CHAPTER

INTRODUCTION

The evolution of digital circuit design

Compelling issues in digital circuit design

How to measure the quality of digital design

Valuable references

1.1 A Historical Perspective
1.2 Issues in Digital Integrated Circuit Design
1.3 Quality Metrics of A Digital Design
1.4 Summary
1.5 To Probe Further
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).]
Section 1.1  A Historical Perspective

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]. 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 I2L—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 manufac-

---

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.
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 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. These technologies only play a very small role in the overall digital integrated circuit design scene. With the ever increasing performance of CMOS, this role is bound to be further reduced with time. Hence the focus of this textbook on CMOS only.

### 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 IC’s 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
Section 1.2 Issues in Digital Integrated Circuit Design

into the GHz range. This is illustrated in Figure 1.4, which plots the microprocessor trends in terms of complexity and performance at the beginning of the 21st century. An 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 1.5a, 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-to-market is one of the crucial factors in the ultimate success of a component.

Figure 1.2 Evolution of integration complexity of logic ICs and memories as a function of time.

Figure 1.3 Historical evolution of microprocessor transistor count and clock frequency (from [Sasaki91]).
INTRODUCTION  Chapter 1

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 1.5b. 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

---

**Figure 1.4** Microprocessor trends at the beginning of the 21st century. Observe how the fraction of the transistors is being devoted to memory is increasing over time ([Young99]).

(a) Trends in transistor count

(b) Trends in clock frequency
Section 1.2 Issues in Digital Integrated Circuit Design

Figure 1.5 Comparing the design methodologies of the Intel 4004 (1971) and Pentium-II™ (1997) microprocessors (reprinted with permission from Intel).

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

(b) The Pentium-II™ microprocessor (see also back cover)
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 1.6. 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 expres-
tion \( Z = A \cdot B \), 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 libraries. 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 1.5b). 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. 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 inductance 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 1.7a).

Consider, for instance, the circuit configuration of Figure 1.7c. 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 sam-
Section 1.2 Issues in Digital Integrated Circuit Design

pled. Under normal circuit operating conditions, this is exactly what happens, as demonstrated in the simulated response of Figure 1.7d. On the rising edge of clock $\phi$, the input $I_n$ is sampled and appears at the output $O_u$.

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 $\phi'$ 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 $O_u'$). 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 an 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 1.8. The circuit under analysis consists of two cascaded registers, both operating on the rising edge of the clock $\phi$. Under normal operating conditions, the input $I_n$ gets sampled into the first register on the rising edge of $\phi$ and appears at the output exactly one clock period later. This is confirmed by the simulations shown in Figure 1.7b (signal $O_u$).

![Diagram of register module and its connections](image1.png)

![Simulated waveforms](image2.png)

**Figure 1.7** Reduced clock slopes can cause a register circuit to fail.
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 indicated by the clock signal differently. Consider the case that the clock signal for the second register is delayed—or skewed—by a value $\delta$. The rising edge of the delayed clock $\phi'$ 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 $\phi'$ 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.

1.3 Quality Metrics of A Digital Design

This section defines a set of basic properties of a digital design. These properties help to quantify the quality of a design from different perspectives: cost, functionality, robustness, performance, and energy consumption. Which one of these metrics is most important depends upon the application. For instance, pure speed is a crucial property in a compute server. On the other hand, energy consumption is a dominant metric for hand-held mobile applications such as cell phones. The introduced properties are relevant at all levels of the design hierarchy, be it system, chip, module, and gate. To ensure consistency in the definitions throughout the design hierarchy stack, we propose a bottom-up approach: we start with defining the basic quality metrics of a simple inverter, and gradually expand these to the more complex functions such as gate, module, and chip.

1.3.1 Cost of an Integrated Circuit

The total cost of any product can be separated into two components: the recurring expenses or the variable cost, and the non-recurring expenses or the fixed cost.

Fixed Cost

The fixed cost is independent of the sales volume, the number of products sold. An important component of the fixed cost of an integrated circuit is the effort in time and manpower it takes to produce the design. This design cost is strongly influenced by the complexity of the design, the aggressiveness of the specifications, and the productivity of the designer. Advanced design methodologies that automate major parts of the design process can help to boost the latter. Bringing down the design cost in the presence of an ever-increasing IC complexity is one of the major challenges that is always facing the semiconductor industry. The Design

Additionally, one has to account for the indirect costs, the company overhead that cannot be billed directly to one product. It includes amongst others the company’s research and development (R&D), manufacturing equipment, marketing, sales, and building infrastructure.

