



Guest Editors' Introduction New VLSI Tools

Daniel D. Gajski, University of Illinois
Robert H. Kuhn, Gould Research Center

This special issue of *Computer* is about the methodologies and tools that may help us solve the VLSI design crisis of the 1980's. Basically, there are three approaches. The first school of thought believes that all design decisions should be made solely by the human designer, who has gained experience through good design practices in the last 10 or 20 years. This approach, called computer-aided design, gives the designer an "efficient paper and pencil" by providing graphic editors, design verification and simulation tools, and efficient databases. This approach is evolutionary and tends to be bottom up, since building blocks are first designed and then later used to realize higher level structures. The resulting design is of high quality, since humans are very good in optimizing designs. On the other hand, the human designer is slow and error-prone. Furthermore, creative designers optimize design by creating new design rules, thereby creating a demand for new verification tools and design description languages for documentation and communication between designers.

The other approaches to dealing with the VLSI design crisis do not model the "mechanical" aspects of the design; rather they focus on the "cerebral" aspects. The second school of thought believes that human knowledge can be captured in the form of design rules and stored in

a knowledge base. A VLSI expert system utilizing the knowledge base can be constructed to help a human create a design. Expert systems disseminate the knowledge of a precious resource, expert human designers. Also, expert systems are less error-prone and, when finely tuned, are more efficient than the human designer. (Moreover, they do not get bored or hired by your competitor!) Also, knowledge can be incrementally added to an expert system, or new technologies and design styles can be accommodated by adding rules to the system's knowledge base. Most existing expert systems, on the other hand, are efficient only in analyzing or critiquing a design. The IC industry is only starting to tap the possibilities of expert systems that synthesize or plan designs.

Supporters of the third approach for design synthesis in the practical IC design world believe that knowledge is algorithmic and that translators can be written to generate (synthesize) the solution or some part of it automatically from a high-level description of the problem. Examples of such programs are placement and routing tools and silicon compilers. This top-down approach tends to be revolutionary, since it displaces human designers instead of assisting them in the design cycle, thus introducing methodology completely opposite to the CAD approach.

Basically, the second and third schools try to capture the same expert knowledge. The difference is in the granularity of the knowledge used. Expert systems tend to have several hundred rules with a small piece of knowledge in each. The underlying structure of the problems to be solved tends to have little regularity, and design space and problem complexity are large. Expert systems mimic human designers who are very efficient at optimizing large design spaces.

In the third approach, however, algorithmic formulation is used to solve problems with predominantly regular structure for which only a few general but complex rules are applicable. The design space is made smaller by formalizing the problem statement; the complexity is low to medium. For example, PLA layouts are simple, regular structures in which only a few rules apply. PLA layouts can be generated by a PLA expert system whose rules are on the level of individual PLA cells (simple transistors, contacts, and connects). This system would be inefficient in comparison with a PLA compiler generating a compact layout from Boolean expressions used to describe the PLA's function. Such a PLA compiler's knowledge has been embedded into the subroutines and modules (such as PLA folding) that implement the algorithm.

A comprehensive view of VLSI CAD tools

How are individual VLSI tools applied to move a design from a high-level specification to the low-level mask data? The principle presented here is based on a tripartite representation of designs. Along each of these three axes—functional, structural, and geometrical—design refinement is conducted in steps or levels.

Functional representation. At the highest level, the customer is interested in what the chip does and how it is built. For example, the Boolean expression $x = a'b + ab'$ indicates only the function of the design whose inputs are a and b and whose output is x . This expression does not

say anything about the implementation or the structure of the cell. The functional representation of a design may be captured on several levels, the most widely accepted being systems, algorithmic, and Boolean expression. (Here, we consider "functional" to be synonymous with "behavioral.")

Structural representation. A structural representation is the bridge between the functional representation and the geometrical representation. It is a mapping of a functional representation onto a set of components and connections under constraints such as cost, area, and time. If, for example, the Boolean expression $x = a'b + ab'$ (Figure 1a) is mapped onto a set of components consisting only of two-input NAND gates, then one of the structural representations consists of the four NAND gates shown in Figure 1b. This representation does not specify any physical parameters, like the positions of the four NAND gates on a printed circuit board or silicon chip. Sometimes, the structural representation may serve as a functional representation. For example, logic or schematic entry is a structural representation that can serve as a functional representation; that is, it can be simulated to determine its timing behavior. Commonly used levels of structural representation are the processor memory switch, the register transfer (operator register bus), and the circuit.

Geometrical representation. The final representation ignores, as much as possible, what the design is supposed to do and binds its structure in space (physical design) or to silicon (geometrical design). For example, if a gate array consisting of two-input NAND gates is used, then a possible binding of the structural representation from Figure 1b is one in which every NAND gate and every connection is assigned a physical location (Figure 1c). Geometrical representation levels are layout planning with arbitrary size blocks, cell, and physical mask geometries.

