generation, designers are having to make trade-offs between improving power and improving delay. Although the cost of printing each transistor goes down, the one-time design costs are increasing exponentially, relegating state-of-the-art processes to chips that will sell in huge quantities or that have cutting-edge performance requirements. 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 2009, we still believe that Moore's Law will continue for at least another decade. The future is yours to invent.

1.2 Preview

As the number of transistors on a chip has 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 functions with hardware description languages and seldom 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. 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 at `www.cmosvlsi.com` to guide you through the design of your own microprocessor chip.

1.3 MOS Transistors

Silicon (Si), a semiconductor, forms the basic starting material for most integrated circuits [Tsividis99]. 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 1.7(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, pure silicon is a poor conductor. The conductivity can be raised by introducing small amounts of impurities, called *dopants*, into the silicon lattice. A dopant from Group V of the periodic table, 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 1.7(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

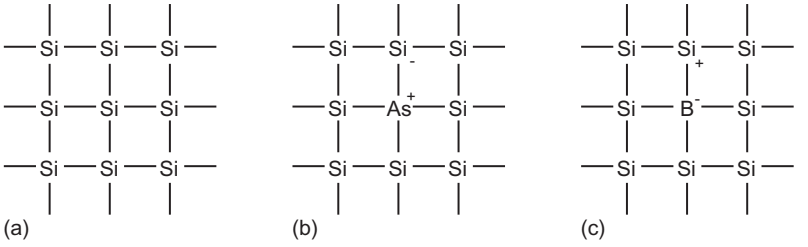

FIGURE 1.7 Silicon lattice and dopant atoms

semiconductor because the free carriers are negatively charged electrons. Similarly, a Group III dopant, such as boron, has three valence electrons, as shown in Figure 1.7(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*, as shown in Figure 1.8. When the voltage on the p-type semiconductor, called the *anode*, is raised above the n-type *cathode*, the diode is *forward biased* and current flows. When the anode voltage is less than or equal to the cathode voltage, the diode is *reverse biased* and very little current flows.

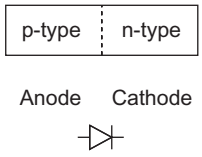

FIGURE 1.8 p-n junction diode structure and symbol

A Metal-Oxide-Semiconductor (*MOS*) 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, selective introduction of dopants, and deposition and etching of metal wires and contacts. Transistors are built on nearly flawless single crystals of silicon, which are available as thin flat circular wafers of 15–30 cm in diameter. CMOS technology provides two types of transistors (also called *devices*): an n-type transistor (*nMOS*) and a p-type transistor (*pMOS*). Transistor operation is controlled by electric fields so the devices are also called Metal Oxide Semiconductor Field Effect Transistors (*MOSFET*s) or simply *FET*s. Cross-sections and symbols of these transistors are shown in Figure 1.9. The n+ and p+ regions indicate heavily doped n- or p-type silicon.

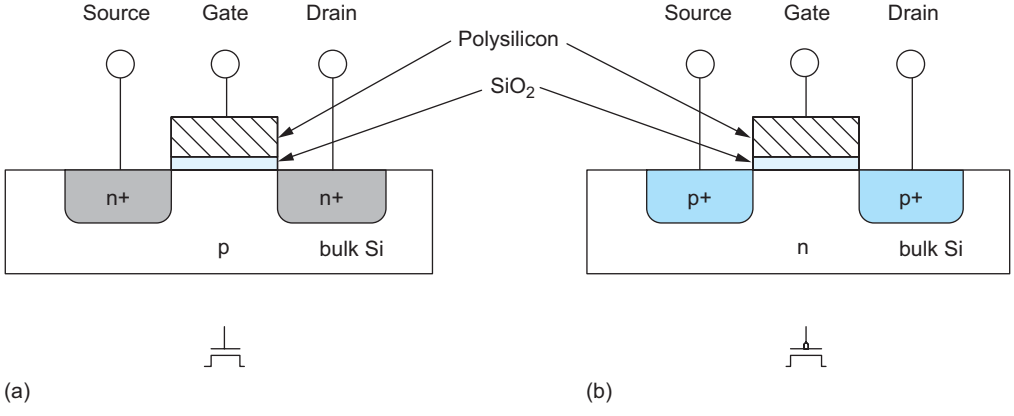

FIGURE 1.9 nMOS transistor (a) and pMOS transistor (b)

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. Since the 1970s, the gate has been formed from polycrystalline silicon (*polysilicon*), but the name stuck. (Interestingly, metal gates reemerged in 2007 to solve materials problems in advanced manufacturing processes.) 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 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 junctions 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–SiO_2 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 positive voltage. When the gate is also at a positive voltage, 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–SiO_2 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.