Variable Cost

This accounts for the cost that is directly attributable to a manufactured product, and is hence proportional to the product volume. Variable costs include the costs of the parts used in the product, assembly costs, and testing costs. The total cost of an integrated circuit is now

\[
\text{cost per IC} = \text{variable cost per IC} + \left( \frac{\text{fixed cost}}{\text{volume}} \right) \tag{1.1}
\]
The impact of the fixed cost is more pronounced for small-volume products. This also
explains why it makes sense to have large design team working for a number of years on a
hugely successful product such as a microprocessor.

While the cost of producing a single transistor has dropped exponentially over the
past decades, the basic variable-cost equation has not changed:

\[
\text{variable cost} = \frac{\text{cost of die} + \text{cost of die test} + \text{cost of packaging}}{\text{final test yield}}
\]  

(1.2)

As will be elaborated on in Chapter 2, the IC manufacturing process groups a number of
identical circuits onto a single \textit{wafer} (Figure). Upon completion of the fabrication, the
wafer is chopped into \textit{dies}, which are then individually packaged after being \textit{tested}. We
will focus on the cost of the dies in this discussion. The cost of packaging and test is the

\[\text{cost of die} = \frac{\text{cost of wafer}}{\text{dies per wafer} \times \text{die yield}} \]  

(1.3)

Figure 1.9 Finished wafer. Each square represents a die.

The die cost depends upon the number of good die on a wafer, and the percentage of
those that are functional. The latter factor is called the \textit{die yield}.

The number of dies per wafer is, in essence, the area of the wafer divided by the die
area. The actual situation is somewhat more complicated as wafers are round, and chips are
square. Dies around the perimeter of the wafer are therefore lost. The size of the wafer has
been steadily increasing over the years, yielding more dies per fabrication run. Eq. (1.3)
also presents the first indication that the cost of a circuit is dependent upon the chip
area—increasing the chip area simply means that less dies fit on a wafer.

The actual relation between cost and area is more complex, and depends upon the
die yield. Both the substrate material and the manufacturing process introduce faults that
can cause a chip to fail. Assuming that the defects are randomly distributed over the wafer,
and that the yield is inversely proportional to the complexity of the fabrication process, we
obtain the following expression of the die yield:
Section 1.3 Quality Metrics of A Digital Design

\[ \text{die yield} = \left( 1 + \frac{\text{defects per unit area} \times \text{die area}}{\alpha} \right)^{-\alpha} \]  

(1.4)

\( \alpha \) is a parameter that depends upon the complexity of the manufacturing process, and is roughly proportional to the number of masks. \( \alpha = 3 \) is a good estimate for today’s complex CMOS processes. The defects per unit area is a measure of the material and process induced faults. A value between 0.5 and 1 defects/cm\(^2\) is typical these days, but depends strongly upon the maturity of the process.

**Example 1.2 Die Yield**

Assume a wafer size of 12 inch, a die size of 2.5 cm\(^2\), 1 defects/cm\(^2\), and \( \alpha = 3 \). Determine the die yield of this CMOS process run.

The number of dies per wafer can be estimated with the following expression, which takes into account the lost dies around the perimeter of the wafer.

\[ \text{dies per wafer} = \frac{\pi \times (\text{wafer diameter}/2)^2}{\text{die area}} - \frac{\pi \times \text{wafer diameter}}{\sqrt{2} \times \text{die area}} \]

This means 252 (= 296 - 44) potentially operational dies for this particular example. The die yield can be computed with the aid of Eq. (1.4), and equals 16%! This means that on the average only 40 of the dies will be fully functional.

The bottom line is that the number of functional of dies per wafer, and hence the cost per die is a strong function of the die area. While the yield tends to be excellent for the smaller designs, it drops rapidly once a certain threshold is exceeded. Bearing in mind the equations derived above and the typical parameter values, we can conclude that die costs are proportional to the fourth power of the area:

\[ \text{cost of die} = f(\text{die area})^4 \]  

(1.5)

The area is a function that is directly controllable by the designer(s), and is the prime metric for cost. Small area is hence a desirable property for a digital gate. The smaller the gate, the higher the integration density and the smaller the die size. Smaller gates furthermore tend to be faster and consume less energy, as the total gate capacitance—which is one of the dominant performance parameters—often scales with the area.

