Editor: Charles Radke, IBM 47A, Route 52, Hopewell Junction, NY 12533

IEEE International Conference on Computer Design

C.E. Radke

Although the IEEE International Conference on Computer Design (ICCD '85) reflected in its attendance the slowdown in the computer and semiconductor industries, it made up for the drop in attendance with high-quality papers. Long known for exemplary invited speakers and simulation panels, this year's event, held in Rye Brook, New York, October 7-10, 1985, maintained the tradition of excellence. The conference was sponsored by the IEEE Computer Society and the IEEE Circuits and Systems Society. This multidisciplinary sponsorship was reflected in the four technical program tracks: technology and VLSI, circuit and computer design, CAD, and computer processor architecture. The conference stressed integration among system design, circuit design, testing, packaging design, VLSI technology and CAD. General chair of this year's meeting was Albert Ruehli. Kenneth Laker, who this year served as technical program chair, will be general chair for ICCD '86.

Testability and manufacturability

It has become a tradition to open ICCD conferences with parallel plenary sessions of invited speakers. This year, two individuals familiar to the computer-aided design and test scene addressed subjects that, upon inspection, appeared similar. In brief, the papers dealt with design for testability, and integration of design with manufacturability and test.

In his talk, "Fault-Tolerant Computer and Microprocessor Design," Constantin Timoc of Spaceborne cited several testability problems inherent in dynamic CMOS circuits. He indicated that these problems were quite different from the testability problems of stuck-opens for static CMOS circuits, in which the resultant capacitance creates a latch out of the logic gate.

Although he classified these as real test problems, he proposed several methods to minimize their effect for PLA designs. He covered such defects as hot electrons, hot holes, source-to-drain short, gate-to-channel shorts (even with polysilicon), and open connections to transistors. He limited his investigation to the solid defects and did not handle delays, which he indicated could be difficult to test. In the four-level circuit, he discussed device stuck-opens, slow-ons, slow-offs, and for connections the usual opens and shorts.

In a dynamic CMOS there is no stuck-open problem in the function or input transistors, as there is for static logic, and the function transistors are completely tested. In the precharge transistor, defects are also fully tested, but by a sequence of three patterns. However, Timoc pointed out, the testability problems increase with defects in the sustainer and discharge transistors.

For the sustainer transistor the stuck-off fault operation depends upon the leakage, which is increased by high temperature or a low clock frequency. It becomes difficult to test for the resulting delays.

For the discharge transistor the stuck-on fault is a real problem, Timoc noted. Only a parametric test of some sort could detect the excessive current. This fault, undetected, poses a potentially serious problem; because of the resultant excessive current involved, it could pose a two-to-three-year reliability factor.

To solve these difficulties, Timoc "designed out" the discharge and sustainer transistors as much as possible. His conclusion that "it is feasible to design a testable dynamic CMOS PLA," while successfully demonstrated, left open the implied question: What about dynamic CMOS in general?

Steve Director, who heads the CAD research center at Carnegie-Mellon University, entitled his plenary presentation "Towards Integrated CAD/CAM/CAT." He indicated that there is interest in automatic processing but little coordination of the process with the circuit design. Similarly, for CAT there is automation at the logic function but no feedback into the circuit design.

He viewed CAD, CAM, and CAT as relatively independent today, but he visualized a manufacturing-based CAD as aiding in yield prediction, yield maximization, statistical design rules generation, and process diagnosis. In a like manner, manufacturing-based CAT allows fault analysis.

Director's efforts aimed at modeling the IC manufacturing process, which he considered the first step. He looked at process disturbances, their physical nature, and electrical effects. Defects, he said, can lead to all types of failures, but these related failures are usually limited; this limitation aids in predicting yield. He talked in terms of functionally and parametrically correct processes and their respective yields.

Functional yield was affected by disturbances that cause hard failures and structural faults, Director said. Key elements were defect statistics such as frequency of occurrence versus diameter, defect densi-

Merlin Smith, former president of the Computer Society, expresses the Society's appreciation for a job well done with a plaque to Albert Ruehli, general chair of ICCD '85.

ties, and radial distributions. CMU's simulator VLASIC is essentially a functional yield calculator given defect distributions. Some of its uses are to optimize design rules, generate meaningful test vectors, and evaluate redundancy.

Parametric yield is calculated using FABRICS, a statistically based process and device simulator, he explained. The objective is to center the design to provide maximum parametric yield. The high computational cost for conducting circuit simulation is one of the limits to achieving satisfactory parametric yield calculations.

