Foreword

Physical design of integrated circuits remains one of the most interesting and challenging arenas in the field of Electronic Design Automation. The ability to integrate more and more devices on our silicon chips requires the algorithms to continuously scale up. Nowadays we can integrate 2e9 transistors on a single 45nm-technology chip. This number will continue to scale for the next couple of technology generations, requiring more transistors to be automatically placed on a chip and connected together. In addition, more and more of the delay is contributed by the wires that interconnect the devices on the chip. This has a profound effect on how physical design flows need to be put together. In the 1990s, it was safe to assume that timing goals of the design could be reached once the devices were placed well on the chip. Today, one does not know whether the timing constraints can be satisfied until the final routing has completed.

As far back as 10 or 15 years ago, people believed that most physical design problems had been solved. But, the continued increase in the number of transistors on the chip, as well as the increased coupling between the physical, timing and logic domains warrant a fresh look at the basic algorithmic foundations of chip implementation. That is exactly what this book provides. It covers the basic algorithms underlying all physical design steps and also shows how they are applied to current instances of the design problems. For example, Chapter 7 provides a great deal of information on special types of routing for specific design situations.

Several other books provide in-depth descriptions of core physical design algorithms and the underlying mathematics, but this book goes a step further. The authors very much realize that the era of individual point algorithms with single objectives is over. Throughout the book they emphasize the multi-objective nature of modern design problems and they bring all the pieces of a physical design flow together in Chapter 8. A complete flow chart, from design partitioning and floorplanning all the way to electrical rule checking, describes all phases of the modern chip implementation flow. Each step is described in the context of the overall flow with references to the preceding chapters for the details.

This book will be appreciated by students and professionals alike. It starts from the basics and provides sufficient background material to get the reader up to speed on the real issues. Each of the chapters by itself provides sufficient introduction and depth to be very valuable. This is especially important in the present era, where experts in one area must understand the effects of their algorithms on the remainder of the design flow. An expert in routing will derive great benefit from reading the chapters on planning and placement. An expert in Design For Manufacturability (DFM) who seeks a better understanding of routing algorithms, and of how these algorithms can be affected by choices made in setting DFM requirements, will benefit tremendously from the chapters on global and detailed routing.

The book is completed by a detailed set of solutions to the exercises that accompany each chapter. The exercises force the student to truly understand the basic physical design algorithms and apply them to small but insightful problem instances.

This book will serve the EDA and design community well. It will be a foundational text and reference for the next generation of professionals who will be called on to continue the advancement of our chip design tools.

Dr. Leon Stok
Vice President, Electronic Design Automation
IBM Systems and Technology Group
Hopewell Junction, NY

Preface

VLSI physical design of integrated circuits underwent explosive development in the 1980s and 1990s. Many basic techniques were suggested by researchers and implemented in commercial tools, but only described in brief conference publications geared for experts in the field. In the 2000s, academic and industry researchers focused on comparative evaluation of basic techniques, their extension to large-scale optimization, and the assembly of point optimizations into multi-objective design flows. Our book covers these aspects of physical design in a consistent way, starting with basic concepts in Chapter 1 and gradually increasing the depth to reach advanced concepts, such as physical synthesis. Readers seeking additional details, will find a number of references discussed in each chapter, including specialized monographs and recent conference publications.

Chapter 2 covers netlist partitioning. It first discusses typical problem formulations and proceeds to classic algorithms for balanced graph and hypergraph partitioning. The last section covers an important application – system partitioning among multiple FPGAs, used in the context of high-speed emulation in functional validation.

Chapter 3 is dedicated to chip planning, which includes floorplanning, power-ground planning and I/O assignment. A broad range of topics and techniques are covered, ranging from graph-theoretical aspects of block-packing to optimization by simulated annealing and package-aware I/O planning.

Chapter 4 addresses VLSI placement and covers a number of practical problem formulations. It distinguishes between global and detailed placement, and first covers several algorithmic frameworks traditionally used for global placement. Detailed placement algorithms are covered in a separate section. Current state of the art in placement is reviewed, with suggestions to readers who might want to implement their own software tools for large-scale placement.