The number of transistors in a gate is indicative for the expected implementation area. Other parameters may have an impact, though. For instance, a complex interconnect pattern between the transistors can cause the wiring area to dominate. The gate complexity, as expressed by the number of transistors and the regularity of the interconnect structure, also has an impact on the design cost. Complex structures are harder to implement and tend to take more of the designers valuable time. Simplicity and regularity is a precious property in cost-sensitive designs.

### 1.3.2 Functionality and Robustness

A prime requirement for a digital circuit is, obviously, that it performs the function it is designed for. The measured behavior of a manufactured circuit normally deviates from the
expected response. One reason for this aberration are the variations in the manufacturing process. The dimensions, threshold voltages, and currents of an MOS transistor vary between runs or even on a single wafer or die. The electrical behavior of a circuit can be profoundly affected by those variations. The presence of disturbing noise sources on or off the chip is another source of deviations in circuit response. The word *noise* in the context of digital circuits means “unwanted variations of voltages and currents at the logic nodes.” Noise signals can enter a circuit in many ways. Some examples of digital noise sources are depicted in Figure 1.10. For instance, two wires placed side by side in an integrated circuit form a coupling capacitor and a mutual inductance. Hence, a voltage or current change on one of the wires can influence the signals on the neighboring wire. Noise on the power and ground rails of a gate also influences the signal levels in the gate.

Most noise in a digital system is internally generated, and the noise value is proportional to the signal swing. Capacitive and inductive cross talk, and the internally-generated power supply noise are examples of such. Other noise sources such as input power supply noise are external to the system, and their value is not related to the signal levels. For these sources, the noise level is directly expressed in Volt or Ampere. Noise sources that are a function of the signal level are better expressed as a fraction or percentage of the signal level. Noise is a major concern in the engineering of digital circuits. How to cope with all these disturbances is one of the main challenges in the design of high-performance digital circuits and is a recurring topic in this book.

The steady-state parameters (also called the *static behavior*) of a gate measure how robust the circuit is with respect to both variations in the manufacturing process and noise disturbances. The definition and derivation of these parameters requires a prior understanding of how digital signals are represented in the world of electronic circuits.

Digital circuits (DC) perform operations on *logical* (or *Boolean*) variables. A logical variable $x$ can only assume two discrete values:

$$x \in \{0,1\}$$

As an example, the inversion (i.e., the function that an inverter performs) implements the following compositional relationship between two Boolean variables $x$ and $y$:

$$y = \overline{x}; \ {x = 0 \Rightarrow y = 1; \ x = 1 \Rightarrow y = 0} \quad (1.6)$$

![Noise sources in digital circuits.](a) Inductive coupling (b) Capacitive coupling (c) Power and ground noise

**Figure 1.10** Noise sources in digital circuits.
A logical variable is, however, a mathematical abstraction. In a physical implementation, such a variable is represented by an electrical quantity. This is most often a node voltage that is not discrete but can adopt a continuous range of values. This electrical voltage is turned into a discrete variable by associating a nominal voltage level with each logic state: \( 1 \Leftrightarrow V_{OH} \) and \( 0 \Leftrightarrow V_{OL} \), where \( V_{OH} \) and \( V_{OL} \) represent the high and the low logic levels, respectively. Applying \( V_{OH} \) to the input of an inverter yields \( V_{OL} \) at the output and vice versa. The difference between the two is called the logic or signal swing \( V_{sw} \).

\[
\begin{align*}
V_{OH} &= (V_{OL}) \\
V_{OL} &= (V_{OH})
\end{align*}
\]  

(1.7)

**The Voltage-Transfer Characteristic**

Assume now that a logical variable \( in \) serves as the input to an inverting gate that produces the variable \( out \). The electrical function of a gate is best expressed by its voltage-transfer characteristic (VTC) (sometimes called the DC transfer characteristic), which plots the output voltage as a function of the input voltage \( V_{out} = f(V_{in}) \). An example of an inverter VTC is shown in Figure 1.11. The high and low nominal voltages, \( V_{OH} \) and \( V_{OL} \), can readily be identified—\( V_{OH} = f(V_{OL}) \) and \( V_{OL} = f(V_{OH}) \). Another point of interest of the VTC is the gate or switching threshold voltage \( V_M \) (not to be confused with the threshold voltage of a transistor), that is defined as \( V_M = f(V_M) \). \( V_M \) can also be found graphically at the intersection of the VTC curve and the line given by \( V_{out} = V_{in} \). The gate threshold voltage presents the midpoint of the switching characteristics, which is obtained when the output of a gate is short-circuited to the input. This point will prove to be of particular interest when studying circuits with feedback (also called sequential circuits).

