LAMBDA

The Magazine of VLSI Design
Fourth Quarter 1980

These animals demonstrate the value of adaptability; ICs designed with procedural tools exhibit equally valuable flexibility.

LAMBDA

Volume I, No. 3
Fourth Quarter 1980

Publisher
Douglas G. Fairbairn

Editor-in-Chief
James A. Rowson

Managing Editor
Barbara S. Haymond

Technical Editors
Barbara Clifford
William D. Jansen

Contributing Editors
Lynn Conway
Steve Domenik

Art Director
Mike Shenon

Designer
Brian Balistreri

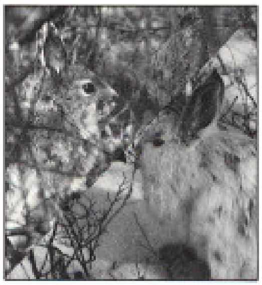
LAMBDA Fourth Quarter, 1980. Volume I, No. 3. Publication of Redwood Systems Group, P.O. Box 50503, Palo Alto, CA. 94303. Telephone: (415) 364-4360.

The subscription rates for LAMBDA are $16/year and $30/2 years (U.S.); $18/year and $34/2 years (Canada and Mexico); $25/year and $48/2 years (Europe and South America); $27/year and $52/2 years (Asia and Middle East). Foreign rates include airmail delivery. All payments must be made in U.S. dollars. Foreign subscriptions must be prepaid.

Application to Mail at Controlled Circulation Postage Rates is Pending at San Jose, CA. POSTMASTER: Send address changes to LAMBDA Magazine, P.O. Box 50503, Palo Alto, CA 94303

LAMBDA was founded to explore, expand, and define the interrelations between very large scale integrated circuits (VLSI) and computer architecture, design strategies, costs, and aids, as well as the electronics industry as a whole. LAMBDA is unique in that it is written by and for the participants in this dynamic field. It is our goal to be the communications focus of a new VLSI design community, encourage its development,and help define the community's directions.

Cover



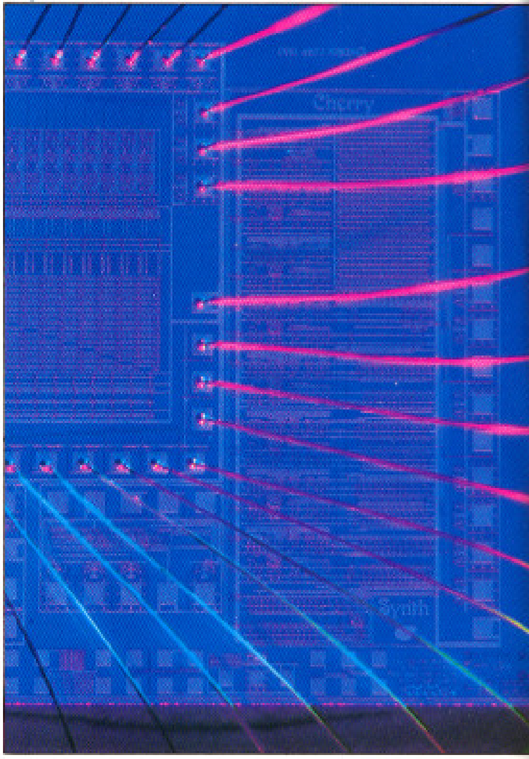
The snowshoe hares pictured on the cover graphically demonstrate the value of environmental adaptation. This ability to change color helps ensure the animals' survival.

The editorial in this issue advises that we learn from nature's example and develop ways to make the component cells of our VLSI designs more adaptable. Cells that can adapt to their "environment" will be useful over many designs, and will contribute to increased designer productivity.

