

Introduction

1

1.1 A Brief History

In 1958, Jack Kilby built the first integrated circuit flip-flop with two transistors at Texas Instruments. In 2003, the Intel Pentium 4 microprocessor contained 55 million transistors and a 512-Mbit dynamic random access memory (DRAM) contained more than half a billion transistors. This corresponds to a compound annual growth rate of 53% over 45 years. No other technology in history has sustained such a high growth rate for so long.

This incredible growth has come from steady miniaturization of transistors and improvements in manufacturing processes. Most other fields of engineering involve tradeoffs between performance, power, and price. However, as transistors become smaller, they also become faster, dissipate less power, and are cheaper to manufacture. This synergy has revolutionized not only electronics, but also society at large.

The processing performance once exclusive to Cray supercomputers is now available in hand-held personal digital assistants. Processing capability that was once necessary for secret military spread-spectrum communications is now available in disposable cellular telephones. Improvements in integrated circuits have enabled space exploration, made automobiles more efficient, revolutionized the nature of warfare, brought vast libraries of information to our Web browsers, and made the world a more interdependent place.

Figure 1.1 shows annual sales in the worldwide semiconductor market. Integrated circuits became a \$100B/year business in 1994. The blip in 2000 is associated with the surge in sales for Y2K upgrades followed by the worldwide recession. In 2003, the industry manufactured more than one quintillion (10^{18}) transistors, or more than 100 million for every human being on the planet. Thousands of engineers have made their fortunes in the field. New fortunes lie ahead for those with innovative ideas and the talent to bring their ideas to reality.

During the first half of the 20th century, electronic circuits used large, expensive, power-hungry, and unreliable vacuum tubes. In 1947, John Bardeen and Walter Brattain built the first functioning point contact transistor at Bell Laboratories, shown in Figure 1.2(a) [Riordan97]. It was nearly classified as a military secret, but Bell Labs publicly announced the device in the following year.

We have called it the Transistor, T-R-A-N-S-I-S-T-O-R, because it is a resistor or semiconductor device which can amplify electrical signals as they are transferred

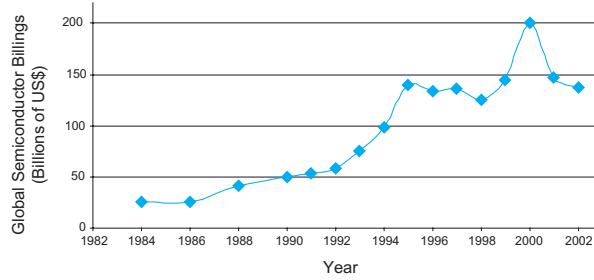


FIG 1.1 Size of worldwide semiconductor market

Source: Semiconductor Industry Association.

through it from input to output terminals. It is, if you will, the electrical equivalent of a vacuum tube amplifier. But there the similarity ceases. It has no vacuum, no filament, no glass tube. It is composed entirely of cold, solid substances.

Ten years later, Jack Kilby at Texas Instruments realized the potential for miniaturization if multiple transistors could be built on a single piece of silicon. Figure 1.2(b) shows his first prototype of an integrated circuit, constructed from a germanium slice and gold wires.

The invention of the transistor earned the Nobel Prize in Physics in 1956 for Bardeen, Brattain, and their co-worker William Shockley. Kilby received the Nobel Prize in Physics in 2000 for the invention of the integrated circuit.

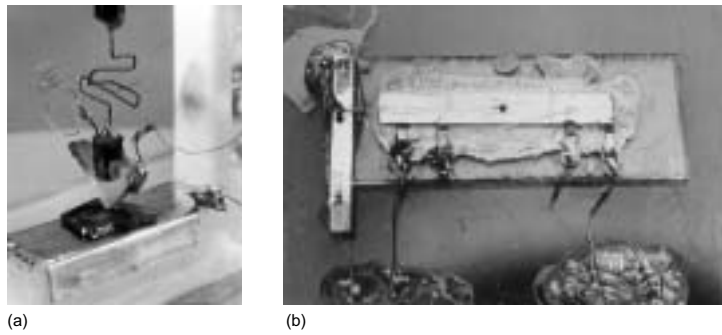


FIG 1.2 First transistor (a) and first integrated circuit (b)

Soon after inventing the point contact transistor, Bell Labs developed the bipolar junction transistor. Bipolar transistors were more reliable, less noisy, and more power-efficient. Early integrated circuits primarily used bipolar transistors. Transistors can be viewed as electrically controlled switches with a control terminal and two other terminals that are connected or disconnected depending on the voltage applied to the control. Bipolar transistors require a small current into the control (base) terminal to switch much larger currents between the other two (emitter and collector) terminals. The quiescent power dissipated by these base currents limits the maximum number of transistors that can be integrated onto a single die. Metal Oxide Semiconductor Field Effect Transistors (MOSFETs) offer the compelling advantage that they draw almost zero control current while idle. They come in two flavors: nMOS and pMOS, using n-type and p-type dopants, respectively. The original idea of field effect transistors dated back to the German scientist Julius Lilienfeld in 1925 [US patent 1745,175] and a structure closely resembling the MOSFET was proposed in 1935 by Oskar Heil [British patent 439,457], but materials problems foiled early attempts to make functioning devices.

Frank Wanlass at Fairchild described the first logic gates using MOSFETs in 1963 [Wanlass63]. His gates used both nMOS and pMOS transistors, earning the name Complementary Metal Oxide Semiconductor, or CMOS. The circuits used discrete transistors but consumed only nanowatts of power, six orders of magnitude less than their bipolar counterparts. With the development of the silicon planar process, MOS integrated circuits became attractive for their low cost because each transistor occupied less area and the fabrication process was simpler [Vadasz69]. Early processes used only pMOS transistors and suffered from poor performance, yield, and reliability. Processes using nMOS transistors became dominant in the 1970s [Mead80]. Intel pioneered nMOS technology with its 1101 256-bit static random access memory and 4004 4-bit microprocessor shown in Figure 1.3. While the nMOS process was less expensive than CMOS, nMOS logic gates still consumed power while idle. Power consumption became a major issue in the 1980s as

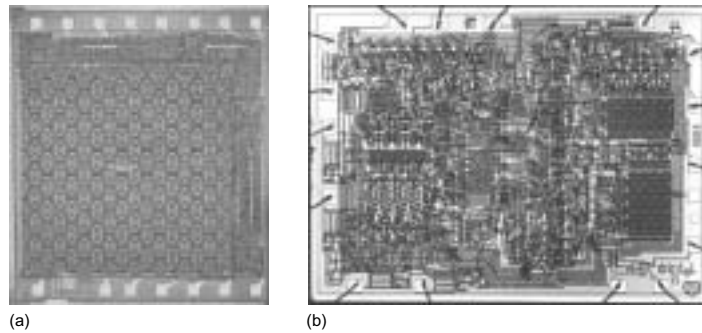


FIG 1.3 Intel 1101 SRAM (a) and 4004 microprocessor (b)

hundreds of thousands of transistors were integrated onto a single die. CMOS processes were widely adopted and have essentially replaced nMOS and bipolar processes for nearly all digital logic applications.