![Figure 1.11 Inverter voltage-transfer characteristic.](image)

Even if an ideal nominal value is applied at the input of a gate, the output signal often deviates from the expected nominal value. These deviations can be caused by noise or by the loading on the output of the gate (i.e., by the number of gates connected to the output signal). Figure 1.12a illustrates how a logic level is represented in reality by a range of acceptable voltages, separated by a region of uncertainty, rather than by nominal levels.
alone. The regions of acceptable high and low voltages are delimited by the $V_{IH}$ and $V_{IL}$ voltage levels, respectively. These represent by definition the points where the gain ($= dV_{out} / dV_{in}$) of the VTC equals $-1$ as shown in Figure 1.12b. The region between $V_{IH}$ and $V_{IL}$ is called the undefined region (sometimes also referred to as transition width, or TW). Steady-state signals should avoid this region if proper circuit operation is to be ensured.

**Noise Margins**

For a gate to be robust and insensitive to noise disturbances, it is essential that the “0” and “1” intervals be as large as possible. A measure of the sensitivity of a gate to noise is given by the noise margins $NM_L$ (noise margin low) and $NM_H$ (noise margin high), which quantify the size of the legal “0” and “1”, respectively, and set a fixed maximum threshold on the noise value:

$$NM_L = V_{IL} - V_{OL}$$
$$NM_H = V_{OH} - V_{IH}$$

The noise margins represent the levels of noise that can be sustained when gates are cascaded as illustrated in Figure 1.13. It is obvious that the margins should be larger than 0 for a digital circuit to be functional and by preference should be as large as possible.

**Regenerative Property**

A large noise margin is a desirable, but not sufficient requirement. Assume that a signal is disturbed by noise and differs from the nominal voltage levels. As long as the signal is within the noise margins, the following gate continues to function correctly, although its output voltage varies from the nominal one. This deviation is added to the noise injected at the output node and passed to the next gate. The effect of different noise sources may accumulate and eventually force a signal level into the undefined region. This, fortunately, does not happen if the gate possesses the regenerative property, which ensures that a dis-

---

**Figure 1.12** Mapping logic levels to the voltage domain.

![Diagram](attachment:image.png)
Section 1.3 Quality Metrics of A Digital Design

A disturbed signal gradually converges back to one of the nominal voltage levels after passing through a number of logical stages. This property can be understood as follows:

An input voltage $v_{in}(v_{in} \in \text{"0"})$ is applied to a chain of $N$ inverters (Figure 1.14a). Assuming that the number of inverters in the chain is even, the output voltage $v_{out}(N \to \infty)$ will equal $V_{OL}$ if and only if the inverter possesses the regenerative property. Similarly, when an input voltage $v_{in}(v_{in} \in \text{"1"})$ is applied to the inverter chain, the output voltage will approach the nominal value $V_{OH}$.

Example 1.3 Regenerative property

The concept of regeneration is illustrated in Figure 1.14b, which plots the simulated transient response of a chain of CMOS inverters. The input signal to the chain is a step-waveform with a degraded amplitude, which could be caused by noise. Instead of swinging from rail to rail,
$v_0$, only extends between 2.1 and 2.9 V. From the simulation, it can be observed that this deviation rapidly disappears, while progressing through the chain; $v_1$, for instance, extends from 0.6 V to 4.45 V. Even further, $v_2$ already swings between the nominal $V_{ol}$ and $V_{oh}$. The inverter used in this example clearly possesses the regenerative property.

