

FUTURE VISION BIE

One Stop for All Study Materials
& Lab Programs



Future Vision

By K B Hemanth Raj

Scan the QR Code to Visit the Web Page



Or

Visit : <https://hemanthrajhemu.github.io>

Gain Access to All Study Materials according to VTU,
CSE – Computer Science Engineering,
ISE – Information Science Engineering,
ECE - Electronics and Communication Engineering
& MORE...

Join Telegram to get Instant Updates: https://bit.ly/VTU_TELEGRAM

Contact: MAIL: futurevisionbie@gmail.com

INSTAGRAM: www.instagram.com/hemanthraj_hemu/

INSTAGRAM: www.instagram.com/futurevisionbie/

WHATSAPP SHARE: <https://bit.ly/FVBIESHARE>

CMOS VLSI Design

A Circuits and Systems Perspective

Fourth Edition

Neil H. E. Weste

*Macquarie University and
The University of Adelaide*

David Money Harris

Harvey Mudd College

Addison-Wesley

Boston Columbus Indianapolis New York San Francisco Upper Saddle River
Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montreal Toronto
Delhi Mexico City Sao Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo

<https://hemanthrajhemu.github.io>

Contents

Preface xxv

Chapter 1 Introduction

1.1	A Brief History	1
1.2	Preview	6
1.3	MOS Transistors	6
1.4	CMOS Logic	9
1.4.1	The Inverter	9
1.4.2	The NAND Gate	9
1.4.3	CMOS Logic Gates	9
1.4.4	The NOR Gate	11
1.4.5	Compound Gates	11
1.4.6	Pass Transistors and Transmission Gates	12
1.4.7	Tristates	14
1.4.8	Multiplexers	15
1.4.9	Sequential Circuits	16
1.5	CMOS Fabrication and Layout	19
1.5.1	Inverter Cross-Section	19
1.5.2	Fabrication Process	20
1.5.3	Layout Design Rules	24
1.5.4	Gate Layouts	27
1.5.5	Stick Diagrams	28
1.6	Design Partitioning	29
1.6.1	Design Abstractions	30
1.6.2	Structured Design	31
1.6.3	Behavioral, Structural, and Physical Domains	31
1.7	Example: A Simple MIPS Microprocessor	33
1.7.1	MIPS Architecture	33
1.7.2	Multicycle MIPS Microarchitecture	34
1.8	Logic Design	38
1.8.1	Top-Level Interfaces	38
1.8.2	Block Diagrams	38
1.8.3	Hierarchy	40
1.8.4	Hardware Description Languages	40
1.9	Circuit Design	42

1.10	Physical Design	45
1.10.1	Floorplanning	45
1.10.2	Standard Cells	48
1.10.3	Pitch Matching	50
1.10.4	Slice Plans	50
1.10.5	Arrays	51
1.10.6	Area Estimation	51
1.11	Design Verification	53
1.12	Fabrication, Packaging, and Testing	54
	Summary and a Look Ahead	55
	Exercises	57

Chapter 2 MOS Transistor Theory

2.1	Introduction	61
2.2	Long-Channel I-V Characteristics	64
2.3	C-V Characteristics	68
2.3.1	Simple MOS Capacitance Models	68
2.3.2	Detailed MOS Gate Capacitance Model	70
2.3.3	Detailed MOS Diffusion Capacitance Model	72
2.4	Nonideal I-V Effects	74
2.4.1	Mobility Degradation and Velocity Saturation	75
2.4.2	Channel Length Modulation	78
2.4.3	Threshold Voltage Effects	79
2.4.4	Leakage	80
2.4.5	Temperature Dependence	85
2.4.6	Geometry Dependence	86
2.4.7	Summary	86
2.5	DC Transfer Characteristics	87
2.5.1	Static CMOS Inverter DC Characteristics	88
2.5.2	Beta Ratio Effects	90
2.5.3	Noise Margin	91
2.5.4	Pass Transistor DC Characteristics	92
2.6	Pitfalls and Fallacies	93
	Summary	94
	Exercises	95

Chapter 3 CMOS Processing Technology

3.1	Introduction	99
3.2	CMOS Technologies	100
3.2.1	Wafer Formation	100
3.2.2	Photolithography	101

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 2008, Intel's Itanium microprocessor contained more than 2 billion transistors and a 16 Gb Flash memory contained more than 4 billion transistors. This corresponds to a compound annual growth rate of 53% over 50 years. No other technology in history has sustained such a high growth rate lasting for so long.

This incredible growth has come from steady miniaturization of transistors and improvements in manufacturing processes. Most other fields of engineering involve trade-offs 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 not only revolutionized electronics, but also society at large.

The processing performance once dedicated to secret government supercomputers is now available in disposable cellular telephones. The memory once needed for an entire company's accounting system is now carried by a teenager in her iPod. Improvements in integrated circuits have enabled space exploration, made automobiles safer and more fuel-efficient, revolutionized the nature of warfare, brought much of mankind's knowledge to our Web browsers, and made the world a flatter place.

Figure 1.1 shows annual sales in the worldwide semiconductor market. Integrated circuits became a \$100 billion/year business in 1994. In 2007, the industry manufactured approximately 6 quintillion (6×10^{18}) transistors, or nearly a billion 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 those ideas to reality.

During the first half of the twentieth 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 introduced the device 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 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.

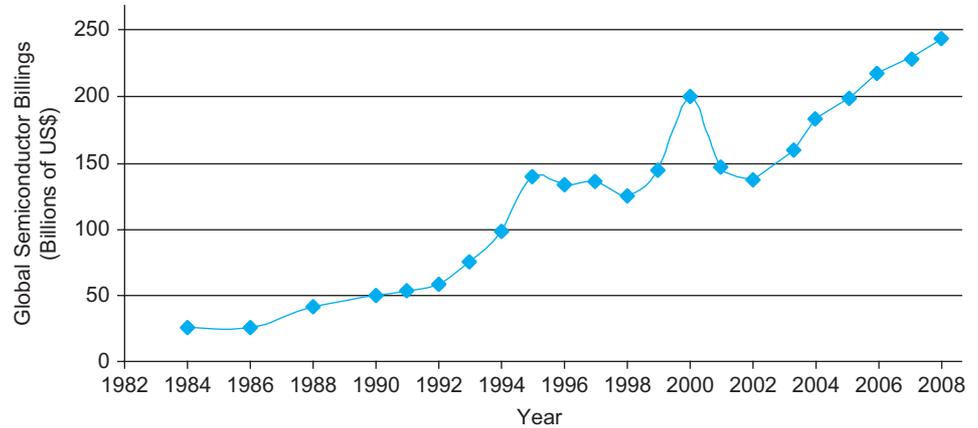
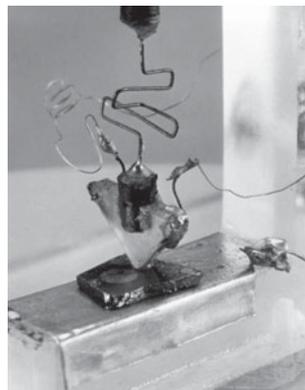


FIGURE 1.1 Size of worldwide semiconductor market (Courtesy of Semiconductor Industry Association.)

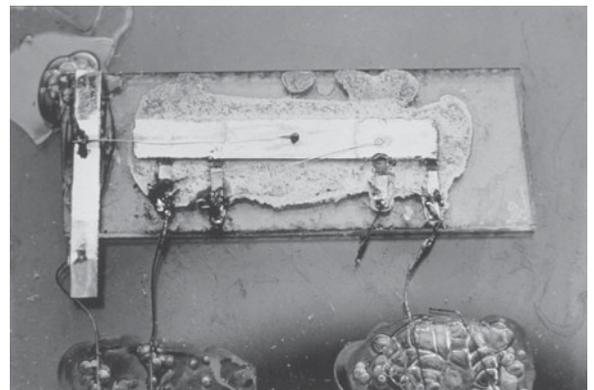
Ten years later, Jack Kilby at Texas Instruments realized the potential for miniaturization if multiple transistors could be built on one piece of silicon. Figure 1.2(b) shows his first prototype of an integrated circuit, constructed from a germanium slice and gold wires.

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

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 or current applied to the control. 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. 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, drawn even when the circuit is not switching,



(a)



(b)

FIGURE 1.2 (a) First transistor (Property of AT&T Archives. Reprinted with permission of AT&T.) and (b) first integrated circuit (Courtesy of Texas Instruments.)

limits the maximum number of transistors that can be integrated onto a single die. By the 1960s, Metal Oxide Semiconductor Field Effect Transistors (MOSFETs) began to enter production. 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 silicon, respectively. The original idea of field effect transistors dated back to the German scientist Julius Lilienfeld in 1925 [US patent 1,745,175] 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.

In 1963, Frank Wanlass at Fairchild described the first logic gates using MOSFETs [Wanlass63]. Fairchild's 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 commercial processes used only pMOS transistors and suffered from poor performance, yield, and reliability. Processes using nMOS transistors became common in the 1970s [Mead80]. Intel pioneered nMOS technology with its 1101 256-bit static random access memory and 4004 4-bit microprocessor, as 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 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.