Gordon Moore observed in 1965 that plotting the number of transistors that can be most economically manufactured on a chip gives a straight line on a semilogarithmic scale [Moore65]. At the time he found transistor count doubling every 18 months. This observation has been called *Moore's Law* and has become a self-fulfilling prophecy. Figure 1.4 shows that the number of transistors in Intel microprocessors has doubled every 26 months since the invention of the 4004.

The level of integration of chips has been classified as small-scale, medium-scale, large-scale, and very large-scale. *Small-scale integration* (SSI) circuits such as the 7404 inverter have fewer than 10 gates, with a conversion of roughly half a dozen transistors per gate. *Medium-scale integration* (MSI) circuits such as the 74161 counter have up to 1000 gates. *Large-scale integration* (LSI) circuits such as simple 8-bit microprocessors have up to 10,000 gates. It soon became apparent that new names would have to be created every five years if this naming trend continued and thus the term *very large-scale integration* (VLSI) is used to describe most integrated circuits from the 1980s onward.

A corollary of Moore's law is that transistors become faster, consume less power, and are cheaper to manufacture each year. Figure 1.5 shows that Intel microprocessor clock frequencies have doubled roughly every 34 months. Remarkably, the improvements have accelerated in recent years. Computer performance has grown even more than raw clock speed. Even though an individual CMOS transistor uses very little energy each time it switches, the enormous numbers of transistors switching at very high rates of speed have made power consumption a major design consideration again.

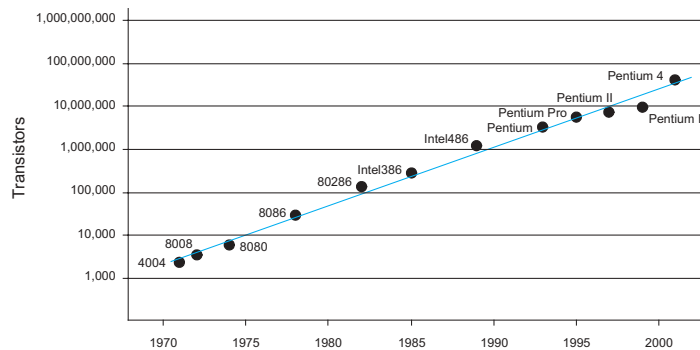


FIG 1.4 Transistors in Intel microprocessors [Intel03]

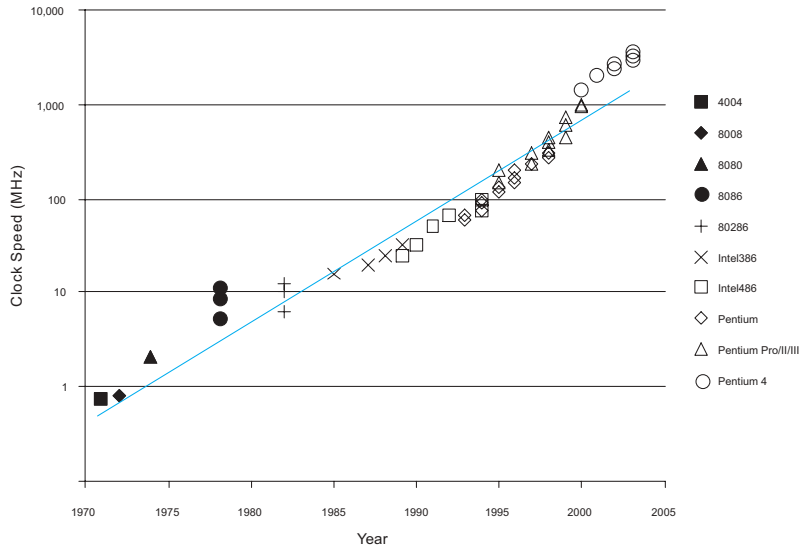


FIG 1.5 Clock frequencies of Intel microprocessors

The 4004 used transistors with minimum dimensions of $10\ \mu\text{m}$ in 1971. The Pentium 4 uses transistors with minimum dimensions of $130\ \mu\text{m}$ in 2003, corresponding to two orders of magnitude in improvement over three decades. Obviously this *scaling* cannot go on forever because transistors cannot be smaller than atoms. However, many predictions of fundamental limits to scaling have already proven wrong. Creative engineers and material scientists have billions of dollars to gain by getting ahead of their competitors. In the early 1990s, experts agreed that scaling would continue for at least a decade but that beyond that point the future was murky. In 2003, we still believe that scaling will continue for at least another decade. The future is yours to invent.

1.2 Book Summary

As VLSI transistor budgets have grown exponentially, designers have come to rely on increasing levels of automation to seek corresponding productivity gains. Many designers spend much of their effort specifying circuits with hardware description languages and sel-

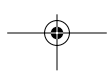


dom look at actual transistors. Nevertheless, chip design is not software engineering. Addressing the harder problems requires a fundamental understanding of circuit and physical design. Therefore, this book focuses on building an understanding of integrated circuits from the bottom up.

In this chapter we will take a simplified view of CMOS transistors as switches. With this model we will develop CMOS logic gates and latches. CMOS transistors are mass-produced on silicon wafers using lithographic steps much like a printing press process. We will explore how to lay out transistors by specifying rectangles indicating where dopants should be diffused, polysilicon should be grown, metal wires should be deposited, and contacts should be etched to connect all the layers together. By the middle of this chapter you will understand all the principles required to design and lay out your own simple CMOS chip. The chapter concludes with an extended example demonstrating the design of a simple 8-bit MIPS microprocessor chip. The processor raises many of the design issues that will be developed in more depth throughout the book. The best way to learn VLSI design is by doing it. A set of laboratory exercises are available on the Web (see the Web Supplement portion of the preface) to guide you through the design of your own microprocessor chip.

