CHAPTER

1 INTRODUCTION

The evolution of digital circuit design

■

Compelling issues in digital circuit design

■

Valuable references

1.1 A Historical Perspective

The concept of digital data manipulation has made a dramatic impact on our society. One has long grown accustomed to the idea of digital computers. Evolving steadily from mainframe and minicomputers, personal and laptop computers have proliferated into daily life. More significant, however, is a continuous trend towards digital solutions in all other areas of electronics. Instrumentation was one of the first noncomputing domains where the potential benefits of digital data manipulation over analog processing were recognized. Other areas such as control were soon to follow. Only recently have we witnessed the conversion of telecommunications and consumer electronics towards the digital format. Increasingly, telephone data is transmitted and processed digitally over both wired and wireless networks. The compact disk has revolutionized the audio world, and digital video is following in its footsteps.

The idea of implementing computational engines using an encoded data format is by no means an idea of our times. In the early nineteenth century, Babbage envisioned large-scale mechanical computing devices, called *Difference Engines* [Swade93]. Although these engines use the decimal number system rather than the binary representation now common in modern electronics, the underlying concepts are very similar. The Analytical Engine, developed in 1834, was perceived as a general-purpose computing machine, with features strikingly close to modern computers. Besides executing the basic repertoire of operations (addition, subtraction, multiplication, and division) in arbitrary sequences, the machine operated in a two-cycle sequence, called "store" and "mill" (execute), similar to current computers. It even used pipelining to speed up the execution of the addition operation! Unfortunately, the complexity and the cost of the designs made the concept impractical. For instance, the design of Difference Engine I (part of which is shown in Figure 1.1) required 25,000 mechanical parts at a total cost of £17,470 (in 1834!).

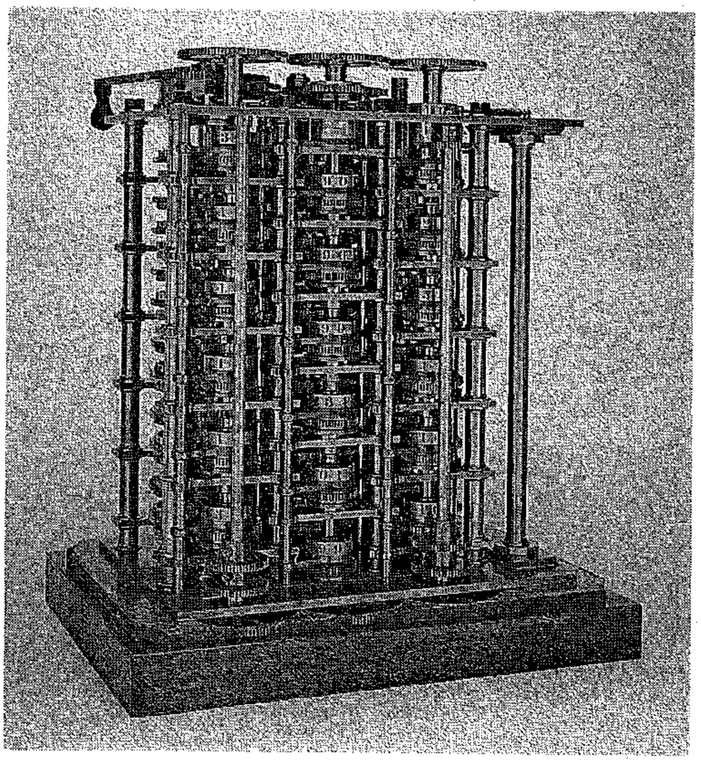


Figure 1.1 Working part of Babbage's Difference Engine I (1832), the first known automatic calculator (from [Swade93], courtesy of the Science Museum of London).

The electrical solution turned out to be more cost effective. Early digital electronics systems were based on magnetically controlled switches (or relays). They were mainly used in the implementation of very simple logic networks. Examples of such are train safety systems, where they are still being used at present. The age of digital electronic computing only started in full with the introduction of the vacuum tube. While originally used almost exclusively for analog processing, it was realized early on that the vacuum tube was useful for digital computations as well. Soon complete computers were realized. The era of the vacuum tube based computer culminated in the design of machines such as the ENIAC (intended for computing artillery firing tables) and the UNIVAC I (the first successful commercial computer). To get an idea about *integration density*, the ENIAC was 80 feet long, 8.5 feet high and several feet wide and incorporated 18,000 vacuum tubes. It became rapidly clear, however, that this design technology had reached its limits. Reliability problems and excessive power consumption made the implementation of larger engines economically and practically infeasible.

All changed with the invention of the *transistor* at Bell Telephone Laboratories in 1947 [Bardeen48], followed by the introduction of the bipolar transistor by Schockley in 1949 [Schockley49][1]. It took till 1956 before this led to the first bipolar digital logic gate, introduced by Harris [Harris56], and even more time before this translated into a set of integrated-circuit commercial logic gates, called the Fairchild Micrologic family [Norman60]. The first truly successful IC logic family, *TTL (Transistor-Transistor Logic)* was pioneered in 1962 [Beeson62]. Other logic families were devised with higher performance in mind. Examples of these are the current switching circuits that produced the first subnanosecond digital gates and culminated in the *ECL (Emitter-Coupled Logic)* family [Masaki74], which is discussed in more detail in this textbook. TTL had the advantage, however, of offering a higher integration density and was the basis of the first integrated circuit revolution. In fact, the manufacturing of TTL components is what spear-headed the first large semiconductor companies such as Fairchild, National, and Texas Instruments. The family was so successful that it composed the largest fraction of the digital semiconductor market until the 1980s.