In 1965, Gordon Moore observed 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. Moore's Law is driven primarily by *scaling* down the size of transistors and, to a minor extent, by building larger chips. 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

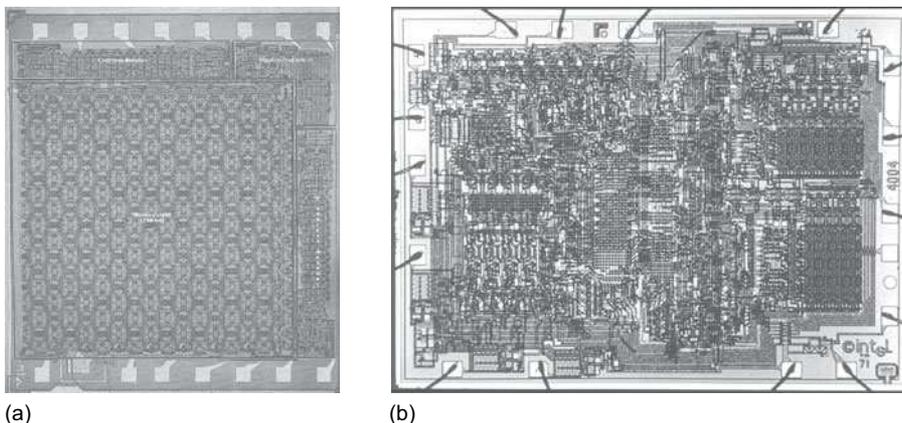


FIGURE 1.3 (a) Intel 1101 SRAM (© IEEE 1969 [Vadasz69]) and (b) 4004 microprocessor (Reprinted with permission of Intel Corporation.)

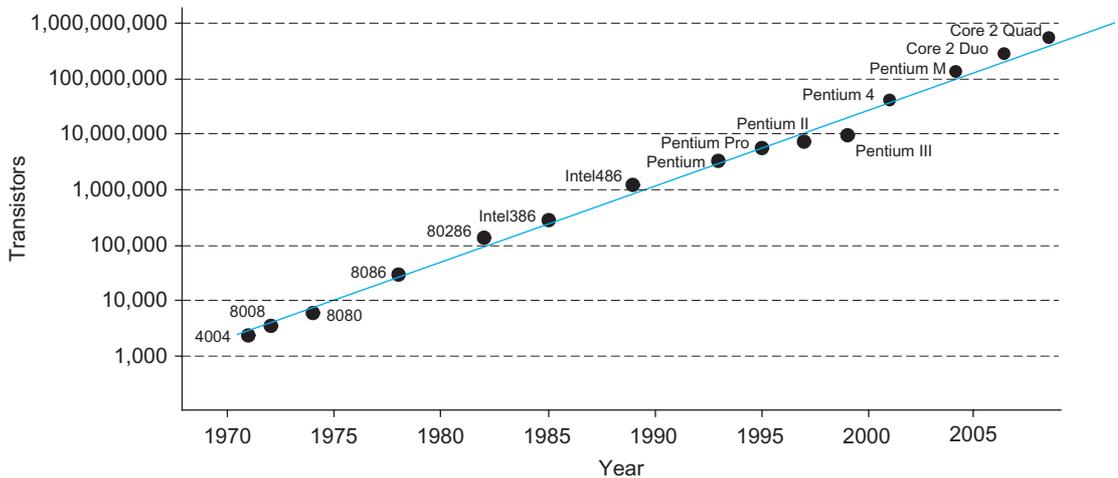


FIGURE 1.4 Transistors in Intel microprocessors [Intel10]

gates, with 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 *Dennard's Scaling Law* [Dennard74]: as transistors shrink, they become faster, consume less power, and are cheaper to manufacture. Figure 1.5 shows that Intel microprocessor clock frequencies have doubled roughly every 34 months. This frequency scaling hit the power wall around 2004, and clock frequencies have leveled off around 3 GHz. Computer performance, measured in time to run an application, has advanced even more than raw clock speed. Presently, the performance is driven by the number of cores on a chip rather than by the clock. Even though an individual CMOS transistor uses very little energy each time it switches, the enormous number of transistors switching at very high rates of speed have made power consumption a major design consideration again. Moreover, as transistors have become so small, they cease to turn completely OFF. Small amounts of current leaking through each transistor now lead to significant power consumption when multiplied by millions or billions of transistors on a chip.

The feature size of a CMOS manufacturing process refers to the minimum dimension of a transistor that can be reliably built. The 4004 had a feature size of $10\ \mu\text{m}$ in 1971. The Core 2 Duo had a feature size of 45 nm in 2008. Manufacturers introduce a new process generation (also called a technology node) every 2–3 years with a 30% smaller feature size to pack twice as many transistors in the same area. Figure 1.6 shows the progression of process generations. Feature sizes down to $0.25\ \mu\text{m}$ are generally specified in microns ($10^{-6}\ \text{m}$), while smaller feature sizes are expressed in nanometers ($10^{-9}\ \text{m}$). Effects that were relatively minor in micron processes, such as transistor leakage, variations in characteristics of adjacent transistors, and wire resistance, are of great significance in nanometer processes.

Moore's Law has become a self-fulfilling prophecy because each company must keep up with its competitors. Obviously, this scaling cannot go on forever because transistors cannot be smaller than atoms. Dennard scaling has already begun to slow. By the 45 nm

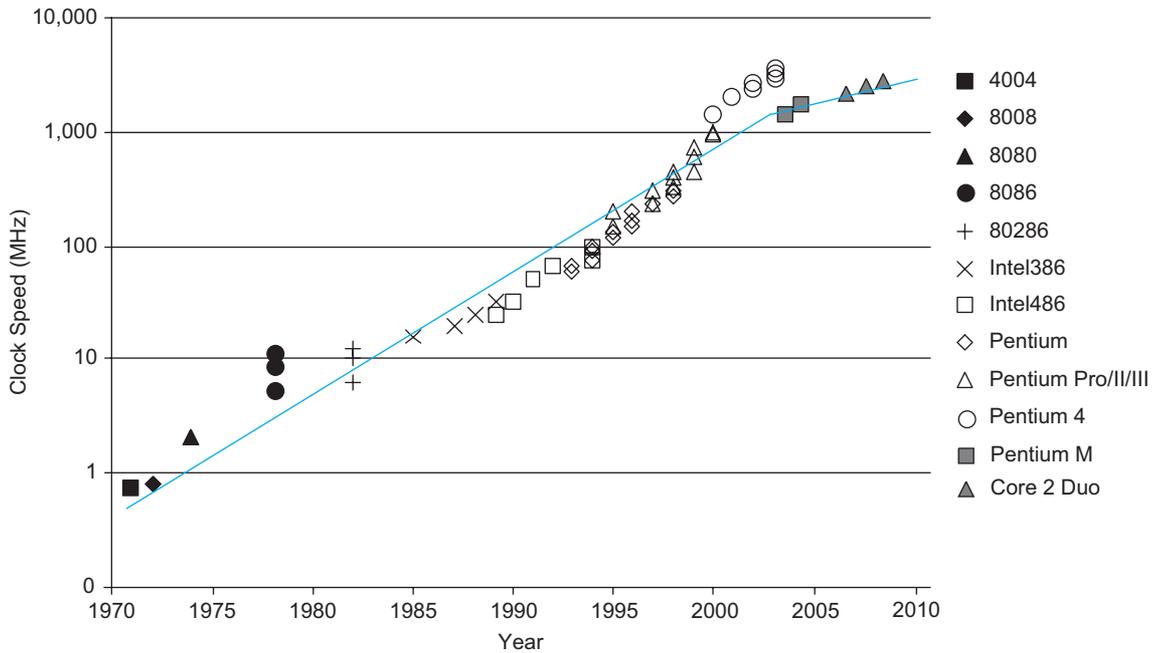


FIGURE 1.5 Clock frequencies of Intel microprocessors

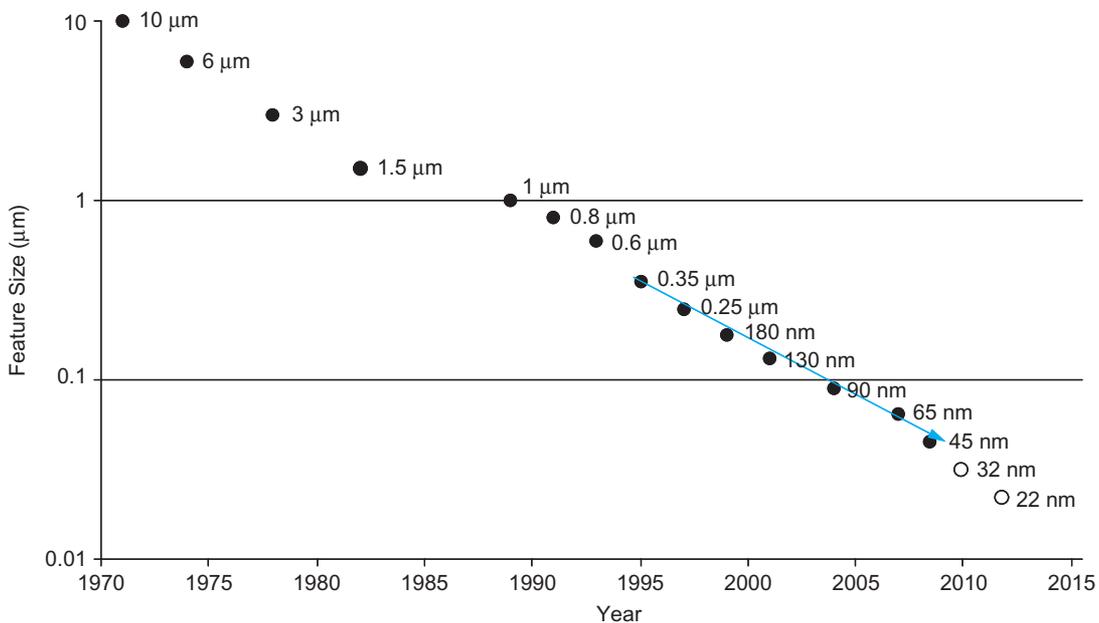


FIGURE 1.6 Process generations. Future predictions from [SIA2007].