Of course, transistors are not simply switches. Chapter 2 develops first-order current-voltage (I-V) and capacitance-voltage (C-V) models for transistors and describes the important second-order effects. These models are used to predict the transfer characteristics of CMOS inverters. A summary of CMOS processing technology is presented in Chapter 3. The basic processes in current use are described along with some interesting process enhancements. Representative layout design rules are also presented. Chapter 4 addresses performance estimation for circuits. The first-order I-V characteristics are both too complicated and too inaccurate to apply by hand to interesting circuits. Fortunately, transistors can be modeled as having an effective resistance and capacitance for the purpose of estimating delay. The relative performance of different gates can be quantified through their logical effort, a technique that we will revisit throughout the book. Wires are also addressed because they are as important as the transistors to overall performance. Simulation is discussed in Chapter 5 and is used to obtain more accurate performance predictions as well as to verify the correctness of circuits and logic. Chapter 6 addresses combinational circuit design. A whole kit of circuit families are available with different tradeoffs in speed, power, complexity, and robustness. Chapter 7 continues with sequential circuit design, including clocking and latching techniques.

The remainder of this book presents a subsystem view of CMOS design. Chapter 8 focuses on a range of current design methods, identifying the issues peculiar to CMOS. Testing and design-for-test techniques are discussed in Chapter 9. Chapter 10 catalogs designs for a host of datapath subsystems including adders, shifters, multipliers, counters, and others. Chapter 11 similarly describes memory subsystems including SRAMs, DRAMs, CAMs, ROMs, and PLAs. Finally, Chapter 12 addresses special-purpose subsystems including clocking, I/O, and mixed-signal blocks. Hardware description languages (HDLs) are used in the design of nearly all digital integrated circuits today. Appendices A and B provide tutorials in Verilog and VHDL, the two dominant HDLs.



A number of sections are marked with an “optional” icon . These sections describe particular subjects in greater detail. You may skip over these sections on a first reading and return to them when they are of practical relevance.



1.3 MOS Transistors

Silicon (Si), a semiconductor, forms the basic starting material for a large class of integrated circuits [Tsivdis99]. Pure silicon consists of a three-dimensional *lattice* of atoms. Silicon is a Group IV element, so it forms covalent bonds with four adjacent atoms, as shown in Figure 1.6(a). The lattice is shown in the plane for ease of drawing, but it actually forms a cubic crystal. As all of its valence electrons are involved in chemical bonds, it is a poor conductor. The conductivity can be raised by introducing small amounts of impurities into the silicon lattice. These impurities are called *dopants*. A Group V dopant such as arsenic has five valence electrons. It replaces a silicon atom in the lattice and still bonds to four neighbors, so the fifth valence electron is loosely bound to the arsenic atom, as shown in Figure 1.6(b). Thermal vibration of the lattice at room temperature is enough to set the electron free to move, leaving a positively charged As^+ ion and a free electron. The free electron can carry current so the conductivity is higher. We call this an *n*-type semiconductor because the free carriers are negatively charged electrons. Similarly, a Group III dopant such as boron is missing one valence electron, as shown in Figure 1.6(c). The dopant atom can borrow an electron from a neighboring silicon atom, which in turn becomes short by one electron. That atom in turn can borrow an electron, and so forth, so the missing electron, or *hole*, can propagate about the lattice. The hole acts as a positive carrier so we call this a *p*-type semiconductor.

A junction between p-type and n-type silicon is called a *diode*, shown in Figure 1.7. When the voltage on the p-type semiconductor, called the *anode*, is raised above the n-type *cathode*, we say the diode is *forward biased* and current will flow. When the anode voltage is less than or equal to the cathode voltage, the diode is *reverse biased* and almost zero current flows.

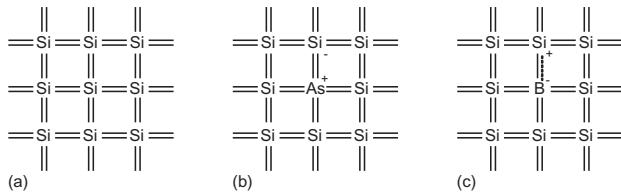


FIG 1.6 Silicon lattice and dopant atoms

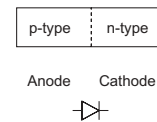


FIG 1.7 p-n junction diode structure and symbol

An *MOS* (Metal-Oxide-Silicon) structure is created by superimposing several layers of conducting and insulating materials to form a sandwich-like structure. These structures are manufactured using a series of chemical processing steps involving oxidation of the silicon, the diffusion of impurities into the silicon to give it certain conduction characteristics, and the deposition and etching of aluminum or other metals to provide interconnection in the same way that a printed wiring board is constructed. This is carried out on a single crystal of silicon, which is available as thin flat circular wafers around 15–30 cm in diameter. CMOS technology provides two types of transistors (also called *devices* in this text): an n-type transistor (nMOS) and a p-type transistor (pMOS). Transistor operation is based on electric fields so the devices are also called Metal Oxide Semiconductor Field Effect Transistors (*MOSFETs*) or simply *FETs*. Cross-sections and symbols of these transistors are shown in Figure 1.8. The n+ and p+ regions indicate heavily doped n- or p-type silicon.

Each transistor consists of a stack of the conducting *gate*, an insulating layer of silicon dioxide (SiO_2 , better known as glass), and the silicon wafer, also called the *substrate*, *body*, or *bulk*. Gates of early transistors were built from metal, so the stack was called metal-oxide-semiconductor, or MOS. Now the gate is typically formed from polycrystalline silicon (*polysilicon*), but the name stuck. An nMOS transistor is built with a p-type body and has regions of n-type semiconductor adjacent to the gate called the *source* and *drain*. They are physically equivalent and for now we will regard them as interchangeable. The body is typically grounded. A pMOS transistor is just the opposite, consisting of p-type source and drain regions with an n-type body. In a CMOS technology with both flavors of transistors, the substrate is either n-type or p-type. The other flavor of transistor must be built in a special well in which dopant atoms have been locally added to form the body of the opposite type.