Ultimately, bipolar digital logic lost the battle for hegemony in the digital design world for exactly the reasons that haunted the vacuum tube approach: the large power consumption per gate puts an upper limit on the number of gates that can be reliably integrated on a single die, package, housing, or box. Although attempts were made to develop high integration density, low-power bipolar families (such as *I^2L—Integrated Injection Logic* [Hart72]), the torch was gradually passed to the MOS digital integrated circuit approach.

The basic principle behind the MOSFET transistor (originally called IGFET) was proposed in a patent by J. Lilienfeld (Canada) as early as 1925, and, independently, by O. Heil in England in 1935. Insufficient knowledge of the materials and gate stability problems, however, delayed the practical usability of the device for a long time. Once these were solved, MOS digital integrated circuits started to take off in full in the early 1970s. Remarkably, the first MOS logic gates introduced were of the CMOS variety [Wanlass63], and this trend continued till the late 1960s. The complexity of the manufacturing process

[1] An intriguing overview of the evolution of digital integrated circuits can be found in [Murphy93]. (Most of the data in this overview has been extracted from this reference). It is accompanied by some of the historically ground-breaking publications in the domain of digital IC's.

delayed the full exploitation of these devices for two more decades. Instead, the first practical MOS integrated circuits were implemented in PMOS-only logic and were used in applications such as calculators. The second age of the digital integrated circuit revolution was inaugurated with the introduction of the first microprocessors by Intel in 1972 (the 4004) and 1974 (the 8080) [Shima74]. These processors were implemented in NMOS-only logic, that has the advantage of higher speed over the PMOS logic. Simultaneously, MOS technology enabled the realization of the first high-density semiconductor memories. For instance, the first 4Kbit MOS memory was introduced in 1970 [Hoff70].

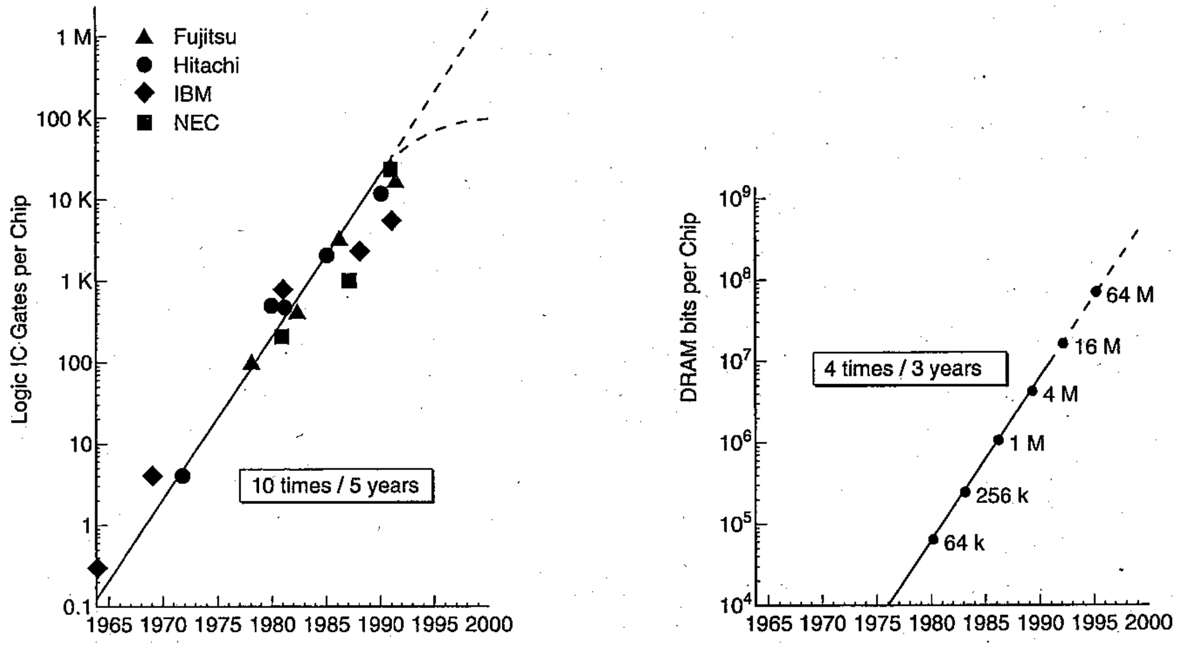
These events were at the start of a truly astounding evolution towards ever higher integration densities and speed performances, a revolution that is still in full swing right now. The road to the current levels of integration has not been without hindrances, however. In the late 1970s, NMOS-only logic started to suffer from the same plague that made high-density bipolar logic unattractive or infeasible: power consumption. This realization, combined with progress in manufacturing technology, finally tilted the balance towards the CMOS technology, and this is where we still are today. Interestingly enough, power consumption concerns are rapidly becoming dominant in CMOS design as well, and this time there does not seem to be a new technology around the corner to alleviate the problem.

Although the large majority of the current integrated circuits are implemented in the MOS technology, other technologies come into play when very high performance is at stake. An example of this is the BiCMOS technology that combines bipolar and MOS devices on the same die. BiCMOS is effectively used in high-speed memories and gate arrays. When even higher performance is necessary, other technologies emerge besides the already mentioned bipolar silicon ECL family—Gallium-Arsenide, Silicon-Germanium and even superconducting technologies. While these circuits only fill in a small niche in the overall digital integrated circuit design scene, it is worth examining some of the issues emerging in the design of these circuits. With the continuing increase in performance of digital MOS circuits, design problems currently encountered in these high-speed technologies might come to haunt CMOS as well in the foreseeable future.

