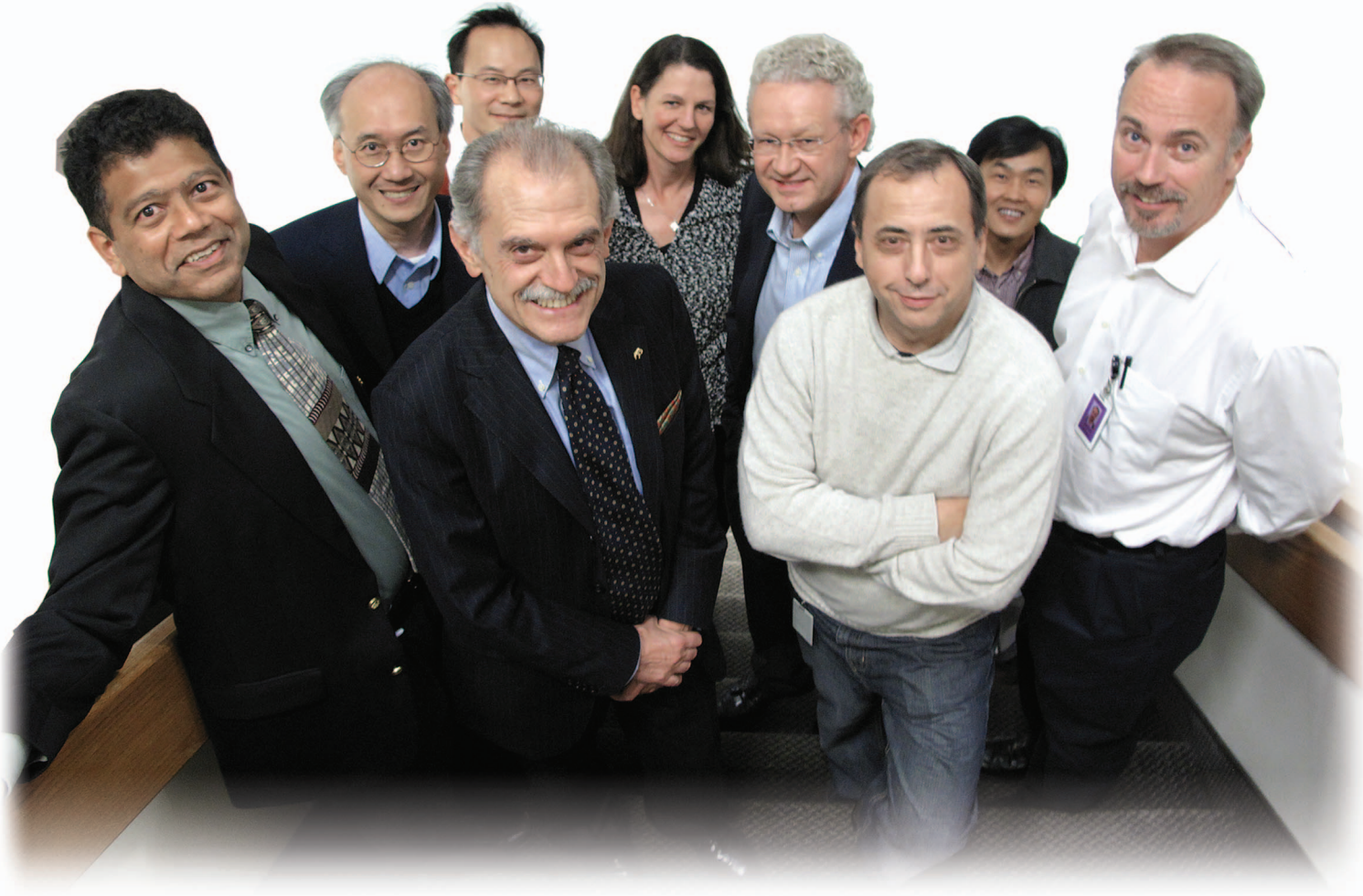
*Giovanni De Micheli*

# Chip Challenge

## *Alberto Sangiovanni-Vincentelli and the birth of logic synthesis.*



**Alberto Sangiovanni-Vincentelli celebrates receiving the Maxwell Award with colleagues and former students from Synopsys at Synopsys headquarters. Celebrating with Alberto in this October 2009 photograph are (from left) Narendra Shenoy, Tony Ma, Henry Sheng, Deirdre Hanford Ryan (Synopsys senior VP, Global Technical Services), Aart deGeus (Synopsys CEO and president), Antun Domic (Synopsys senior VP and general manager, Implementation Group), Wilsin Gosti, and Michael Jackson. Narendra, Tony, Henry, Deirdre, Wilsin, and Michael are Alberto's former students.**


1943-0582/10/$26.00©2010IEEE

Back in 1981, Alberto Sangiovanni-Vincentelli took me and two other graduate students to Harris Semiconductor, in Melbourne, Florida, as part of his sabbatical leave from the University of California, Berkeley. Harris was at that time a major provider of analog and digital semiconductors, and the management was farsighted in looking at avenues to design more, better, and cheaper integrated circuits (ICs). It is important to plunge back for a moment into the eighties to remember what design was: large-scale schematics drawn by hand and then entered into computing systems and medium-scale hand-drawn layouts painfully captured by "Calma operators" in specialized "design rooms" (see Figure 1). Most computer-aided design (CAD) at the time was related to simulating and optimizing transistor-level schematics. This important activity was already mature at that time: robust mathematical methods and efficient algorithms were implemented in computer programs such as SPICE and its derivatives.

The growth of the semiconductor sector in the 1980s was related to the capability of creating designs from high-level specifications, yet the bottleneck was the human capacity to create large schematics. The potential of semicustom design styles, like gate arrays and standard cell design, was hampered by the lack of methodologies to automate the