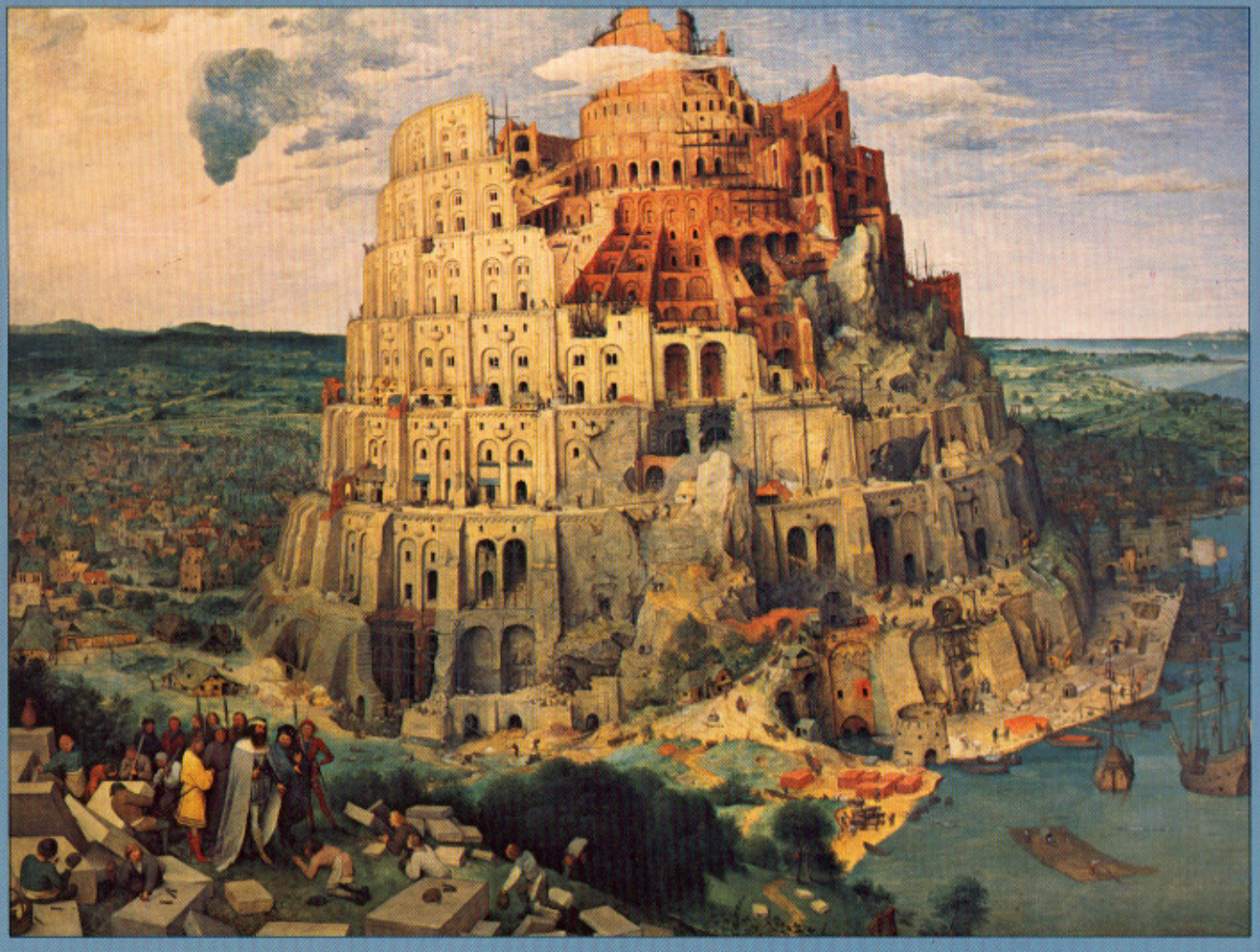
# LAMBDA

The Magazine of VLSI Design
Second Quarter 1980



VLSI and CAD systems designers must guard against the problem that halted the Babel project: poor communications.