The conditions under which a gate is regenerative can be intuitively derived by analyzing a simple case study. Figure 1.15(a) plots the VTC of an inverter $V_{out} = f(V_{in})$ as well as its inverse function $finv()$, which inverts the function of the x- and y-axis and is defined as follows:

\[ in = f(out) \Rightarrow in = finv(out) \] (1.9)

Assume that a voltage $v_0$, deviating from the nominal voltages, is applied to the first inverter in the chain. The output voltage of this inverter equals $v_1 = f(v_0)$ and is applied to the next inverter. Graphically this corresponds to $v_1 = finv(v_2)$. The signal voltage gradually converges to the nominal signal after a number of inverter stages, as indicated by the arrows. In Figure 1.15(b) the signal does not converge to any of the nominal voltage levels but to an intermediate voltage level. Hence, the characteristic is nonregenerative. The difference between the two cases is due to the gain characteristics of the gates. To be regenerative, the VTC should have a transient region (or undefined region) with a gain greater than 1 in absolute value, bordered by the two legal zones, where the gain should be smaller than 1. Such a gate has two stable operating points. This clarifies the definition of the $V_{IH}$ and the $V_{IL}$ levels that form the boundaries between the legal and the transient zones.

**Noise Immunity**

While the noise margin is a meaningful means for measuring the robustness of a circuit against noise, it is not sufficient. It expresses the capability of a circuit to “overpower” a noise source. Noise immunity, on the other hand, expresses the ability of the system to pro-
Section 1.3 Quality Metrics of A Digital Design

cess and transmit information correctly in the presence of noise [Dally98]. Many digital circuits with low noise margins have very good noise immunity because they reject a noise source rather than overpower it.

To study the noise immunity of a gate, we have to construct a noise budget that allocates the power budget to the various power sources. As discussed earlier, the noise sources can be divided into sources that are proportional to the signal swing \((V_{Np} = g V_{sw})\), and others that are fixed \((V_{Ni})\). We assume, for the sake of simplicity, that the noise margin equals half the signal swing (for both H and L). To operate correctly, the noise margin has to be larger than the sum of the noise values.

\[
V_{NM} = \frac{V_{sw}}{2} \geq \sum_{i} V_{Ni} + \sum_{j} g_j V_{sw}
\]  

(1.10)

Given a set of noise sources, we can derive the minimum signal swing necessary for the system to be operational,

\[
V_{sw} \geq \frac{2 \sum V_{Ni}}{1 - \sum g_j}
\]

(1.11)

This makes it clear that the signal swing (and the noise margin) has to be large enough to overpower the fixed sources. On the other hand, the impact of the internal sources is strongly dependent upon the noise suppressing capabilities of the gates, i.e. the proportionality or gain factors \(g_j\), which should be as small as possible. In later chapters, we will discuss some differential logic families that suppress most of the internal noise, and hence can get away with very small noise margins and signal swings.

Directivity

The directivity property requires a gate to be unidirectional, that is, changes in an output level should not appear at any unchanging input of the same circuit. If not, an output-signal transition reflects to the gate inputs as a noise signal, affecting the signal integrity.

In real gate implementations, full directivity can never be achieved. Some feedback of changes in output levels to the inputs cannot be avoided. Capacitive coupling between inputs and outputs is a typical example of such a feedback. It is important to minimize these changes so that they do not affect the logic levels of the input signals.

Fan-In and Fan-Out

The fan-out denotes the number of load gates \(N\) that are connected to the output of the driving gate (Figure 1.16). Increasing the fan-out of a gate can affect its logic output levels. From the world of analog amplifiers, we know that this effect is minimized by making the input resistance of the load gates as large as possible (minimizing the input currents) and by keeping the output resistance of the driving gate small (reducing the effects of load currents on the output voltage). When the fan-out is large, the added load can deteriorate the dynamic performance of the driving gate. For these reasons, many generic and library
components define a maximum fan-out to guarantee that the static and dynamic performance of the element meet specification.

The fan-in of a gate is defined as the number of inputs to the gate (Figure 1.16b). Gates with large fan-in tend to be more complex, which often results in inferior static and dynamic properties.

The Ideal Digital Gate

Based on the above observations, we can define the ideal digital gate from a static perspective. The ideal inverter model is important because it gives us a metric by which we can judge the quality of actual implementations.