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

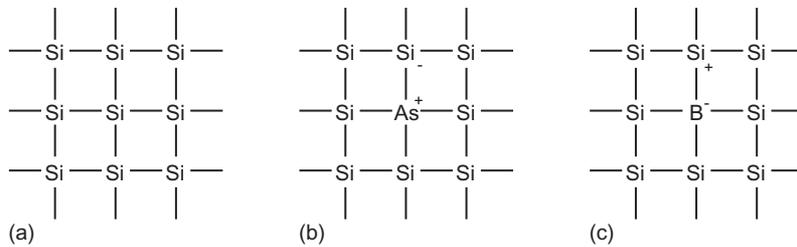


FIGURE 1.7 Silicon lattice and dopant atoms

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.

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 (*MOSFETs*) or simply *FETs*. 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.

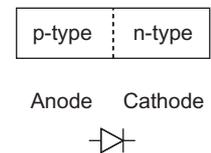


FIGURE 1.8 *p*-*n* junction diode structure and symbol

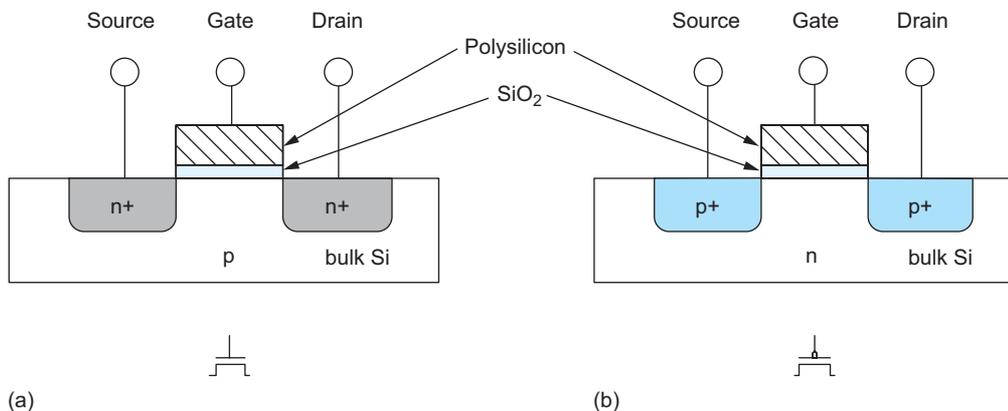


FIGURE 1.9 *nMOS* transistor (a) and *pMOS* transistor (b)

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–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 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–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.

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.

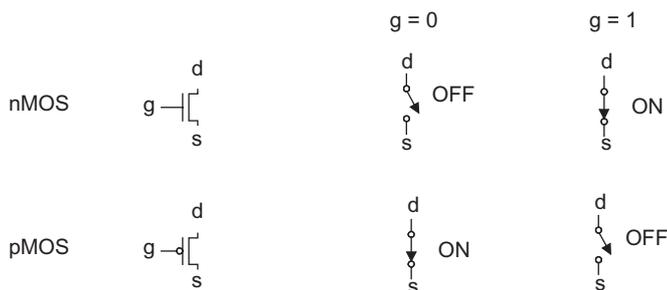


FIGURE 1.10 Transistor symbols and switch-level models

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

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.

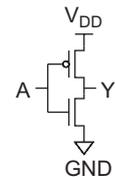
TABLE 1.2 NAND gate truth table

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

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.

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.



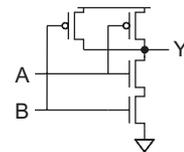
(a)



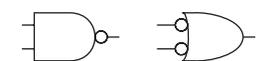
(b)

FIGURE 1.11

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



(a)



(b)

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

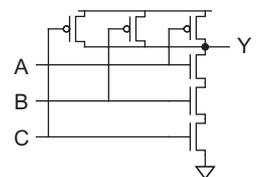


FIGURE 1.13 3-input NAND gate schematic $Y = \bar{A} \cdot \bar{B} \cdot \bar{C}$

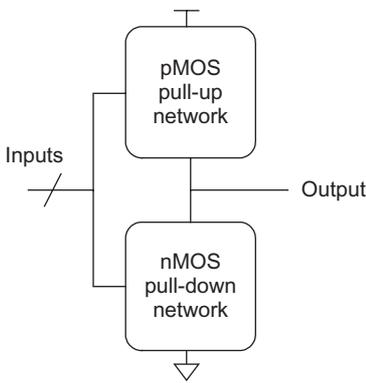


FIGURE 1.14 General logic gate using pull-up and pull-down networks

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.

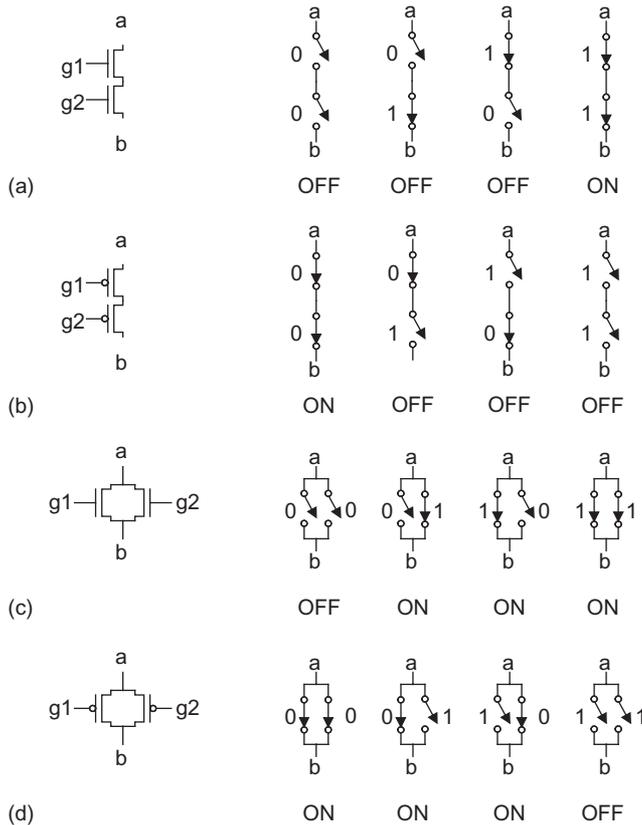


FIGURE 1.15 Connection and behavior of series and parallel transistors

TABLE 1.3 Output states of CMOS logic gates

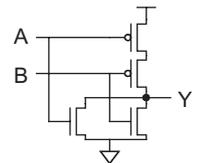
	pull-up OFF	pull-up ON
pull-down OFF	Z	1
pull-down ON	0	crowbarred (X)

1.4.4 The NOR Gate

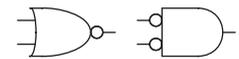
A 2-input NOR gate is shown in Figure 1.16. The nMOS transistors are in parallel to pull the output low when either input is high. The pMOS transistors are in series to pull the output high when both inputs are low, as indicated in Table 1.4. The output is never crowbarred or left floating.

TABLE 1.4 NOR gate truth table

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0



(a)



(b)

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

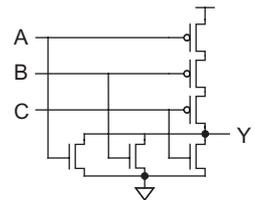


FIGURE 1.17 3-input NOR gate schematic $Y = \overline{A + B + C}$

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.

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).

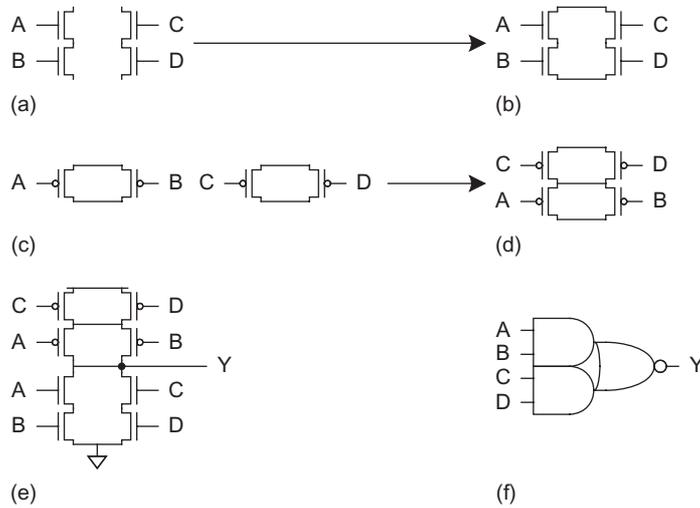


FIGURE 1.18 CMOS compound gate for function $Y = \overline{(A \cdot B) + (C \cdot D)}$

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.

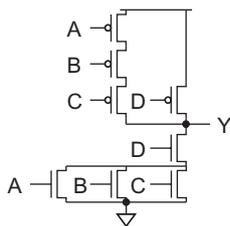


FIGURE 1.19 CMOS compound gate for function $Y = \overline{(A + B + C) \cdot D}$

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.

