VLSI CAD Tools and Applications

Wolfgang Fichtner
Martin Morf

Kluwer Academic Publishers

VLSI CAD TOOLS AND APPLICATIONS

edited by

Wolfgang Fichtner
Martin Morf

Institute for Integrated Systems
Swiss Federal Institute of Technology
Zurich, Switzerland

KLUWER ACADEMIC PUBLISHERS
Boston/Dordrecht/Lancaster

Distributors for North America:
Kluwer Academic Publishers
101 Philip Drive
Assinippi Park
Norwell, MA 02061, USA

Distributors for the UK and Ireland:
Kluwer Academic Publishers
MTP Press Limited
Falcon House, Queen Square
Lancaster LA1 1RN, UNITED KINGDOM

Distributors for all other countries:
Kluwer Academic Publishers Group
Distribution Centre
Post Office Box 322
3300 AH Dordrecht, THE NETHERLANDS

Library of Congress Cataloging-in-Publication Data

VLSI CAD tools and applications.

(The Kluwer international series in engineering and computer science ; SECS 24)
Papers presented at a summer school held from July 21 to Aug. 1, 1986 at Beatenberg, Switzerland.
Includes bibliographies.
1. Integrated circuits—Very large scale integration —Design and construction—Data processing.
2. Computer-aided design. I. Fichtner, Wolfgang.
II. Morf, Martin. III. Series
TK7874.V5572 1987 621.395 86-21148
ISBN 0-89838-193-2

Printed in the United States of America

VLSI CAD TOOLS AND APPLICATIONS

THE KLUWER INTERNATIONAL SERIES IN ENGINEERING AND COMPUTER SCIENCE

VLSI, COMPUTER ARCHITECTURE AND DIGITAL SIGNAL PROCESSING

Consulting Editor

Jonathan Allen

Other books in the series:

Logic Minimization Algorithms for VLSI Synthesis, R.K. Brayton, G.D. Hachtel, C.T. McMullen, and A.L. Sangiovanni-Vincentelli. ISBN 0-89838-164-9.

Adaptive Filters: Structures, Algorithms, and Applications, M.L. Honig and D.G. Messerschmitt. ISBN: 0-89838-163-0.

Computer-Aided Design and VLSI Device Development, K.M. Cham, S.-Y. Oh, D. Chin and J.L. Moll. ISBN 0-89838-204-1.

Introduction to VLSI Silicon Devices: Physics, Technology and Characterization, B. El-Kareh and R.J. Bombard. ISBN 0-89838-210-6.

Latchup in CMOS Technology: The Problem and Its Cure, R.R. Troutman. ISBN 0-89838-215-7.

Digital CMOS Circuit Design, M. Annaratone. ISBN 0-89838-224-6.

The Bounding Approach to VLSI Circuit Simulation, C.A. Zukowski. ISBN 0-89838-176-2.

Multi-Level Simulation for VLSI Design, D.D. Hill, D.R. Coelho. ISBN 0-89838-184-3.

Relaxation Techniques for the Simulation of VLSI Circuits, J. White and A. Sangiovanni-Vincentelli. ISBN 0-89838-186-X.

PREFACE

The summer school on *VLSI CAD Tools and Applications* was held from July 21 through August 1, 1986 at Beatenberg in the beautiful Bernese Oberland in Switzerland. The meeting was given under the auspices of IFIP WG 10.6 VLSI, and it was sponsored by the Swiss Federal Institute of Technology Zurich, Switzerland. Eighty-one professionals were invited to participate in the summer school, including 18 lecturers. The 81 participants came from the following countries: Australia (1), Denmark (1), Federal Republic of Germany (12), France (3), Italy (4), Norway (1), South Korea (1), Sweden (5), United Kingdom (1), United States of America (13), and Switzerland (39).

Our goal in the planning for the summer school was to introduce the audience into the realities of CAD tools and their applications to VLSI design. This book contains articles by all 18 invited speakers that lectured at the summer school. The reader should realize that it was not intended to publish a textbook. However, the chapters in this book are more or less self-contained treatments of the particular subjects. Chapters 1 and 2 give a broad introduction to VLSI Design. Simulation tools and their algorithmic foundations are treated in Chapters 3 to 5 and 17. Chapters 6 to 9 provide an excellent treatment of modern layout tools. The use of CAD tools and trends in the design of 32-bit microprocessors are the topics of Chapters 10 through 16. Important aspects in VLSI testing and testing strategies are given in Chapters 18 and 19.

We would like to thank all of the invited speakers for the time and effort they had to invest into the preparation of their talks and papers. It would have been impossible to organize the summer school and to edit this book without the help of the members of the Institute of Integrated Systems at the Swiss Federal Institute of Technology Zurich. In particular, we would like to thank L. Heusler, Dr. H. Kaeslin, P. Lamb, R. Meyer, and M. Raths for their assistance. Dr. A. Aemmer was especially helpful in the planning and organization. He deserves special credit for the success of the meeting and the existence of this book. Mrs. Bourquin was a perfect summer school secretary. It was a special pleasure to work with Carl Harris from Kluwer Publishing in the preparation of this book.

