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 to address increasing design complexities. To achieve the best results, a *user* of this software needs a high-level understanding of the implemented algorithms. In addition, a *developer* of such software must have a keen understanding of relevant computer science aspects, including algorithmic performance bottlenecks and how various algorithms operate and interact.

This book introduces and compares the fundamental algorithms that are used during the IC physical design phase, wherein a geometric chip layout is produced starting 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 the target functionality during the initial stages of physical design
– Geometric *placement* (Chap. 4) and *routing* (Chaps. 5 and 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*, to ensure 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 references such as [1].

This book emphasizes *digital* circuit design for *very large-scale integration* (*VLSI*); the degree of automation for digital circuits is significantly higher than that for *analog* circuits. In particular, our 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

A. B. Kahng et al., *VLSI Physical Design: From Graph Partitioning to Timing Closure*, https://doi.org/10.1007/978-3-030-96415-3_1

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 a functionally correct layout produced from a netlist?
– How does software for VLSI physical design work to produce such a layout?
– How do we develop and improve software for VLSI physical design?

More information about this book is at https://www.ifte.de/books/eda/index.html.

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* [8], has led to integrated circuits (1) comprised of billions 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 dependent on automated tools. That is, computer software is able to automate design steps such as logic design, simulation, physical design, and verification, to a very high degree.

EDA first appeared in the 1960s in the form of simple programs to convert layout information into tapes for the Gerber photoplotter and to automate placement of a very small number of devices on a circuit board. Over the next few years, the advent of the integrated circuit enabled increasingly complex circuits to be implemented, a trend that has only accelerated. These evermore complex circuits, which now can contain billions of devices, have driven the need for EDA software that can handle such large circuits, optimize the resulting physical layouts for speed, space, and power efficiency, and ensure correctness and manufacturability. In the modern VLSI design flow, 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 a healthy revenue growth rate and employs tens of thousands of people around the world. 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 EDA in industry and academia. The most notable one is the *Design Automation Conference* (*DAC*),

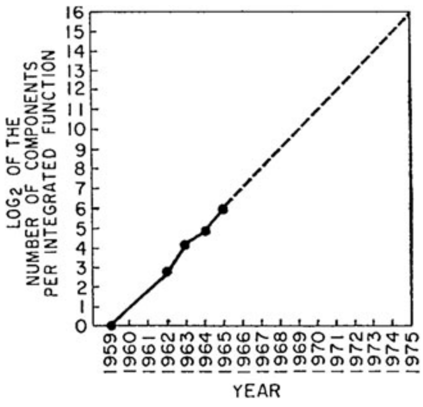

Fig. 1.1 Moore's law and the original graph from Gordon Moore's article [8] predicting the increase in the number of transistors. In 1965, Gordon Moore (Fairchild) stated that the number of transistors on an IC would double every year. Ten years later, he revised his statement, asserting that doubling would occur every 2 years. Since then, this "rule" has been famously known as Moore's law

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 advancements. 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 worldwide engineering association *Institute of Electrical and Electronic Engineers* (*IEEE*) publishes the monthly journal *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 doubling every 2 years. This corresponds to an annual *compounded* increase of 41.4% in the number of transistors per chip.

However, chip designs produced by prominent semiconductor companies suggest a different trend. The annual *productivity*, measured by the number of transistors and the required number of designers (person-years), has an annual compounded growth of "only" around 21% per year, leading to a *design productivity gap* [4]. Hence, this gap symbolizes the divergence between technology capabilities (transistors per IC) and design capabilities (transistors per person-year). Since the number of transistors is highly context specific—analog versus digital or memory versus logic—this statistic, introduced by SEMATECH in the mid-1990s, refers to the design productivity for so-called *standardized transistors*.

