TN402
R963

8960063

Computer aided tools for VLSI system design

Edited by
G. Russell

Peter Peregrinus Ltd. on behalf of the Institution of Electrical Engineers

Published by: Peter Peregrinus Ltd., London, United Kingdom

British Library Cataloguing in Publication Data

Computer aided tools for VLSI system design.
—(IEE computing series; 9)
1. Integrated circuits—Very large scale integration—Data processing
I. Russell, G. II. Series
621.381'73 TK7874

ISBN 0-86341-093-6

Printed in England by Billing and Sons Ltd.

IEE COMPUTING SERIES 9
Series Editors: Dr. B. Carré, Prof. S. L. Hurst, Dr. D. A. H. Jacobs, M. Sage, Prof. I. Sommerville

Computer aided tools for VLSI system design

List of Contributors

Chapters 1, 4 and 5
G. Russell, University of Newcastle upon Tyne, UK.

Chapter 2
H. G. Adshead, ICL, Manchester, UK.

Chapter 3
T. J. Kazmierski, University of Southampton, UK (on leave from Technical University of Warsaw, Poland).

Chapter 5
K. Baker, Hirst Research Centre, Middlesex, UK.

Chapter 6
D. J. Kinniment, University of Newcastle upon Tyne, UK.

Chapter 7
P. Ivey, BTRL, Ipswich, UK.

Chapter 8
R. A. Cottrell, UMIST, Manchester, UK.

Chapter 9
M. R. McLauchlan, University of Newcastle upon Tyne, UK.

Chapter 10
E. G. Chester, University of Newcastle upon Tyne, UK.

Chapter 11
S. G. Smith, University of Edinburgh, UK.

Chapter 12
J. D. Wilcock, Plessey Research (Caswell) Ltd, UK.

Chapter 13
D. Warburton, ICL, Manchester, UK.

Chapter 14
M. H. Gill, STC Technology, Harlow, UK.

Chapter 15
A. P. Ambler, Brunel University, Middlesex, UK.

Preface

Over the past decade the use of Computer Aided Design (CAD) Tools in the design of integrated circuits has become well established. However, in order to gain acceptance, in the design community at large, these tools had to be introduced gradually. After a short period of time it was realised that these tools were becoming inadequate as circuit complexities increased due to the advancements made in fabrication technology. Subsequently, CAD research activities were focused on attempting to develop more efficient algorithms to cope with the problem of complexity. This approach, however, could only provide a temporary solution, since the fundamental problem was not that the CAD tools were inefficient, necessarily, but the design style into which they were introduced was simply incapable of supporting the design of complex circuits.

Recently, design styles based on hierarchical decomposition or 'divide and conquer' techniques have evolved which are more amenable to CAD techniques, since they limit the amount of data and design constraints to be processed at each stage of the design cycle, resulting in reduced design times and costs through the more efficient use of man/machine resources.

Although, CAD tools have now been integrated into each stage of the design process, the IC design community, which is continually increasing, is not fully aware of the arsenal of CAD tools available for integrated circuit design. Consequently the objectives of the 1st IEE Vacation School on CAD Tools for VLSI System Design, held at the University of Newcastle upon Tyne in July 1985, were twofold. First, to review the range of CAD Tools available for design and, thereafter, in the light of projected increases in chip complexity, anticipated through improvements in device technology, to discuss the limitations of these tools. Second, to introduce the next generation of design tools and associated design methods which should restrict, to within acceptable limits, the major obstacles to the pervasive use of VLSI, namely design times and costs. The contents of this book comprise the lectures given during the course, which are self contained and cover a wide range of topics related to the use of CAD tools in the design of integrated circuits.

G. Russell

Contents

Chapter 1

Introduction

What is CAD? CAD, in the context of the design of integrated circuits can be defined as the use of computers for the collection, manipulation and storage of all the data associated with the design. More specifically, CAD tools are used to evaluate design alternatives and tradeoffs, to construct the basic components used in the design, to assemble and interconnect these components into the final layout of the design and finally to generate the data and test files used in mask making and testing the device after fabrication.