Chapters 5 and 6 discuss global and detailed routing, which have received significant attention in research literature due to their interaction with manufacturability and chip-yield optimizations. Topics covered include representing layout with graph models and performing routing, for single and multiple nets, in these models. State-of-the-art global routers are discussed, as well as yield optimizations performed in detailed routing to address specific types of manufacturing faults.

Chapter 7 deals with several specialized types of routing which do not conform with the global-detailed paradigm followed by Chapters 5 and 6. These include non-Manhattan area routing, commonly used in PCBs, and clock-tree routing required for every synchronous digital circuit. In addition to algorithmic aspects, we explore the impact of process variability on clock-tree routing and means of decreasing this impact.

Chapter 8 focuses on timing closure, and its perspective is particularly unique. It offers a comprehensive coverage of timing analysis and relevant optimizations in placement, routing and netlist restructuring. Section 8.6 assembles all these techniques, along with those covered in earlier chapters, into an extensive design flow, illustrated in detail with a flow chart and discussed step-by-step with several figures and many references.

This book does not assume prior exposure to physical design or other areas of EDA. It introduces the reader to the EDA industry and basic EDA concepts, covers key graph concepts and algorithm analysis, carefully defines terms and specifies basic algorithms with pseudocode. Many illustrations are given throughout the book, and every chapter includes a set of exercises, solutions to which are given in one of the appendices. Unlike most other sources on physical design, we made an effort to avoid impractical and unnecessarily complicated algorithms. In many cases we offer comparisons between several leading algorithmic techniques and refer the reader to publications with additional empirical results.

Some chapters are based on material in the book *Layoutsynthese elektronischer Schaltungen – Grundlegende Algorithmen für die Entwurfsautomatisierung*, which was published by Springer in 2006.

We are grateful to our colleagues and students who proofread earlier versions of this book and suggested a number of improvements (in alphabetical order): Matthew Guthaus, Kwangok Jeong, Johann Knechtel, Andreas Krinke, Nancy MacDonald, Jarrod Roy, Yen-Kuan Wu and Hailong Yao.

Images for global placement and clock routing in Chapter 8 were provided by Myung-Chul Kim and Dong-Jin Lee. Cell libraries in Appendix B were provided by Bob Bullock, Dan Clein and Bill Lye from PMC Sierra; the layout and schematics in Appendix B were generated by Matthias Thiele. The work on this book was partially supported by the National Science Foundation (NSF) through the CAREER award 0448189 as well as by Texas Instruments and Sun Microsystems.

We hope that you will find the book interesting to read and useful in your professional endeavors.

Sincerely,

Andrew, Jens, Igor and Jin

Table of Contents

Chapter 1

1 Introduction

1

1 Introduction

The design and optimization of *integrated circuits* (*ICs*) are essential to the production of new semiconductor chips. Modern chip design has become so complex that it is largely performed by specialized software, which is frequently updated to reflect improvements in semiconductor technologies and increasing design complexities. A *user* of this software needs a high-level understanding of the implemented algorithms. On the other hand, a *developer* of this software must have a strong computer-science background, including a keen understanding of how various algorithms operate and interact, and what their performance bottlenecks are.

This book introduces and evaluates algorithms used during *physical design* to produce a geometric chip layout from an abstract circuit design. Rather than list every relevant technique, however, it presents the essential and fundamental algorithms used within each physical design stage.

- *Partitioning* (Chap. 2) and *chip planning* (Chap. 3) of design functionality during the initial stages of physical design
- Geometric *placement* (Chap. 4) and *routing* (Chaps. 5-6) of circuit components
- Specialized routing and *clock tree synthesis* for synchronous circuits (Chap. 7)
- Meeting specific technology and performance requirements, i.e., *timing closure*, such that the final fabricated layout satisfies system objectives (Chap. 8)

Other design steps, such as *circuit design*, *logic synthesis*, *transistor-level layout* and *verification*, are not discussed in detail, but are covered in such references as [1.1].

This book emphasizes *digital* circuit design for *very large-scale integration* (*VLSI*); the degree of automation for digital circuits is significantly higher than for *analog* circuits. In particular, the focus is on algorithms for digital ICs, such as system partitioning for *field-programmable gate arrays* (*FPGAs*) or *clock network synthesis* for *application-specific integrated circuits* (*ASICs*). Similar design techniques can be applied to other implementation contexts such as *multi-chip modules* (*MCMs*) and *printed circuit boards* (*PCBs*).