The gate is a control input: It affects the flow of electrical current between the source and drain. Consider an nMOS transistor. The body is generally grounded so the p–n junc-

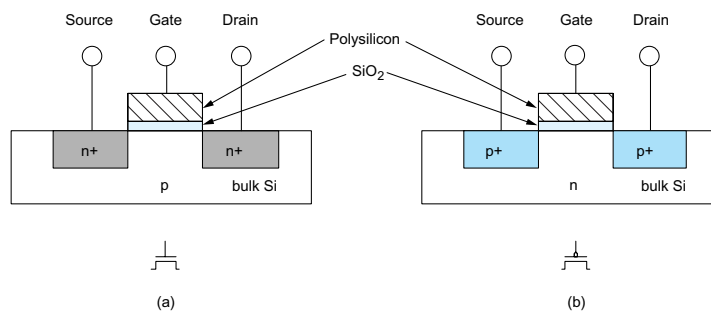


FIG 1.8 nMOS transistor (a) and pMOS transistor (b)

tions of the source and drain to body are reverse-biased. If the gate is also grounded, no current flows through the reverse-biased junctions. Hence, we say the transistor is OFF. If the gate voltage is raised, it creates an electric field that starts to attract free electrons to the underside of the Si-SiO₂ interface. If the voltage is raised enough, the electrons outnumber the holes and a thin region under the gate called the *channel* is inverted to act as an n-type semiconductor. Hence, a conducting path of electron carriers is formed from source to drain and current can flow. We say the transistor is ON.

For a pMOS transistor, the situation is again reversed. The body is held at a high potential. When the gate is also at a high potential, the source and drain junctions are reverse-biased and no current flows so the transistor is OFF. When the gate voltage is lowered, positive charges are attracted to the underside of the Si-SiO₂ interface. A sufficiently low gate voltage inverts the channel and a conducting path of positive carriers is formed from source to drain, so the transistor is ON. Notice that the symbol for the pMOS transistor has a bubble on the gate, indicating that the transistor behavior is the opposite of the nMOS.

Throughout this book, we will call the high potential V_{DD} or POWER and assume it represents a logic '1' value in digital circuits. In many logic families of the 1970s and 1980s, V_{DD} was set to 5 volts or occasionally higher. Smaller, more recent transistors are unable to withstand such high voltages and have used supplies of 3.3 V, 2.5 V, 1.8 V, 1.5 V, and so forth. The low potential is called GROUND (GND) or V_{SS} and represents a logic '0.' It is normally 0 volts.

In summary, the gate of an MOS transistor controls the flow of current between the source and drain. Simplifying this to the extreme allows the MOS transistors to be viewed as simple on/off switches. When the gate of an nMOS transistor is '1,' the transistor is ON and there is a conducting path from source to drain. When the gate is low, the nMOS transistor is OFF and almost zero current flows from source to drain. A pMOS transistor is just the opposite, being ON when the gate is low and OFF when the gate is high. This switch model is illustrated in Figure 1.9, where g , s , and d indicate gate, source, and drain. It is so useful that it will be our most common model for thinking about circuit behavior.

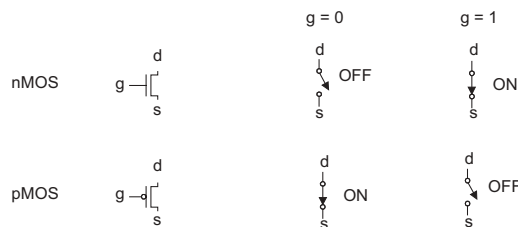


FIG 1.9 Transistor symbols and switch-level models

1.4 CMOS Logic

1.4.1 The Inverter

Figure 1.10(a) shows a CMOS inverter or NOT gate using one nMOS transistor and one pMOS transistor. The horizontal bar at the top indicates V_{DD} and the triangle at the bottom indicates GND. When the input A is '0,' the nMOS transistor is OFF and the pMOS transistor is ON. Thus the output Y is pulled up to '1' because it is connected to V_{DD} but not to GND. Conversely, when A is '1,' the nMOS is ON, the pMOS is OFF, and the Y is pulled down to '0.' This is summarized in the truth table of Table 1.1. The symbol is given in Figure 1.10(b).

Table 1.1 Inverter truth table

| A | Y |
|---|---|
| 0 | 1 |
| 1 | 0 |

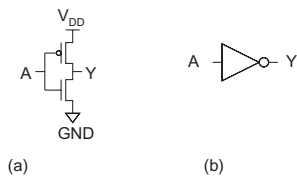


FIG 1.10 Inverter schematic (a) and symbol (b) $Y = \bar{A}$

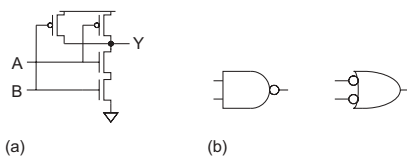


FIG 1.11 2-input NAND gate schematic (a) and symbol (b) $Y = A \cdot B$

1.4.2 The NAND Gate

Figure 1.11(a) shows a 2-input CMOS NAND gate. It consists of two series nMOS transistors between Y and GND and two parallel pMOS transistors between Y and V_{DD} . If either input A or B is '0,' at least one of the nMOS transistors will be OFF, breaking the path from Y to GND. But at least one of the pMOS transistors will be ON, creating a path from Y to V_{DD} . Hence, the output Y will be '1.' If both inputs are '1,' both of the nMOS transistors will be ON and both of the pMOS transistors will be OFF. Hence, the output will be '0.' The truth table is given in Table 1.2 and the symbol is shown in Figure 1.11(b). Note that by DeMorgan's Law, the inversion bubble may be placed on either side of the gate. In the figures in this book, two lines intersecting at a T-junction are connected. Two lines crossing are connected if and only if a dot is shown.

k -input NAND gates are constructed using k series nMOS transistors and k parallel pMOS transistors. For example, a 3-input NAND gate is shown in Figure 1.12. When any of the inputs are 0, the output is pulled high through the parallel pMOS transistors. When all of the inputs are '1,' the output is pulled low through the series nMOS transistors.