It is well established that CAD techniques are essential to the design of Very Large Scale Integrated (VLSI) Circuits in order to process, efficiently and cost effectively, the vast amount of data associated with the design of these complex devices. The reason given for using CAD in VLSI circuit design, to some extent, belittles the important role that CAD now plays in ensuring that IC products from a given company retain the competitive edge in the market place. The factors which make an integrated circuit competitive are performance, functionality, cost and turn around time. In general performance is a function of the processing technology, this has reached a very advanced state of development, to the extent that further gains in performance through improvements in technology are extremely expensive. Functionality reflects the ingenuity of the designer, again good designers are expensive; the functional capability of a circuit can be duplicated and superseded, hence the competitive edge gained in this way is marginal. The remaining factors are cost and turn around time, these are directly associated with CAD systems; hence the efficiency of the CAD systems used in IC design is now playing an extremely important role in maintaining a competitive edge in the market place for IC products. In order to maintain a competitive edge it is essential that the CAD system has a wide range of CAD tools, the ability to support various design styles and have an extensive library of functional cells. If the CAD system is not to fade into obsolescence it must also contain a flexible database management system, since this is the foundation on which all efficient CAD systems are built. The database system permits new tools to be interfaced, readily to the system, it controls the versions of cell libraries used in a design, it ensures the integrity of design changes throughout the various representations of a design and also allows changes in design style or technology to be accommodated, readily, in the system. A further factor which affects the efficiency of the CAD system is the types of tools that it contains. If the system has many analysis tools, it infers that the design time will be long and expensive, since the CAD system will impose few constraints on the designer, requiring long verification runs to ensure that the layouts are error free. Although the efficiency of these systems can be improved by developing analysis algorithms which exploit the structure of the layout in the verification procedures or have the analysis algorithms mapped onto special purpose computer architectures, the competitive edge gained in this way only lasts a short time. However, if the system contains synthesis tools, design times are drastically reduced since the designer is constrained; the essence of these tools is 'correctness by construction'. Furthermore, the competitive edge gained will last longer since the capabilities of these systems are not fully explored, leaving ample opportunity for improvement and also the capabilities of synthesis systems are more proprietary hence more difficult to duplicate.

Although the concept of using CAD tools in the design of digital systems has been around since the mid 1950's, it has only been used in IC design since the mid 1960's. The reason for this is twofold; first the concept of using computers in the design of digital systems was brought into disrepute, throughout the late 50's and early 60's due to the inadequacy of the computer systems (in terms of computational power, storage facilities and availability) to solve the problems which occurred in the design of digital systems. Second, the complexity of the task in designing integrated circuits at that time, was still amenable to manual methods, and there was a reluctance to change to CAD techniques which, in the past, had been shown to be inadequate. However, as circuit complexity increased manual methods were becoming inadequate and alternative design techniques were sought. By this time there had been a considerable improvement in computer technology, with respect to both hardware and software. Subsequently, some simple layout tools evolved which permitted the designer to digitise a layout into a computer and have a plot produced to determine what errors had been made, and thereafter have the drive tapes for the mask making equipment generated automatically. The introduction of interactive graphics greatly enhanced the designers capability to edit circuit layouts stored in the computer. At the same time designers were becoming interested in the use of circuit simulation as a means of verifying the operational characteristics of their circuits. Previously breadboarding techniques had been employed to simulate the behaviour of the circuit, however due to the differences in device size and parasitic capacitances, particularly as IC's became more complex, breadboarding could no longer be used as a means of checking, satisfactorily the performance of the circuit. As the complexity of integrated circuits increased, simple functions which were interconnected on a printed circuit board were now integrated into the same circuit. Consequently