W. Fichtner M. Morf

TABLE OF CONTENTS

1

VLSI DESIGN STRATEGIES

Carlo H. Séquin

Computer Science Division
Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720

ABSTRACT

The growing complexity of VLSI chips creates a need for better CAD tools and data management techniques. The rapidly changing nature of the field requires a modular toolbox approach — rather than a fixed monolithic design system — and the involvement of the designer in the tool-building process. A short overview over the Berkeley design environment and our recent Synthesis Project is also given.

1. INTRODUCTION

Very large scale integration (VLSI) has made it economically viable to place several hundred thousand devices on a single chip, and the technological evolution will continue to increase this number by more than an order of magnitude within a decade. While the limits on chip growth imposed by technology and materials are still another three orders of magnitude away,[1] the design of the present-day chips already causes tremendous problems. G. Moore coined the term "complexity barrier".[2] This is the major hurdle faced today in the construction of ever larger integrated systems. In Section 2 the nature of the VLSI complexity problem will be discussed.

In order to deal with this complexity and to exploit fully the technological potential of VLSI, some structure has to be introduced into the design process; the resulting design styles are reviewed in Section 3, and the general nature of the VLSI design process is discussed. The size of the task is such that it cannot be done without tools; new tools and new ways of managing the information associated with the design of a VLSI chip must be developed (Section 4). This changes the role of the designer (Section 5).

Section 6 illustrates with the example of the Berkeley Synthesis Project how we think the art of VLSI design is going to evolve.

2. VLSI COMPLEXITY

In the early 1980's. VLSI complexity became a hot topic for concern and discussion.[3] This may appear surprising if one compares the complexity of the chips of that period with other technological structures that mankind has built in the

past. Certainly the number of components on a VLSI chip does not exceed the number of parts in a telephone switching station or in the space shuttle, and mainframe computers with an even larger number of transistors have been built for at least a decade before they were integrated onto a chip. System complexity should not differ markedly whether a circuit is contained within a cabinet, on a printed circuit board, or on a single silicon chip.

It is the "large", potentially unstructured space of the VLSI chip that causes the concern. Nobody would dare to insert a million discrete devices into a large chassis using discrete point-to-point wiring. Large systems built from discrete devices are broken down into sub-chassis, mother-boards, and module-boards carrying the actual components. This physical partitioning encourages careful consideration of the logical partitioning and of the interfaces between the modules at all levels of the hierarchy. Since such systems are typically designed by large teams, early top-down decisions concerning the partitioning and the interfaces must be made and enforced rather rigidly — for better or for worse. This keeps the total complexity in the scope of each individual designer limited in magnitude, and thus manageable.

VLSI permits the whole system to be concentrated in a basically unstructured domain of a single silicon chip which does not *a priori* force any partitioning or compartmentalization. On the positive side, this freedom may be exploited for significant performance advantages. On the negative side, it may result in a dangerous situation where the complexity within a large, unstructured domain simply overwhelms the designer.

A similar crisis was faced by software engineers when unstructured programs started to grow to lengths in excess of 10,000 lines of code. The crisis was alleviated by the development of suitable design methodologies, structuring techniques, and documentation styles. Many of the lessons learned in the software domain are also applicable to the design of VLSI systems.[3]

Furthermore, the field of VLSI is rather interdisciplinary in nature. To achieve optimal results, we need a tight interaction of algorithms, architecture, circuit design, IC technology, etc. However, designers who are experts in all these fields are rarely found. How can ordinary mortals attempt to do a reasonable VLSI design? Here again, suitable abstractions have to be found, so that the details of processing are hidden from the layout designer, and the details of the circuit implementation are hidden from the microarchitect. Models that are accurate enough to permit sound decisions based on them need to be created, and clean interfaces between the various domains of responsibility need to be defined. For instance, the semantic meaning of the geometry specified in a layout has to be defined carefully: Is this the geometry of the fabrication masks? Is this the desired pattern on the silicon chip? Or is this a symbolic representation of some of the desired device parameters? These questions still lead to much discussion and often to bad chips. The emergence of silicon brokerage services such as MOSIS[4] has forced clarification of many of these issues.

3. THE DESIGN SPECTRUM

To make the task of filling the void on a VLSI chip manageable, some widely accepted abstractions have emerged that lead to a hierarchy (or rather a multidimensional space) of *views* of a particular design. The different representations generally address different concerns. A typical list of design levels and of the concerns they address is shown in Table 1.

Design Level	Concerns Addressed
Behavior	Functionality
Functional blocks, Linked module abstraction[5]	Resource allocation sequencing, causality
Register-transfer level Clocked register and logic[5]	Testability timing, synchronization
Gate Level, Clocked primitive switches[5]	Implementation with proper digital behavior
Circuit Level	Performance, noise margins
Sticks Level	Layout topology
Mask Geometry	Implementation, yield

Table 1. *Levels of abstraction in chip design.*

The other saving notion is that of prefabricated parts. The same functions at various levels of the design space are needed again and again. Successful designs of frequently used parts can thus be saved in libraries for the reuse by many customers. The nature of these parts and the level to which they are predefined or even prefabricated leads to a variety of different design styles.