design process itself in the front end. Structured layouts, as realized by programmable logic arrays (PLAs) [1] and other regular macrocells, were suffering from the lack of good optimization and instantiation methods. Manual design, as it was painfully effected back then, was a tedious, error-prone, and time-consuming process. Moreover, some conservative designers—reminiscent of those workers who destroyed the first automatic looms at the birth of the industrial revolution 200 years earlier—were very skeptical about any design automation methodology. I well remember, as a highlight of my stay at Harris, a meeting with a top-notch designer who showed us a hand-crafted folded PLA. When I naively asked him about a detail while pointing at a transistor on the schematic, he was startled: by accident I had discovered a missing contact that would have created a catastrophic fault in the circuit. Alberto shook his head and said, "The time has come to move on."

Much has changed since then. In less than 30 years, logic synthesis has enabled the semiconductor industry to design chips with many millions of transistors. Whereas conceptual, high-level design is performed by skilled designers, the mapping of the high-level specifications to the mask geometries is fully supported by logic and physical synthesis. The importance of the synthesis methodology cannot be understated: popular products like cell phones and portable computers are available today at a modest cost thanks to the "industrial revolution" enabled by synthesis technology.

***The pioneering work on logic synthesis of Alberto Sangiovanni-Vincentelli and his coworkers had a profound impact on the community.***

The vision Alberto brought to the community can be articulated in three points:

1) Designs must be captured by high-level primitives based on sound and unequivocal mathematical models.
2) The synthesis process must be carried forward by software tools implementing algorithms with guaranteed properties.
3) The synthesis output must feed seamlessly into physical design tools that can generate mask layouts.