The following broad questions, of interest to both students and designers, are addressed in the upcoming chapters.

- How is functionally correct layout produced from a netlist?
- How does software for VLSI physical design work?
- How do we develop and improve software for VLSI physical design?

More information about this book is at http://vlsicad.eecs.umich.edu/KLMH/.

1.1 1.1 Electronic Design Automation (EDA)

The *Electronic Design Automation* (*EDA*) industry develops software to support engineers in the creation of new integrated-circuit (IC) designs. Due to the high complexity of modern designs, EDA touches almost every aspect of the IC design flow, from high-level system design to fabrication. EDA addresses designers' needs at multiple levels of electronic system hierarchy, including integrated circuits (ICs), multi-chip modules (MCMs), and printed circuit boards (PCBs).

Progress in semiconductor technology, based on *Moore's Law* (Fig. 1.1), has led to integrated circuits (1) comprised of hundreds of millions of transistors, (2) assembled into packages, each having multiple chips and thousands of pins, and (3) mounted onto *high-density interconnect* (*HDI*) circuit boards with dozens of wiring layers. This design process is highly complex and heavily depends on automated tools. That is, computer software is used to mostly automate design steps such as logic design, simulation, physical design, and verification.

EDA was first used in the 1960s in the form of simple programs to automate placement of a very small number of blocks on a circuit board. Over the next few years, the advent of the integrated circuit created a need for software that could reduce the total number of gates. Current software tools must additionally consider electrical effects such as signal delays and capacitive coupling between adjacent wires. In the modern VLSI design flow, nearly all steps use software to automate optimizations.

In the 1970s, semiconductor companies developed in-house EDA software, specialized programs to address their proprietary design styles. In the 1980s and 1990s, independent software vendors created new tools for more widespread use. This gave rise to an independent EDA industry, which now enjoys annual revenues of approximately five billion dollars and employs around twenty thousand people. Many EDA companies have headquarters in Santa Clara county, in the state of California. This area has been aptly dubbed the *Silicon Valley*.

Several annual conferences showcase the progress of the EDA industry and academia. The most notable one is the *Design Automation Conference* (*DAC*), which holds an industry trade show as well as an academic symposium. The *International Conference on Computer-Aided Design* (*ICCAD*) places emphasis on academic research, with papers that relate to specialized algorithm development. PCB developers attend *PCB Design Conference West* in September. Overseas, Europe and Asia host the *Design, Automation and Test in Europe* (*DATE*) conference and the *Asia and South Pacific Design Automation Conference* (*ASP-DAC*), respectively. The world-wide engineering association *Institute of Electrical and Electronic Engineers* (*IEEE*) publishes the monthly *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (*TCAD*), while the *Association for Computing Machinery* (*ACM*) publishes *ACM Transactions on Design Automation of Electronic Systems* (*TODAES*).

Impact of EDA. According to Moore's Law (Fig. 1.1), the number of transistors on a chip is increasing at an exponential rate. Historically, this corresponds to an annual *compounded* increase of 58% in the number of transistors per chip.

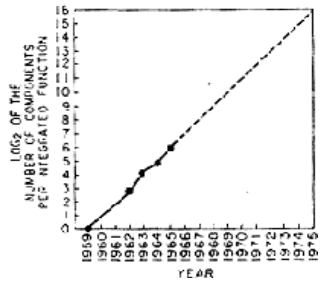

Fig. 1.1 Moore's Law and the original graph from Gordon Moore's article [1.10] predicting the increase in number of transistors. In 1965, Gordon Moore (Fairchild) stated that the number of transistors on an IC would double every year. Ten years later, he revised his statement, asserting that doubling would occur every 18 months. Since then, this "rule" has been famously known as Moore's Law.

However, chip designs produced by prominent semiconductor companies suggest a different trend. The annual *productivity*, measured by the number of transistors, of designers, and (fixed-size) design teams has an annual compounded growth of only around 21% per year, leading to a *design productivity gap* [1.5]. Since the number of transistors is highly context-specific – analog versus digital or memory versus logic – this statistic, due to SEMATECH in the mid-1990s, refers to the design productivity for *standardized transistors*.