1.2 Issues in Digital Integrated Circuit Design

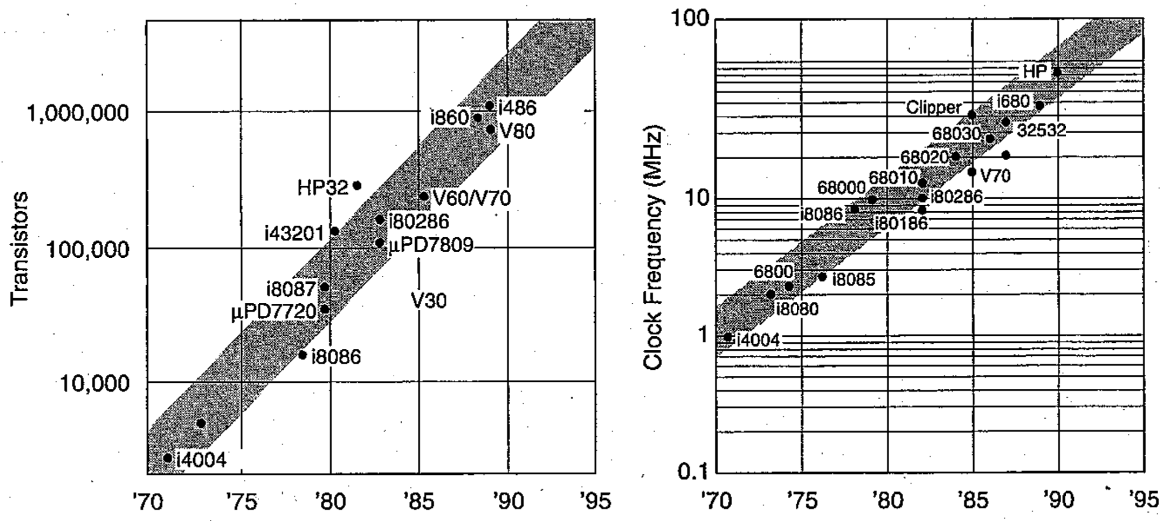
Integration density and performance of integrated circuits have gone through an astounding revolution in the last couple of decades. In the 1960s, Gordon Moore, then with Fairchild Corporation and later cofounder of Intel, predicted that the number of transistors that can be integrated on a single die would grow exponentially with time. This prediction, later called *Moore's law*, has proven to be amazingly visionary. Its validity is best illustrated with the aid of a set of graphs. Figure 1.2 plots the integration density of both logic ICs and memory as a function of time. As can be observed, integration complexity doubles approximately every 1 to 2 years. As a result, memory density has increased by more than a thousandfold since 1970.

An intriguing case study is offered by the microprocessor. From its inception in the early seventies, the microprocessor has grown in performance and complexity at a steady and predictable pace. The number of transistors and the clock frequency for a number of landmark designs are collected in Figure 1.3. The million-transistor/chip barrier was crossed in the late eighties. Clock frequencies double every three years and have reached



(a) Trends in logic IC complexity (b) Trends in memory complexity

Figure 1.2 Evolution of integration complexity of logic ICs and memories as a function of time (from [Masaki92]).