This vision was embodied in the founding of the two major design automation companies, Cadence and Synopsys, and in the development of

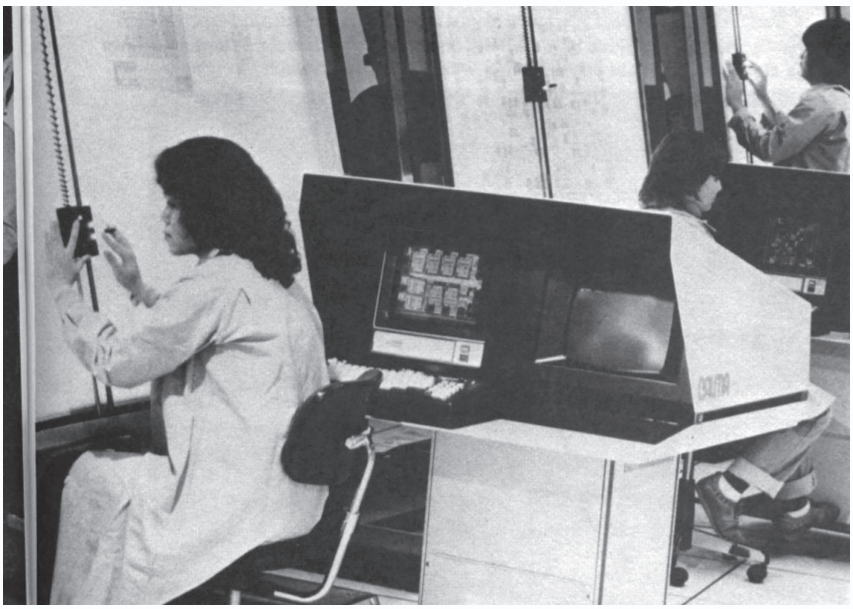


**FIGURE 1:** Digital design in the early 1980s using Calma stations. Figure courtesy of W. Harper.

*Digital Object Identifier 10.1109/MSSC.2010.938348*

***The growth of the semiconductor sector in the 1980s was related to the capability of creating designs from high-level specifications, yet the bottleneck was the human capacity to create large schematics.***

the synthesis systems at certain major corporations, such as IBM and Intel. Nevertheless, the successful development of this technology still required years of research. I was fortunate to witness this rich period of discoveries and the fruitful interaction of Alberto and Richard Newton at Berkeley with Robert Brayton and Gary Hachtel at IBM's T.J. Watson lab.

The early years of synthesis were characterized by two technological paths. The former was related to a design flow based on two-level logic macros, such as PLAs, and macrocell design. This approach lost its appeal with the replacement of NMOS by CMOS technology; a possible avenue for its resurrection may be driven by the recent development of regular fabrics for nanotechnologies (e.g., molecular electronics). The latter path was geared toward multilevel logic synthesis and addressed best gate array and standard-cell design. Most digital chips manufactured in the last 25 years have been designed with the methods developed at the University of California, Berkeley, and at IBM in the early 1980s.

The eighties were characterized by the arrival in the United States of Italian espresso in the coffee shops, offering students, researchers, and designers more dignified working conditions after-hours. At the same time, the ESPRESSO program became the standard logic synthesis tool for two-level logic optimization and to support multilevel synthesis as well. ESPRESSO was remarkably successful because it addressed an engineering issue: providing a reasonably good solution within a limited computing time. Moreover, this solution was always guaranteed to be prime and irredundant (the crucial properties with respect to testability) and was often globally optimal as well. ESPRESSO used smart heuristics that defeated the monstrous computational complexity of the problem. I remember the vivid discussions on Alberto's terrace with Bob Brayton, Gary Hachtel, and Curtis McMullen that led to the first landmark book on logic synthesis [2]. I would also like to credit Alberto and Rick Rudell for creating an incredibly efficient implementation of ESPRESSO and for putting it in the public domain, thus enabling the worldwide growth of the logic synthesis sector of design automation and supporting thousands of digital designs.