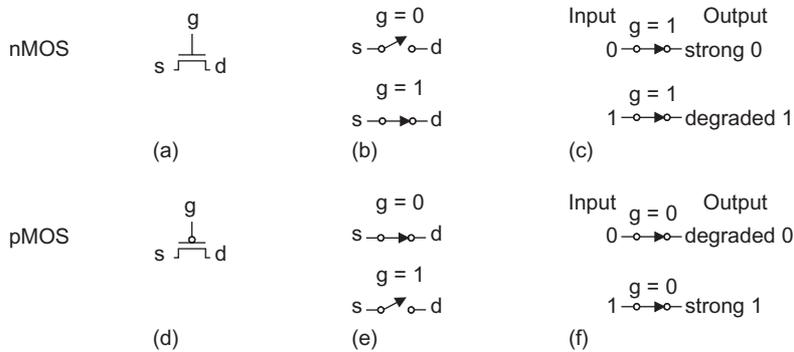


FIGURE 1.20 Pass transistor strong and degraded outputs

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.

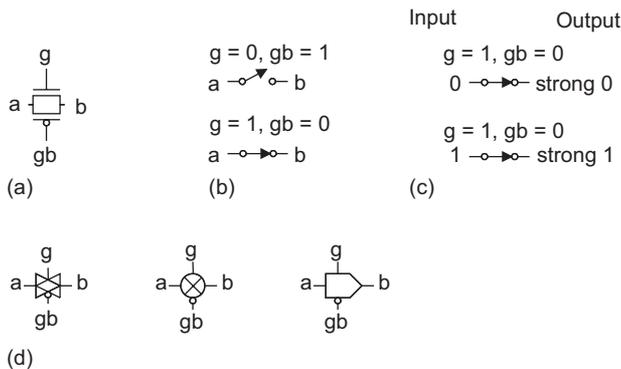


FIGURE 1.21 Transmission gate

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.

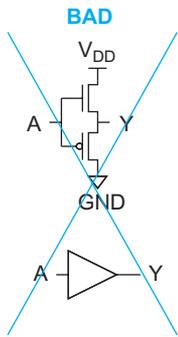


FIGURE 1.22 Bad noninverting buffer

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.

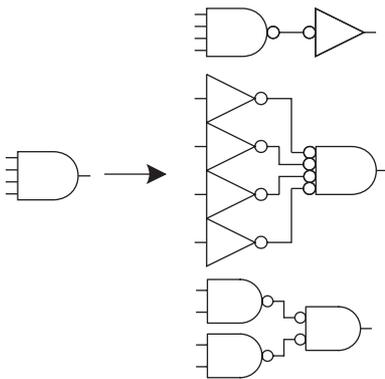


FIGURE 1.23 Various implementations of a CMOS 4-input AND gate

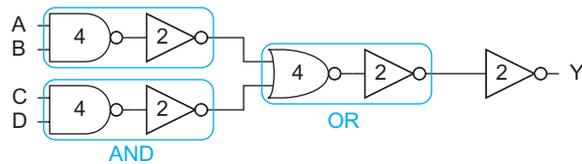


FIGURE 1.24 Inefficient discrete gate implementation of AOI22 with transistor counts indicated

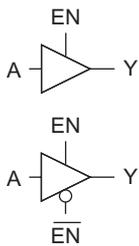


FIGURE 1.25 Tristate buffer symbol

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

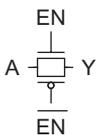


FIGURE 1.26 Transmission gate

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.

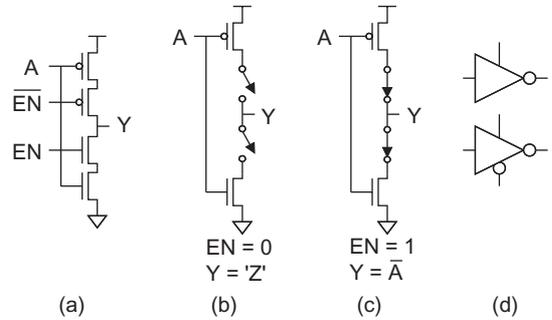


FIGURE 1.27 Tristate Inverter

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

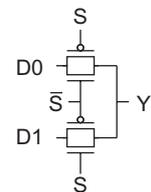
Multiplexers are key components in CMOS memory elements and data manipulation structures. A *multiplexer* chooses the output from among several inputs based on a select signal. A 2-input, or 2:1 multiplexer, chooses input $D0$ when the select is 0 and input $D1$ when the select is 1. The truth table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

TABLE 1.6 Multiplexer truth table

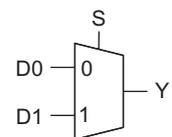
S / \bar{S}	$D1$	$D0$	Y
0 / 1	X	0	0
0 / 1	X	1	1
1 / 0	0	X	0
1 / 0	1	X	1

Two transmission gates can be tied together to form a compact 2-input multiplexer, as shown in Figure 1.28(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 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



(a)



(b)

FIGURE 1.28 Transmission gate multiplexer

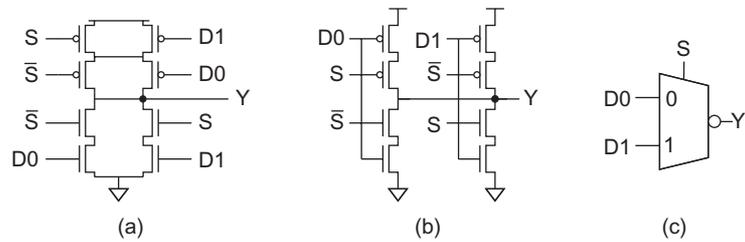


FIGURE 1.29 Inverting multiplexer

requires less internal wire. Again, if the complementary select is generated within the cell, it is omitted from the symbol (Figure 1.29(c)).

Larger multiplexers can be built from multiple 2-input multiplexers or by directly ganging together several tristates. The latter approach requires decoded enable signals for each tristate; the enables should switch simultaneously to prevent contention. 4-input (4:1) multiplexers using each of these approaches are shown in Figure 1.30. In practice, both inverting and noninverting multiplexers are simply called multiplexers or muxes.

1.4.9 Sequential Circuits

So far, we have considered *combinational circuits*, whose outputs depend only on the current inputs. *Sequential circuits* have memory: their outputs depend on both current and previous inputs. Using the combinational circuits developed so far, we can now build sequential circuits such as latches and flip-flops. These elements receive a clock, CLK , and a data input, D , and produce an output, Q . A D latch is *transparent* when $CLK = 1$, meaning that Q follows D . It becomes *opaque* when $CLK = 0$, meaning Q retains its previous value and ignores changes in D . An *edge-triggered flip-flop* copies D to Q on the rising edge of CLK and remembers its old value at other times.

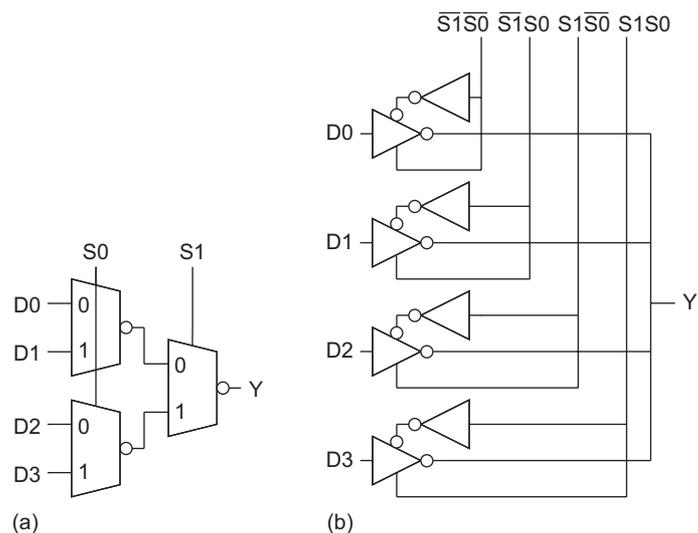


FIGURE 1.30 4:1 multiplexer

1.4.9.1 Latches A D latch built from a 2-input multiplexer and two inverters is shown in Figure 1.31(a). The multiplexer can be built from a pair of transmission gates, shown in Figure 1.31(b), because the inverters are restoring. This latch also produces a complementary output, \bar{Q} . When $CLK = 1$, the latch is transparent and D flows through to Q (Figure 1.31(c)). When CLK falls to 0, the latch becomes opaque. A feedback path around the inverter pair is established (Figure 1.31(d)) to hold the current state of Q indefinitely.

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 1.31(e). The latch shown is a positive-level-sensitive latch, represented by the symbol in Figure 1.31(f). By inverting the control connections to the multiplexer, the latch becomes negative-level-sensitive.

1.4.9.2 Flip-Flops By combining two level-sensitive latches, one negative-sensitive and one positive-sensitive, we construct the edge-triggered flip-flop shown in Figure 1.32(a–b). 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 1.32(c)). When the clock transitions from 0 to 1, the master latch becomes opaque and holds the D value at the time of the clock transition. The slave latch becomes transparent, 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 (Figure 1.32(d)). When the clock transitions from 1 to 0, the slave latch holds its value and the master starts sampling the input again.