The positive voltage is usually called V_{DD} or POWER and represents a logic 1 value in digital circuits. In popular logic families of the 1970s and 1980s, V_{DD} was set to 5 volts. 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, 1.2 V, 1.0 V, and so forth. The low voltage 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 1.10, where *g*, *s*, and *d* indicate gate, source, and drain. This model will be our most common one when discussing circuit behavior.

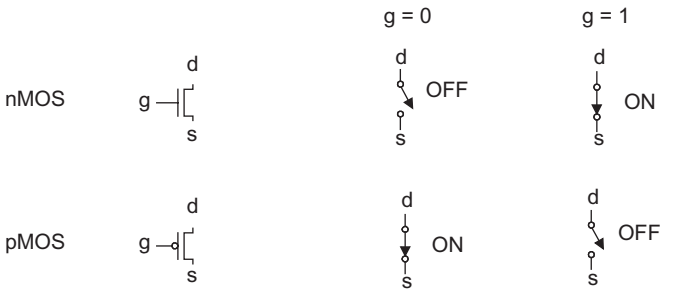

FIGURE 1.10 Transistor symbols and switch-level models

1.4 CMOS Logic

1.4.1 The Inverter

Figure 1.11 shows the schematic and symbol for a CMOS inverter or NOT gate using one nMOS transistor and one pMOS transistor. The 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 Y is pulled down to '0.' This is summarized in Table 1.1.

TABLE 1.1 Inverter truth table

A	Y
0	1
1	0

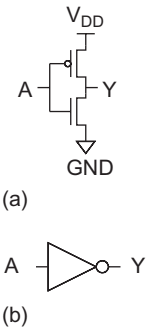

FIGURE 1.11 Inverter schematic (a) and symbol (b) $Y = \overline{A}$

1.4.2 The NAND Gate

Figure 1.12(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 1.12(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.

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

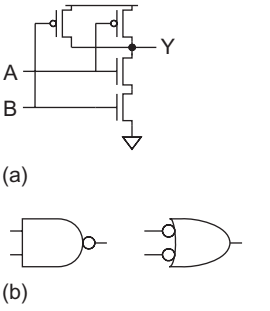

FIGURE 1.12 2-input NAND gate schematic (a) and symbol (b) $Y = \overline{A \cdot B}$

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 1.13. 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.

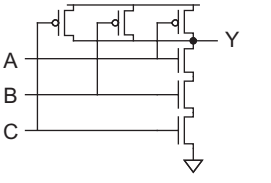

FIGURE 1.13 3-input NAND gate schematic $Y = \overline{A \cdot B \cdot C}$

1.4.3 CMOS Logic Gates

The inverter and NAND gates are examples of *static CMOS logic gates*, also called *complementary CMOS gates*. In general, a static 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 1.14. The networks are arranged such that one is ON and the other OFF for any input pattern.

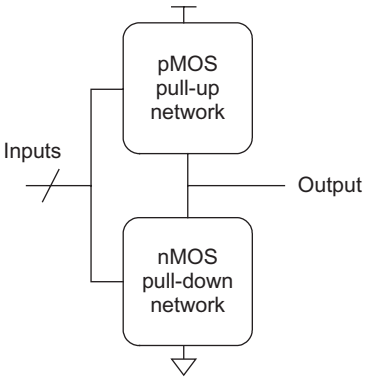

FIGURE 1.14 General logic gate using pull-up and pull-down networks

The pull-up and pull-down networks in the inverter each consist of a single transistor. The NAND gate uses 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 1.15 for nMOS and pMOS transistor pairs. By using combinations of these constructions, CMOS combinational gates can be constructed. Although such static CMOS gates are most widely used, Chapter 9 explores alternate ways of building gates with transistors.

In general, when we join a pull-up network to a pull-down network to form a logic gate as shown in Figure 1.14, 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 tristate bus drivers. The *crowbarred* (or *contention*) X level exists when both pull-up and pull-down are simultaneously turned ON. Contention between the two networks results in an indeterminate output level and dissipates static power. It is usually an unwanted condition.