Addressing multilevel logic synthesis was much harder. This problem is multifaceted, and even defining the objective functions (e.g., for timing and power dissipation) is difficult. Nevertheless, the experience acquired with the heuristic iterative optimization in ESPRESSO helped in the task of creating an environment for the synthesis and optimization of multilevel circuits (see Figure 2). The methods developed by Brayton and McMullen at IBM within the Yorktown Logic Editor (YLE) blended well with Alberto's algorithmic contributions and led to

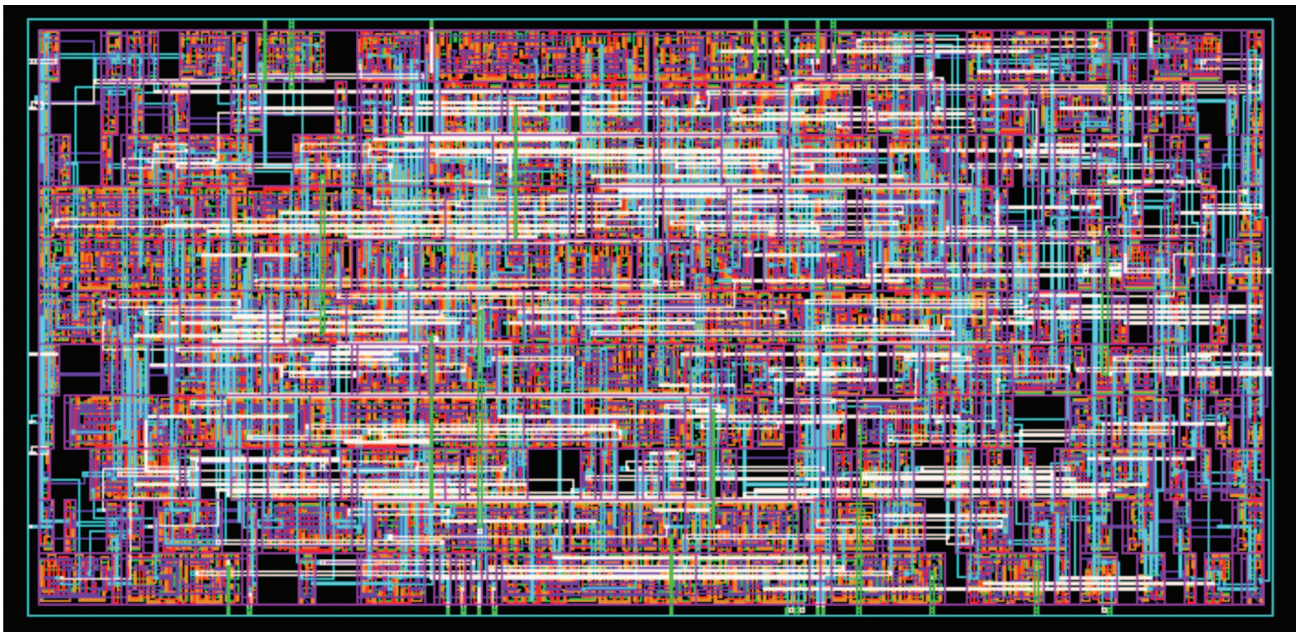


**FIGURE 2:** The power of logic synthesis: a complete multiplier layout generated from a few lines in a hardware description language.

the Multilevel Interactive Synthesis (MIS) program, the mother of many commercial synthesis toolsets. MIS was extremely influential in evangelizing logic synthesis by making it accessible to designers and researchers as an open framework.

## The Technology

Logic synthesis and optimization have been important subjects of research since the inception of computing, i.e., since the 1950s. In the beginning, logic synthesis was plagued by the lack of good physical models: indeed, it was assumed that the number of logic stages was the sole factor influencing delay. As a result, two-level logic design, in sum-of-products or product-of-sum forms, was praised as the best implementation. This model lasted until the 1980s when IC fabrication technology enabled submicron transistors. Since then, fan-out capacitance and wire delays became the bottlenecks for achieving high performance.

Two-level logic optimization was pioneered by McCluskey [3], who formulated the problem in terms of selection of prime implicants while exploiting a previous result by Quine showing that primes are sufficient to determine an optimum solution. McCluskey's covering problem was solved by branching procedures, which he programmed on the computers available at that time. The effectiveness of the Quine-McCluskey algorithm was marred by two factors: 1) computers were too weak at that time to provide solutions to problems of significant size and 2) the theory of computational complexity [4] had yet to be developed. A posteriori we know that there is almost no hope of solving the prime implicant covering problem in polynomial time, as this problem is NP-complete.