IC designers had a requirement for placement and routing programs, which they subsequently borrowed from the printed circuit board designers and modified to suit their own requirements. Also, the complexity of the integrated circuits being designed was becoming too large to be efficiently simulated at circuit level, consequently gate level simulators were also borrowed from the designers implementing systems on printed circuit boards. However, by this time some designers had become interested in developing their own suites of CAD tools and several tools evolved which were directed at the time consuming tasks, for example, design rule checking, connectivity checking, layout to function verification. The increased functionality of integrated circuits, also, permitted the development of computer systems with better performance characteristics, which allowed computational intensive CAD tools for example, fault simulators and automatic test pattern generators to be developed. A major factor which influenced the acceptance of CAD tools into the IC design process was that they were introduced incrementally and were oriented towards a psuedo-manual design style. In this design style the layout is considered to be a monolithic entity without any structure, consequently as circuit complexity increased the capabilities of the CAD tools were far exceeded due to the vast amounts of data to be processed. To combat the complexity problem a new generation of CAD tools evolved, which could be classed as revolutionary, since they imposed a design style upon the user. It constrained the designer to using function blocks, for example ROMs, RAMs, PLAs and Shift Registers, whose regularity could be used to advantage in improving the efficiency of the CAD tools which would process the layouts. Simple design languages also evolved to describe the layout of these regular structures at a high level, from which a detailed layout description could be generated automatically to prescribed design rules. PLA layouts could be derived directly from their logical specification. This style of design required the designer to consider the circuit to comprise functions more complex than basic gates, this resulted in the development of high-level simulation tools, which also improved the simulation efficiency of large circuits. The basic trend in the CAD tools, currently under development, is that they perform more of a synthesis rather than an analysis function, for example PLA generators and more recently silicon compilers, whose objective is to automatically generate the complete layout of a circuit from its behavioural description. The trend towards the development of synthesis emphasises the importance of these tools in maintaining the competitive edge in a CAD system.

The individual chapters in this book describe some of the tools discussed in the preceding paragraphs in more detail.

Chapter 2 discusses, in general terms, the need to use computers in the design of VLSI circuits. The VLSI design process is described as a major data processing problem involving vast amount of data, complex algorithms, CPU intensive processes and many aspects of man machine interaction. Typical CAD tasks are outlined and the distinction is drawn between Design

Automation and Computer Aided Design. The Chapter ends with a brief survey of future trends in CAD tools for VLSI design.

Chapter 3 is the first of three chapters discussing tools which may be classified as behavioural analysis tools for both faulty and fault free circuits. This chapter is concerned with techniques of circuit simulation and describes the mathematical modelling and equation formulation of non-linear circuits with lumped elements. Techniques used to provide approximate solutions to systems of non-linear differential equations are also discussed and compared with respect to their numerical efficiency and stability properties. The transformation of a system of non-linear ordinary differential equations into non-linear algebraic equations and a final to a set of linear algebraic equations is also described and a simplified algorithm, which may be used in a classical circuit level simulator for this purpose, is outlined. Finally, in view of the increase in the size of circuit to which this level of simulation is applied and the subsequent increase in solution time, Relaxation Methods are introduced as a means of reducing equation solution time.

Continuing with the topic of simulation Chapter 4 describes the 'anatomy' of a gate level simulator. The basic components which make up a simulator are described together with two techniques for modelling a circuit for the purpose of simulation, namely the Compiled Code and Table Driven Model. The basic simulation algorithm is outlined, together with the basic techniques for evaluating the change in logic state on the output of gate and scheduling events or gate changes in the simulator. The factors which affect the accuracy of the simulation results are also discussed. The problems of using standard simulators to model devices, for example pass transistors, which can have a dynamic state or exhibit a bilateral switching characteristic are outlined. The chapter is concluded with a section outlining the advantages of describing functions using a Behavioural Modelling language and the possibility of using this language to describe analogue functions so that they can be simulated in a digital environment.

A major issue in the design of VLSI circuits is that of testing the devices after fabrication, this topic is addressed in Chapter 5. The basic problems in testing VLSI devices are discussed and this is followed by a brief description of the various Design for Testability (DFT) techniques, namely Ad-hoc, Classical Structured and Neo-Structured methods. Although DFT facilitates the testing of circuits, the problem, in general of generating the test patterns in the first instance, still remains. An important aspect of test pattern generation is fault modelling, the major categories of faults considered in testing are outlined, namely stuck-at-faults, bridging faults, stuck-open faults and pattern sensitive faults. An essential adjunct to test pattern generation is Fault Simulation which is used to determine the fault coverage of the test patterns, the techniques used, namely Parallel, Deductive and Concurrent Simulation are outlined. The basic technique for generating tests in combinational circuits using the Boolean Difference Method and the D-Algorithm are also described, together with a technique called PODEM which was designed to generate tests in very large combinational circuits. The chapter is concluded with a brief description of Testability Analysis Tools and some comments on Automatic Test Equipment.