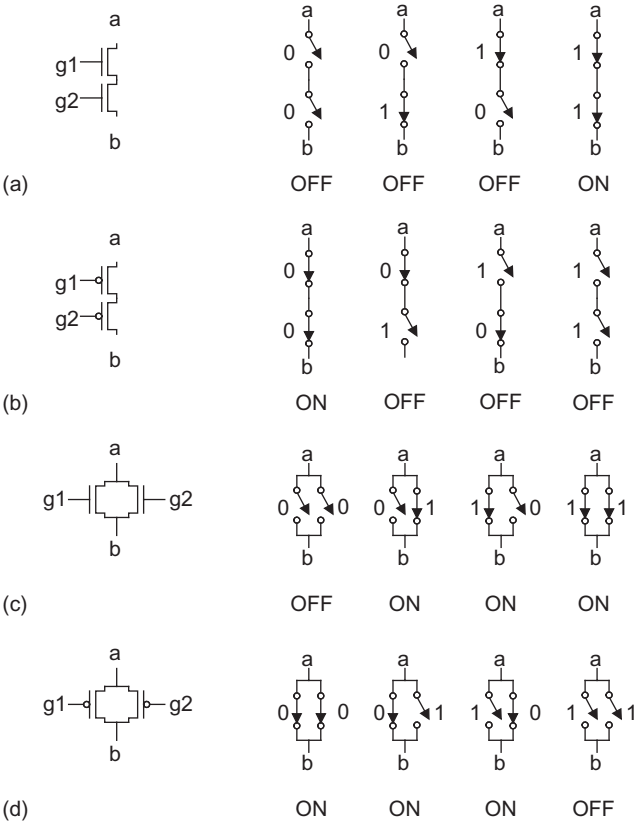

FIGURE 1.15 Connection and behavior of series and parallel transistors

TABLE 1.3 Output states of CMOS logic gates

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

1.4.4 The NOR Gate

A 2-input NOR gate is shown in Figure 1.16. 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 in Table 1.4. The output is never 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

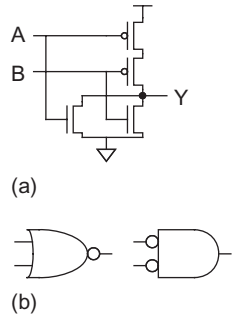

FIGURE 1.16 2-input NOR gate schematic (a) and symbol (b) $Y = \overline{A + B}$

Example 1.1

Sketch a 3-input CMOS NOR gate.

SOLUTION: Figure 1.17 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.

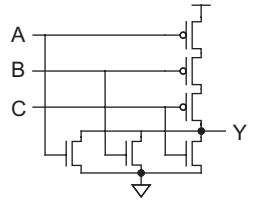

FIGURE 1.17 3-input NOR gate schematic $Y = \overline{A + B + C}$

1.4.5 Compound Gates

A *compound gate* performing a more complex logic function in a single stage of logic is formed by using a combination of series and parallel switch structures. For example, the derivation of the circuit for the function $Y = \overline{(A \cdot B) + (C \cdot D)}$ is shown in Figure 1.18. 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 1.18(a). Now ORing the result requires the parallel connection of these two structures, which is shown in Figure 1.18(b). For the pMOS pull-up network, we must compute the complementary expression using switches that turn on with inverted 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 been used 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 1.18(c) and Figure 1.18(d). Putting the networks together yields the full schematic (Figure 1.18(e)). The symbol is shown in Figure 1.18(f).

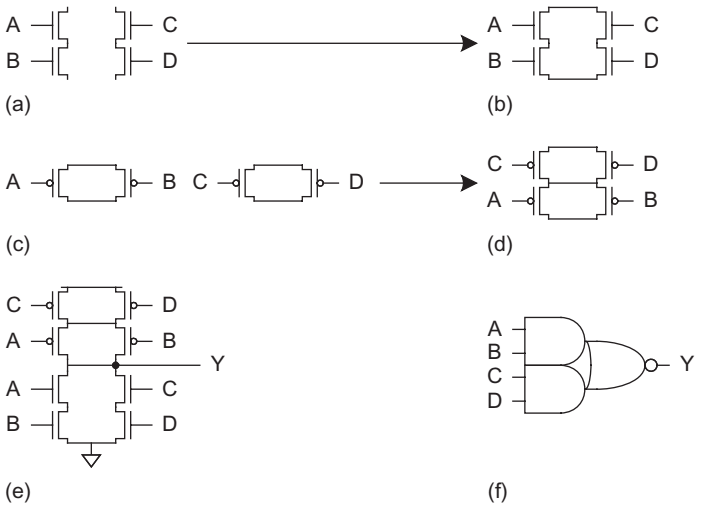

FIGURE 1.18 CMOS compound gate for function $Y = \overline{(A \cdot B) + (C \cdot D)}$

This AOI22 gate can be used as a 2-input inverting multiplexer by connecting $C = \overline{A}$ as a select signal. Then, $Y = \overline{B}$ if C is 0, while $Y = \overline{D}$ if C is 1. Section 1.4.8 shows a way to improve this multiplexer design.