Table 1.2 NAND gate truth table

| A | B | pull-down network | pull-up network | Y |
|---|---|-------------------|-----------------|---|
| 0 | 0 | OFF | ON | 1 |
| 0 | 1 | OFF | ON | 1 |
| 1 | 0 | OFF | ON | 1 |
| 1 | 1 | ON | OFF | 0 |

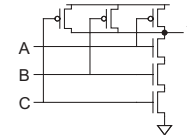


FIG 1.12 3-input NAND gate schematic
 $Y = A \cdot B \cdot C$

1.4.3 Combinational Logic

The inverter and NAND gates are examples of *complementary CMOS logic gates*, also called *static CMOS gates*. In general, a fully complementary CMOS gate has an nMOS *pull-down network* to connect the output to '0' (GND) and pMOS *pull-up network* to connect the output to '1' (V_{DD}), as shown in Figure 1.13. The networks are arranged such that one is ON and the other OFF for any input pattern.

The pull-up and pull-down networks in the inverter each consisted of a single transistor. The NAND gate used a series pull-down network and a parallel pull-up network. More elaborate networks are used for more complex gates. Two or more transistors in series are ON only if all of the series transistors are ON. Two or more transistors in parallel are ON if any of the parallel transistors are ON. This is illustrated in Figure 1.14 for nMOS and pMOS transistor pairs. By using combinations of these constructions, CMOS combinational gates can be constructed.

In general when we join a pull-up network to a pull-down network to form a logic gate as shown in Figure 1.13, they both will attempt to exert a logic level at the output. The possible levels at the output are shown in Table 1.3. From this table it can be seen that the output of a CMOS logic gate can be in four states. The '1' and '0' levels have been encountered with the inverter and NAND gates, where either the pull-up or pull-down is OFF and the other structure is ON. When both pull-up and pull-down are OFF, the *high-impedance* or *floating Z* output state results. This is of importance in multiplexers, memory elements, and bus drivers. The *crowbarred X* level exists when both pull-up and pull-down are simultaneously turned ON. This causes an indeterminate level and also static power to be dissipated. It is usually an unwanted condition in any CMOS digital circuit.

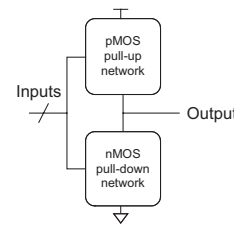


FIG 1.13 General logic gate using pull-up and pull-down networks

Table 1.3 Output states of CMOS logic gate

| | pull-up OFF | pull-up ON |
|---------------|-------------|----------------|
| pull-down OFF | Z | 1 |
| pull-down ON | 0 | crowbarred (X) |

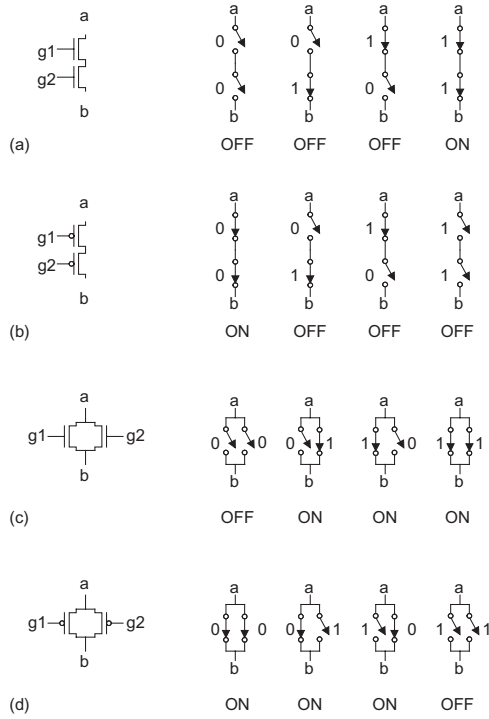


FIG 1.14 Connection and behavior of series and parallel transistors

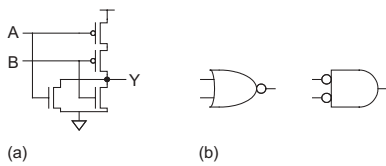


FIG 1.15 2-input NOR gate schematic (a) and symbol (b) $Y = A + B$

1.4.4 The NOR Gate

A 2-input NOR gate is shown in Figure 1.15. The nMOS transistors are in parallel to pull the output low when either input is high. The pMOS transistors are in series to pull the output high when both inputs are low, as indicated by the truth table of Table 1.4. As with the NAND gate, there is never a case in which the output is crowbarred or left floating.

Table 1.4 NOR gate truth table

| A | B | Y |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

Example

Sketch a 3-input CMOS NOR gate.

Solution: Figure 1.16 shows such a gate. If any input is high, the output is pulled low through the parallel nMOS transistors. If all inputs are low, the output is pulled high through the series pMOS transistors.

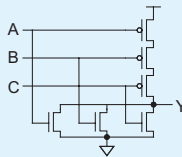


FIG 1.16 3-input NOR gate schematic
 $Y = \overline{A + B + C}$

1.4.5 Compound Gates

A compound gate is formed by using a combination of series and parallel switch structures. For example, the derivation of the switch connection diagram for the function $Y = \overline{(A \cdot B) + (C \cdot D)}$ is shown in Figure 1.17. This function is sometimes called AND-OR-INVERT-22, or AOI22 because it performs the NOR of a pair of 2-input ANDs. For the nMOS pull-down network, take the uninverted expression $((A \cdot B) + (C \cdot D))$ indicating when the output should be pulled to '0.' The AND expressions $(A \cdot B)$ and $(C \cdot D)$ may be implemented by series connections of switches as shown in Figure 1.17(a). Now taking these as subswitches and ORing the result requires the parallel connection of these two structures, which is shown in Figure 1.17(b). For the pMOS pull-up network we must compute the complementary expression using switches that turn on with inversed polarity. By DeMorgan's Law, this is equivalent to interchanging AND and OR operations. Hence, transistors that appear in series in the pull-down network must appear in

parallel in the pull-up network. Transistors that appear in parallel in the pull-down network must appear in series in the pull-up network. This principle is called *conduction complements* and has already appeared in the design of the NAND and NOR gates. In the pull-up network, the parallel combination of A and B is placed in series with the parallel combination of C and D . This progression is evident in Figure 1.17(c) and Figure 1.17(d). Putting the networks together yields the connection diagram (Figure 1.17(e)). The schematic icon is shown in Figure 1.17(f), which shows that this gate can be used in a 2-input multiplexer. If $C = \bar{B}$, then $Y = A$ if B is true, while $Y = D$ if B is false.