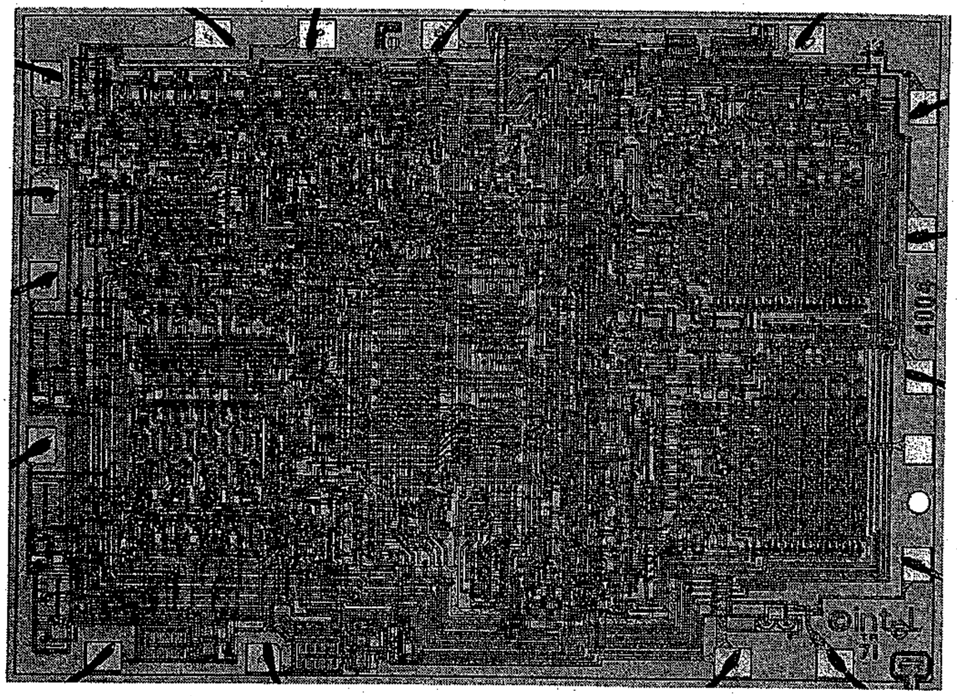


(a) Trends in transistor count (b) Trends in clock frequency

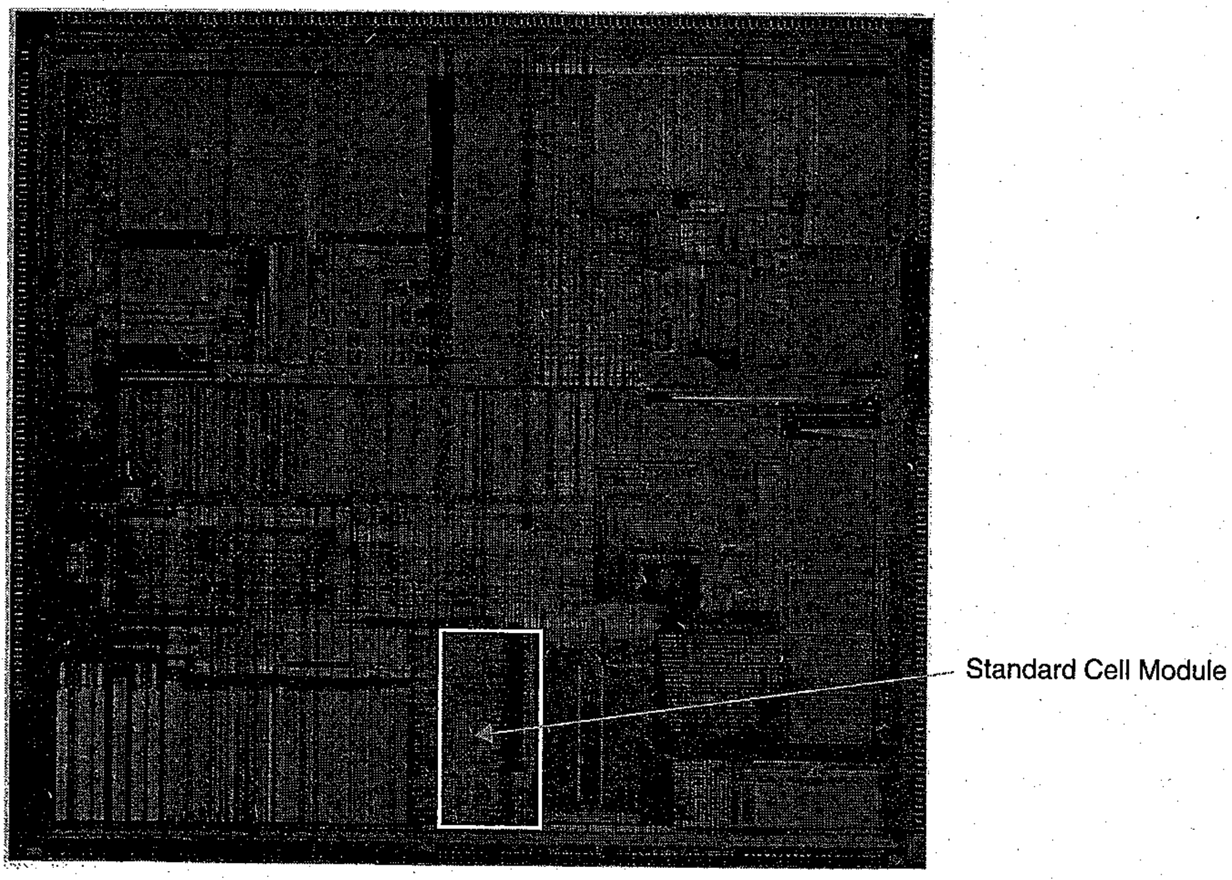
Figure 1.3 Evolution of microprocessor transistor count and clock frequency (from [Sasaki91]).

into the 100 MHz range. An even more important observation is that, as of now, these trends have not shown any signs of a slow-down.

It should be no surprise to the reader that this revolution has had a profound impact on how digital circuits are designed. Early designs were truly hand-crafted. Every transistor was laid out and optimized individually and carefully fitted into its environment. This is adequately illustrated in Figure 1.4a, which shows the design of the Intel 4004 microprocessor. This approach is, obviously, not appropriate when more than a million devices have to be created and assembled. With the rapid evolution of the design technology, time-



(a) The 4004 microprocessor (see also back cover)



(b) The Pentium™ microprocessor (see also back cover)

Figure 1.4 Comparing the design methodologies of the Intel 4004 (1971) and Pentium™ (1994) microprocessors (reprinted with permission from Intel).

to-market is one of the crucial factors in the ultimate success of a component. Designers have, therefore, increasingly adhered to rigid design methodologies and strategies that are more amenable to design automation. The impact of this approach is apparent from the layout of one of the later Intel microprocessors, the Pentium, shown in Figure 1.4b. Instead of the individualized approach of the earlier designs, a circuit is constructed in a hierarchical way: a processor is a collection of modules, each of which consists of a number of cells on its own. Cells are reused as much as possible to reduce the design effort and to enhance the chances for a first-time-right implementation. The fact that this hierarchical approach is at all possible is the key ingredient for the success of digital circuit design and also explains why, for instance, very large scale analog design has never caught on.

The obvious next question is why such an approach is feasible in the digital world and not (or to a lesser degree) in analog designs. The crucial concept here, and the most important one in dealing with the complexity issue, is *abstraction*. At each design level, the internal details of a complex module can be abstracted away and replaced by a *black box view* or *model*. This model contains virtually all the information needed to deal with the block at the next level of hierarchy. For instance, once a designer has implemented a multiplier module, its performance can be defined very accurately and can be captured in a model. The performance of this multiplier is in general only marginally influenced by the way it is utilized in a larger system. For all purposes, it can hence be considered a black box with known characteristics. As there exists no compelling need for the system designer to look inside this box, design complexity is substantially reduced. The impact of this *divide and conquer* approach is dramatic. Instead of having to deal with a myriad of elements, the designer has to consider only a handful of components, each of which are characterized in performance and cost by a small number of parameters.