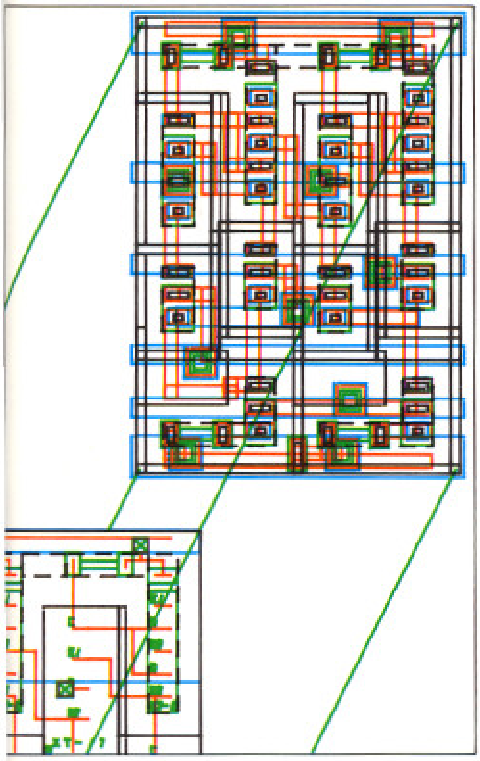
Photographers: S.J. Krasemann, Leonard Lee Rue III/BRUCE COLEMAN INC.

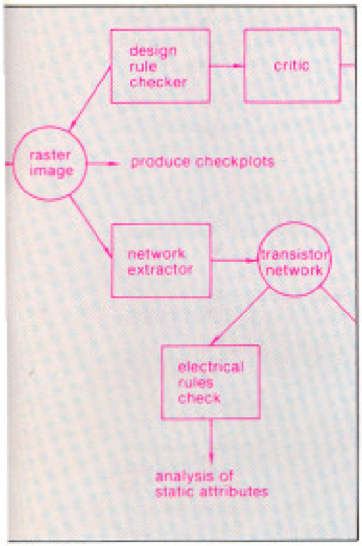
Departments

Page 71



Page 32





age 22

Articles

From the Editor

Procedural vs. Graphical Design of Integrated Circuits

The CAD tools most commonly used to aid the layout of integrated circuits today are vastly improved over the first such tools developed a decade or more ago. We have advanced from awkward gridded mylar and digitizing tables to high-speed, interactive color display systems aimed at on-line design.

It's interesting to note what has changed, and what hasn't, as these design systems have evolved. Certainly the *user interface* is very different and, most would agree, greatly improved. The cost has also changed, increasing by perhaps an order of magnitude. What hasn't changed much at all is the kind of data entered into these systems. The basic primitives are still rectangles, polygons, circles, etc. The geometric layout is stored in terms of absolute coordinates. Higher-level structures, such as repeated cells, have been present since the earliest systems (although nesting is a relatively new construct in some).

Thus we see that the user interface (and price) have changed, but other important qualities have not. The constancy of the data representation indicates that the "expressiveness" of our design tools has not improved. Expressiveness is a measure of the power of the language we use to describe our designs. This editorial suggests that the CAD community shift its primary focus from improving user interfaces to improving the design languages themselves. There has certainly been some effort and progress in this area over the years, but not nearly as much as is required.

Each design tool provides a language (possibly graphical) that can be compared with the languages of other tools. While not everything is understood about the relative expressive power of languages, enough is agreed upon to support some interesting conclusions. Consider the language provided by most graphics-based, mask geometry editing systems. The graphic operations usually involve adding and editing rectangles, wires and polygons, as well as defining cells (or symbols or macros), and making instances of cells, perhaps in an array. In most such interactive graphics systems, all coordinates in a design are constants.

Now, consider what kind of programming language such a graphics system would make. It does provide a procedure mechanism, in the guise of cell definitions and instances. However, only one parameter can be passed into a procedure: the geometric transformation placing the cell in the plane. FOR loops are usually tied tightly to the instantiation construct: a single data-base construct specifying an entire array of instances. The other primitives are rectangle, wire and polygon. Analogous constructs in a programming language are, built-in routines, such as printing a line of text. Notice that these built-in routines do not return a value. That's it. Note the lack of variables, assignment statements, recursion, conditionals, while loops, and expressions.