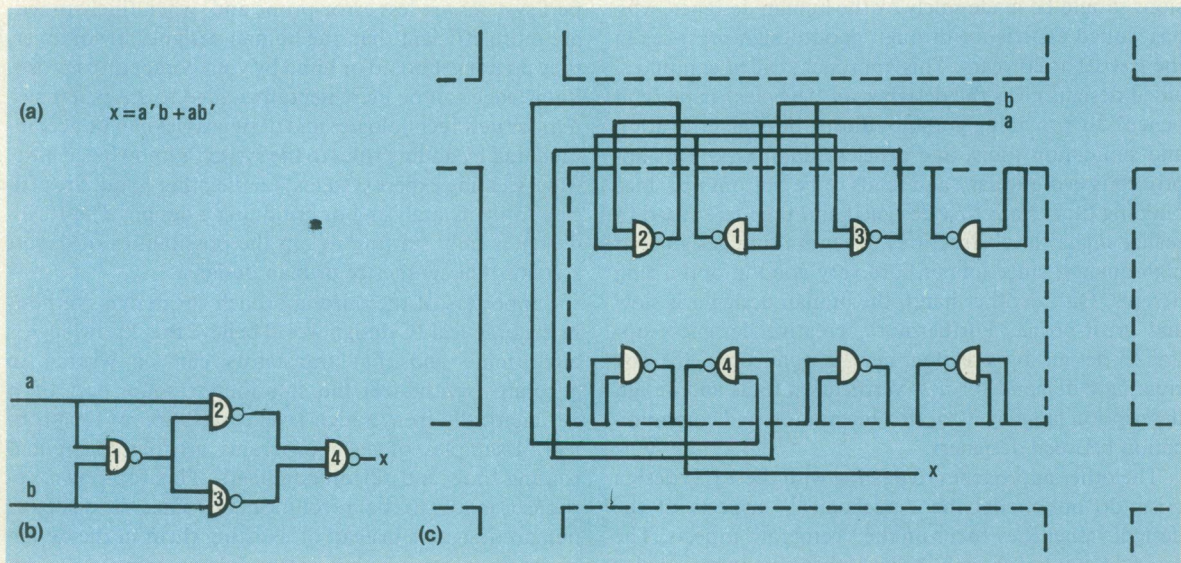


Figure 1. Functional (a), structural (b), and geometrical (c) representation of a Boolean expression.

Figure 2 shows the levels of design refinement of the tripartite design representations (Y chart). In synthesizing a design from the top down, the designer focuses in, from a high-level representation of what the chip is supposed to do to the geometries that go into the mask. Sometimes, a design does not need all three representations especially when the designer is starting from a

schematic diagram that implicitly contains a functional description. However, all three representations are used in a well-rounded chip design, and it is useful to examine the design from several complementary viewpoints.

Synthetic CAD tools are represented in this view as arcs from one representation and level to another representation or to a lower level. For example, an arc from circuit to mask geometries represents a synthetic tool that generates circuit geometry while a reverse arc corresponds to an analytical tool such as a circuit extractor. A loop on the circuit level represents an analytical tool such as a circuit simulator while a similar loop on the mask geometries level represents a design rule checker.

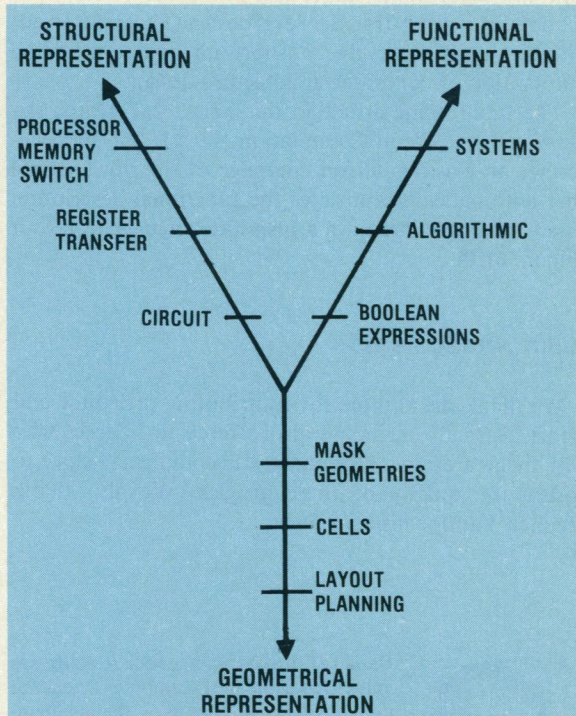


Figure 2. Tripartite representation of design and various design levels. The arcs in the figure represent two feedback loops a designer can use to verify the design.