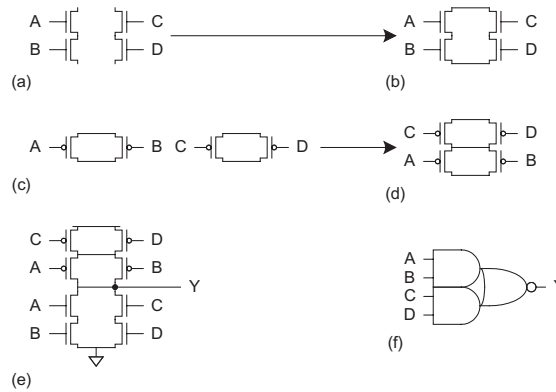


FIG 1.17 CMOS compound gate for function $Y = (A \bullet B) + (C \bullet D)$

1.4.6 Pass Transistors and Transmission Gate

The *strength* of a signal is measured by how closely it approximates an ideal voltage source. In general, the stronger a signal, the more current it can source or sink. The power supplies, or *rails*, (V_{DD} and GND) are the source of the strongest '1's and '0's.

An nMOS transistor is an almost perfect switch when passing a '0' and thus we say it passes a *strong* '0.' However, the nMOS transistor is imperfect at passing a '1.' The high voltage level is somewhat less than V_{DD} , as will be explained in Section 2.3.2. We say it passes a *degraded* or *weak* '1.' A pMOS transistor again has the opposite behavior, passing strong '1's but degraded '0's. The transistor symbols and behaviors are summarized in Figure 1.19 with g , s , and d indicating gate, source, and drain. The symbols with arrows represent switches in the OFF and ON positions.

Example

Sketch a complementary CMOS gate computing $Y = \overline{(A + B + C)} \cdot D$.

Solution: Figure 1.18 shows such an OR-AND-INVERT-3-1 (OAI31) gate. The nMOS pull-down network pulls the output low if D is '1' and either A or B or C are '1', so D is in series with the parallel combination of A , B , and C . The pMOS pull-up network is the conduction complement, so D must be in parallel with the series combination of A , B , and C .

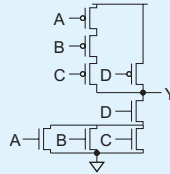


FIG 1.18 CMOS compound gate for function $Y = (A + B + C) \cdot D$

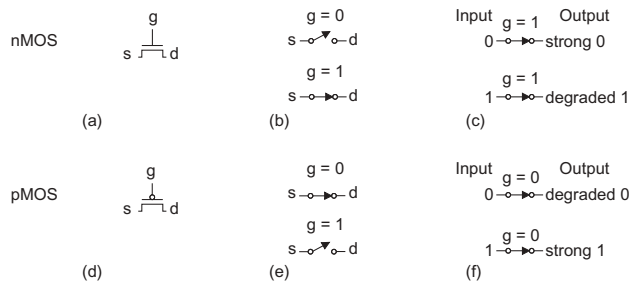


FIG 1.19 Pass transistor strong and degraded outputs

When an nMOS or pMOS is used alone as an imperfect switch, we sometimes call it a *pass transistor*. By combining an nMOS and a pMOS transistor in parallel (Figure 1.20(a)), we obtain a switch that turns on when a '1' is applied to g (Figure 1.20(b)) in which '0's and '1's are both passed in an acceptable fashion (Figure 1.20(c)). We term this a

transmission gate or pass gate. In a circuit where only a '0' or a '1' has to be passed, the appropriate transistor (n or p) can be deleted, reverting to a single nMOS or pMOS device. Note that both the control input and its complement are required by the transmission gate. This is called *double rail logic*. Some circuit symbols for the transmission gate are shown in Figure 1.20(d).¹ None are easier to draw than the simple schematic, so we will use the schematic to represent a transmission gate in this book.

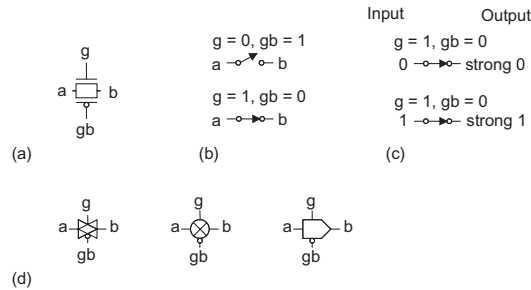


FIG 1.20 Transmission gate

In all of our examples so far, the inputs drive the gate terminals of nMOS transistors in the pull-down network and pMOS transistors in the complementary pull-up network, as was shown in Figure 1.13. Thus the nMOS transistors only need to pass 0's and the pMOS only pass 1's, so the output is always strongly driven and the levels are never degraded. This is called a *fully restored logic gate* and simplifies circuit design considerably. In contrast to other forms of logic, where the pull-up and pull-down switch networks have to be ratioed in some manner, complementary CMOS gates operate correctly independently of the physical sizes of the transistors. Moreover, there is never a path through 'ON' transistors from the '1' to the '0' supplies for any combination of inputs (in contrast to single-channel MOS, GaAs technologies, or bipolar). As we will learn in subsequent chapters, this is the basis for the low static power dissipation in CMOS.

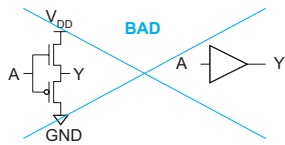


FIG 1.21 Bad noninverting buffer

A consequence of the design of complementary CMOS gates is that they must be inverting. The nMOS pull-down network turns ON when inputs are '1,' leading to '0' at the output. We might be tempted to turn the transistors upside down to build a noninverting gate. For example, Figure 1.21 shows a noninverting buffer. Unfortunately, now both the nMOS and pMOS transistors produce degraded outputs, so the technique should be avoided. Instead, we

¹We call the left and right terminals *a* and *b* because each is technically the source of one of the transistors and the drain of the other.

can build noninverting functions from multiple stages of noninverting gates. Figure 1.22 shows a 4-input AND gate built from two levels of inverting complementary CMOS gates. In isolation, the NAND design is simpler. In the context of a larger system, one can optimize the gates depending on the speed and density required.