Fig 1.2, reproduced from the *International Technology Roadmap for Semiconductors* (*ITRS*) [1.5], demonstrates that cost-feasible IC products require innovation in EDA technology. Given the availability of efficient design technologies to semiconductor design teams, the hardware design cost of a typical portable system-on-chip, e.g., a baseband processor chip for a cell phone, remains manageable at $15.7 million (2009 estimate). With associated software design costs, the overall chip design project cost is $45.3 million. Without the design technology innovations between 1993 and 2007, and the resulting design productivity improvements, the design cost of a chip would have been $1,800 million, well over a *billion* dollars.

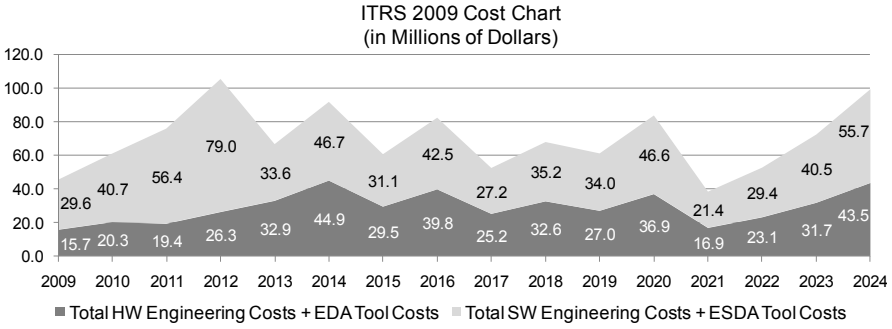

Fig. 1.2 Recent editions of the semiconductor technology roadmap project total hardware (HW) engineering costs + EDA tool costs (dark gray) and total software (SW) engineering costs + electronic system design automation (ESDA) tool costs (light gray). This shows the impact of EDA technologies on overall IC design productivity and hence IC design cost [1.5].

History of EDA. After tools for schematic entry of integrated circuits were developed, the first EDA tool, a placer that optimized the physical locations of devices on a circuit board, was created in the late 1960s. Shortly thereafter, programs were written to aid circuit layout and visualization. The first integrated-circuit *computer-aided design* (*CAD*) systems addressed the physical design process and were written in the 1970s. During that era, most CAD tools were proprietary – major companies such as IBM and AT&T Bell Laboratories relied on software tools designed for internal use only. However, beginning in the 1980s, independent software developers started to write tools that could serve the needs of multiple semiconductor product companies. The electronic design automation (EDA) market grew rapidly in the 1990s, as many design teams adopted commercial tools instead of developing their own in-house versions. The largest EDA software vendors today are, in alphabetical order, *Cadence Design Systems*, *Mentor Graphics*, and *Synopsys*.

EDA tools have always been geared toward automating the entire design process and linking the design steps into a complete design flow. However, such integration is challenging, since some design steps need additional degrees of freedom, and scalability requires tackling some steps independently. On the other hand, the continued decrease of transistor and wire dimensions has blurred the boundaries and abstractions that separate successive design steps – physical effects such as signal delays and coupling capacitances need to be accurately accounted for earlier in the design cycle. Thus, the design process is moving from a sequence of atomic (independent) steps toward a deeper level of integration. Tab. 1.1 summarizes a timeline of key developments in circuit and physical design.

Tab. 1.1 Timeline of EDA progress with respect to circuit and physical design.

Time Period	Circuit and Physical Design Process Advancements
1950-1965	Manual design only.
1965-1975	Layout editors, e.g., place and route tools, first developed for printed circuit boards.
1975-1985	More advanced tools for ICs and PCBs, with more sophisticated algorithms.
1985-1990	First performance-driven tools and parallel optimization algorithms for layout; better understanding of underlying theory (graph theory, solution complexity, etc.).
1990-2000	First over-the-cell routing, first 3D and multilayer placement and routing techniques developed. Automated circuit synthesis and routability-oriented design become dominant. Start of parallelizing workloads. Emergence of physical synthesis.
2000-now	Design for Manufacturability (DFM), optical proximity correction (OPC), and other techniques emerge at the design-manufacturing interface. Increased reusability of blocks, including intellectual property (IP) blocks.