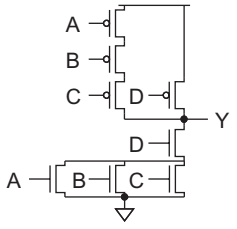

FIGURE 1.19 CMOS compound gate for function $Y = \overline{(A + B + C) \cdot D}$

Example 1.2

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

SOLUTION: Figure 1.19 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.

1.4.6 Pass Transistors and Transmission Gates

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 1s and 0s.

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 2.5.4. We say it passes a *degraded* or *weak* 1. A pMOS transistor again has the opposite behavior, passing strong 1s but degraded 0s. The transistor symbols and behaviors are summarized in Figure 1.20 with *g*, *s*, and *d* indicating gate, source, and drain.

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 1.21(a)), we obtain a switch that turns on when a 1 is applied to *g* (Figure 1.21(b)) in which 0s and 1s are both passed in an acceptable fashion (Figure 1.21(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.

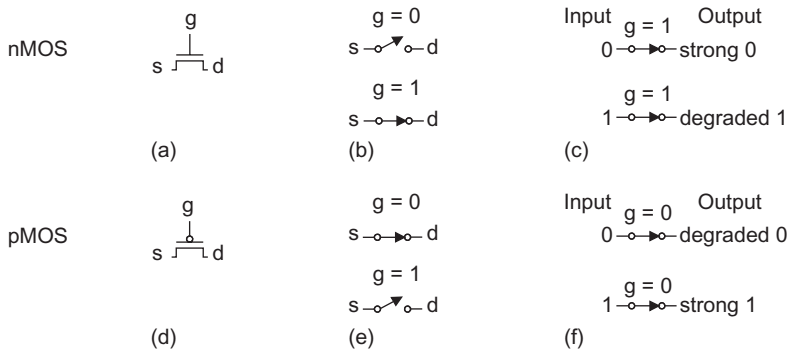

FIGURE 1.20 Pass transistor strong and degraded outputs

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 1.21(d).[1] None are easier to draw than the simple schematic, so we will use the schematic version to represent a transmission gate in this book.

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 1.14. Thus, the nMOS transistors only need to pass 0s and the pMOS only pass 1s, 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, static 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 find in subsequent chapters, this is the basis for the low static power dissipation in CMOS.

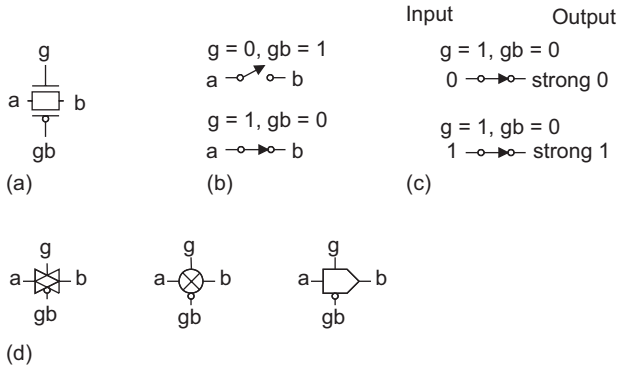

FIGURE 1.21 Transmission gate

[1]We 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.

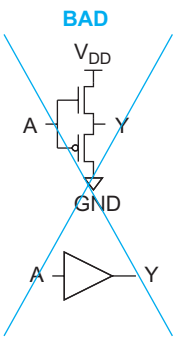

FIGURE 1.22
Bad noninverting buffer

A consequence of the design of static 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 1.22 shows a noninverting buffer. Unfortunately, now both the nMOS and pMOS transistors produce degraded outputs, so the technique should be avoided. Instead, we can build noninverting functions from multiple stages of inverting gates. Figure 1.23 shows several ways to build a 4-input AND gate from two levels of inverting static CMOS gates. Each design has different speed, size, and power trade-offs.