**Publisher**
Redwood Systems Group

**Editor-in-Chief**
James A. Rowson

**Managing Editor**
Barbara S. Baird

**Contributing Editor**
Lynn Conway

**Production Manager**
Ann Harris

**Production Coordinator**
Carol Nash

**Art Director**
Kathleen Thorne-Thomsen

**Staff**
Donna Glaviano
Bernice Lifton

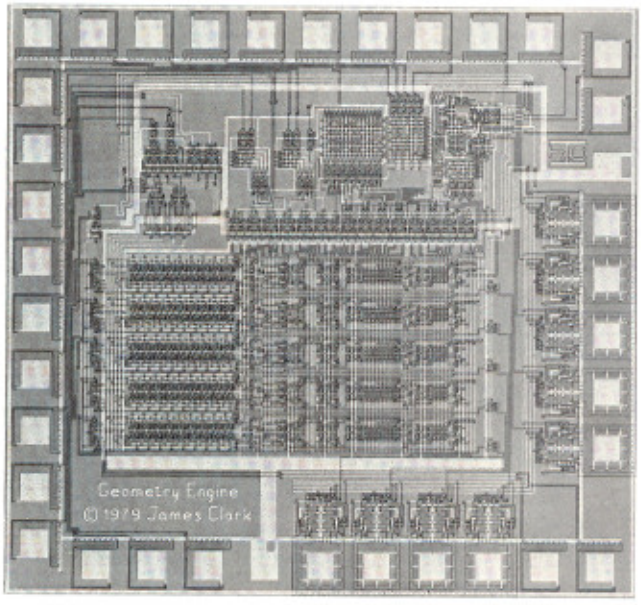
**Photographers**
David Bickford
Bart Locanthi
Blair Proctor

Composed through the facilities of the Electronic Publishing Laboratories of XEOS, Pasadena, CA.

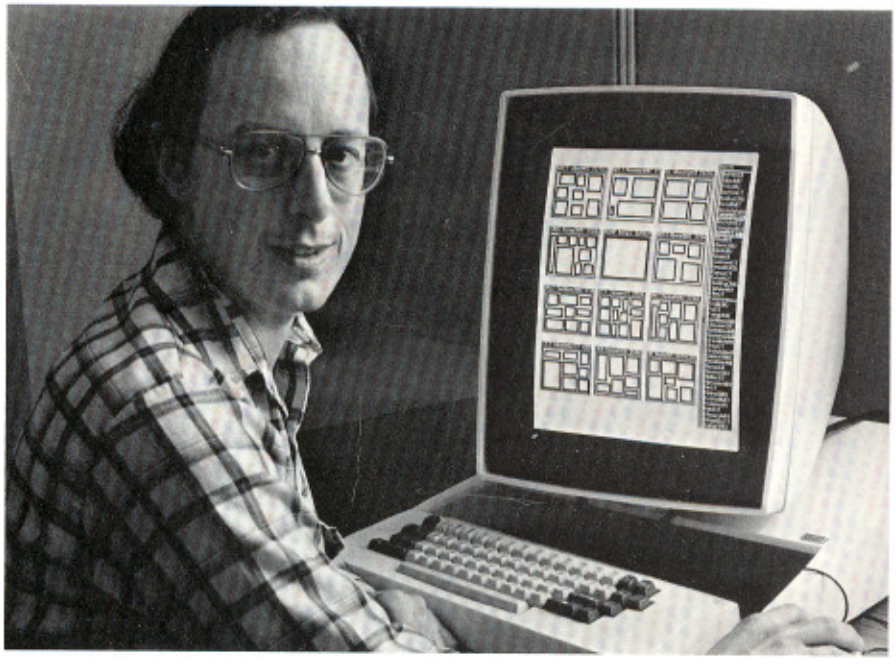
LAMBDA Second Quarter 1980. Volume I, No. 2. Publication of Redwood Systems Group, P.O. Box 50503, Palo Alto, CA 94303. Telephone: (415)364-4360.  The subscription rate for LAMBDA is $16/year and $30/2 years (US); $18/year and $34/2 years (Canada and Mexico); $25/year and $48/2 years (Europe and South America); $27/year and $52/2 years (Japan and Middle East). Foreign rates include airmail delivery. All payments must be made in US dollars. Foreign subscriptions must be prepaid.

LAMBDA was founded to explore, expand, and define the interrelationships 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.

**page 25**




**page 10**



### About the cover

*The Tower of Babel* (Pieter Bruegel, 1563, Kunsthistorisches Museum, Vienna) may seem an unusual choice for the cover of a VLSI design magazine. However, the lesson it teaches is an important one for all research and engineering groups. Genesis 11:1-9 relates that all went well on this massive engineering project until the Lord decided that Man had gone far enough. The undertaking came to a halt when the Lord caused the builders to speak different languages and scattered them about the earth. Lack of communication killed the Babel project and the same problem besets the LSI/VLSI design community today. As the need for interdisciplinary research and development grows, so does the need for careful communication. We must all make it our goal when we write or speak to define our terms carefully, and describe what relevance our work has to that of others.