Director's main objectives were to classify process disturbances and faults, relate those disturbances to those faults, and thereby develop intelligent testing procedures. "What tests are important?" he asked, and visualized four levels: process, structure, circuitry, and logic.

There is a need to integrate CAD, CAM, and CAT, Director said. The first step is to develop a manufacturing-based CAD and CAT. Yield prediction is important during design, and yield estimation is important during manufacturing. Director stressed that semiconductor companies need to ensure that the right people interface. This is a political rather than a technical problem, he concluded.

Editor's note: ICCD '86, organized in cooperation with the IEEE Electron Devices Society, will be held October 6-9, 1986, in Rye Brook, New York.

CAD: Myth or Methodology?

Computer-aided design holds great promise, and many engineers believe that it will take us into the future without substantial problems. But can CAD really do so, given that package density increases tenfold every five years?

Yes, according to Peter Bottorff, manager of design systems at IBM's Endicott facility. Speaking during the opening luncheon at ICCD '85, Bottorff cited the 50 percent growth rate projected for workstations within the next two years to support his claim.

Workstations number some 4540 in the US in 1985, with today's market estimated at $367 million; those figures are expected to rise to 49,500 workstations and a $1.6 billion market by 1987, he said. Bottorff cautioned, however, that those estimates could be affected by the recent slowdown in the semiconductor and computer industries.

Each design phase holds both good news and bad news, he noted. Design capture, for instance, offers the many improvements that have greatly increased productivity while reducing mistakes. Unfortunately, design capture retains its poor data handling capabilities and poor compatibility with other design tasks, and it is adversely affected by the general lack of horsepower encountered in current workstations; in general, he indicated, the design capture phase has been incomplete.

Design validation offers a wide variety of simulation tools, most of which are easy to use and relatively accurate. Hardware accelerators also are coming on board. However, he added, the tools still are not fast enough, they need hierarchical models to handle densities and complexity, and no one has established a measure of when "enough is enough" in simulation.

Peter Bottorff, keynote luncheon speaker, received thanks and a plaque from Walt Luciw, special sessions chairman for ICCD '85. In his presentation, Bottorff questioned whether or not some of the accomplishments reported in CAD were really myths.

Bottorff lamented the lack of specific tools for timing analysis, noting also that despite substantial interest and progress in logic synthesis, an automatic process is still a long way off. The output is not optimized to include timing, physical realization, and testing, and synthesis tools to assure correctness have yet to be developed. Much unnecessary time and effort is needed to correct automated synthesized design, he said, questioning the real productivity.

Bottorff did praise the state of the art of physical layout, citing these tools as the most dependable in the designers' kit. He cautioned, however, that there are signs of strain appearing: too much effort is already required to route overflows, and the amount is increasing rapidly. It becomes ridiculous to take hours to route one wire, and the situation is causing missed schedules. Despite increased densities, the same old tools are being used.

Stressing that most of his career has been spent in the test area, Bottorff left the "good news" category blank—there is none! Test generation, he said, is a rough area because current tools have run times that are equal to a factor raised to a power equal to the current density. The ATG programs are purely and simply "running out of gas." Built-in self-test is a promising alternative, he added, but that doesn't work yet.

Systems and databases were not left unscathed as Bottorff stressed the lack of user interfaces and commented that a collection of data sets hardly passes as a database. He emphasized an overall lack of focus.

In conclusion, he indicated that there must be a focus on long-term goals: system solutions are needed, and automatic tools are a key to productivity, which is really what CAD is all about. The need, he concluded, is not for more expensive "toys," but for efficient, economical design systems—"tools, not toys."

—C.E.R.

S. Joy Mountford

A talking workstation

S. Joy Mountford's ICCD paper "Application of Speech Technology in the CAD Workstation" advocates several potential uses of speech techniques in the CAD environment. Mountford, now with the Microelectronics and Computer Technology Corporation of Austin, Texas, has spent years trying to apply speech technology to various human factor situations including the airplane cockpit, which she compared to a designer's workstation. She discussed speech recognition, generation, recording (in digital form), understanding (AI), and identification.

One reason given to use voice recognition in aiding eyes and hands is to improve the handling of large graphic display menus in CAD. An interesting use of speech recording is to keep a designer's notebook since one is often too busy to write down notes. Within the design layout on the workstation screen a "voice mark" may be inserted so that "notes" may be revisited later. Speech generation and understanding are useful in warning of design violations, or in verbal prompting (for example, "where was I?").