Its VTC is shown in Figure 1.17 and has the following properties: infinite gain in the transition region, and gate threshold located in the middle of the logic swing, with high and low noise margins equal to half the swing. The input and output impedances of the ideal gate are infinity and zero, respectively (i.e., the gate has unlimited fan-out). While this ideal VTC is unfortunately impossible in real designs, some implementations, such as the static CMOS inverter, come close.
Section 1.3 Quality Metrics of A Digital Design

Example 1.4 Voltage-Transfer Characteristic

Figure 1.18 shows an example of a voltage-transfer characteristic of an actual, but outdated gate structure (as produced by SPICE in the DC analysis mode). The values of the dc-parameters are derived from inspection of the graph.

- \( V_{OH} = 3.5 \text{ V} \)
- \( V_{OL} = 0.45 \text{ V} \)
- \( V_{IH} = 2.35 \text{ V} \)
- \( V_{IL} = 0.66 \text{ V} \)
- \( V_M = 1.64 \text{ V} \)
- \( NM_H = 1.15 \text{ V} \)
- \( NM_L = 0.21 \text{ V} \)

The observed transfer characteristic, obviously, is far from ideal: it is asymmetrical, has a very low value for \( NM_L \), and the voltage swing of 3.05V is substantially below the maximum obtainable value of 5 V (which is the value of the supply voltage for this design).

1.3.3 Performance

From a system designer's perspective, the performance of a digital circuit expresses the computational load that the circuit can manage. For instance, a microprocessor is often characterized by the number of instructions it can execute per second. This performance metric depends both on the architecture of the processor—for instance, the number of instructions it can execute in parallel—and the actual design of logic circuitry. While the former is crucially important, it is not the focus of this textbook. We refer the reader to the many excellent books on this topic [for instance, Patterson96]. When focusing on the pure design, performance is most often expressed by the duration of the clock period (clock cycle time), or its rate (clock frequency). The minimum value of the clock period for a given technology and design is set by a number of factors such as the time it takes for the signals to propagate through the logic, the time it takes to get the data in and out of the
registers, and the uncertainty of the clock arrival times. Each of these topics will be discussed in detail on the course of this text book. At the core of the whole performance analysis, however, lays the performance of an individual gate.

The propagation delay $t_p$ of a gate defines how quickly it responds to a change at its input(s). It expresses the delay experienced by a signal when passing through a gate. It is measured between the 50% transition points of the input and output waveforms, as shown in Figure 1.19 for an inverting gate.\footnote{The 50% definition is inspired the assumption that the switching threshold $V_{th}$ is typically located in the middle of the logic swing.} Because a gate displays different response times for rising or falling input waveforms, two definitions of the propagation delay are necessary. The $t_{p_{LH}}$ defines the response time of the gate for a low to high (or positive) output transition, while $t_{p_{HL}}$ refers to a high to low (or negative) transition. The propagation delay $t_p$ is defined as the average of the two.

\[
 t_p = \frac{t_{p_{LH}} + t_{p_{HL}}}{2} \tag{1.12}
\]

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure1.19}
\caption{Definition of propagation delays and rise and fall times.}
\end{figure}

CAUTION: Observe that the propagation delay $t_p$, in contrast to $t_{p_{LH}}$ and $t_{p_{HL}}$, is an artificial gate quality metric, and has no physical meaning per se. It is mostly used to compare different semiconductor technologies, or logic design styles.

The propagation delay is not only a function of the circuit technology and topology, but depends upon other factors as well. Most importantly, the delay is a function of the slopes of the input and output signals of the gate. To quantify these properties, we introduce the rise and fall times $t_r$ and $t_f$, which are metrics that apply to individual signal waveforms rather than gates (Figure 1.19), and express how fast a signal transits between the different levels. The uncertainty over when a transition actually starts or ends is avoided by defining the rise and fall times between the 10% and 90% points of the wave-
forms, as shown in the Figure. The rise/fall time of a signal is largely determined by the strength of the driving gate, and the load presented by the node itself, which sums the contributions of the connecting gates (fan-out) and the wiring parasitics.