The design gap in digital logic design has been written about extensively. It is visualized in Fig. 1.2 by the upper shaded area and the brown vertical arrow, and is one of the toughest and most urgent problems in microelectronics. The analog design gap and the gap between analog and digital design productivity (red vertical arrow)

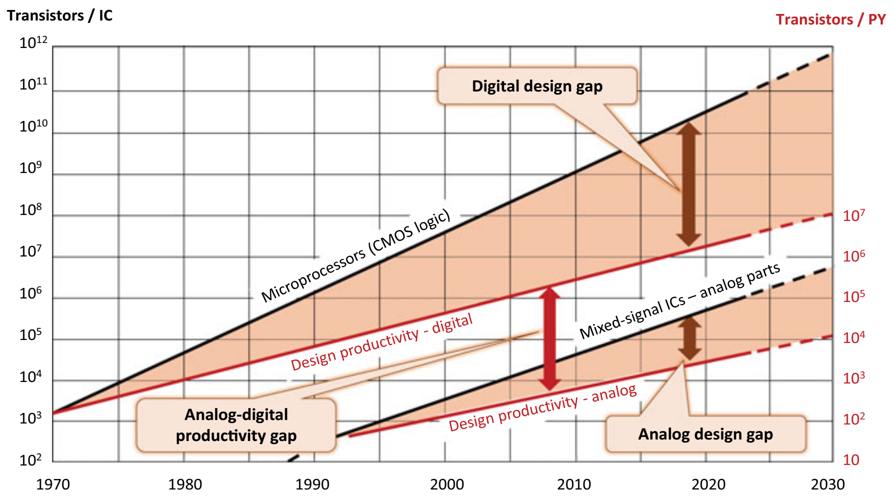

Fig. 1.2 Approximate growth rates for standardized transistors per IC (*black*, left scale) and design productivity in standardized transistors per person-year (*red*, right scale) for digital chips (above) and the analog part of mixed-signal ICs (*bottom*) [5]. The digital design gap, the analog design gap, and the gap between analog and digital design productivity (*red vertical arrow*) are also shown [5]

are also shown in Fig. 1.2 [5]. As we will elaborate upon in later chapters, EDA plays a crucial role in mitigating these design gaps through improvements that further automate the electronic design process.

History of EDA Before EDA, integrated circuits and printed circuit boards were designed by hand. Automation started in the mid-1960s with geometric software to generate the tapes for the layout photoplotter by copying digital recordings of manually drawn components. The first integrated circuit *computer-aided design* (*CAD*) systems were established in the 1970s to address the physical design process. More and more programs were written to aid circuit layout generation, such as place and route, and visualization. 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.

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 require 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

Table 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	First software to generate the tapes for the layout photoplotter by copying digital recordings of manually drawn components. Layout editors, e.g., place and route tools, first developed for PCBs
1975–1985	More advanced tools for ICs and PCBs, with more sophisticated algorithms. First annealing algorithms
1985–1990	First performance-driven tools and parallel optimization algorithms for layout; better understanding of the 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. Quadratic algorithms. Layout hotspot detection
2000–2010	Design for manufacturability (DFM). Design for test (DFT). Optical proximity correction (OPC) and other techniques emerge at the design-manufacturing interface. Increased reusability of blocks, including intellectual property (IP) blocks. Low-power methods such as voltage islands and clock gating
2010–2020	Built-in self-test (BIST). Emergence of layout post-processing and migration-mitigating design methodologies, monolithic 3D design, and more-than-Moore integration. Renewal of procedural automation (generator) methods. Hyperoptimization
2020–now	Machine learning (ML) boosting of algorithms. Loss of orthogonalizations and abstractions. Fusions. Co-optimizations

in the design cycle. Thus, the design process is moving from a sequence of segmented (independent) steps toward a deeper level of integration. Table 1.1 summarizes a timeline of key developments in circuit and physical design.

1.2 VLSI Design Flow

The process of designing a very-large-scale integrated (VLSI) circuit is highly complex. It can be separated into distinct steps (Fig. 1.3). Earlier steps are at a high level of abstraction, typically focusing on overall system functionality and logical design, while later design steps are at lower levels of abstraction, where the focus moves toward electrical properties of physical components (devices and interconnecting wires) and how these affect signal delays and other characteristics of the chip (e.g., power consumption, reliability). At the end of the process, before fabrication, tools and algorithms operate on detailed information about each circuit element's geometric shape and electrical properties.

The steps of the VLSI design flow in Fig. 1.3 are discussed in detail below and throughout the chapters in this book. For further reading on specialized topics in physical design algorithms, see [1, 5, 9].