The editorial in this issue takes the lead by treating two of the most-used but least-understood terms in the LSI/CAD area: "structured" and "hierarchical" design. I'm sure this editorial will spark some reaction, but that's what LAMBDA is here for. Let's here from you!

LAMBDA

Volume I, No. 2 Second Quarter 1980

## Articles



## Departments

# From the Editor

# A Modern Day Tower of Babel

The recent International Solid State Circuits Conference in San Francisco featured an evening panel session on VLSI design tools and methodologies. The participants had much of value to say, but unfortunately, the discussion resembled a tower of Babel. Because their words and contexts were so different, little communication took place. Each speaker had a different view and nomenclature for common problems, ideas and solutions. If we are going to speed progress in this critical area of design methodologies and tools, we must learn to communicate better.

A common base for understanding is the first requirement. New methodologies, implementations and the metrics for comparing both need to be developed and interchanged. LAMBDA can be the medium for this communication. A potentially powerful use of this magazine is to help to define new words and phrases and to disseminate the ideas they represent.

The communication vacuum is vividly illustrated by the misuse of the currently popular buzzwords *hierarchical design* and *structured design*. In order to stimulate discussion, I want to propose definitions for these commonly used terms. These terms are often exchanged between people who have different or poorly defined definitions. What do these terms really mean? How are they related? How do they relate to structured programming?

David Bickford

**Hierarchical design** implies recursively splitting a system into less complicated, independent parts that communicate through well-defined interfaces. By posing the solution to a complex problem in terms of a few powerful submodules, the designer can work at a level of abstraction appropriate to the problem. With the right abstractions, the complexity of any particular module is independent of the complexity of the whole. Module independence is the most important tenet of hierarchical design. Systems designed with unrestrained interaction between modules, even though the modules are small and well documented, do not, in general, work. The reason? The human mind can only deal with about 7 short-term ideas at once. When modules interact in ill-defined ways, the number of elements the designer must consider is much greater than the total number of modules. Even if this added complexity can be handled during the design process, when a change is made to one of those modules a year later, the interdependency will be waiting. Patiently.

Currently available integrated circuit design aids exploit very little of the power of hierarchical design. Often, some method of hierarchical breakdown is included in a design aid purely as a method for replicating common submodules. Replication is a useful property, in fact essential for designing large memories, but it certainly doesn't exhaust the power of hierarchical design. Although to many, hierarchies mean replication, they actually have much greater potential.

**Structured programming** is the result of applying hierarchical design to software systems. Structured programming advocates the recursive breakdown of complex problems and also restricts "pathological connections" between pieces of code. A pathological connection is one that cuts across the hierarchical structure in a dangerous way. For instance, the"go to" programming construct is considered

pathological because a "go to" can force entry into a module at a place other than the standard interface.

**Structured design** not only includes the fundamentals of hierarchical design but also adds a set of principles appropriate to the design of VLSI. The progressive splitting of complex systems into simpler systems, the careful specification of interfaces between modules, and the relative independence of modules are principles carried over from hierarchical design. Structured design expands on these by dealing with the constraints of the 2.5-dimensional world of integrated circuits and by recognizing the significant costs associated with communications on the chip.

Because VLSI systems must map onto flat silicon, the system design must reflect the difficulties of managing a two dimensional area. The "random logic" approach of carefully packing each logic function into its smallest possible area is useless as chip complexities rise. Therefore, the structured design methodology advocates the replacement of random logic by a more regular approach. By regularizing a design geometrically, using standardized interfaces, and matching cell pitches, the floor plan of a design becomes its actual implementation. The process of design becomes one of tiling: placing instances of similar cells in regular patterns on the plane.

In recognition of the relative expense of wires, structured design also advocates designing algorithms in terms of their communication requirements. Implementations are designed first in terms of the number and positions of the control and data wires involved. Often, a design optimizing communication will result in a simple network of wires with occasional transistors at their intersections.

Part of the purpose of this editorial is to get the VLSI community aroused enough to respond, presenting other views of the design problem. I want to encourage you to make LAMBDA the vehicle for sharing ideas which help develop new design methodologies. We welcome letters to the editor, one-page papers for a FORUM department, or full articles describing methodologies that have been implemented and tested.

Any ideas?

James A. Rowson Editor-in-Chief