Similarly, the compound gate of Figure 1.18 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, as shown in Figure 1.24, using a total of 20 transistors, as compared to eight in Figure 1.18. Good CMOS logic designers exploit the efficiencies of compound gates rather than using large numbers of AND/OR gates.

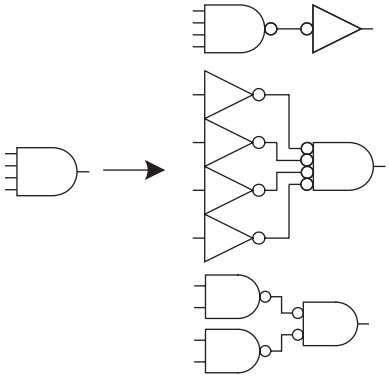

FIGURE 1.23 Various implementations of a CMOS 4-input AND gate

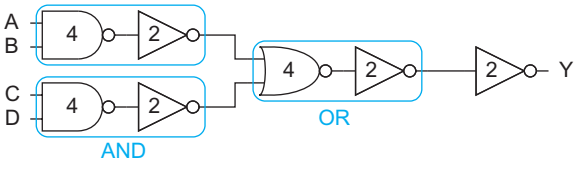

FIGURE 1.24 Inefficient discrete gate implementation of AOI22 with transistor counts indicated

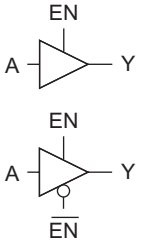

FIGURE 1.25
Tristate buffer symbol

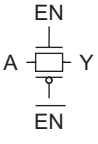

FIGURE 1.26
Transmission gate

1.4.7 Tristates

Figure 1.25 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.

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

The transmission gate in Figure 1.26 has the same truth table as a tristate buffer. It only requires two transistors but it is a *nonrestoring* circuit. If the input is noisy or otherwise degraded, the output will receive the same noise. We will see in Section 4.4.2 that the delay of a series of nonrestoring gates increases rapidly with the number of gates.

Figure 1.27(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 allows the output to float under certain input combinations. When EN is 0 (Figure 1.27(b)), both enable transistors are OFF, leaving the output floating. When EN is 1 (Figure 1.27(c)), both enable transistors are ON. They are conceptually removed from the circuit, leaving a simple inverter. Figure 1.27(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 an ordinary inverter followed by a tristate inverter.

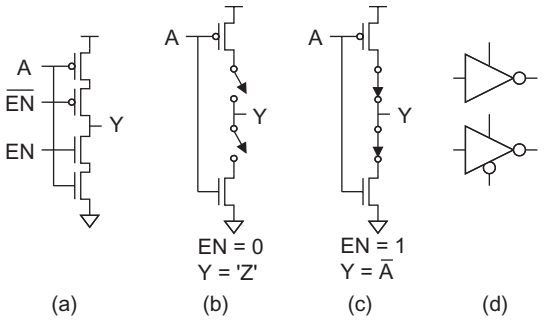

FIGURE 1.27 Tristate Inverter

Tristates were once commonly used to allow multiple units to drive a common bus, as long as exactly one unit is enabled at a time. If multiple units drive the bus, contention occurs and power is wasted. If no units drive the bus, it can float to an invalid logic level that causes the receivers to waste power. Moreover, it can be difficult to switch enable signals at exactly the same time when they are distributed across a large chip. Delay between different enables switching can cause contention. Given these problems, multiplexers are now preferred over tristate busses.

1.4.8 Multiplexers

Multiplexers are key components in CMOS memory elements and data manipulation structures. A *multiplexer* chooses the output from among several inputs based on a select signal. A 2-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 $Y = \overline{S} \cdot D0 + S \cdot D1$.

TABLE 1.6 Multiplexer truth table

S / $\overline{S}$	$D1$	$D0$	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 1.28(a). The select and its complement enable exactly one of the two transmission gates at any given time. The complementary select $\overline{S}$ is often not drawn in the symbol, as shown in Figure 1.28(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 1.18(e), connected as shown in Figure 1.29(a). Another is to gang together two tristate inverters, as shown in Figure 1.29(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 1.29(b). This is possible because the select and its complement are mutually exclusive. The tristate approach is slightly more compact and faster because it

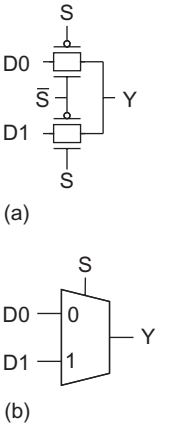

FIGURE 1.28 Transmission gate multiplexer