Part I - Application to Projection Lithography", IEEE Transactions on Electron Devices, vol. ED-26, no 4, pp. 717-722, April 1979.

[3] C. P. Ho, J. D. Plummer, S. E. Hansen, and R. W. Dutton, "VLSI process modeling -- SUPREM-III", IEEE Transactions on Electron Devices, vol. ED-30, no 11, pp. 1438-1453, November 1983.

[4] S. R. Nassif, A. J. Strojwas, and S. W. Director, "FABRICS II: A statistically based IC fabrication process simulator", IEEE Transactions on Computer-Aided Design, vol. CAD-3, pp. 20-46, January 1984.

[5] P. Lloyd, H. K. Dirks, E. J. Pendergast, and K. Singhal, "Technology CAD for competitive products", IEEE Transaction on Computer-aided Design, vol. 9, no 11, pp. 1209-1216, November 1990.

[6] E. W. Scheckler, A. S. Wong, R. H. Wang, G. Chin, J. R. Camanga, K. K. H. Toh, K. H. Tadros, R. A. Ferguson, A. R. Neureuther, and R. W. Dutton, "A Utility Based Integrated Process Simulation System", Proceedings of the IEEE Symposium on VLSI Technology, Honolulu, Hawaii, Digest of Technical Papers, pp. 97-98, June 1990.

[7] T. Luan, "Manufacturing-Based IC Process and Device Simulation", ERL Memorandum UCB/ERL M91/55, Electronics Research Laboratory, University of California at Berkeley, May 1991.

[8] E. W. Scheckler, A. S. Wong, R. H. Wang, G. Chin, and J. R. Camanga, "SIMPL-IPX User's Manual", University of California at Berkeley, 1992.

[9] R. A. Hughes and J. D. Shott, "The Future of Automation for High-Volume Wafer Fabrication and ASIC Manufacturing", Proceedings of the IEEE, vol. 74, no 12, pp. 1775-1793, December 1986.

[10] A. J. MacDonald, A. J. Walton, J. M. Robertson, and R. J. Holwill, "Integrating

CAM and Process Simulation to Enhance On-line Analysis and Control of IC Fabrication," IEEE Transactions on Semiconductor Manufacturing, vol. 3, no 2, pp. 72-79, May 1990.

[11] A. Wong, M. Karasick, V. T. Rajan, G. Chin, M. Giles, L. Nackman, and M. Law, "Semiconductor Wafer Representation Procedural interface", TCAD-91-G-2, 1991.

[12] D. Boning, "Semiconductor Process Representation: Bibliography", TCAD-91-T-xx, 1991.

[13] CFI TCAD framework prototype demonstration, Stanford University, 14 August 1991.

[14] S. Duval, "An Interchange Format for process and device simulation", IEEE Trans. CAD, vol. 7, pp. 741-754, July 1988.

[15] D. Boning, G. Chin, R. Cottle, W. Ditrich, S. Duval, M. Giles, R. Harris, M. Karasick, N. Khalil, M. Law, M. J. McLennan, P. K. Mozmder, L. Nackman, S. Nassif, V. T. Rajan, D. Shroeder, R. Tremain, D. M. H. Walker, R. Wang, and A. Wong, "Developing and Integrating TCAD Applications with the Semiconductor Wafer Representation", CFI TCAD Framework group, Semiconductor Wafer Representation Working Group, CAD Framework Initiative, Inc., February1992

[16] B. Bombay, "The BCAM Control and Monitoring Environment", Memorandum No. UCB/ERL M92/113, Electronics Research Laboratory, University of California at Berkeley, September 1992.

[17] E. W. Scheckler, J. Camagna, R. Wong, and A. Wong, "SIMPL-IPX User's Manual", The U. C. Berkeley SIMPL Group, Electronics Research Laboratory, University of California, Berkeley, January 1990.

[18] B. Bombay and C. J. Spanos, "Application of Adaptive Equipment Models to a Photolithographic Process," SPIE Technical Symposium on Microelectronic Processing Integration 1991, September 1991.

[19] K. Wieskamp and B. Flaming, The Complete C++ Primer, Academic Press, San Diego, CA, 1989.

[20] K. K. Lin and C. J. Spanos, "Statistical Equipment Modeling for VLSI Manufacturing: an Application for LPCVD", IEEE Transactions on Semiconductor Manufacturing, vol. 3, no. 4, pp. 216-229, November 1990.

[21] Z. M. Ling, S. Leang, and C. J. Spanos, "A Lithographic Workcell Monitoring and

Modelling Scheme," Microcircuit Engineering Conference, Leuven, Belgium, September 1990.