This is analogous to a software designer using a library of software routines such as input/output drivers. Someone writing a large program does not bother to look inside those library routines. The only thing he cares about is the intended result of calling one of those modules. Imagine what writing software programs would be like if one had to fetch every bit individually from the disk and ensure its correctness instead of relying on handy "file open" and "get string" operators.

Typically used abstraction levels in digital circuit design are, in order of increasing abstraction, the device, circuit, gate, functional module (e.g., adder) and system levels (e.g., processor), as illustrated in Figure 1.5. A semiconductor device is an entity with a very complex behavior. No circuit designer will ever seriously consider the solid-state physics equations governing the behavior of the device when designing a digital gate. Instead he will use a simplified model that adequately describes the input-output behavior of the transistor. For instance, an AND gate is adequately described by its Boolean expression ($Z = A.B$), its bounding box, the position of the input and output terminals, and the delay between the inputs and the output.

This design philosophy has been the enabler for the emergence of elaborate *computer-aided design* (CAD) frameworks for digital integrated circuits; without it the current design complexity would not have been achievable. Design tools include simulation at the various complexity levels, design verification, layout generation, and design synthesis. An overview of these tools and design methodologies is given in Chapter 11 of this textbook.

Furthermore, to avoid the redesign and reverification of frequently used cells such as basic gates and arithmetic and memory modules, designers most often resort to *cell librar-*

STE-101-106

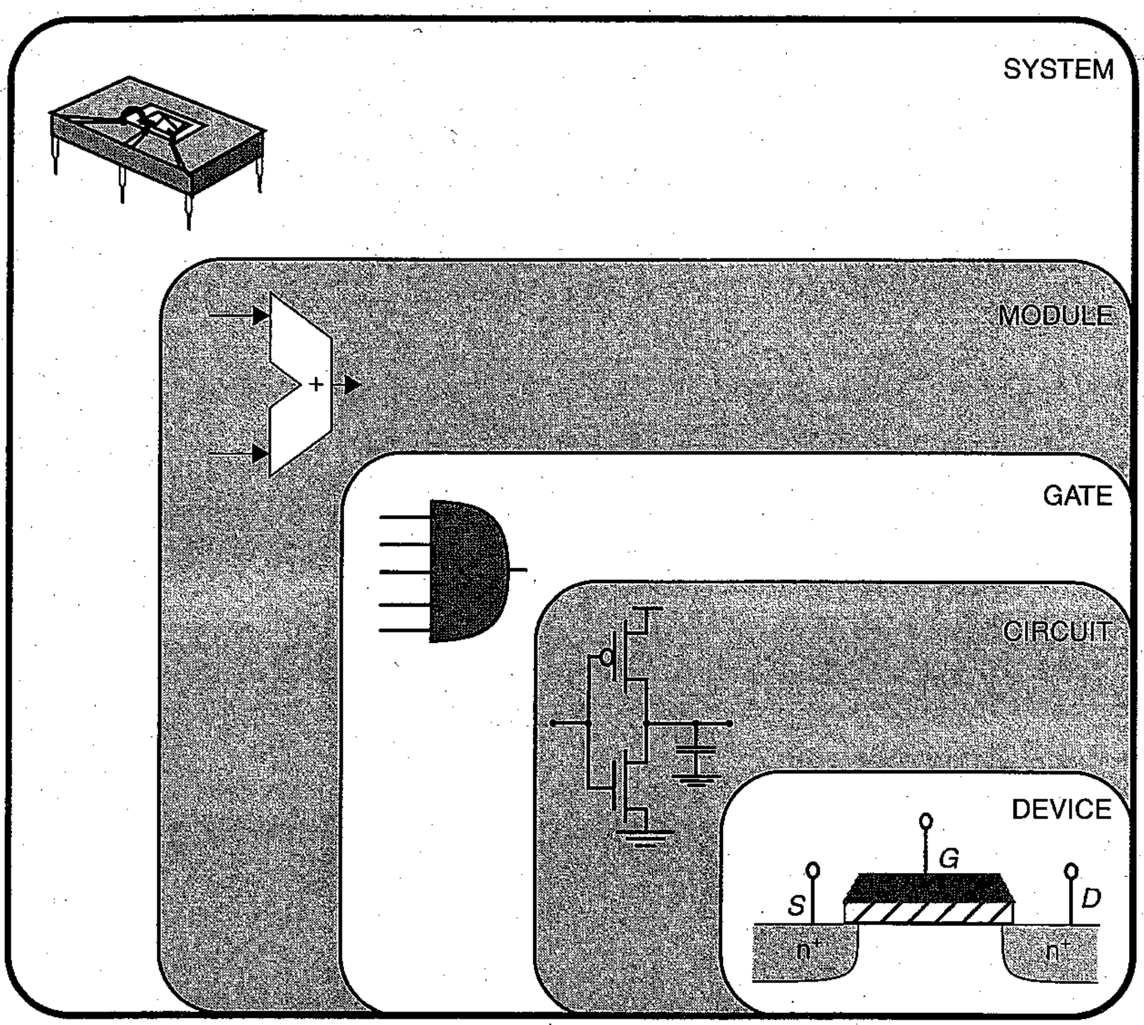


Figure 1.5 Design abstraction levels in digital circuits.

ies. These libraries contain not only the layouts, but also provide complete documentation and characterization of the behavior of the cells. The use of cell libraries is, for instance, apparent in the layout of the Pentium processor (Figure 1.4b). The integer and floating-point unit, just to name a few, contain large sections designed using the so-called *standard cell approach*. In this approach, logic gates are placed in rows of cells of equal height and interconnected using routing channels. The layout of such a block can be generated automatically given that a library of cells is available.