When comparing the performance of gates implemented in different technologies or circuit styles, it is important not to confuse the picture by including parameters such as load factors, fan-in and fan-out. A uniform way of measuring the $t_p$ of a gate, so that technologies can be judged on an equal footing, is desirable. The de-facto standard circuit for delay measurement is the ring oscillator, which consists of an odd number of inverters connected in a circular chain (Figure 1.20). Due to the odd number of inversions, this circuit does not have a stable operating point and oscillates. The period $T$ of the oscillation is determined by the propagation time of a signal transition through the complete chain, or $T = 2 \times t_p \times N$ with $N$ the number of inverters in the chain. The factor 2 results from the observation that a full cycle requires both a low-to-high and a high-to-low transition. Note that this equation is only valid for $2Nt_p \gg t_f + t_r$. If this condition is not met, the circuit might not oscillate—one “wave” of signals propagating through the ring will overlap with a successor and eventually dampen the oscillation. Typically, a ring oscillator needs at least five stages to be operational.

**Figure 1.20** Ring oscillator circuit for propagation-delay measurement.

**CAUTION:** We must be extremely careful with results obtained from ring oscillator measurements. A $t_p$ of 20 psec by no means implies that a circuit built with those gates will operate at 50 GHz. The oscillator results are primarily useful for quantifying the differences between various manufacturing technologies and gate topologies. The oscillator is an idealized circuit where each gate has a fan-in and fan-out of exactly one and parasitic loads are minimal. In more realistic digital circuits, fan-ins and fan-outs are higher, and interconnect delays are non-negligible. The gate functionality is also substantially more complex than a simple invert operation. As a result, the achievable clock frequency on average is 50 to a 100 times slower than the frequency predicted from ring oscillator mea-
measurements. This is an average observation; carefully optimized designs might approach the ideal frequency more closely.

**Example 1.5  Propagation Delay of First-Order RC Network**

Digital circuits are often modeled as first-order RC networks of the type shown in Figure 1.21. The propagation delay of such a network is thus of considerable interest.

![First-order RC network](image)

**Figure 1.21** First-order RC network.

When applying a step input (with $v_{in}$ going from 0 to $V$), the transient response of this circuit is known to be an exponential function, and is given by the following expression (where $\tau = RC$, the time constant of the network):

$$v_{out}(t) = (1 - e^{-t/\tau}) V$$  \hspace{1cm} (1.13)

The time to reach the 50% point is easily computed as $t = \ln(2) = 0.69 \tau$. Similarly, it takes $t = \ln(9) = 2.2 \tau$ to get to the 90% point. It is worth memorizing these numbers, as they are extensively used in the rest of the text.

### 1.3.4 Power and Energy Consumption

The power consumption of a design determines how much energy is consumed per operation, and much heat the circuit dissipates. These factors influence a great number of critical design decisions, such as the power-supply capacity, the battery lifetime, supply-line sizing, packaging and cooling requirements. Therefore, power dissipation is an important property of a design that affects feasibility, cost, and reliability. In the world of high-performance computing, power consumption limits, dictated by the chip package and the heat removal system, determine the number of circuits that can be integrated onto a single chip, and how fast they are allowed to switch. With the increasing popularity of mobile and distributed computation, energy limitations put a firm restriction on the number of computations that can be performed given a minimum time between battery recharges.

Depending upon the design problem at hand, different dissipation measures have to be considered. For instance, the peak power $P_{\text{peak}}$ is important when studying supply-line sizing. When addressing cooling or battery requirements, one is predominantly interested in the average power dissipation $P_{\text{av}}$. Both measures are defined in equation Eq. (1.14):

$$P_{\text{peak}} = i_{\text{peak}} V_{\text{supply}} = \max[p(t)]$$

$$P_{\text{av}} = \frac{1}{T} \int_{0}^{T} p(t) dt = \frac{V_{\text{supply}}}{T} \int_{0}^{T} i_{\text{supply}}(t) dt$$  \hspace{1cm} (1.14)
Section 1.3 Quality Metrics of A Digital Design

where \( p(t) \) is the instantaneous power, \( i_{\text{supply}} \) is the current being drawn from the supply voltage \( V_{\text{supply}} \) over the interval \( t \in [0, T] \), and \( i_{\text{peak}} \) is the maximum value of \( i_{\text{supply}} \) over that interval.

The dissipation can further be decomposed into static and dynamic components. The latter occurs only during transients, when the gate is switching. It is attributed to the charging of capacitors and temporary current paths between the supply rails, and is, therefore, proportional to the switching frequency: the higher the number of switching events, the higher the dynamic power consumption. The static component on the other hand is present even when no switching occurs and is caused by static conductive paths between the supply rails or by leakage currents. It is always present, even when the circuit is in stand-by. Minimization of this consumption source is a worthwhile goal.