In the early 1980s, researchers attempted to solve logic optimization with algorithms that provide solutions in minutes of computation time for problems of engineering relevance. To set a reference point, let us say that a solver should handle at least some thousands of primes—today, the million mark has been well surpassed. The cornerstone of heuristic logic optimization is to solve the problem by iterative improvement, i.e., by successively transforming a logic cover into a sequence of smaller ones that are still valid implementations. A smart termination rule would put the solution close to the optimum, if not at the optimum itself.

**Logic synthesis and optimization have been important subjects of research since the inception of computing, i.e., since the 1950s.**

The effective manipulation of logic covers required procedures for solving some fundamental problems, such as, for example, tautology check, containment, and complementation. Alberto and his coworkers invented the unate recursive paradigm to design effective algorithms for the aforementioned problems [2]. This paradigm enables a solver to split a problem into two problems of smaller dimensions; the recursive application of the paradigm eventually maps the initial problem into simple terminal-level problems that can be solved in a straightforward way. The recursive paradigm may still generate an exponential number of terminal-level problems, thus hampering the effectiveness of the solution. The unate recursive paradigm trims the solution space by using the properties of unate functions and covers: in each recursion step, it aims to generate unate subproblems that are faster and easier to solve. Though these ideas are easy to portray in broad terms, their correctness can be proved only by going through a complex formulation in discrete mathematics, reported in [2].

In summary, the success of ESPRESSO as a tool is based on its strong mathematical basis and on algorithms with guaranteed properties: the tool manipulates sequences of covers that are guaranteed to be prime and irredundant (i.e., locally optimal) by two related algorithms, called *expand* and *irrcover*. While others attempted to develop logic minimizers, no program is superior to ESPRESSO in terms of the quality of its results and computing time.

Multilevel synthesis and optimization is a much harder problem to deal with. Overall, it can be stated as the path of mapping a logic-level specification into a set of interconnected cells from a given library, such as those used for standard cells and mask- or field-programmable gate arrays. This problem is even hard to formulate mathematically, as there are many flavors in which a network can be cast. Again, this problem was solved efficiently by heuristics that yield good solutions with limited computing time, even though in this case it is difficult to quantify the distance from the optimum solution because it is often unknown and hard to bound.

The MIS program achieves logic optimization through iterative improvement of an initial network, obtained by applying operators to the network itself. Each operator is a realization of an algorithm with guaranteed properties. For example, some operators merge and split (i.e., decompose) logic functions while others extract the common subexpressions from logic functions. These operations are supported by a polynomial algebraic model, where logic functions are dealt with as polynomials for speed of processing. Moreover, another set of transformations reduces logic functions through Boolean transformations that exploit the notion of global and local "don't care" conditions. Again, the engine of these

***In less than 30 years, logic synthesis has enabled the semiconductor industry to design chips with many millions of transistors.***