It's fairly obvious that such a language isn't worth much at all.

There are many advantages of using a more expressive language to design IC's. The placement of features can be done in a relative way, so that if one item or cell moves or grows, others follow. Cells can be parameterized so that they can adapt to their instance's environment (much as the hare on the cover can change its color in reaction to its environment). Transistors can change size to react to a change in load capacitance. Cells can be designed to route wires automatically between instances of unmatched cells. The list is as long as the number of designers involved. Adaptable cells designed in this parameterized fashion will "live" longer and be more useful as building blocks in later designs (which may have different requirements). They also allow decisions about detailed characteristics of the cell to be delayed until later in the design cycle.

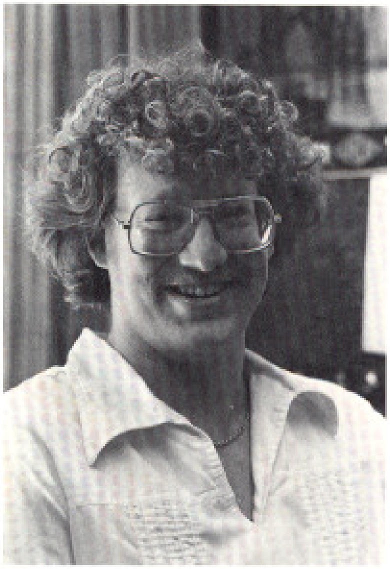
In order to gain more expressiveness, the first urge may be to jazz up a graphics system to "add the needed features." The trick is that nobody has yet figured out how to graphically specify high-level constructs like conditionals or WHILE loops. Also, by the time the "needed features" have been added, *you end up with a complete programming language.*

Why not just start with a programming language? Almost any language, like FORTRAN or LISP, can be used to design mask geometry simply by adding some procedures to output primitives of the Caltech Intermediate Form (CIF). A simple example of such a procedure is "Rectangle". This procedure, when called, outputs the CIF code for a rectangle. The coordinates are the parameters of the procedure. Most programming languages allow those parameters to be expressions, enhancing the adaptability of the procedure. A more complex procedure might be "PLA." The parameters of this procedure are the coordinates of the origin, number of minterms, inputs, and outputs, and the names of the files holding programs for the AND and OR planes. These procedures can then be called by the designer in the midst of a program that "computes" the chip. Such procedures create a kind of "imbedded language" that can be used as a design language.

An excellent example of procedural description applied to chip design can be found in Ron Rivest's article in this issue of LAMBDA. Ron has implemented the RSA encryption standard in an nMOS chip. The entire chip was designed using an "imbedded" language (within LISP). The very regular nature of his silicon algorithm makes it possible to compute the chip layout with a very small program, and to design it in a relatively short period of time.

But, I hear you cry, everything must be digitized by hand since the geometry is to be produced by a program! Sure, it looks powerful, but it's so hard to use! Well, yes and no. Some of the time, the program must produce detailed geometry that can be somewhat tedious to encode. This tedious geometry is generally in the small cells, as in a memory cell or ALU cell. The amount of time to *code* these cells is not a significant fraction of the overall design time, however. It has been said that actual digitization time is less than 10% of the chip design time. Much of the time, new cells can be designed using bits and pieces of other cells, perhaps as parameterized subroutines.

On the other hand, if you just *can't* give up the luxurious, feel of a slick interactive graphics editor (and I must admit, I like them too), an imbedded design language can usually incorporate designs done graphically. There are many ways to allow a procedural design description to incorporate a design done by an interactive graphics editor. One way is to simply write a program that translates the graphic file format into a piece of program that produces the same geometry. This will produce a program full of constants, which can be edited to parameterize it, or used as is.



CHRIS TSCHOEGL

It's time to concentrate our efforts on developing more powerful design methodologies and their complementary design languages. We have gone as far as we can with basic digitizing systems featuring sophisticated user interfaces. More expressive design tools are the key to easing the VLSI design problem.