Mountford believes that improved productivity in the CAD environment is achieved through the use of speech technologies; furthermore, she envisions the application of all speech techniques. Through human engineering and speech technology, efficiency can be substantially increased, errors monitored, and documentation facilitated. Best of all, increased productivity can be realized. The applications of a talking workstation sound more appealing than a talking car—and are certainly more productive.

—*C.E.R.*

CAD Database Without Serialization

Serialization of transactions in design is not always required in CAD databases, according to Hank Korth of the Computer Science Department at the University of Texas, Austin.

In presenting his paper, "Transaction and Concurrency Control in CAD" to ICCD '85 attendees, Korth contended that CAD databases are quite different from traditional databases such as those found in banking. Although in CAD there is a need for shared access and collaboration among team members, the basic difference is that the transactions are significantly longer. In addition, people are involved *within* the transaction rather than *between* transactions.

In CAD, databases are distributed; there is a need to support separate workstations in which data is checked out from the public database into a private database, then checked back in after modification. There are multiple levels of transactions as well as levels of recoverability, hence Korth's conclusion that a hierarchy of databases is required.

Korth pointed out that published work in the area is generally unclear. Current models do not allow recovery upon sharing. Independent projects must be protected. The general overview of designs and designers must be maintained. The concept of coordinating transactions—or sharing of incomplete results—must be introduced: one needs, possibly, to initiate subcontracting a subtask to another design, and to provide for regulation of that process.

In serialization, a two-phase locking takes place, that is, the transaction cannot request a lock after it has released a lock. Korth proposed a 3-tuple: transactions or operators, a partial order, and a set of integrity constraints preserved by a transaction. As a result, he said, designers are allowed to work as a team and yet preserve consistency. This concept or model, he noted, allows for new techniques for transaction management and permits recognition of the difference between the traditional database environment and that found in CAD.

—*C.E.R.*

Bryan Ackland, presenter of the paper "CADRE—A System of Cooperating VLSI Design Experts," accepts a plaque and audience recognition for the best paper in the CAD session. The plaque was presented by Kenneth Laker, technical program chair for ICCD '85. Ackland's coauthors were A. Dickenson, R. Ensor, J. Gabbe, P. Kollaritsch, T. London, C. Poirier, P. Subrahmanyam, and H. Watanabe. Best paper winner in the VLSI Technology session was author/presenter John Y. Chen, for "The Emerging VLSI Technology." Authors Tam-Anh Chu, T. Wanuga, and C. K. C. Leung won best paper honors in the Design & Test session for their paper, "A Design Methodology for Concurrent VLSI Systems," while in the Architectures & Algorithms session, "An Algorithmically Flexible Systolic Array," by M. K. Williams and D. A. Carlson, was selected.

Simulation and test generation environments workshop

Saied Bozorgui-Nesbat and William Lee, Factron/Schlumberger

This was the first of a series of biennial workshops dealing with simulation and test generation environments, and was suggested by the Automatic Test Generation workshop series example. The new workshop series addressed issues outside the core of simulation and test generation algorithms, serving as a forum for the discussion of issues concerning test engineers and managers.

Some of these issues were: algorithms used in performance simulation and test, support software including languages used and maintenance techniques, and hardware—specifically the increasingly popular workstations and means for networking them.

This workshop was sponsored by the IEEE Computer Society and the Test Technology Technical Committee, was chaired by Cihan Tinaztepe of Factron/ Schlumberger, and was held in San Francisco September 17-18, 1985.

Programming languages. Keynote speaker Peter Deutsch of Xerox defined an environment as more than a set of software tools and consisting as well of the social/organizational context, the programmer's knowledge, the physical environment, software and hardware user interfaces, libraries, languages, operating systems, and hardware. Deutsch emphasized that an environment may be viewed as the totality of these elements; to improve the quality of the programming environment, therefore, programming tools and other elements must be both examined and improved.

The trend is towards larger programs with greater capabilities; to remain manageable and productive, we must integrate the programming language with the tools. As computing power becomes cheaper, tool-set and language integration becomes more feasible, more affordable, and more necessary. Ultimately, a good environment should fade, so that the programmer can concentrate on problem solving rather than programming.

