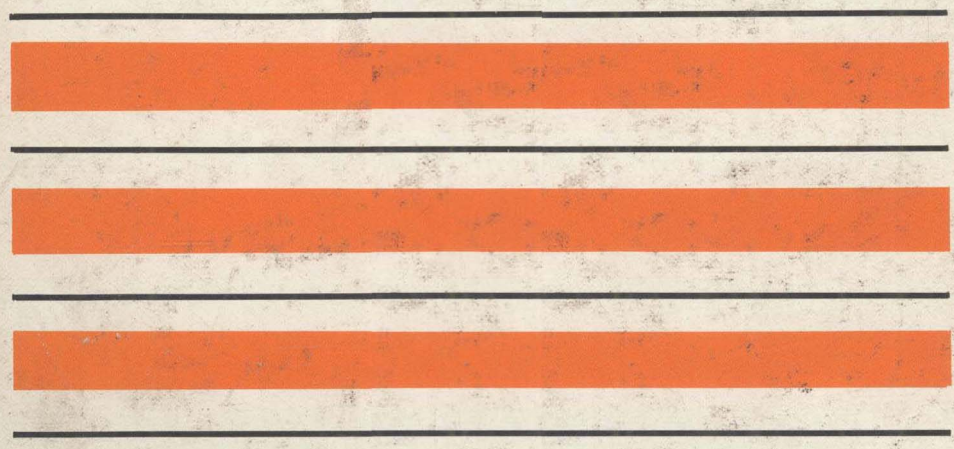
An Artificial Intelligence Approach to VLSI Design

Thaddeus J. Kowalski



Kluwer Academic Publishers

AN ARTIFICIAL INTELLIGENCE APPROACH TO VLSI DESIGN

THADDEUS J. KOWALSKI
Bell Telephone Laboratories, Inc.
Murray Hill, New Jersey

KLUWER ACADEMIC PUBLISHERS
Boston/Dordrecht/Lancaster

Distributors for North America:
Kluwer Academic Publishers
190 Old Derby Street
Hingham, MA 02043, USA

Distributors outside North America:
Kluwer Academic Publishers Group
Distribution Centre
P O Box 322
3300 AH Dordrecht
THE NETHERLANDS

Library of Congress Cataloging in Publication Data

Kowalski, Thaddeus J
An artificial intelligence approach to VLSI design

(The Kluwer international series in engineering and computer science, SECS 4)
Bibliography p
Includes index
1 Integrated circuits — Very large scale integration
2 Artificial intelligence 3 Expert systems
4 Digital electronics I Title II Series
TK7874 K675 1985 621 3819'5835 85-7581
ISBN 0-89838-169-X

Printed in the United States of America

AN ARTIFICIAL INTELLIGENCE APPROACH TO VLSI DESIGN

THE KLUWER INTERNATIONAL SERIES IN ENGINEERING AND COMPUTER SCIENCE

KNOWLEDGE REPRESENTATION, LEARNING AND EXPERT SYSTEMS

Consulting Editor

Tom M. Mitchell

To William Shelley, my fourth grade teacher . . .

He made learning fun.

PREFACE

VLSI design synthesis is a method for designing hardware that starts with an algorithmic description and uses interactive computer programs to create a finished design. This structured approach can decrease the time it takes to design a chip, automatically provide multi-level documentation for the finished design, and create reliable and testable designs. VLSI design synthesis is a difficult problem because the huge number of facts and implicit dynamic constraints do not lend themselves to a recipe-like solution. However, the Knowledge Based Expert System, KBES, approach provides a framework for organizing solutions to problems that are currently solved by experts using large amounts of domain-specific knowledge. Therefore, the goals of my research are to extract and codify the knowledge of expert designers for a better understanding of VLSI design-synthesis, to implement a KBES capable of creating efficient, testable and usable designs, and to determine the usefulness of the KBES technique for implementing design automation tools.

A series of acquisition interviews and an initial prototype system have been used to bootstrap a system that generates a technology-independent list of operators, registers, data paths and control signals from an algorithmic description. The Design Automation Assistant, DAA, has been used to design an IBM System 370 and was favorably evaluated by an IBM System/370 designer.

The advantages of using the KBES approach are the ease of validating the knowledge gathered from interviews with experts, the ease of incrementally adding to the knowledge base, the ability to query that knowledge during the design task, and the replacement of extensive backtracking by domain-specific knowledge techniques. The disadvantage is the difficulty of extracting knowledge from the experts.

This thesis adds to knowledge in the digital design synthesis domain by compiling and testing the set of rules used by expert designers, and to knowledge in the expert system domain by providing another system for researchers to examine. This knowledge will also aid in the teaching of design by making explicit knowledge that is now passed on primarily through apprenticeship.

ACKNOWLEDGEMENT

First and foremost, my profound thanks go to my parents, Henry and Ann Kowalski. None of this could have been done without their indulgence, support, enthusiasm, and love. Thank you.