The preceding analysis demonstrates that design automation and modular design practices have effectively addressed some of the complexity issues incurred in contemporary digital design. This leads to the following pertinent question. If design automation solves all our design problems, why should we be concerned with digital circuit design at all? Will the next-generation digital designer ever have to worry about transistors or parasitics, or is the smallest design entity he will ever consider the gate and the module?

The truth is that the reality is more complex, and various reasons exist as to why an insight into digital circuits and their intricacies will still be an important asset for a long time to come.

- First of all, someone still has to *design and implement the module libraries*. Semiconductor technologies continue to advance from year to year, as demonstrated in

STE-101-107

Figure 1.2, where the minimum MOS device dimensions are plotted as a function of time. Until one has developed a fool-proof approach towards "porting" a cell from one technology to another, each change in technology—which happens approximately every two years—requires a redesign of the library.

- Creating an adequate *model* of a cell or module requires an in-depth understanding of its internal operation. For instance, to identify the dominant performance parameters of a given design, one has to recognize the critical timing path first.
- The library-based approach works fine when the design constraints (speed, cost or power) are not stringent. This is the case for a large number of *application-specific designs*, where the main goal is to provide a more integrated system solution, and performance requirements are easily within the capabilities of the technology. Unfortunately for a large number of other products such as microprocessors, success hinges on high performance, and designers therefore tend to push technology to its limits. At that point, the hierarchical approach tends to become somewhat less attractive. To resort to our previous analogy to software methodologies, a programmer tends to "customize" software routines when execution speed is crucial; compilers—or design tools—are not yet to the level of what human sweat or ingenuity can deliver.
- Even more important is the observation that the abstraction-based approach is only correct to a certain degree. The performance of, for instance, an adder can be substantially influenced by the way it is connected to its environment. The interconnection wires themselves contribute to delay as they introduce parasitic capacitances, resistances and even inductances. The impact of the *interconnect parasitics* is bound to increase in the years to come with the scaling of the technology.
- Scaling tends to emphasize some other deficiencies of the abstraction-based model. Some design entities tend to be *global or external* (to resort anew to the software analogy). Examples of global factors are the clock signals, used for synchronization in a digital design, and the supply lines. Increasing the size of a digital design has a profound effect on these global signals. For instance, connecting more cells to a supply line can cause a voltage drop over the wire, which, in its turn, can slow down all the connected cells. Issues such as clock distribution, circuit synchronization, and supply-voltage distribution are becoming more and more critical. Coping with them requires a profound understanding of the intricacies of digital circuit design.
- Another impact of technology evolution is that *new design issues* and constraints tend to emerge over time. A typical example of this is the periodical reemergence of power dissipation as a constraining factor, as was already illustrated in the historical overview. Another example is the changing ratio between device and interconnect parasitics. To cope with these unforeseen factors, one must at least be able to model and analyze their impact, requiring once again a profound insight into circuit topology and behavior.
- Finally, when things can go wrong, they do. A fabricated circuit does not always exhibit the exact waveforms one might expect from advance simulations. Deviations can be caused by variations in the fabrication process parameters, or by the induc-

tance of the package, or by a badly modeled clock signal. *Troubleshooting* a design requires circuit expertise.

For all the above reasons, it is my belief that an in-depth knowledge of digital circuit design techniques and approaches is an essential asset for a digital-system designer. Even though she might not have to deal with the details of the circuit on a daily basis, the understanding will help her to cope with unexpected circumstances and to determine the dominant effects when analyzing a design.

Example 1.1 Clocks Defy Hierarchy

To illustrate some of the issues raised above, let us examine the impact of deficiencies in one of the most important global signals in a design, the *clock*. The function of the clock signal in a digital design is to order the multitude of events happening in the circuit. This task can be compared to the function of a traffic light that determines which cars are allowed to move. It also makes sure that all operations are completed before the next one starts—a traffic light should be green long enough to allow a car or a pedestrian to cross the road. Under ideal circumstances, the clock signal is a periodic step waveform with abrupt transitions between the low and the high values (Figure 1.6a).