Deutsch distinguished between functions that are timeless mathematical expressions, and procedures that are chronological reflections of the real world. He went on to note that most programming is procedurization, which he defined as the mapping of highly functional mathematical entities to procedures and programs through the incorporation of time and state. These procedurized parts must be integrated to create a large system, but there is currently little theory for building systems out of parts in this way. Nevertheless, Deutsch pointed out, program development environments are already migrating to support application programming. Programs need to be designed in common frameworks decided upon early in the project. On this subject, Deutsch recommended Paul Heckel's book, *Design of User-Friendly Software*.

To obtain customer satisfaction, the ultimate measure of product quality, we need to balance theory and practice. Deutsch suggested using programming apprenticeships to achieve this goal. He predicted the coming domination of bit-mapped terminals and workstations; consequently, rather than building for the lowest common dominator in available software and hardware, it is better to anticipate the widespread use of powerful workstations where the computer waits for user response rather than the reverse.

Marla Bollak of Bell Laboratories underscored Deutsch's position by pointing out that changes made by software developers working on a large program may produce some unforeseen side effects, and that changes made by several software developers working on the same program may produce many clashing side effects.

Simulation, test, and hardware acceleration. Hardware acceleration can reduce verification problems to manageable levels, according to second keynote speaker Rob Mathews of Silicon Solutions. Verification problems exist in each of the three areas of design, manufacture, and test. During design, hardware acceleration can enhance circuit and fault simulation. During verification for manufacture and test, hardware acceleration can offer substantial computing power—once solutions to verification problems have been found.

Machines like Daisy's MegaLogician offer highly flexible, improved speed systems at modest cost through the use of general-purpose multiprocessors. Zycad's Logic Evaluator offers high- speed, inflexible systems at high cost through the use of custom-dedicated hardware. According to Mathews, Silicon Solution's Mach 1000 mixes the two approaches by using custom ICs to provide speed and general-purpose control processors to provide flexibility.

There is a bottleneck in generating cost-effective and efficient hardware accelerators for currently available tools. In fault simulation, translating available practical tools into usable ones for the marketplace is the bottleneck. But for automated test pattern generation, or ATPG, the bottleneck is more severe, as it joins the realm of research and the pragmatic world. Many fundamental problems must be solved before a usable ATPG tool can be made available.

During the Mathews question and answer period, John Newkirk observed that, as logic and fault simulation times decrease, netlist compilation will become the next bottleneck (hence the next target) for hardware acceleration.

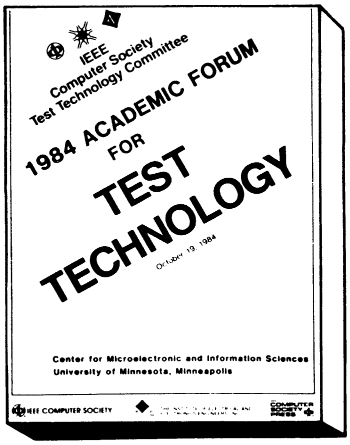
Design and test integration. Test development can no longer be considered separately from design; design and test planning must proceed concurrently. Further, as posited by Ken Van Egmond of VLSI Technology, it is a misconception that design simulation output can be processed to generate a test program since tester limitations are currently ignored by the simulation.

In general, much information is lost by translating design simulation results into a test program; a solution is to familiarize design engineers with test procedures and tester limitations. While the worlds of design and test remain apart, and improvements tend to be made in each world separately, a system linking design and test at three distinct levels has been developed. These levels are: (1) better communications between the two domains, (2) better education of the design engineer concerning test issues, and (3) better application tools to bridge the design/test gap.

Testability checking and design. Knowledge and rule-based systems are used in a testability rule checker and test generator—the main advantages being that testability rules are checkable, expandable, and reusable. Kyushik Son of GenRad Design Engineering observed that these rules are in frame-based structures containing either topological or semantic information, a frame being a data structure for knowledge representation. The rules determine how sequential circuits can be broken into pseudocombinational circuits and how test patterns can be chained. A similar structure is also used for test generation, unstructured design testability rule enforcement, and tester rule verification and checking.

Stanford Professor Edward J. McCluskey's address dealt with a design for testability using a cell library. Many industrial semicustom and custom parts are designed in the absence of strong design rules checking discipline. For these circuits, a library of cells and macrofunctions can be provided that include scan path features. Replacing existing cells of the chip with their corresponding parts from this library will create the scan path required for testability. In a recent application of this method, the size of a chip was actually reduced.