Similarly, the compound gate of Figure 1.17 could be built with two AND gates, an OR gate, and an inverter. The AND and OR gates in turn could be constructed from NAND/NOR gates and inverters, shown in Figure 1.23, using a total of 20 transistors, as compared to 8 in Figure 1.17. CMOS logic designers must learn to take advantage of the efficiencies of compound gates and rather than using large numbers of AND/OR gates.

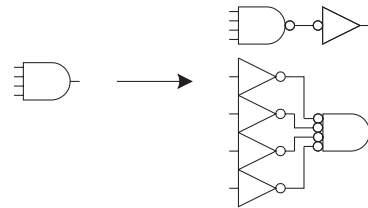


FIG 1.22 Various implementations of a CMOS 4-input AND gate

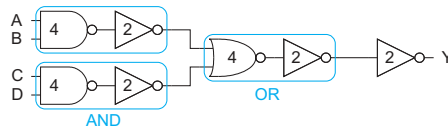


FIG 1.23 Inefficient discrete gate implementation of AOI22 indicating transistor counts

1.4.7 Transmission Gates and Tristates

Figure 1.24 shows symbols for a *tristate buffer*. When the enable input EN is '1,' the output *Y* equals the input *A*, just as in an ordinary buffer. When the enable is '0,' *Y* is left floating (a 'Z' value). This is summarized in Table 1.5. Sometimes both true and complementary enable signals EN and \overline{EN} are drawn explicitly, while sometimes only EN is shown.



FIG 1.24 Tristate buffer symbol

| Table 1.5 Truth table for tristate | | |
|------------------------------------|---|---|
| EN / \overline{EN} | A | Y |
| 0/1 | 0 | Z |
| 0/1 | 1 | Z |
| 1/0 | 0 | 0 |
| 1/0 | 1 | 1 |

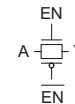


FIG 1.25 Transmission gate

The transmission gate in Figure 1.25 has the same truth table as a tristate buffer. It only requires two transistors but it is a *nonrestoring* circuit. If the input is a noisy or other-

wise degraded signal, the output will receive the same noise. After several stages of nonrestoring logic, a signal can become too degraded to recognize. We will see in Section 4.2 that the delay of a series of nonrestoring gates also increases quadratically with the number of gates in series.

Figure 1.26(a) shows a *tristate inverter*. The output is actively driven from V_{DD} or GND, so it is a restoring logic gate. Unlike any of the gates considered so far, the tristate inverter does not obey the conduction complements rule because it must allow the output to float under certain input combinations. When EN is '0' (Figure 1.26(b)), both enable transistors are OFF, leaving the output floating. When EN is '1' (Figure 1.26(c)), both enable transistors are ON. They are conceptually removed from the circuit, leaving a simple inverter. Figure 1.26(d) shows symbols for the tristate inverter. The complementary enable signal can be generated internally or can be routed to the cell explicitly. A tristate buffer can be built as a tristate inverter following an ordinary inverter.

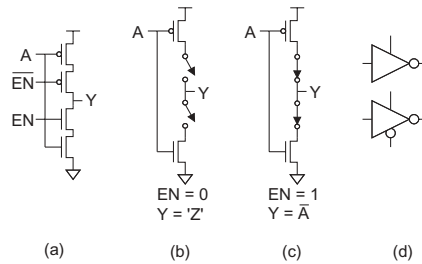


FIG 1.26 Tristate inverter

Tristates were once commonly used to allow multiple units to drive a common bus, as long as more than one are not simultaneously enabled. Distributing mutually exclusive enable signals in a timely fashion across a large chip is becoming more difficult, so multiplexers are now preferred.

1.4.8 Multiplexers

Multiplexers are key components in CMOS memory elements and data manipulation structures. A multiplexer chooses the output to be one of several inputs based on a select signal. A two-input, or 2:1 multiplexer, chooses input $D0$ when the select is '0' and input $D1$ when the select is '1.' The truth table is given in Table 1.6; the logic function is $Y = \bar{S} \cdot D0 + S \cdot D1$.

Table 1.6 Multiplexer truth table

| S/\bar{S} | $D1$ | $D0$ | Y |
|-------------|------|------|-----|
| 0/1 | X | 0 | 0 |
| 0/1 | X | 1 | 1 |
| 1/0 | 0 | X | 0 |
| 1/0 | 1 | X | 1 |

Two transmission gates can be tied together to form a compact 2-input multiplexer, as shown in Figure 1.27(a). The select and its complement enable exactly one of the two transmission gates at any given time. The complementary select \bar{S} is often not drawn in the symbol, as in Figure 1.27(b).

Again, the transmission gates produce a nonrestoring multiplexer. We could build a restoring, inverting multiplexer out of gates in several ways. One is the compound gate of Figure 1.17, connected as shown in Figure 1.28(a). Another is to gang together two tristate inverters, as shown in Figure 1.28(b). Notice that the schematics of these two approaches are nearly identical, save that the pull-up network has been slightly simplified and permuted in Figure 1.28(b). This is possible because the select and its complement are mutually exclusive. The tristate approach is slightly more compact and faster because it requires less internal wire. Again, if the complementary select is generated within the cell, it is omitted from the symbol (Figure 1.28(c)).

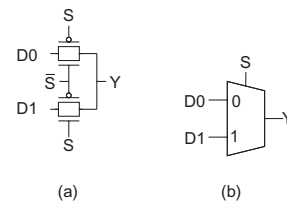


FIG 1.27 Transmission gate multiplexer

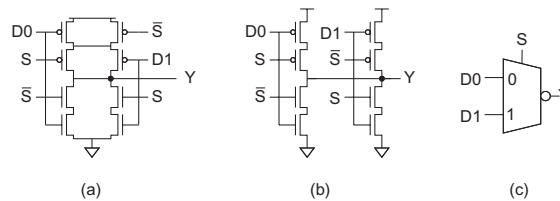


FIG 1.28 Inverting multiplexer

Larger multiplexers can be built from multiple 2-input multiplexers or by directly ganging together several tristates. The latter approach requires decoded select signals for each tristate. 4-input (4:1) multiplexers using each of these approaches are shown in Figure 1.29.

