Microelectronic Engineering 15 (1991) 217–220
Elsevier

A NEW OPEN TECHNOLOGY CAD SYSTEM

F. Fasching, C. Fischer, S. Halama, H. Pimingstorfer, H. Read†,
S. Selberherr, H. Stippel, W. Tuppa, P. Verhas, K. Wimmer

Institute for Microelectronics, Technical University of Vienna, Austria
†ECE Department, Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

An implementation of a technology CAD environment is presented, using a level concept for the system: a data level provides simulation data retrieval facilities, a tool level is used for tool integration and a task level aids complex developments and user interaction.

1 Introduction

The recent changes in computation systems structure from centralized mainframes to distributed workstations have a strong impact on task methodologies applied in process, e.g. [1], and device technology optimization. Therefore our system as shown in Fig. 1 is controlled through a LISP interaction language, providing homogeneous integration of the data, tool and task levels. The format upon which the system is built is an enhanced and extended intertool mode of a widely used profile interchange format (PIF) proposed in [2].

2 Task Level - The TCAD Shell

Modules or tools are directly callable as shell functions of the interaction language, thus enabling programs written in this language to call these tools. This structure allows arbitrarily complex tasks to be performed, ranging from simply calling a single module interactively over coupling simulators via a shell function to running whole optimization loops as background processes. Starting tools or shell functions on different machines is also possible (distributed processing). A major influence in the decision to use LISP as the interaction language is that there is no distinction between program and data structures. This allows, for example, a process flow representation to be either executed directly in the shell as a program, or to be stored in the database as data. Task level programs can be executed in both interactive and batch mode. An example of a task level program for minimizing the bulk current of a device by means of varying the LDD implant dose is presented in Fig. 2.

A powerful extension is the User Interface Agent (UIA), shown in Fig. 1 on top of the TCAD shell, which allows graphical control of the TCAD system including editing, manipulating and viewing geometries, simulation results and symbolic process flow representations. Again, customizeability and homogeneous integration are ensured by a tight LISP coupling both to the windowing system (X11) and the TCAD shell. In addition, the experienced user can directly use the shell language to create new functions or modify existing ones. The environment does not depend on the graphical interface which is inherently system dependent. It could as well be used without it (terminal capability is enough), although it is more convenient to use the UIA.