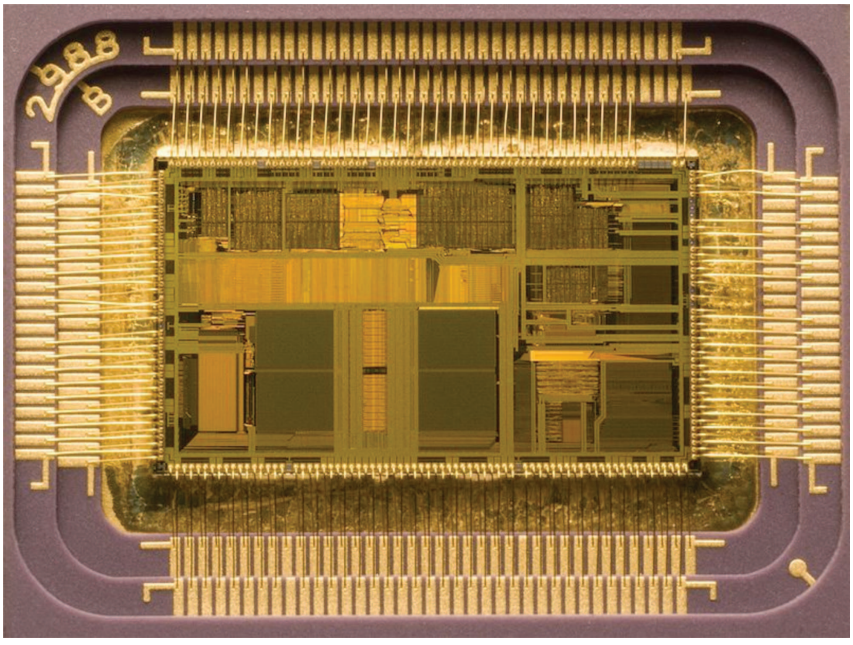


**FIGURE 3:** Intel 80486. The top part was implemented by standard cells and designed using logic synthesis. Figure courtesy of Intel Corp.

transformations is the ESPRESSO program, modified to act as a subroutine within MIS. The repeated application of logic minimization of local functions, with appropriate don't care conditions, leads to networks that are prime and irredundant, i.e., such that no literal and no product term in their representation can be dropped without changing the overall nature of the network. This local optimality condition correlates to some important quality factors for testing the network, such as being fully testable for stuck-at faults. Achieving these results—and formally validating their properties—is a complex task that can only be explained through a thorough mathematical analysis of logic models, algorithms, and their properties [5].

## The Impact

The pioneering work on logic synthesis of Alberto Sangiovanni-Vincentelli and his coworkers had a profound impact on the community. From an academic and scientific standpoint, the work presented in [2] and [6] is the first scientific source of information on contemporary logic synthesis. From a commercial standpoint, MIS paved the way for the release of Synopsys's Design Compiler and other commercial tools. Indeed, while the implementation may differ, the algorithmic core goes back to MIS. From an industrial product standpoint, logic synthesis profoundly changed the way in which chips were designed. Early adopters, like IBM and Intel, reaped a strong benefit from using logic synthesis and tying it to the physical design tool chain. Logic synthesis was instrumental in the design of a major portion of Intel's 80486 processor (see Figure 3). Thereafter, logic synthesis was used in virtually all digital integrated circuits that have been brought to market.

## References


[1] H. Fleishe and L. Maissel, "An introduction to array logic," *IBM J. Res. Develop.*, vol. 19, pp. 98–109, Mar. 1975.

[2] R. Brayton, G. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*. Norwell, MA: Kluwer, 1984.

[3] E. McCluskey, "Minimization of Boolean functions," *Bell Syst. Tech. J.*, vol. 35, pp. 1417–1444, Nov. 1956.

[4] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. Freeman, 1979.

[5] G. De Micheli, *Synthesis and Optimization of Digital Circuits*. New York: McGraw-Hill, 1994.

[6] R. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, and A. Wang, "MIS: A multiple-level logic optimization system," *IEEE Trans. Computer-Aided Design*, vol. CAD-6, no. 6, pp. 1062–1081, Nov. 1987.


## About the Author


**Giovanni De Micheli** is a professor and director of the Institute of Electrical Engineering at École Polytechnique Fédérale de Lausanne, Switzerland. Previously, he was a professor of electrical engineering at Stanford University, in Palo Alto, California. His research interests include several aspects of design technologies for integrated circuits and systems, such as synthesis, networks on chip, and low-power design. He is the author of *Synthesis and Optimization of Digital Circuits* (McGraw-Hill, 1994) and coauthor and/or coeditor of eight other books and more than 400 technical articles. He is a Fellow of ACM and IEEE. He has served IEEE as Division 1 director (2008–2009), cofounder and president-elect of the IEEE Council on EDA (2005–2007), president of the IEEE CAS Society (2003), editor in chief of *IEEE Transactions on Computer-Aided Design* (1987–2001), and general chair of DATE (2010) and DAC (2000).


SSC