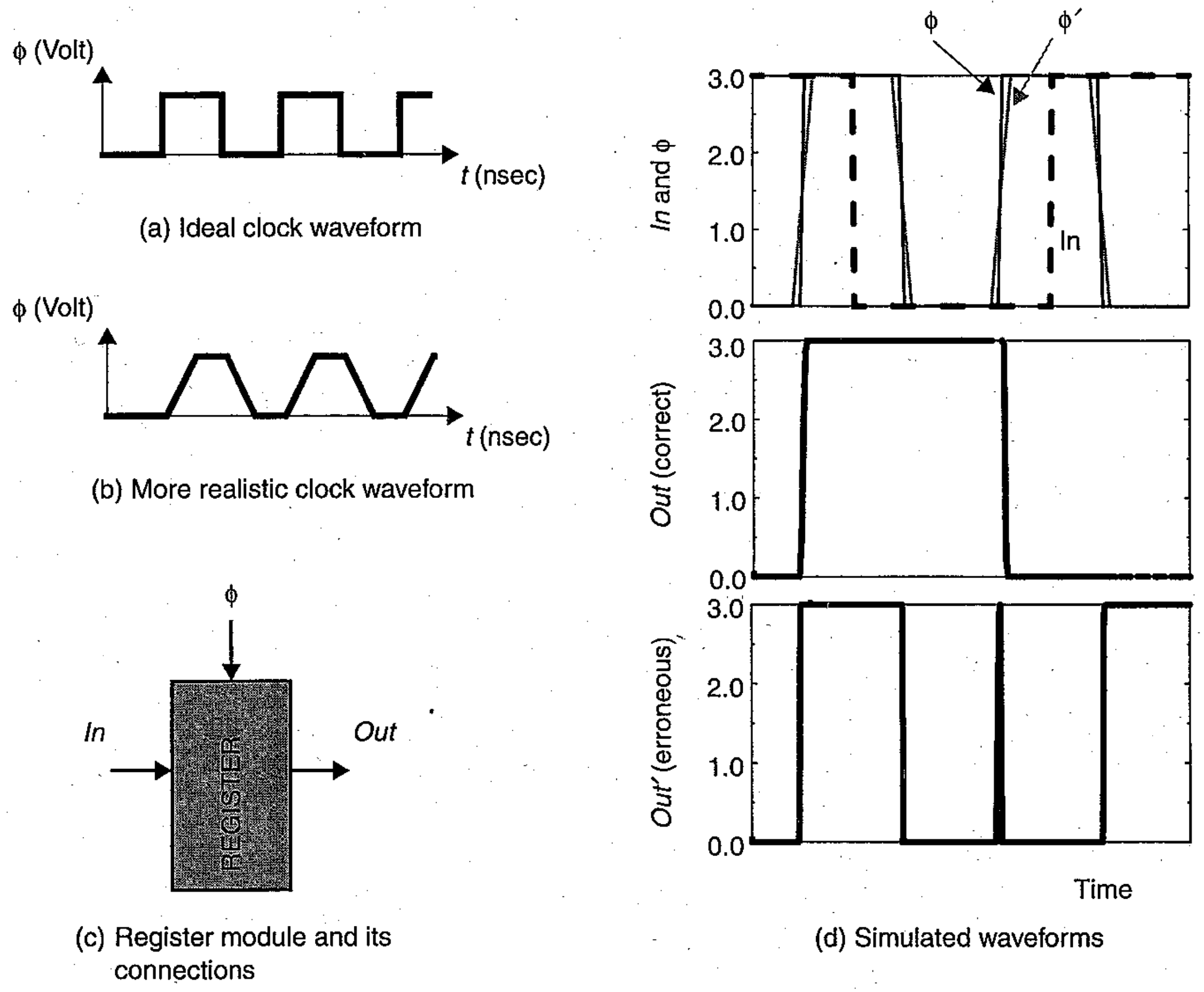


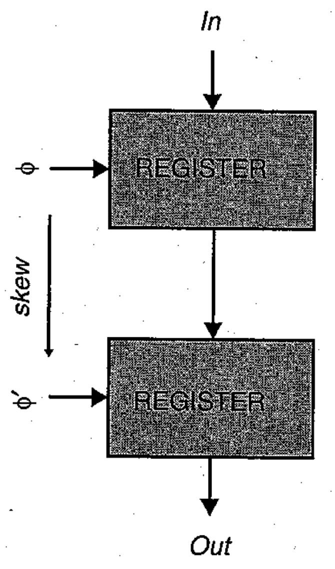
Figure 1.6 Reduced clock slopes can cause a register circuit to fail.

Consider, for instance, the circuit configuration of Figure 1.6c. The *register* module samples the value of the input signal at the rising edge of the clock signal ϕ. This sampled value is preserved and appears at the output until the clock rises anew and a new input is sampled. Under normal circuit operating conditions, this is exactly what happens, as demonstrated in the simulated response of Figure 1.6d. On the rising edge of clock ϕ, the input *In* is sampled and appears at the output *Out*.

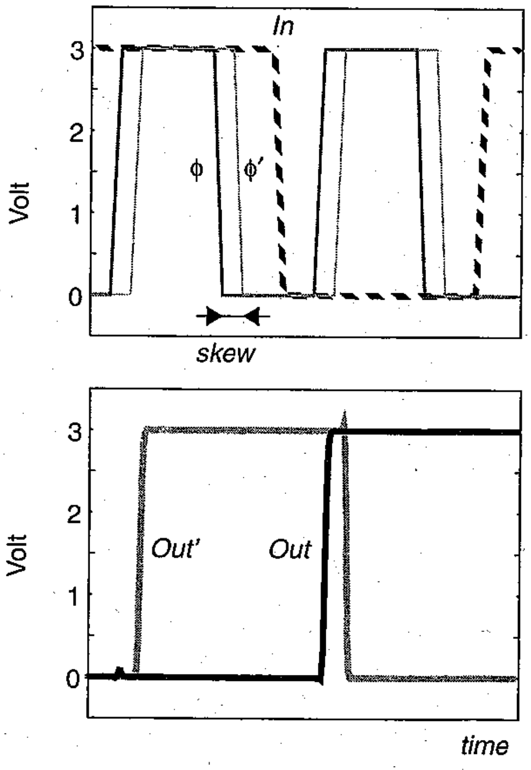
Assume now that, due to added loading on the clock signal (for instance, connecting more latches), the clock signal is degenerated, and the clock slopes become less steep (clock ϕ' in Figure 1.6d). When the degeneration is within bounds, the functionality of the latch is not impacted. When these bounds are exceeded the latch suddenly starts to malfunction as shown in Figure 1.6d (signal *Out'*). The output signal makes unexpected transitions at the falling clock edge, and extra spikes can be observed as well. Propagation of these erroneous values can cause the digital system to go into a unforeseen mode and crash. This example clearly shows how global effects, such as adding extra load to a clock, can change the behavior of an individual module. Observe that the effects shown are not universal, but are a property of the register circuit used.

Besides the requirement of steep edges, other constraints must be imposed on clock signals to ensure correct operation. A second requirement related to *clock alignment*, is illustrated in Figure 1.7. The circuit under analysis consists of two cascaded registers, both operating on the rising edge of the clock ϕ. Under normal operating conditions, the input *In* gets sampled into the first register on the rising edge of ϕ and appears at the output exactly one clock period later. This is confirmed by the simulations shown in Figure 1.7b (signal *Out*).