In this issue

This special issue of *Computer* is a collection of articles depicting all three approaches to VLSI design tools, each representing a different methodology in VLSI design.

The first article, "Masterimage Approach to VLSI Design," by R. Donze and G. Sporzynski represents a state-of-the-art traditional CAD, the first school of thought (Figure 3). The emphasis is on generating mask data from a structural description of cells and macrocells that are taken from a cell library. Some of the macrocells, such as PLAs, are compiled from a functional description. (Their PLA compiler is a component from the third school.)

In the second article, "Managing the Chip Design Database," R. Katz describes the requirements for management of design information in a traditional CAD environment. (There is no Y chart for this article, since it

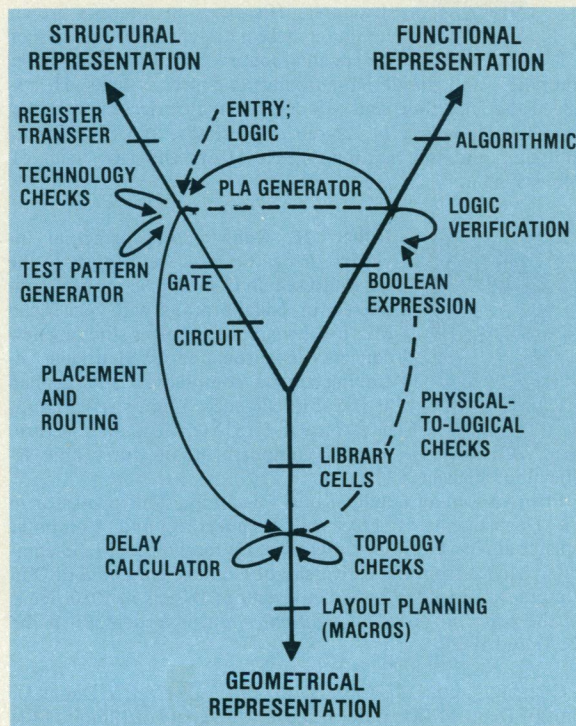


Figure 3. Masterimage methodology.

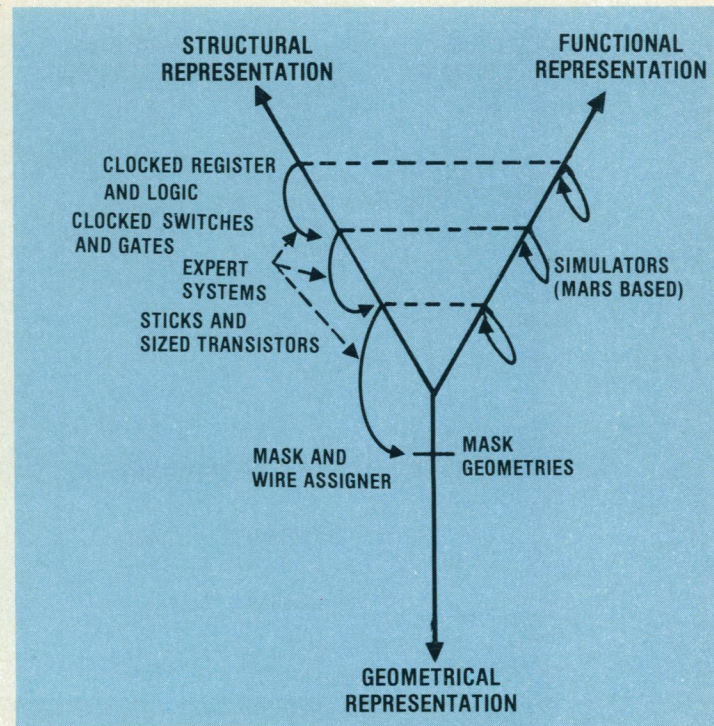


Figure 4. A possible design methodology in Palladio.

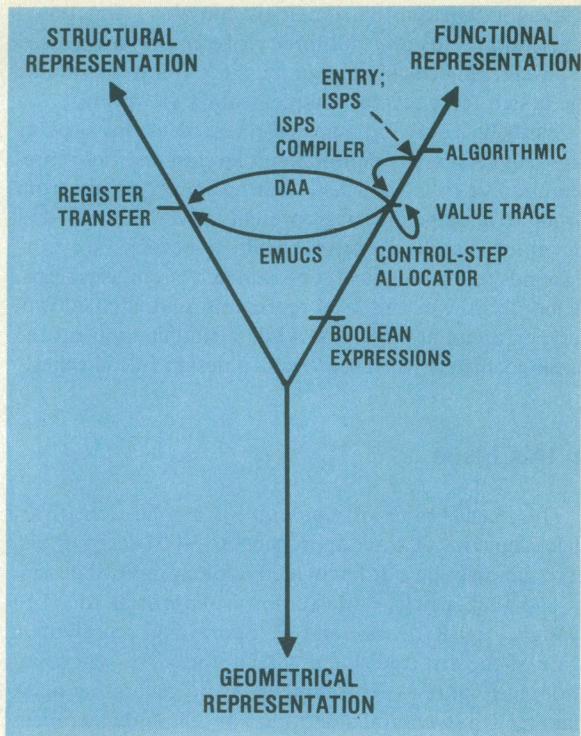


Figure 5. CMU's data-path methodology.