While we have shown a transmission gate multiplexer as the input stage, good design practice would buffer the input and output with inverters, as shown in Figure 1.32(e), to

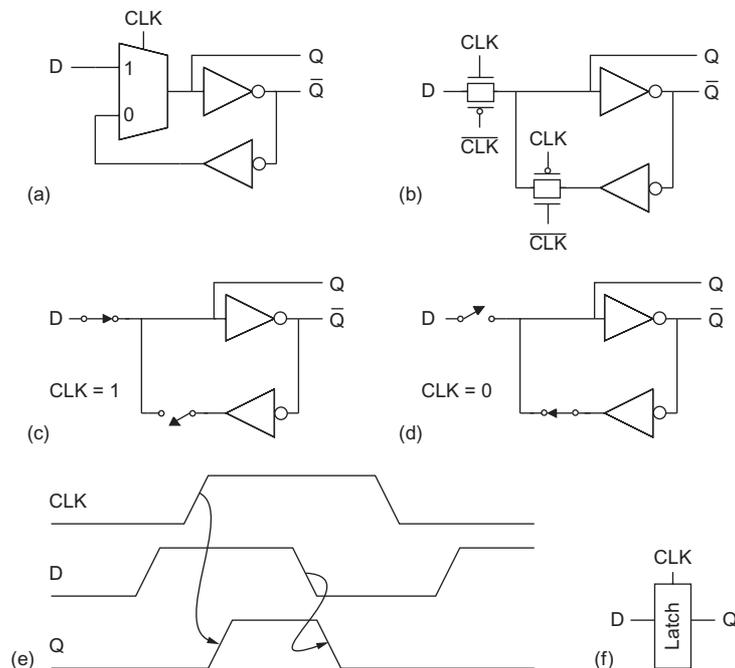


FIGURE 1.31 CMOS positive-level-sensitive D latch

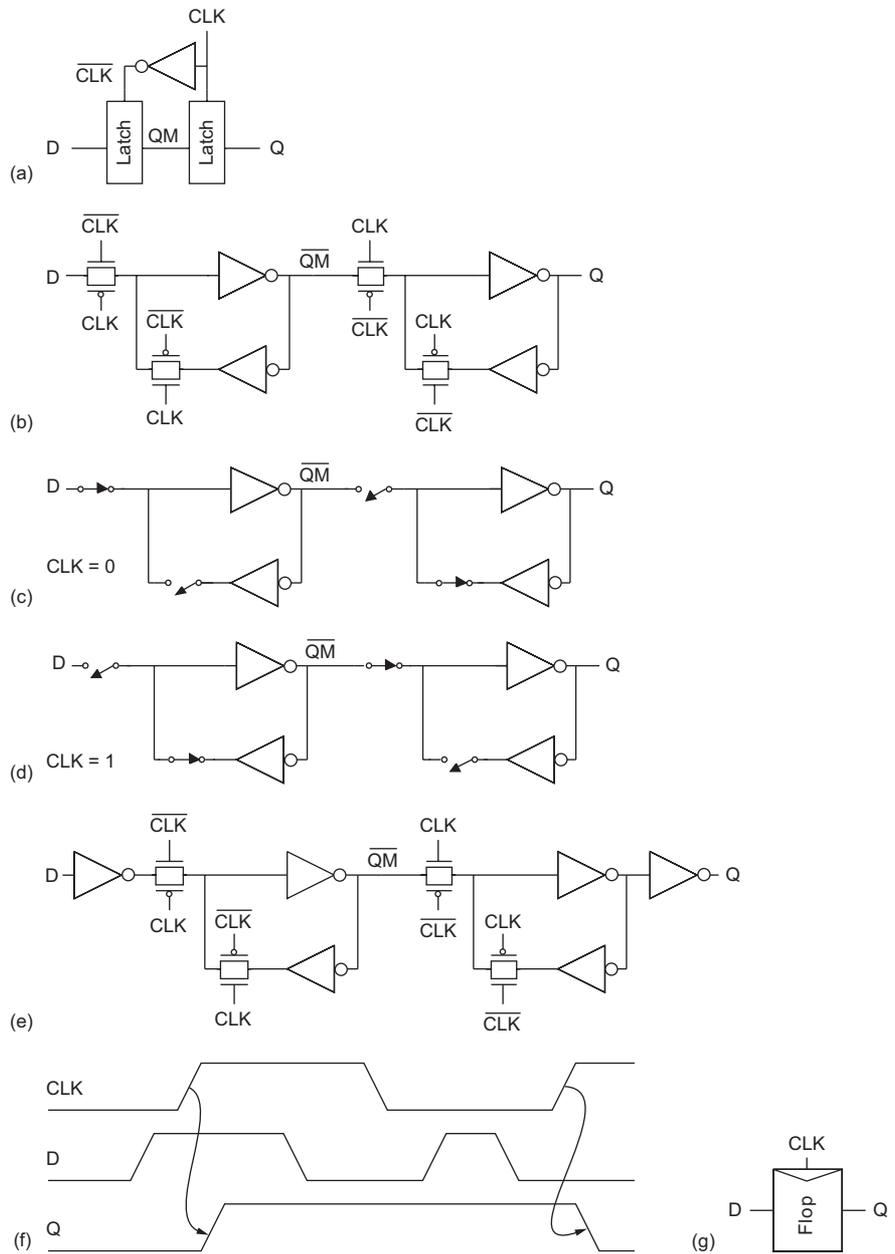


FIGURE 1.32 CMOS positive-edge-triggered *D* flip-flop

preserve what we call “modularity.” Modularity is explained further in Section 1.6.2 and robust latches and registers are discussed further in Section 10.3.

In summary, this flip-flop copies *D* to *Q* on the rising edge of the clock, as shown in Figure 1.32(f). 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 1.32(g) 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 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 10.2.5 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 class projects), hold-time problems can be avoided altogether by distributing a two-phase nonoverlapping clock. Figure 1.33 shows the flip-flop clocked with two nonoverlapping phases. As long as the phases never overlap, at least one latch will be opaque at any given time and hold-time problems cannot occur.

1.5 CMOS Fabrication and Layout

Now that we can design logic gates and registers from transistors, let us consider how the transistors are built. Designers need to understand the physical implementation of circuits because it has a major impact on performance, power, and cost.

Transistors are fabricated on thin silicon wafers that serve as both a mechanical support and an electrical common point called the *substrate*. We can examine the physical layout of transistors from two perspectives. One is the top view, obtained by looking down on a wafer. The other is the cross-section, obtained by slicing the wafer through the middle of a transistor and looking at it edgewise. We begin by looking at the cross-section of a complete CMOS inverter. We then look at the top view of the same inverter and define a set of masks used to manufacture the different parts of the inverter. The size of the transistors and wires is set by the mask dimensions and is limited by the resolution of the manufacturing process. Continual advancements in this resolution have fueled the exponential growth of the semiconductor industry.

1.5.1 Inverter Cross-Section

Figure 1.34 shows a cross-section and corresponding schematic of an inverter. (See the inside front cover for a color cross-section.) In this diagram, the inverter is built on a p-type substrate. The pMOS transistor requires an n-type body region, so an n-well is diffused into the substrate in its vicinity. As described in Section 1.3, the nMOS transistor

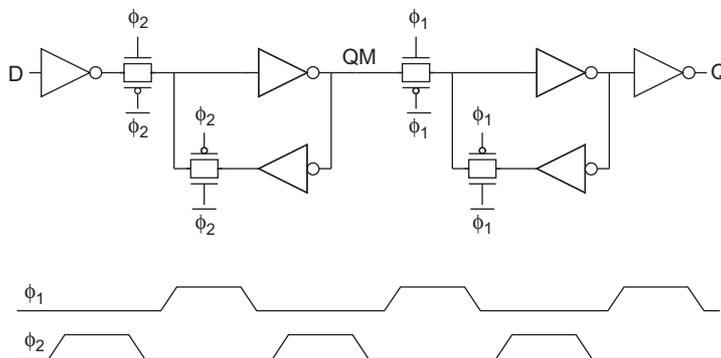


FIGURE 1.33 CMOS flip-flop with two-phase nonoverlapping clocks

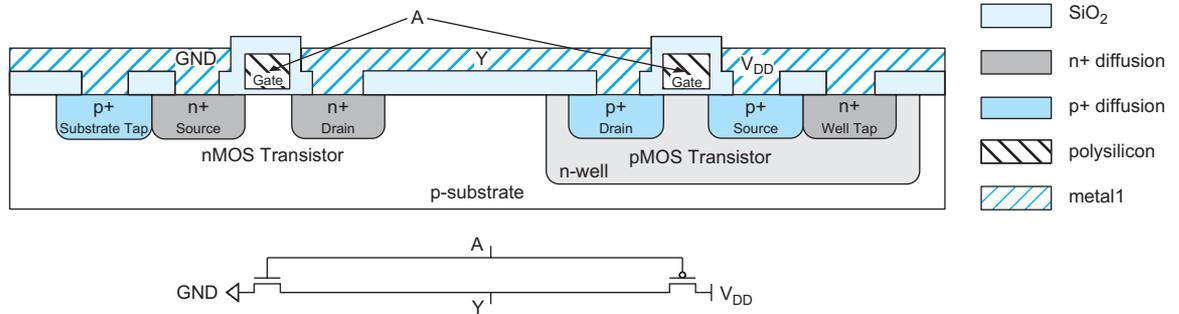


FIGURE 1.34 Inverter cross-section with well and substrate contacts. Color version on inside front cover.

has heavily doped n-type source and drain regions and a polysilicon gate over a thin layer of silicon dioxide (SiO_2 , also called *gate oxide*). n+ and p+ diffusion regions indicate heavily doped n-type and p-type silicon. The pMOS transistor is a similar structure with p-type source and drain regions. The polysilicon gates of the two transistors are tied together somewhere off the page and form the input A . The source of the nMOS transistor is connected to a metal ground line and the source of the pMOS transistor is connected to a metal V_{DD} line. The drains of the two transistors are connected with metal to form the output Y . A thick layer of SiO_2 called *field oxide* prevents metal from shorting to other layers except where contacts are explicitly etched.