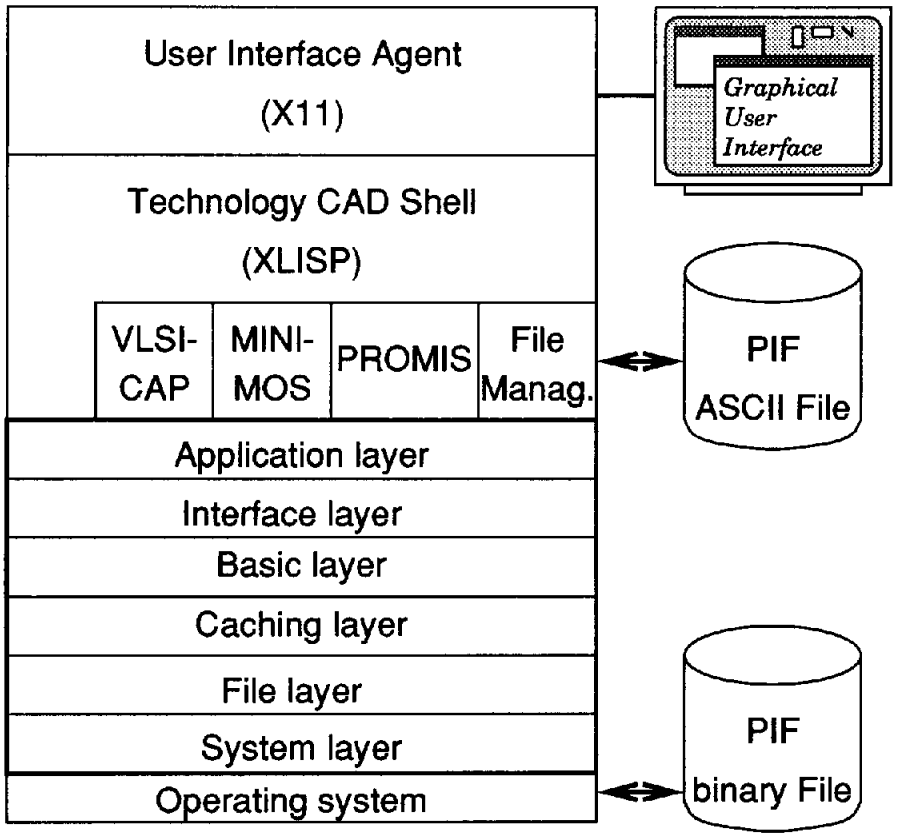

Figure 1: TCAD System Overview

3 Tool Level - The Workhorses

Modules callable at the tool level include all kinds of simulators (process, device, circuit), grid manipulators, discretizers, solvers, measurement data translators, optimizers, graphical editors, previewers, etc., some of which are incorporated in a PIF toolbox which also provides for intersite to intertool format conversion and vice versa. Until now the

```
;;; sample TCAD shell task level function
(defun minimize-i-b (LDDimpl-dose obj-hdl)
   (run-minimos obj-hdl)
   (setq i-b-act (extract obj-hdl "BulkCurrent"))
   (do ((test-criterion i-b-act LDDimpl-dose obj-hdl))
      (setq LDDimpl-dose (compute-new-dose i-b-act LDDimpl-dose obj-hdl))
      (run-minimos obj-hdl)
      (setq i-b-act (extract obj-hdl "BulkCurrent"))
   )
)

;;; sample usage
(setq obj-handle (ped))              ; edit a geometry
(minimize-i-bulk 5e12 obj-handle)    ; call the function
0.01132                              ; the output i-bulk/i-drain [%]
```

Figure 2: Sample TCAD function

device simulator MINIMOS, e.g. [3], the process simulator PROMIS, e.g. [4], and the interconnect capacitance simulator VLSICAP, e.g. [5], have been integrated.

New tools can be added very easily by replacing input and output functions with corresponding application layer functions (see below). This small change yet allows data level integration into the TCAD system. The full power of the system can be exploited if the new simulator is provided with a shell language interface which allows the required data to be passed in form of PIF object handles. Additional flexibility can be gained by splitting the simulator (e.g. separating grid generation, discretization and solver parts) and by combining the new modules with existing TCAD tools into task level programs almost arbitrarily. To do so, the modules must adhere to a data format convention [6]. The major advantage when building a new simulator is that it is no longer necessary to provide a specific grid generator, solver, etc., since these tools are readily available on the shell level. Therefore, simulator designers are able to concentrate on the specialized parts of simulator construction. The executable modules are usually small and can be run (in parallel) on different machines under control of the TCAD shell, thus yielding considerable speed improvement. When modularized appropriately, the most time consuming parts (e.g. linear solvers) can be executed on a supercomputer communicating with the TCAD shell running on a graphics workstation using our PIF linear solver communications protocol [6] and the networking facilities of the PIF Application Interface described below.

4 Data Level - The Database System

The full flexibility of the TCAD shell extension language is retained even in the binary file structure of the TCAD database, since it resembles LISP structures but is also designed

to deal with (compressed) arrays thus ensuring compatibility with existing applications from the very bottom of the system. This allows also the LISP-like syntax of the intersite PIF to be readily transformed into its binary (intertool) representation.

The TCAD database is accessed from programs with the help of an application interface (Fig. 1). Our implementation of this interface is strictly layered thus conforming to the most recent software engineering standards. A *system layer* at the very bottom is used to hide system specifics from the rest of the application interface. Thus the interface is open to all operating systems and not restricted to UNIX. Networking capabilities are included in the file layer, allowing the realization of server-client concepts not only for TCAD data but also between applications. Additionally, distributed processing facilities can be exploited through communication between TCAD shells. A *caching layer* implementing a segment-buffer caching algorithm sits on top of the file layer, which significantly enhances access speed and memory utilization, allowing data security aspects to be taken into account through write-through files as well as full memory caching e.g. for temporary data. The *basic layer* then is used to access primitive objects which hold primitive data types as well as efficiently compressed arrays. The *interface layer* deals with PIF objects which are made up of primitive objects. This interface layer is designed to work with C applications specifically designed for PIF. To provide a consistent interface with existing C and FORTRAN applications, an *application layer* has been designed which deals with all TCAD objects based on PIF. Therefore adding any new TCAD tool (simulator, measurement interpreter, correlator, etc.) is a simple and straightforward task.

Acknowledgements

This project is supported by the research laboratories of: AUSTRIAN INDUSTRIES – AMS at Unterpremstätten, Austria; DIGITAL EQUIPMENT at Hudson, USA; SIEMENS at Munich, FRG; and SONY at Atsugi, Japan.

References

[1] E.W. Scheckler *et al.*, *A Utility-Based Integrated Process Simulation System*, Symp. on VLSI Technology, pp. 97-98, 1990.

[2] S. Duvall, *An Interchange Format for Process and Device Simulation*, IEEE Trans. CAD, Vol. 7, pp. 489-500, 1988.

[3] S. Selberherr, *Three Dimensional Device Modeling with MINIMOS 5*, Proc. VLSI Workshop, pp. 40-41, 1989.

[4] G. Hobler *et al.*, *RTA-Simulation with the 2D Process Simulator PROMIS*, Proc. NUPAD III, pp. 13-14, 1990.

[5] F. Straker *et al. Capacitance Computation for VLSI Structures*, Proc. EUROCON, pp. 602-608, 1986.

[6] F. Fasching *et al.*, *Viennese Integrated System for TCAD Applications*, Institute for Microelectronics, 1990.