deals with data management and not a particular design methodology.) Next is "Palladio: An Exploratory Environment for Circuit Design," in which H. Brown, C. Tong, and G. Foyster explain how expert system technology can be used in VLSI design. Their system, representative of the second school of thought, has design levels

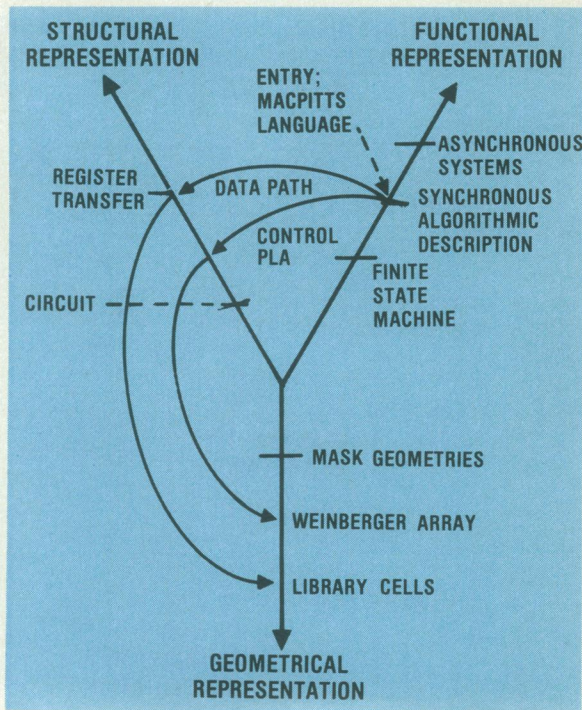


Figure 6. Methodology of the MacPitts silicon compiler.

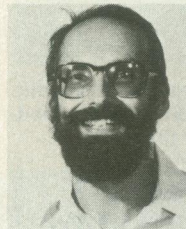
that can be defined by the user (Figure 4). Although translation between levels is primarily the designer's responsibility, expert systems can be used to assist in the translations as well as to refine a design within a level.

"Methods of Automatic Data Path Synthesis" by D. Thomas, C. Y. Hitchcock, T. J. Kowalski, J. V. Rajan, and R. A. Walker compares the expert system approach and the algorithmic approach, the second and third schools, in register-transfer-level design (Figure 5). In addition, the authors give a short introduction to the automation of computer-architecture design.

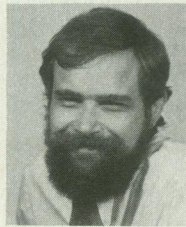
The final theme article in this issue, "MacPitts: An Approach to Silicon Compilation," by J. Southard describes an existing silicon compiler of the third school that automatically translates the functional description of an integrated circuit in a Lisp-like language to layout (Figure 6). ■

Acknowledgments

We thank the authors for contributing their time and effort. Special thanks are also offered to referees who with their careful reading and useful comments helped to ensure the quality of these articles. We also thank Stephen Yau for his assistance.



Daniel D. Gajski is an associate professor in the Department of Computer Science at the University of Illinois, Urbana-Champaign. Before joining the university in 1978, he had 10 years of industrial experience in digital circuits, switching systems, supercomputer design, and VLSI structures. His research interests are in computer system design, algorithm design for supercomputers, hardware and silicon compilers, and the design-automation expert systems. He received the Dipl. Ing. and MS degrees in electrical engineering from the University of Zagreb, Yugoslavia, and the PhD in computer and information sciences from the University of Pennsylvania.



Robert H. Kuhn is the principal investigator on the IC CAD project at the Gould Research Center. The project, conducted in collaboration with American Microsystems, Inc., involves studying new concepts in automated VLSI design, including silicon compilation and applied artificial intelligence. Kuhn's research interests are in IC CAD, VLSI architecture, parallel computer architecture, and artificial intelligence.

Prior to joining Gould, Kuhn was an assistant professor in the Department of Electrical Engineering and Computer Science at Northwestern University. He received an MS in computer science from the University of Connecticut and a PhD in computer science from the University of Illinois in 1980. He is on the editorial board of *Computer* and is a member of the ACM and IEEE.

Questions about this special issue can be directed to Daniel D. Gajski, Dept. of Computer Science, University of Illinois, 1304 W. Springfield, Urbana, IL 61801, or Robert H. Kuhn, Gould Research Center, 40 Gould Center, Rolling Meadows, IL 60008.