A junction between metal and a lightly doped semiconductor forms a *Schottky diode* that only carries current in one direction. When the semiconductor is doped more heavily, it forms a good ohmic contact with metal that provides low resistance for bidirectional current flow. The substrate must be tied to a low potential to avoid forward-biasing the p-n junction between the p-type substrate and the n+ nMOS source or drain. Likewise, the n-well must be tied to a high potential. This is done by adding heavily doped substrate and well contacts, or *taps*, to connect GND and V_{DD} to the substrate and n-well, respectively.

1.5.2 Fabrication Process

For all their complexity, chips are amazingly inexpensive because all the transistors and wires can be printed in much the same way as books. The fabrication sequence consists of a series of steps in which layers of the chip are defined through a process called *photolithography*. Because a whole wafer full of chips is processed in each step, the cost of the chip is proportional to the chip area, rather than the number of transistors. As manufacturing advances allow engineers to build smaller transistors and thus fit more in the same area, each transistor gets cheaper. Smaller transistors are also faster because electrons don't have to travel as far to get from the source to the drain, and they consume less energy because fewer electrons are needed to charge up the gates! This explains the remarkable trend for computers and electronics to become cheaper and more capable with each generation.

The inverter could be defined by a hypothetical set of six masks: n-well, polysilicon, n+ diffusion, p+ diffusion, contacts, and metal (for fabrication reasons discussed in Chapter 3, the actual mask set tends to be more elaborate). Masks specify where the components will be manufactured on the chip. Figure 1.35(a) shows a top view of the six masks. (See also the inside front cover for a color picture.) The cross-section of the inverter from Figure 1.34 was taken along the dashed line. Take some time to convince yourself how the top view and cross-section relate; this is critical to understanding chip layout.

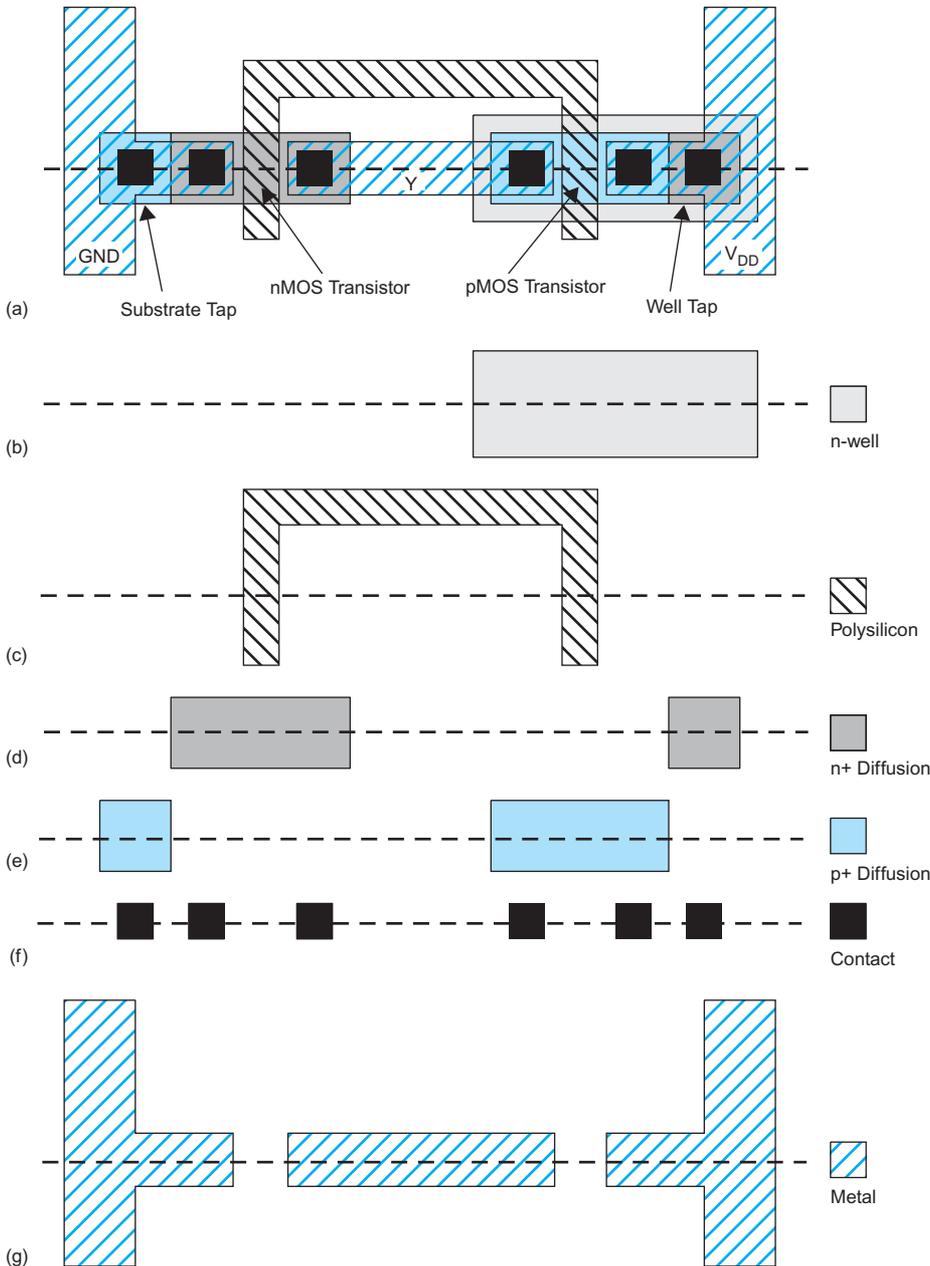


FIGURE 1.35 Inverter mask set. Color version on inside front cover.

Consider a simple fabrication process to illustrate the concept. The process begins with the creation of an n-well on a bare p-type silicon wafer. Figure 1.36 shows cross-sections of the wafer after each processing step involved in forming the n-well; Figure 1.36(a) illustrates the bare substrate before processing. Forming the n-well requires adding enough Group V dopants into the silicon substrate to change the substrate from p-type to n-type in the region of the well. To define what regions receive n-wells, we grow a protective layer of

oxide over the entire wafer, then remove it where we want the wells. We then add the n-type dopants; the dopants are blocked by the oxide, but enter the substrate and form the wells where there is no oxide. The next paragraph describes these steps in detail.

The wafer is first *oxidized* in a high-temperature (typically 900–1200 °C) furnace that causes Si and O₂ to react and become SiO₂ on the wafer surface (Figure 1.36(b)). The oxide must be *patterned* to define the n-well. An organic photoresist² that softens where

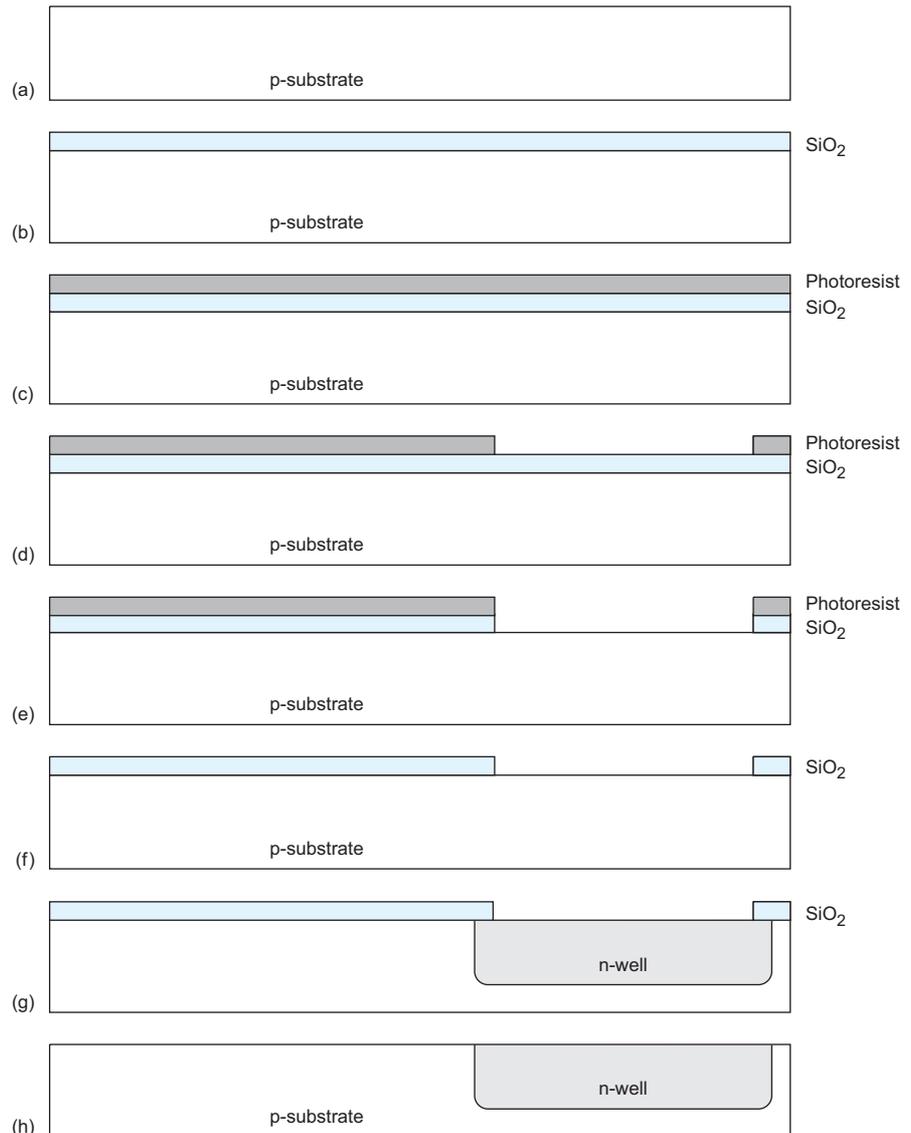


FIGURE 1.36 Cross-sections while manufacturing the n-well

²Engineers have experimented with many organic polymers for photoresists. In 1958, Brumford and Walker reported that Jello™ could be used for masking. They did extensive testing, observing that “various Jellos™ were evaluated with lemon giving the best result.”