My sincere thanks goes to my adviser, Donald Thomas for his patience and support through many phone calls and meetings. I would like to thank Stephen Director, Allen Newell, John McDermott, and Daniel Siewiorek for their encouragement, support, and critical review of my research. Also, the thoughtful criticisms and suggestions of friends and colleagues have added greatly to this thesis and to my pleasure in writing it. In particular, David Ditzel, Mitch Marcus, Thomas Mitchell, Lisa Masakowski, and Myron Wish all read multiple versions with care. Marcia Derr, Charles Forgy, Karen Kukich, Walter MacWilliams, Michael McFarland, Sharon Murrel, Peg Schafer, Sherri Shulman, and Sandra Pruzansky have made helpful comments at various stages of the research. The many members of the CMU/DA community have provided many ideas and probing questions, especially David Gatenby, David Geiger, Charles Hitchcock, John Lertola, Matt Mathis, John Nestor, Jayanth Rajan, and Robert Walker. Invaluable assistance with the mechanics of my thesis was provided by the UNIX operating system, the TROFF text processing program, the writers workbench, and the OPS5 knowledge-based expert-system writing tool.

I would like to thank Ken Chong, Claud Davis, David Ditzel, Michael Maul, Gil Mowery, Allen Ross, Clayton Schneider, Glen Williams, and Andrew Wilson for donating time to critique the various designs. Lastly, I would like to thank the INTEL and IBM corporations for providing designers to critique the designs, with special thanks to AT&T Bell Laboratories for computer resources and designer's critiques. This work was supported in part by an IBM fellowship and the National Science Foundation.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

Recent advances in integrated circuit fabrication technology have allowed larger and more complex designs to form complete systems[1] on single VLSI chips. These chips use one-micron to five-micron features to achieve complexities equivalent to 100,000 to 250,000 transistors. This level of design complexity has created a combinatorial explosion of details — a major limitation in realizing cost-effective, low-volume, special-purpose VLSI systems. To overcome this limitation, design tools and methodologies capable of automating more of the digital synthesis process must be built.

We have been developing just such synthesis tools[2,3] in the Carnegie-Mellon University design automation, CMU/DA, community. These tools help the designer develop the algorithmic description of the system and interactively add the details required to produce a finished design. This structured approach can decrease the time it takes to design a chip, automatically provide multi-level documentation for the finished design, and create reliable and testable designs.

This thesis focuses on the synthesis, or allocation, of the implementation-design space as it advances from an algorithmic description

of a VLSI system to a list of technology-independent registers, operators, data paths and control signals. Our approach is aimed at aiding the designer by producing data paths and control sequences that implement the algorithmic system description within supplied constraints. Thus, the designer can consider many alternatives before deciding on a final design.

This task has inspired a variety of approaches, ranging from the most simplistic backtracking methods through the most complicated constraint propagation methods.[4,5,6,7,8] Owing to the complexity of design synthesis, simplistic backtracking schemes consume large amounts of CPU time, and the constraint propagation method is too cumbersome for large designs. Because of the combinatorial explosion of details and implicit dynamic constraints involved in choosing an implementation, this problem does not lend itself to these algorithmic solutions. An alternate approach to design synthesis uses a large amount of design knowledge to eliminate backtracking; whenever possible, the focus is on specific design details and constraints. Artificial intelligence researchers have called systems developed under this heuristic approach knowledge-based expert systems, KBESs.[9] This chapter motivates the research, provides background on the CMU/DA system, lists the related research in the area, states the problem, and provides a road map of the rest of the thesis.

1.1 Motivation

If Moore's law[10] continues to hold, within this decade the size of the smallest VLSI feature will be reduced ten times. This increasing potential to fabricate more complex systems will cause at *least* a linear increase in information that must be managed. From information management studies of large software projects,[11] it can be shown that the time to design and implement future VLSI systems will be considerably more than ten times that required at present, which is prohibitive for low-volume special-purpose VLSI systems. It can be shown further that as the cost of designing special-purpose VLSI systems decreases, the demand for expert designers will increase. Thus, design methods should be developed to deal effectively with the magnitude and complexity of VLSI design. Such methods would increase the potential productivity of designers, while also making it possible for more people to design systems that are both testable and reliable. These are areas even good designers often forget, but can mean the difference between working and marginal designs.

Many people are working on problems related to computer-aided design, or CAD, of VLSI systems. For example, if we look in the proceedings of the Twentieth Design Automation Conference, we see topics

of hardware description languages, testing, simulation, layout and placement, PLA minimization, and synthesis, to name a few. Although synthesis was considered to be strictly in the realm of the creative designer, automatic and computer-aided synthesis programs are now being developed for many levels of VLSI system design. In general, the quality of the designs produced by automatic synthesis programs is not adequate for complete automation of the design process for production use. However, these programs are beginning to find use as an interactive aid during the design process.[2,12,13,14] Toward this end the development of synthesis tools to aid in the creative design process has become an important area of research.