Melvin Breuer of USC described the development of a system, ADAM, to assist in choosing among testability enhancement techniques of VLSI chips—PLA circuits in particular. ADAM has a knowledge base containing information about seven of the approximately 30 design-for-testability methods for PLAs. Accepting as input a description of the PLA as well as user requirements and restrictions, ADAM is failure driven; given stringent requirements, it announces the failure to find a completely acceptable solution and then helps the user modify some of his requirements to reach the ‘nearest‘ solution.


IEEE Computer Society Test Technology Committee
1984 ACADEMIC FORUM FOR
TEST TECHNOLOGY
October 19, 1984
Center for Microelectronic and Information Sciences
University of Minnesota, Minneapolis
IEEE COMPUTER SOCIETY

Test generation. Sentry Digital Test System's Kerry Kurasaki presented the merits of test definition language, or TDL, the basis of an automatic test generation tool. Using TDL, test program generation software increases test engineer productivity by transferring test-system expertise to a rule-based system.

Consider the following inadequacies: current test programs are not transportable; there is usually no DUT definition; tester limitations are hopelessly intermingled with DUT parameters; test engineers need to be expert with both the DUT and the tester; finally, there is no tester-independent test language on the market. TDL will solve most of these problems by providing a non-procedural description medium with the emphasis on DUT definition. TDL's structured tabular format provides a natural form for test parameters, DUT definition, and simulation and test strategy definition.

A growing problem—the cost of test generation—was addressed by Major Phil Gordon of the USAF. The Department of Defense proposes, as a solution, the integration of test specification and generation. Although still in the early design stage, the tester independent support software system, or TISSS, will be tester and simulator-independent, deriving its data directly from CAD tools, automatically translating the specifications to test programs.

Mark Shirley of MIT discussed the impracticality of classical test generation methods for large and complex circuits; the experiences of a test programmer can be encapsulated in the form of cliches from which test program fragments can be generated—fragments that can then be automatically combined within applicable temporal, structural, and resource constraints to produce a test program.

Alternatively, a simulator such as MARS can be used to convert testing goals into event patterns from which code can be extracted. Both techniques emphasize test program construction rather than the currently pervasive use of random test pattern generation.

To alleviate design-test cycle difficulties and to reduce testing costs, Louis McDonald of Hughes pointed out, software tools have been developed to assist users in test generation and evaluation tasks. A preprocessor, used in design rule checking and circuit analysis, partitions the circuit into combinational parts and set-scannable sequential parts; combinational circuits are further divided if necessary. An ATPG module then creates tests, which are graded using a fault simulator. This system, according to McDonald, is currently being refined.

Traditionally, ATPG programs depend on the sensitization of a single path that leads to numerous searches the algorithm must conduct to produce tests for all elements in the DUT. Based on research he has conducted with Mahieddine Ladjadj, Renssalaer Professor John McDonald said that gang testing accelerates the rate of test pattern generation by finding patterns that simultaneously sensitize many paths for many circuits—an acceleration achieved by using the subscripted D-algorithm that sensitizes all control and observation paths of a gate and assigns flexible values (signals that can be set to 0 or 1 as the need arises) to these paths.

To reduce conflicts while preserving the independent identities of the mergers, merging of the observation paths at various gates is allowed. Examples of gang testing, the D-algorithm, and the improved D-algorithm demonstrated gang testing's superiority—improvements quantified by the number of test patterns needed and the time required to achieve a desired fault coverage.

Causal reasoning diagnosis. Using primary input values and observed inputs, logical inference can be used to identify potentially faulty elements—elements that can be tested and eliminated based on logical inconsistency between expected and observed values. Therefore, according to Bruce Havlicsek of Westinghouse, system fault diagnosis can be performed without the use of explicit fault models or test programming.

Diagnosis covers several inference phases. Primary inputs are set in the first phase and the correct values of circuit nodes are determined. Observations are then entered and potential fault candidates are determined by noting inconsistencies between expected value and inferred value. During the next inference phase, the behavior of each candidate is disabled; if inconsistencies disappear, then the candidate can be faulty—if inconsistencies persist, then the disabled candidate cannot be faulty.

Some audience members noted the use of a fault model despite the no-fault-model claim, since only single faults are handled and no time dependencies are assumed.

Workshop conclusions. Some of the topics discussed in the 1983 ATPG workshop on expert systems appeared as prototype working systems in this year's workshop. At the next workshop, we may hear about hardware-accelerated test generation algorithms.

An interest was expressed in conducting panel discussions, and in continuing to conduct the workshop every other year rather than annually.