exposed to light is spun onto the wafer (Figure 1.36(c)). The photoresist is exposed through the n-well mask (Figure 1.35(b)) that allows light to pass through only where the well should be. The softened photoresist is removed to expose the oxide (Figure 1.36(d)). The oxide is etched with hydrofluoric acid (HF) where it is not protected by the photoresist (Figure 1.36(e)), then the remaining photoresist is stripped away using a mixture of acids called *piranha etch* (Figure 1.36(f)). The well is formed where the substrate is not covered with oxide. Two ways to add dopants are diffusion and ion implantation. In the *diffusion* process, the wafer is placed in a furnace with a gas containing the dopants. When heated, dopant atoms diffuse into the substrate. Notice how the well is wider than the hole in the oxide on account of *lateral diffusion* (Figure 1.36(g)). With *ion implantation*, dopant ions are accelerated through an electric field and blasted into the substrate. In either method, the oxide layer prevents dopant atoms from entering the substrate where no well is intended. Finally, the remaining oxide is stripped with HF to leave the bare wafer with wells in the appropriate places.

The transistor gates are formed next. These consist of polycrystalline silicon, generally called *polysilicon*, over a thin layer of oxide. The thin oxide is grown in a furnace. Then the wafer is placed in a reactor with silane gas (SiH_4) and heated again to grow the polysilicon layer through a process called *chemical vapor deposition*. The polysilicon is heavily doped to form a reasonably good conductor. The resulting cross-section is shown in Figure 1.37(a). As before, the wafer is patterned with photoresist and the polysilicon mask (Figure 1.35(c)), leaving the polysilicon gates atop the thin gate oxide (Figure 1.37(b)).

The n⁺ regions are introduced for the transistor active area and the well contact. As with the well, a protective layer of oxide is formed (Figure 1.37(c)) and patterned with the n-diffusion mask (Figure 1.35(d)) to expose the areas where the dopants are needed (Figure 1.37(d)). Although the n⁺ regions in Figure 1.37(e) are typically formed with ion implantation, they were historically diffused and thus still are often called *n-diffusion*. Notice that the polysilicon gate over the nMOS transistor blocks the diffusion so the source and drain are separated by a channel under the gate. This is called a *self-aligned* process because the source and drain of the transistor are automatically formed adjacent to the gate without the need to precisely align the masks. Finally, the protective oxide is stripped (Figure 1.37(f)).

The process is repeated for the p-diffusion mask (Figure 1.35(e)) to give the structure of Figure 1.38(a). Oxide is used for masking in the same way, and thus is not shown. The field oxide is grown to insulate the wafer from metal and patterned with the contact mask (Figure 1.35(f)) to leave contact cuts where metal should attach to diffusion or polysilicon (Figure 1.38(b)). Finally, aluminum is sputtered over the entire wafer, filling the contact cuts as well. Sputtering involves blasting aluminum into a vapor that evenly coats the wafer. The metal is patterned with the metal mask (Figure 1.35(g)) and plasma etched to remove metal everywhere except where wires should remain (Figure 1.38(c)). This completes the simple fabrication process.

Modern fabrication sequences are more elaborate because they must create complex doping profiles around the channel of the transistor and print features that are smaller than the wavelength of the light being used in lithography. However, masks for these elaborations can be automatically generated from the simple set of masks we have just examined. Modern processes also have 5–10+ layers of metal, so the metal and contact steps must be repeated for each layer. Chip manufacturing has become a commodity, and many different foundries will build designs from a basic set of masks.

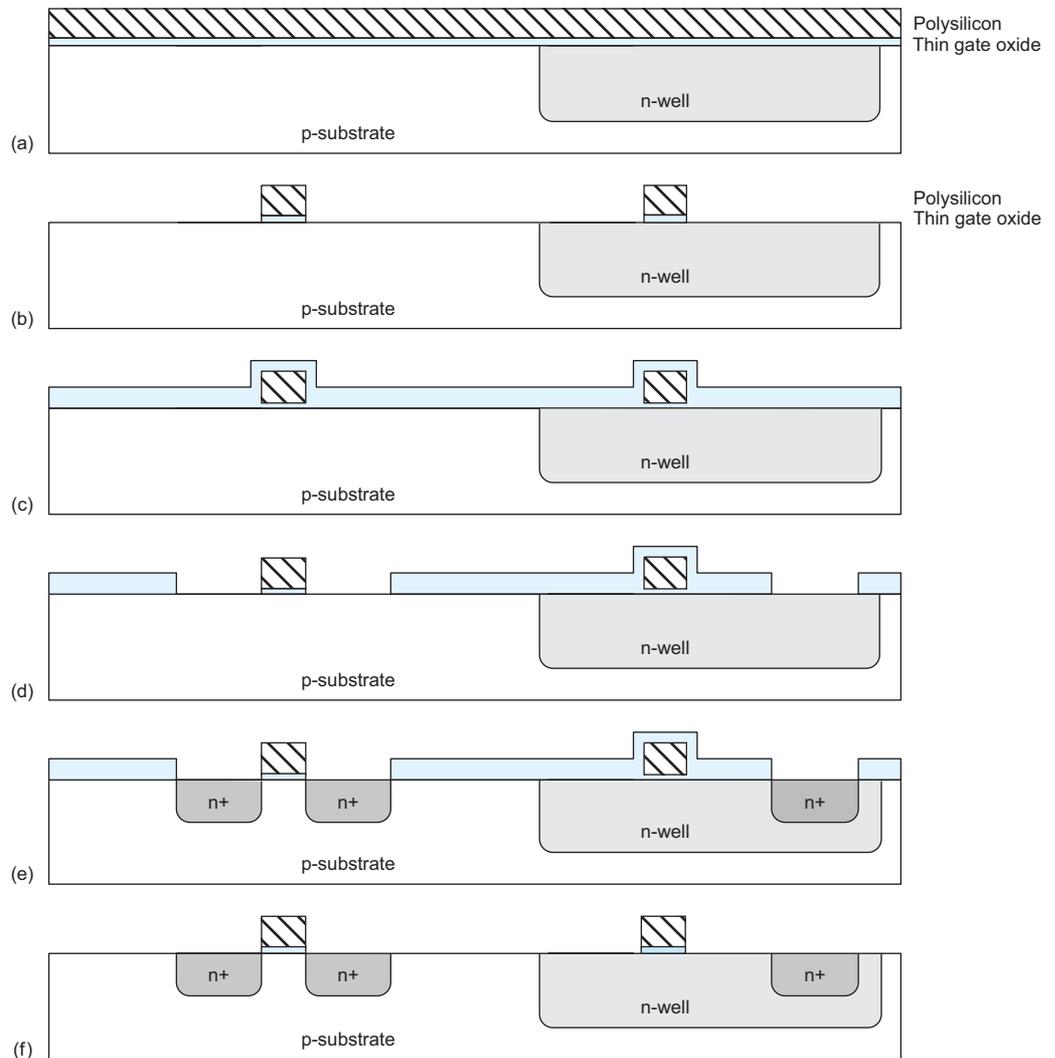


FIGURE 1.37 Cross-sections while manufacturing polysilicon and n-diffusion

1.5.3 Layout Design Rules

Layout design rules describe how small features can be and how closely they can be reliably packed in a particular manufacturing process. Industrial design rules are usually specified in microns. This makes migrating from one process to a more advanced process or a different foundry's process difficult because not all rules scale in the same way.