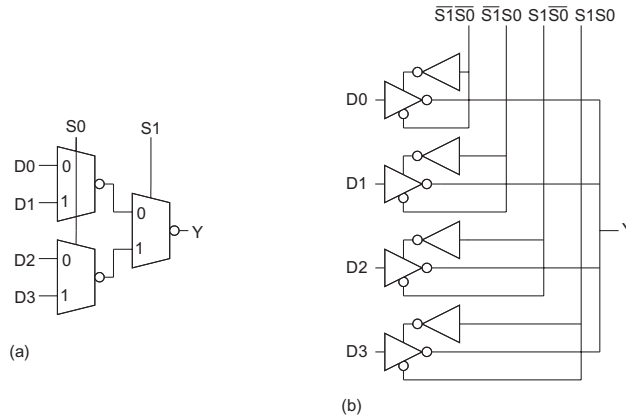


FIG 1.29 4:1 multiplexer

1.4.9 Latches and Flip-Flops

Using the combinational circuits developed so far, we can now build sequential circuits such as latches and flip-flops. A *D latch* using one 2-input multiplexer and two inverters is shown in Figure 1.30(a). It consists of a data input, D , a clock input, CLK , and true and complementary outputs Q and \bar{Q} . When $CLK = '1'$, the latch is *transparent*. $Q = D$ and $\bar{Q} = \bar{D}$ (Figure 1.30(c)). When CLK is switched to '0,' the latch is *opaque*. A feedback path around the inverter pair is established (Figure 1.30(d)) to hold the current state of Q indefinitely. While the latch is opaque, the input D is ignored. The multiplexer can be constructed from a pair of transmission gates, shown in Figure 1.30(b).

The D latch is also known as a *level-sensitive latch* because the state of the output is dependent on the level of the clock signal, as shown in Figure 1.30(e). The latch shown is a positive-level-sensitive latch, represented by the symbol in Figure 1.30(f). By inverting the control connections to the multiplexer, a negative-level-sensitive latch may be constructed.

By combining two level-sensitive latches, one positive-sensitive and one negative-sensitive, we construct an *edge-triggered flip-flop* as shown in Figure 1.31(a-b). By convention, the first latch stage is called the *master* and the second is called the *slave*.

While CLK is low, the master negative-level-sensitive latch output ($\bar{Q}M$) follows the D input while the slave positive-level-sensitive latch holds the previous value (Figure 1.31(c)). When the clock transitions from 0 to 1, the master latch ceases to sample the input and holds the D value at the time of the clock transition. The slave latch opens, passing the stored master value ($\bar{Q}M$) to the output of the slave latch (Q). The D input is blocked from affecting the output because the master is disconnected from the D input

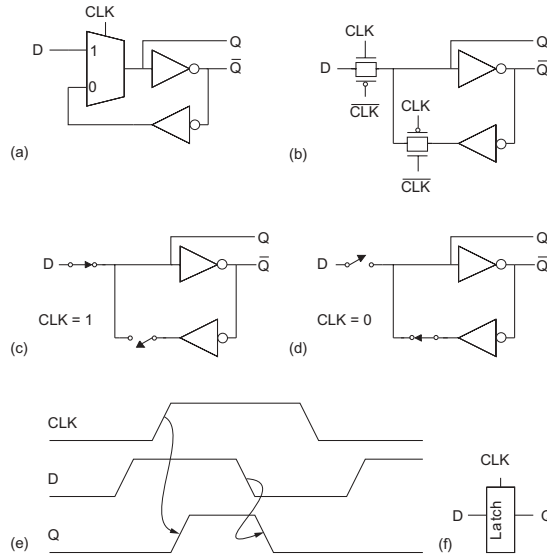


FIG 1.30 CMOS positive-level-sensitive D latch

(Figure 1.31(d)). When the clock transitions from 1 to 0, the slave latch holds its value and the master starts sampling the input again.

In summary, this flip-flop copies D to Q on the rising edge of the clock, as shown in Figure 1.31(e). Thus this device is called a positive-edge triggered flip-flop (also called a D flip-flop, D register, or master-slave flip-flop). Figure 1.31(f) shows the circuit symbol for the flip-flop. By reversing the latch polarities, a negative edge triggered flip-flop may be constructed. A collection of two or more D flip-flops sharing a common clock input is called a register. A register is often drawn as a flip-flop with multi-bit D and Q busses.

In Section 7.2.3 we will see that flip-flops may experience hold-time failures if the system has too much clock skew, i.e., if one flip-flop triggers early and another triggers late because of variations in clock arrival times. In industrial designs, a great deal of effort is devoted to timing simulations to catch hold-time problems. When design time is more important (e.g., in academic class projects), hold time problems can be avoided altogether by distributing a two-phase nonoverlapping clock. Figure 1.32 shows the flip-flop clocked with two nonoverlapping phases. As long as the phases do not overlap even with worst-case skews, at least one latch will be opaque at any given time and hold-time problems will never occur.

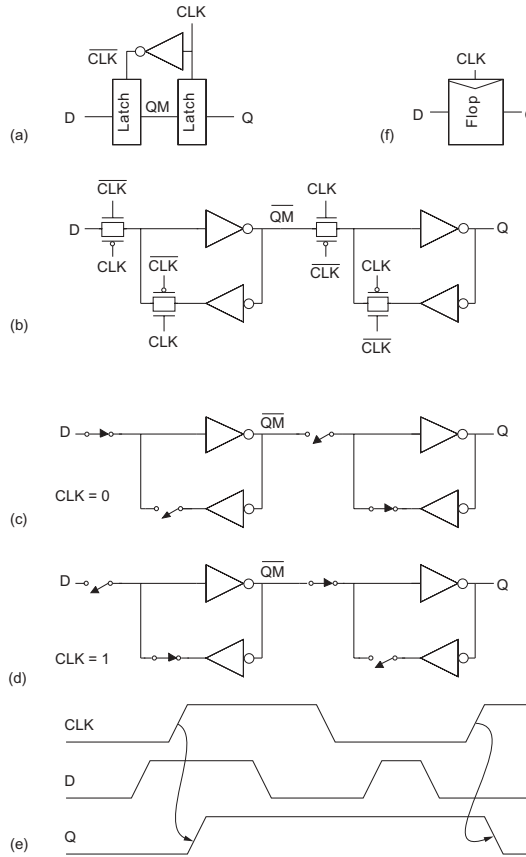


FIG 1.31 CMOS positive-edge-triggered D flip-flop

It is often useful to provide reset and/or enable signals to flip-flops and latches. Such modifications are straightforward and are discussed in Section 7.3.