Chapter 6 is the first of six chapters which deals with topics related to the physical synthesis of a layout. As the complexity of an integrated circuit increased the functions normally connected together on a printed circuit board were now integrated at chip level and designers had a requirement for placement and routing tools, these topics are discussed in Chapter 6. In order to make the layout and interconnections of modules on either an integrated circuit or printed circuit board a tractable problem, the processes of placement and routing are considered separately. The placement algorithms which are outlined in this chapter are the Cluster Development and Force Directed techniques, techniques to improve placements are also discussed together with methods of avoiding wire congestion during the routing phase. An essential part of the layout problem is that of routing the interconnections between the placed modules, the basic method described, to determine the interconnection path, is Lee's Algorithm, together with several heuristics to reduce the routing time. The chapter concludes with a description of Channel and Hierarchical routers.

The traditional method of designing basic logic elements in a layout comprised drawing out the individual shapes on each mask layer required to realise a transistor, contact etc., in the physical circuit. This process is very time consuming and error prone. Recently, basic cell design has been made more efficient by the use of symbolic design tools, these are discussed in Chapter 7. The major techniques described are the Fixed and Relative grid approaches, in the Fixed Grid technique all the design rules are obeyed on entry of the cell description, however using the Relative Grid approach it is necessary to modify the layout to obey various design rules, hence a necessary adjunct to this approach is a Compactor program; several compaction algorithms are briefly described. The technique of using symbolic methods at a higher constructional level are discussed in the form of Floor Planning. The chapter is concluded by a brief description of chip assembly techniques and cell abutment algorithms used in the formation of complex functional blocks from subcells.

Chapter 8 discusses the range of layout analysis tools used in IC design to detect any physical design errors which will either reduce the yield on the fabricated circuit or realise a logic function other than that intended by the designer. The layout tools are classified as either semi-custom or custom, since the semi-custom design style obviates the necessity of using certain analysis tools required in custom design. The major analysis tools used in semi-custom design are wire delay extractors required to determine circuit delay for post layout simulation and circuit verifiers which check that the interconnection of modules on the layout matches that described at a higher level of abstraction.

In a full custom design the designer has fewer constraints, and hence the layout is more likely to contain errors, consequently a wider range of tools are available, for example layout rule checkers, circuit extractors which perform layout to circuit verification, parameter extractors for circuit simulation, electrical rule checkers and netlist to layout verifiers. Techniques to improve the efficiency of these tools by exploiting hierarchy in the circuit are also discussed together with the problems encountered in designs which have overlapping cells. The difficulties in checking the correspondence between the actual and extracted circuits are discussed, together with the role that a good database plays in this function.

Chapter 9 describes the use of High Level languages in the design of integrated circuit layouts. The chapter starts with a brief review of the manual techniques of designing IC layouts and the description of some low level mask layout description languages, outlining their limitations. Some of the unique aspects to be considered when attempting to define a high level language for IC design are identified together with advantages to be gained by using high level languages in the design process. The Bristle Block system is then identified as an example of a first attempt at using high level languages in the design process. The basic disadvantages of the Bristle Block approach are subsequently outlined, and the use of procedural design languages, in overcoming these disadvantages, are discussed. The issues involved in attempting to capture structural and behavioural aspects of a design using procedural types of design languages are also discussed and several examples of existing languages are given. The chapter concludes with a section on special purpose high level languages for IC design in contrast to the use of standard languages with embedded procedures.

A major problem in the design of datapath circuits is the implementation of the control logic for the datapath. The most effective way of implementing this logic is to use a PLA design style. This task has been made easier by the introduction of PLA generators, which may be described as a crude form of silicon compiler and are discussed in Chapter 10. The standard input to a PLA generator is a set of Boolean equations and the output is an regular array structure which realises the Boolean equations. A straight implementation of the Boolean equations, however, results in an inefficient structure in terms of size and performance. Subsequently optimisation techniques, for example logic minimisation and PLA folding methods, to improve the performance of the PLA are discussed. Variations of the basic PLA structure are also outlined, for example Weinberger Arrays and Storage Logic Arrays together with the formation of FSMs from PLAs by including latches between the secondary input and outputs. The chapter concludes with a section dealing with the testing of PLAs and methods used to enhance their testability.

Chapter 11 is the last of the six chapters considering the different aspects of the synthesis of layouts and is concerned with a description of Silicon Compilers. The introduction of the concept of silicon compilers into the design