1.2 CMU/DA Levels

At CMU we look at synthesis as the creation of a detailed representation from an abstract representation. The VLSI design synthesis task can be decomposed into several subtasks, each providing an increasing level of detail from the abstract representation. This section describes the four levels of increasing detail used in the CMU/DA system.[2]

1.2.1 The algorithmic level. The first level is an algorithmic description of the design. At this level of detail the high-level intent of hardware can be understood and simulated[15] regardless of the target design style (for example pipeline, multiplexer, microprocessor, or bus) or technology. This level is represented in the instruction set processor language, ISPS,[16] and a value trace data and control flow description language, VT.[17,18] ISPS is a programming language similar to ALGOL, while VT is an extraction of the data flow and control flow information present in the designer's ISPS description. VT is easy for computer programs to manipulate and is felt to be less sensitive to different styles of writing the same algorithmic description.

1.2.2 The technology-independent-hardware-network level. The second level is a technology-independent-hardware-network description of the design. At this level of detail the functionality and connectivity of the hardware can be understood regardless of the target design technology (TTL, ECL, NMOS, or CMOS). This level is represented in the technology-independent-hardware-network language, SCS,[19] which describes technology-independent registers, operators, data paths, and control signals.† Data-memory allocation and control allocation are

defined as the creation of this level from the algorithmic level while applying classical compiler optimizations[21] and design styles.[22]

1.2.3 The technology-dependent-hardware-network level. The third level is a technology-dependent-hardware-network description of the design. At this level of detail the logic and circuits of the hardware can be understood and simulated[23] regardless of physical placement. This level is represented in the technology-dependent-hardware-network language, DIF,[20] which describes feature assignment detail. Feature binding is the creation of this level from the technology-independent level by selection of registers, operators, data paths and control signals that match available feature data-base entries.[24, 25, 26]

1.2.4 The fabrication-dependent-hardware-network level. The fourth level is a fabrication-dependent-hardware-network description of the design. At this level of detail the physical placement of hardware can be understood and simulated.[27] This level is represented in a fabrication-dependent-hardware-network language, which describes feature placement assignment detail. Layout is the creation of this level from the technology-dependent level using geometric information to guide routing and placement of features.

1.3 Problem Statement

Now that we understand how the CMU/DA system divides the task into synthesis levels, let us examine how expert VLSI designers make the transition from the algorithmic description to a hardware implementation. This thesis examines how expert VLSI designers choose a hardware implementation for MOS-microcomputers and whether a KBES can mimic their results. The goals of the research are to extract, codify and test the knowledge of expert designers for a better understanding of VLSI design-synthesis, to implement a KBES capable of creating efficient, testable and usable designs, and to determine the usefulness of the KBES technique for implementing design automation tools. This adds knowledge in the digital design synthesis domain by compiling and testing the set of rules used by expert designers, and to knowledge in the expert system domain by providing another system for researchers to examine. This knowledge will also aid in the teaching of design by making explicit knowledge that is now

† SCS is soon to be replaced by DIF.[20]

passed on primarily through apprenticeship.

1.4 Approach

Through a series of detailed structured interviews, the knowledge that expert VLSI designers use to go from the algorithmic level to the technology-independent hardware-network level has been extracted. The codified knowledge has been tested for completeness and correctness by implementing an interactive system, DAA, that adds register, operator, data path and control signal detail to the VT description of hardware to form the SCS description. DAA is implemented as a production system using the OPS5[28] KBES writing system. This KBES tool is based on the premise that humans solve problems by recognizing familiar sub-problems and apply past solutions. This domain is appropriate for a KBES because there are human experts available whose knowledge has been gained through experience and who can teach this knowledge through apprenticeship. Furthermore, the knowledge required to do the task is extensive and requires the type of organization provided by the KBES approach. The advantages of using the KBES approach are the ease of validating the knowledge gathered from interviews with experts, the ease of incrementally adding to the knowledge base, the ability to query that knowledge during the design task, and the replacement of extensive backtracking by domain-specific knowledge techniques. The disadvantage is the difficulty of extracting knowledge from the experts.

The DAA system has been used to design many computers including the MOS Technology Incorporated MCS6502 and the IBM System/370. The design and many redesigns of the MCS6502 provided stimulus for critiquing the implementation design knowledge contained in DAA. A design of each of the small ISPS descriptions maintained at CMU (RK11, HP21MX, F8, I8080, MOD91, 1802, AM2903, H316, VIDEO, TI1200, PDPTTY, SCF3, PDP4, FP, AM2910, AM2901, ELEV, CHANGE, PQ, MINI, MINIS, AM2909, F9407, AM2902, MARK1) showed that the system would produce a functionally correct design for a large number of test cases. Finally, the knowledge in DAA was tested for generality and robustness by designing a much larger processor, with a completely different instruction set, than DAA had ever seen before, and critiquing this design with a designer not used to develop the knowledge base.

1.5 Related Research

The work done thus far in digital design synthesis is reminiscent of the early chess playing programs.[29] It is easy to get a computer to play a legal