Due to delays associated with routing the clock wires, it may happen that the clocks become misaligned with respect to each other. As a result, the registers are interpreting time



(a) Two cascaded registers



(b) Simulated waveforms

Figure 1.7 Impact of clock misalignment.

indicated by the clock signal differently. Consider the case that the clock signal for the second register is delayed—or skewed—by a value δ. The rising edge of the delayed clock ϕ' will postpone the sampling of the input of the second register. If the time it takes to propagate the output of the first register to the input of the second is smaller than the clock delay, the latter will sample the wrong value. This causes the output to change prematurely, as clearly illustrated in the simulation, where the signal *Out'* goes high at the first rising edge of ϕ' instead of the second one.

Clock misalignment, or *clock skew*, as it is normally called, is another example of how global signals may influence the functioning of a hierarchically designed system. Clock skew is actually one of the most critical design problems facing the designers of large, high-performance systems.

The purpose of this textbook is to provide *a bridge between the abstract vision of digital design and the underlying digital circuit and its peculiarities.* While starting from a solid understanding of the operation of electronic devices and an in-depth analysis of the nucleus of digital design—the inverter—we will gradually channel this knowledge into the design of more complex entities, such as complex gates, datapaths, registers, controllers, and memories. The persistent quest for a designer when designing each of the mentioned modules is to identify the dominant design parameters, to locate the section of the design he should focus his optimizations on, and to determine the specific properties that make the module under investigation (e.g., a memory) different from any others.

The text also addresses other compelling (global) issues in modern digital circuit design such as *power dissipation, interconnect, timing, and synchronization.* While most of the text concentrates on CMOS design, it is worthwhile exploring the boundaries of what can be achieved with semiconductor digital circuit technology. A short discussion on bipolar, BiCMOS, GaAs, and cryogenic circuit approaches is therefore included.

1.3 To Probe Further

The design of digital integrated circuits has been the topic of a multitude of textbooks and monographs. To help the reader find more information on some selected topics, an extensive list of reference works is listed below. The state-of-the-art developments in the area of digital design are generally reported in technical journals or conference proceedings, the most important of which are listed.

JOURNALS AND PROCEEDINGS

IEEE Journal of Solid-State Circuits
IEICE Transactions on Electronics (Japan)
Proceedings of The International Solid-State and Circuits Conference (ISSCC)
Proceedings of the Integrated Circuits Symposium
European Solid-State Circuits Conference (ESSCIRC)

REFERENCE BOOKS

General

H. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI*, Addison-Wesley, 1990.

J. Buchanan, *CMOS/TTL Digital Systems Design*, McGraw-Hill, 1990.

J. Di Giacoma, *VLSI Handbook*, McGraw-Hill, 1989.

E. Friedman, ed., *Clock Distribution Networks in VLSI Circuits and Systems*, IEEE Press, 1995.

H. Haznedar, *Digital Micro-Electronics*, Benjamin/Cummings, 1991.

D. Hodges, *Semiconductor Memories*, IEEE Press, 1972.

D. Hodges and H. Jackson, *Analysis and Design of Digital Integrated Circuits*, 2nd ed., McGraw-Hill, 1988.

R. K. Watts, *Submicron Integrated Circuits*, Wiley, 1989.

Design Tools and Methodologies

W. Bhanzhaf, *Computer Aided Circuit Analysis Using SPICE*, Prentice Hall, 1992.

G. De Micheli, *Synthesis and Optimization of Digital Circuits*, McGraw-Hill, 1994.

E. Elmasry, ed., *Digital VLSI Systems*, IEEE Press, 1985.

S. Rubin, *Computer Aids for VLSI Design*, Addison-Wesley, 1987.

P. Tuinenga, *A Guide to Circuit Simulation & Analysis using PSpice*, Prentice Hall, 1988.

W. Wolf, *Modern VLSI Design: A Systems Approach*, Prentice Hall, 1994.

MOS

M. Annaratone, *Digital CMOS Circuit Design*, Kluwer, 1986.

T. Dillinger, *VLSI Engineering*, Prentice Hall, 1988.

E. Elmasry, ed., *Digital MOS Integrated Circuits*, IEEE Press, 1981.

E. Elmasry, ed., *Digital MOS Integrated Circuits II*, IEEE Press, 1992.

Glasser and Dopperpuhl, *The Design and Analysis of VLSI Circuits*, Addison-Wesley, 1985.

Mead and Conway, *Introduction to VLSI Systems*, Addison-Wesley, 1980.

D. Pucknell and K. Eshraghian, *Basic VLSI Design*, Prentice Hall, 1988.

M. Shoji, *CMOS Digital Circuit Technology*, Prentice Hall, 1988.

J. Uyemura, *Fundamentals of MOS Digital Integrated Circuits*, Addison-Wesley, 1988.

J. Uyemura, *Circuit Design for CMOS VLSI*, Kluwer, 1992.

H. Veendrick, *MOS IC's: From Basics to ASICS*, VCH, 1992.

Weste and Eshraghian, *Principles of CMOS VLSI Design*, Addison-Wesley, 1985, 1993.

Bipolar and BiCMOS

A. Alvarez, *BiCMOS Technology and Its Applications*, Kluwer, 1989.

M. Elmasry, *Digital Bipolar Integrated Circuits*, Wiley, 1983.

M. Elmasry, ed., *BiCMOS Integrated Circuit Design*, IEEE Press, 1994.

S. Embabi, A. Bellaouar, and M. Elmasry, *Digital BiCMOS Integrated Circuit Design*, Kluwer, 1993.

Lynn et al., eds., *Analysis and Design of Integrated Circuits*, McGraw-Hill, 1967.