3.1. VLSI Design Styles

A large spectrum of possibilities for the design of a VLSI chip has evolved, offering wide ranges of expected turn-around time, resulting performance, and required design effort. Table 2 gives a strongly simplified view of the spectrum of possibilities.

On one end of the spectrum are the Gate Array and Standard Cell technologies. Predesigned logic cells at the SSI and MSI level permit the engineers to use functional blocks that they are already familiar with from TTL breadboard designs. The abstraction and prefabrication of these cells lead to minimal design effort and faster turn-around time but at the price of less functionality per chip and less performance for a given technology.

Method	Complexity	Effort	Main Strength	Automation
Gate Array	20,000	4-8 weeks	Fast changes	Yes
Standard Cell	40,000	4-8 weeks	Resuse of logic	Yes
Macro Cell	100,000	1-2 years	Good area use	Almost
Flexible Modules	200,000	1-2 years	High density	Almost
Standard Functions	200,000	2-8 years	Testability	Not yet
Optimized Layout	400,000	$\geq$8 years	High performance	Not so soon

Table 2. *Styles of IC chip design.*

At the other end of the spectrum is the full custom chip in which all modules have been hand-designed with the utmost care for performance and density and have been integrated and packed onto the chip in a tailor-made fashion. This design style can lead to spectacular results in terms of functionality and performance of an individual chip, but it comes at the price of an exorbitant design effort.

Somewhere in the middle between these two extremes are mixed approaches in which the crucial cells have been hand-designed with great care — particularly the cells that are in the critical path determining performance and the cells that are used in large arrays, as they will make the dominant contribution to the size of the chip. Uncritical "glue" logic that is used only once may be generated by a program either in the form of a PLA or as a string of standard cells. These macro cells of varying sizes and shapes are then placed and wired by hand or by emerging CAD programs.[6] This approach leads to higher densities than standard cells, since the degree of integration in the macro cells is typically higher and since a smaller amount of area is wasted in partly filled wiring channels. If the macro cells are procedurally generated and suitably parameterized so that they can be adjusted to the available space, even higher densities can be achieved. When properly used, these intermediate approaches can compete with a full custom design in terms of performance but typically result in a somewhat larger chip size.

Concerns of modularity and testability may outweigh aims for density and performance; functional modules designed for testability with clean interfaces are then used. This is in analogy to the use of properly abstracted and encapsulated software modules. This approach has started to gain acceptance also in VLSI. A good VLSI design environment will permit the designer to mix these various design styles in appropriate ways.

3.2. The Design Process

Even with an agreed-upon set of hierarchical levels, an extensive library of predefined parts, and a chosen design style, the design process can still be rather

involved. It is rarely a single forward pass through all the transformation steps that takes a high-level behavioral description through register-transfer and logic level descriptions into a symbolic representation capturing the topology and finally into a dense layout suitable for implementation with a particular technology (Table 1). The overall problem may be structured in a top down manner into simpler subtasks with clearly defined functions. But in parallel, designers intimately familiar with the implementation technology will explore good solutions for generic functions in the given technology in a bottom-up fashion. This effort will result in an understanding of what functions can best be implemented in this environment and produce a set of efficient building blocks.

Hopefully, the top-down decomposition and the bottom-up provision of solutions will meet in the middle and permit completion of the design. However, for a new technology, it is unlikely that this will happen on the first try. The natural building blocks must first be discovered; only then can the architectures be modified and partitioned appropriately. Thus there is an iteration of top-down and bottom-up moves in a Yo-Yo like fashion until the optimal path linking architecture to technology has been found.

It should also be pointed out that the design process is often a mixture of solid established procedures and of free associations and 'trial-and-error'. The guessing part plays a role in finding good partitioning schemes as well as in the definition of generic functions that might constitute worthwhile building block in the given technology. Proven checking methods are then used to evaluate objectively whether the guesses made are indeed usable: Is the decomposition functionally correct? Is it appropriate — or does it cut through some inner loop, causing unnecessary communications penalties? Are the building blocks of general use? How many algorithms, tasks, or architectures can actually make use of them? Is their performance reasonable?

In the next section we explore to what extent this design process can be supported by the computer, and for which part the human intelligence might be hard to replace.

4. THE ROLE OF CAD TOOLS

Good tools help man to achieve more, to obtain better results, or to reach given goals more effortlessly. VLSI design is no exception. I like to split the CAD tools useful in the design of ICs into five classes:

1) *Checking and Verification Tools* typically answer questions such as: Are there any errors? Are the connections between blocks consistent? Does this function behave as specified?

2) *Analysis Tools* tell the designer: How well does a particular approach work? How much power does this circuit consume? What is the worst case settling time?