James A Rowson

James A. Rowson Editor-in-Chief

From the Publisher

Welcome, new readers! Our circulation has quadrupled since our last issue, so this is the first issue that many of you will have received. LAMBDA is published to serve you, with the latest chip architectures, design and testing aids and strategies, and a variety of other topics of interest to the VLSI design community. We cover general news in the field and spotlight the newsmakers, and we always encourage your comments, criticism, and suggestions for articles. If you would like to write a paper for us, call us—we'll be happy to discuss it with you.

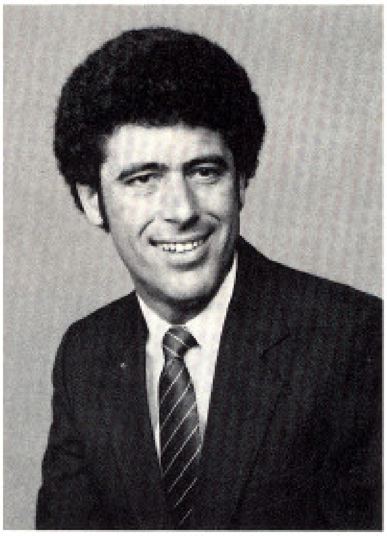
Long-time readers will observe that several new departments bow in this issue. *Technology Insight* focuses on what the new semiconductor technologies mean to the *designer,* instead of reporting them from a purely technical viewpoint. The design implications of much worthwhile research are unclear; we're here to illuminate these issues.

We also solicit readers' suggestions and contributions to the *Designer's Corner.* Steve Domenik, former Intel designer and co-founder of Integrated Solutions, Inc., brings his judgment and experience to the editor's desk. In this issue, he and Ed Cheng of Intel examine the subtleties of laying out those useful but ever-tricky substrate bias generators. Some of the data on second-order effects has never been published before.

People features interviews with acknowledged and future industry leaders. In this issue, Gene Potter, Chairman of the Board of Silicon Systems, Inc., speaks out on foundries, gate arrays, design automation, and custom IC's.

At $10 per line ($50 minimum), our *Classified Ads* are a bargain. You write the copy, we set the type. Do you have a service to sell, or need financing or a partner for your new high-tech start-up? Give LAMBDA Classifieds a try.

Our expanded editorial coverage will address *gate arrays* regularly. This decade-old semi-custom technology has just stepped into the limelight. Over a dozen large and small companies are jockeying for shares of a $50 million market, which has little second-sourcing, inadequate CAD, and no clear "leader" to set styles and standards. Gate-array design problems are much like those of full-custom work. Both design approaches call for defining unique layers on silicon chips, differing only in the number of customized layers. Many of our readers must make technical trade-offs between these methods, and (as we've said before!) LAMBDA wants to be your one-stop source for IC-design information.



The most striking changes may be *color* inside the magazine, and our many new *advertisers.* We are delighted to offer color, which is so vital in conveying IC design information. Our advertisers are important not only as financial supporters, but also as very valuable educational resources. I hope you'll find our advertising and editorial material equally interesting and informative.

There have been behind-the-scenes changes, too. Barbara Clifford and Bill Jansen have joined us as technical editors. Barbara has extensive technical writing and editing experience, and is largely responsible for the readability of this issue. Bill Jansen's wide-ranging background in electronic systems and IC design makes him ideal for a variety of editorial and administrative responsibilities. As noted above, Steve Domenik is our new Contributing Editor of the *Designer's Corner.* I'm also happy that Jim Rowson, Barb Haymond (formerly Barb Baird), and Lynn Conway are continuing with us. We couldn't have done it without any of them, but we especially appreciate Barb Haymond's day-in, day-out dedication.

If this is not your personal copy of LAMBDA, don't wait another minute—fill out the enclosed subscription card so you won't miss our next, special International Solid State Circuits Conference (ISSCC) issue.

Douglas G. Fairbairn Publisher