[22] F. H. Dill, W. P. Hornberger, P. S. Hauge, J. M. Shaw, "Characterization of Positive Resist", IEEE Transactions on Electron Devices, vol. 22, no. 7, July 1976.

[23] S. Leang, "Supervisory Control System for a Photolithographic Workcell", Memorandum No. UCB/ERL M92/70, Electronics Research Laboratory, University of California at Berkeley, July 1992.

[24] S. Leang and C. J. Spanos, "Statistically Based Feedback Control of Photoresist Application", ASMC, Boston, 1991.

[25] M. Watts, T. Perera, B. Ozaski, D. Meyer, and R.Tan, "Photoresist As its Own Process Monitor", Solid State Technology, p. 59, July 1988.

[26] Z. M. Ling and C. J. Spanos, "In-line Supervisory Control in a Photolithographic Workcell," Proc. SRC/DARPA CIMIC Workshop, August 1990.

[27] G. S. May, J. Huang, and C. J. Spanos, "Experimental Modeling of the Etch Characteristics of Polysilicon in CCl4/He/O2 Plasmas," Proc. IEEE Int. Electronics Manufacturing Technology Symp., October 1990.

[28] G. S. May, J. H. Huang, and C. J. Spanos, "Statistical Experimental Design in Plasma Etch Modeling", IEEE Transactions on Semiconductor Manufacturing, vol. 4, no. 2, May 1991.

APPENDIX A

Source Code Modifications in SIMPL-IPX

The following is a listing of SIMPL-2 and SIMPL-DIX source code files modified to allow the wrapper fit into SIMPL-IPX. All the modifications have been commented and initialed ("M. H.") by the author of this report:

- In SIMPL-2:

  ~/NEW_SIMPL/simpl.system.5/simpl-2/SYSTEM/Do_Process.c

  ~/NEW_SIMPL/simpl.system.5/simpl-2/SYSTEM/Initialize.c

  ~/NEW_SIMPL/simpl.system.5/simpl-2/SYSTEM/main.cc

  ~/NEW_SIMPL/simpl.system.5/simpl-2/PROCESS/COMMAND/DEPO.c

  ~/NEW_SIMPL/simpl.system.5/simpl-2/PROCESS/COMMAND/ETCN.c

  ~/NEW_SIMPL/simpl.system.5/simpl-2/MakeSIMPL2

- The "wrapper" is in:

  ~/NEW_SIMPL/simpl.system.5/simpl-2/

- The list of the files created for this purpose are:

  ~/NEW_SIMPL/simpl.system.5/simpl-2/EQUIP/Makefile

~/NEW_SIMPL/simpl.system.5/simpl-2/EQUIP/EQUIPFILE

~/NEW_SIMPL/simpl.system.5/simpl-2/EQUIP/do_models.cc

~/NEW_SIMPL/simpl.system.5/simpl-2/EQUIP/do_recipe.cc

~/NEW_SIMPL/simpl.system.5/simpl-2/EQUIP/do_functions.cc

- Some of the functions used in EQUIP directory are defined and supported by the codes in the following directories (developed by Bart Bombay):

  /home/radon1/users/bcamdev/BCAM/

- In SIMPL-DIX:

  ~/NEW_SIMPL/simpl.system.5/simpl-dix/src_x11/command_control.c

  ~/NEW_SIMPL/simpl.system.5/simpl-dix/src_x11/simpl_action.c

  ~/NEW_SIMPL/simpl.system.5/simpl-dix/src_x11/Include/command.h

In order to add new equipment models, the following directory is the place to start:

~/NEW_SIMPL/simpl.system.5/simpl-2/PROCESS/COMMAND/*

Most of the IC manufacturing processes such as expose, develope, etch, deposit, ion implant, diffusion, oxidation, etc. are represented in their appropriate source codes. The following is a listing of the existing source codes in that directory:

DEPO.c, DEVL.c, DIFF.c, ETCH.c, ETCN.c, ETCU.c, EXPO.c, IMPL.c, and OXID.c.

For example, if in future one decides to add a newly developed ion implantation equipment model to the MTCAD family of equipment models, the following files in SIMPL-IPX will be first rate candidates to begin with:

~/NEW_SIMPL/simpl.system.5/simpl-dix/src_x11/command_control.c

~/NEW_SIMPL/simpl.system.5/simpl-dix/src_x11/simpl_action.c

~/NEW_SIMPL/simpl.system.5/simpl-dix/src_x11/Include/command.h

~/NEW_SIMPL/simpl.system.5/simpl-2/PROCESS/COMMAND/IMPL.c

The changes in SIMPL-IPX are necessary, otherwise the newly developed equipment model that has already been stored in BCAM database will not be activated in MTCAD.