The propagation delay and the power consumption of a gate are related—the propagation delay is mostly determined by the speed at which a given amount of energy can be stored on the gate capacitors. The faster the energy transfer (or the higher the power consumption), the faster the gate. For a given technology and gate topology, the product of power consumption and propagation delay is generally a constant. This product is called the power-delay product (or PDP) and can be considered as a quality measure for a switching device. The PDP is simply the energy consumed by the gate per switching event. The ring oscillator is again the circuit of choice for measuring the PDP of a logic family.

An ideal gate is one that is fast, and consumes little energy. The energy-delay product (E-D) is a combined metric that brings those two elements together, and is often used as the ultimate quality metric. From the above, it should be clear that the E-D is equivalent to power-delay\(^2\).

**Example 1.6 Energy Dissipation of First-Order RC Network**

Let us consider again the first-order RC network shown in Figure 1.21. When applying a step input (with \( V_{\text{in}} \) going from 0 to \( V \)), an amount of energy is provided by the signal source to the network. The total energy delivered by the source (from the start of the transition to the end) can be readily computed:

\[
E_{\text{in}} = \int_{0}^{\infty} i_{\text{in}}(t) v_{\text{in}}(t) \, dt = V \int_{0}^{\infty} C \frac{dv_{\text{out}}}{dt} \, dt = (CV) \int_{0}^{V} dv_{\text{out}} = CV^2
\]  

(1.15)

It is interesting to observe that the energy needed to charge a capacitor from 0 to \( V \) volt with a step input is a function of the size of the voltage step and the capacitance, but is independent of the value of the resistor. We can also compute how much of the delivered energy gets stored on the capacitor at the end of the transition.

\[
E_{C} = \int_{0}^{\infty} i_{C}(t)v_{\text{out}}(t) \, dt = \int_{0}^{\infty} C \frac{dv_{\text{out}}}{dt} v_{\text{out}} \, dt = C \int_{0}^{V} v_{\text{out}} \, dv_{\text{out}} = CV^2
\]  

(1.16)

This is exactly half of the energy delivered by the source. For those who wonder happened with the other half—a simple analysis shows that an equivalent amount gets dissipated as heat in the resistor during the transaction. We leave it to the reader to demonstrate that dur-
ing the discharge phase (for a step from $V$ to 0), the energy originally stored on the capacitor gets dissipated in the resistor as well, and turned into heat.

### 1.4 Summary

In this introductory chapter, we learned about the history and the trends in digital circuit design. We also introduced the important quality metrics, used to evaluate the quality of a design: cost, functionality, robustness, performance, and energy/power dissipation. At the end of the Chapter, you can find an extensive list of reference works that may help you to learn more about some of the topics introduced in the course of the text.

### 1.5 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

- **MOS**
Section 1.5 To Probe Further

H. Veendrick, MOS IC’s: From Basics to ASICs, VCH, 1992.

High-Performance Design
M. Shoji, High-Speed Digital Circuits, Addison-Wesley, 1996.

Low-Power Design

Memory Design

Interconnections and Packaging

Design Tools and Methodologies
D. Klein, CMOS IC Layout, Newnes, 2000.

Bipolar and BiCMOS
INTRODUCTION

Chapter 1

General

REFERENCES

1.6 Exercises

1. [E, None, 1.2] Based on the evolutionary trends described in the chapter, predict the integration complexity and the clock speed of a microprocessor in the year 2010. Determine also how much DRAM should be available on a single chip at that point in time, if Moore’s law would still hold.

2. [D, None, 1.2] By scanning the literature, find the leading-edge devices at this point in time in the following domains: microprocessor, SRAM, and DRAM. Determine for each of those, the number of integrated devices, the overall area and the maximum clock speed. Evaluate the match with the trends predicted in section 1.2.

3. [D, None, 1.2] Find in the library the latest November issue of the *Journal of Solid State Circuits*. For each of the papers, determine its application class (such as microprocessor, signal processor, DRAM, SRAM, Gate Array), the type of manufacturing technology used (MOS, bipolar, etc.), the minimum feature size, the number of devices on a single die, and the maximum clock speed. Tabulate the results along the various application classes.

4. [E, None, 1.2] Provide at least three examples for each of the abstraction levels described in Figure 1.6.