Universities sometimes simplify design by using scalable design rules that are conservative enough to apply to many manufacturing processes. Mead and Conway [Mead80] popularized scalable design rules based on a single parameter, λ , that characterizes the resolution of the process. λ is generally half of the minimum drawn transistor channel length. This length is the distance between the source and drain of a transistor and is set by the minimum width of a polysilicon wire. For example, a 180 nm process has a minimum polysilicon width (and hence transistor length) of $0.18 \mu\text{m}$ and uses design rules with

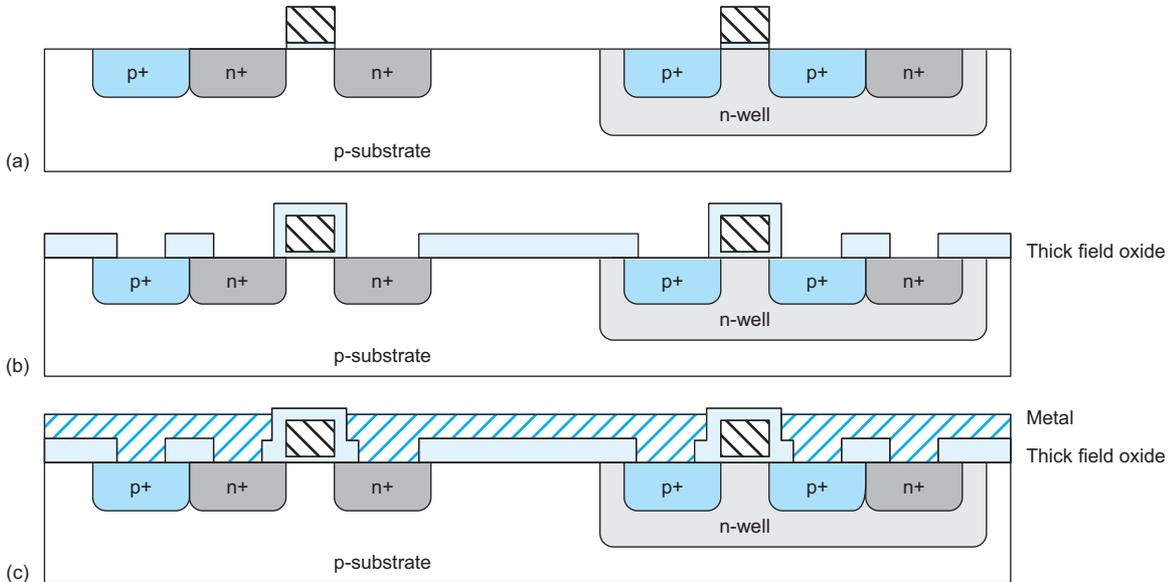


FIGURE 1.38 Cross-sections while manufacturing p-diffusion, contacts, and metal

$\lambda = 0.09 \mu\text{m}$.³ Lambda-based rules are necessarily conservative because they round up dimensions to an integer multiple of λ . However, they make scaling layout trivial; the same layout can be moved to a new process simply by specifying a new value of λ . This chapter will present design rules in terms of λ . The potential density advantage of micron rules is sacrificed for simplicity and easy scalability of lambda rules. Designers often describe a process by its *feature size*. Feature size refers to minimum transistor length, so λ is half the feature size.

Unfortunately, below 180 nm, design rules have become so complex and process-specific that scalable design rules are difficult to apply. However, the intuition gained from a simple set of scalable rules is still a valuable foundation for understanding the more complex rules. Chapter 3 will examine some of these process-specific rules in more detail.

The MOSIS service [Piña02] is a low-cost prototyping service that collects designs from academic, commercial, and government customers and aggregates them onto one mask set to share overhead costs and generate production volumes sufficient to interest fabrication companies. MOSIS has developed a set of scalable lambda-based design rules that covers a wide range of manufacturing processes. The rules describe the minimum width to avoid breaks in a line, minimum spacing to avoid shorts between lines, and minimum overlap to ensure that two layers completely overlap.

A conservative but easy-to-use set of design rules for layouts with two metal layers in an n-well process is as follows:

- Metal and diffusion have minimum width and spacing of 4λ .
- Contacts are $2 \lambda \times 2 \lambda$ and must be surrounded by 1λ on the layers above and below.
- Polysilicon uses a width of 2λ .

³Some 180 nm lambda-based rules actually set $\lambda = 0.10 \mu\text{m}$, then shrink the gate by 20 nm while generating masks. This keeps 180 nm gate lengths but makes all other features slightly larger.

- Polysilicon overlaps diffusion by 2λ where a transistor is desired and has a spacing of 1λ away where no transistor is desired.
- Polysilicon and contacts have a spacing of 3λ from other polysilicon or contacts.
- N-well surrounds pMOS transistors by 6λ and avoids nMOS transistors by 6λ .

Figure 1.39 shows the basic MOSIS design rules for a process with two metal layers. Section 3.3 elaborates on these rules and compares them with industrial design rules.

In a three-level metal process, the width of the third layer is typically 6λ and the spacing 4λ . In general, processes with more layers often provide thicker and wider top-level metal that has a lower resistance.

Transistor dimensions are often specified by their Width/Length (W/L) ratio. For example, the nMOS transistor in Figure 1.39 formed where polysilicon crosses n-diffusion has a W/L of $4/2$. In a $0.6\mu\text{m}$ process, this corresponds to an actual width of $1.2\mu\text{m}$ and a length of $0.6\mu\text{m}$. Such a minimum-width contacted transistor is often called a unit transistor.⁴ pMOS transistors are often wider than nMOS transistors because holes move more slowly than electrons so the transistor has to be wider to deliver the same current. Figure 1.40(a) shows a unit inverter layout with a unit nMOS transistor and a double-sized pMOS transistor. Figure 1.40(b) shows a schematic for the inverter annotated with Width/Length for each transistor. In digital systems, transistors are typically chosen to have the minimum possible length because short-channel transistors are faster, smaller, and consume less power. Figure 1.40(c) shows a shorthand we will often use, specifying multiples of unit width and assuming minimum length.

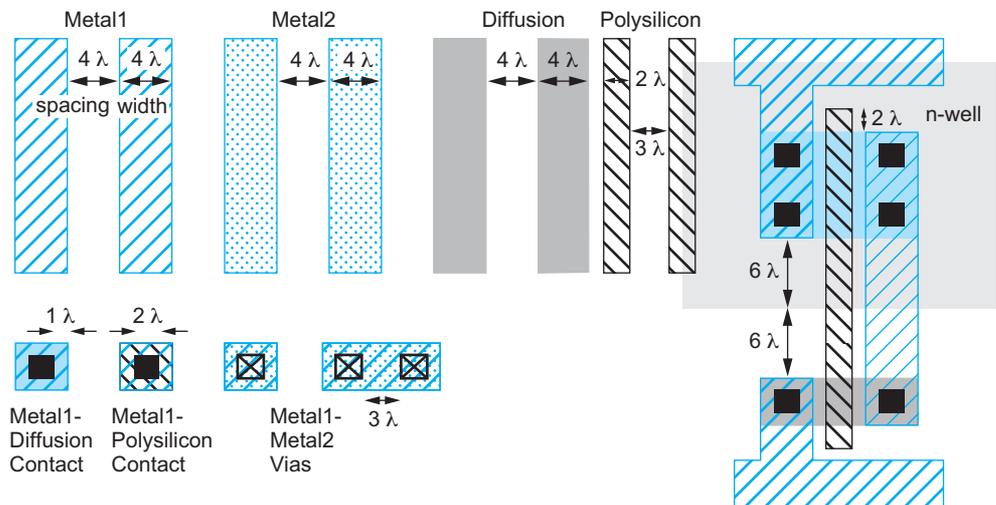


FIGURE 1.39 Simplified λ -based design rules

⁴Such small transistors in modern processes often behave slightly differently than their wider counterparts. Moreover, the transistor will not operate if either contact is damaged. Industrial designers often use a transistor wide enough for two contacts (9λ) as the unit transistor to avoid these problems.

1.5.4 Gate Layouts

A good deal of ingenuity can be exercised and a vast amount of time wasted exploring layout topologies to minimize the size of a gate or other *cell* such as an adder or memory element. For many applications, a straightforward layout is good enough and can be automatically generated or rapidly built by hand. This section presents a simple layout style based on a “line of diffusion” rule that is commonly used for standard cells in automated layout systems. This style consists of four horizontal strips: metal ground at the bottom of the cell, n-diffusion, p-diffusion, and metal power at the top. The power and ground lines are often called *supply rails*. Polysilicon lines run vertically to form transistor gates. Metal wires within the cell connect the transistors appropriately.

Figure 1.41(a) shows such a layout for an inverter. The input *A* can be connected from the top, bottom, or left in polysilicon. The output *Y* is available at the right side of the cell in metal. Recall that the p-substrate and n-well must be tied to ground and power, respectively. Figure 1.41(b) shows the same inverter with well and substrate taps placed under the power and ground rails, respectively. Figure 1.42 shows a 3-input NAND gate. Notice how the nMOS transistors are connected in series while the pMOS transistors are connected in parallel. Power and ground extend 2λ on each side so if two gates were abutted the contents would be separated by 4λ , satisfying design rules. The height of the cell is 36λ , or 40λ if the 4λ space between the cell and another wire above it is counted. All these examples use transistors of width 4λ . Choice of transistor width is addressed further in Chapters 4–5 and cell layout styles are discussed in Section 14.7.

These cells were designed such that the gate connections are made from the top or bottom in polysilicon. In contemporary standard cells, polysilicon is generally not used as a routing layer so the cell must allow metal2 to metal1 and metal1 to polysilicon contacts

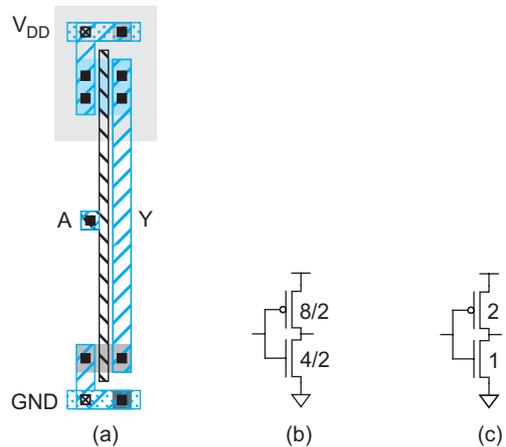


FIGURE 1.40 Inverter with dimensions labeled

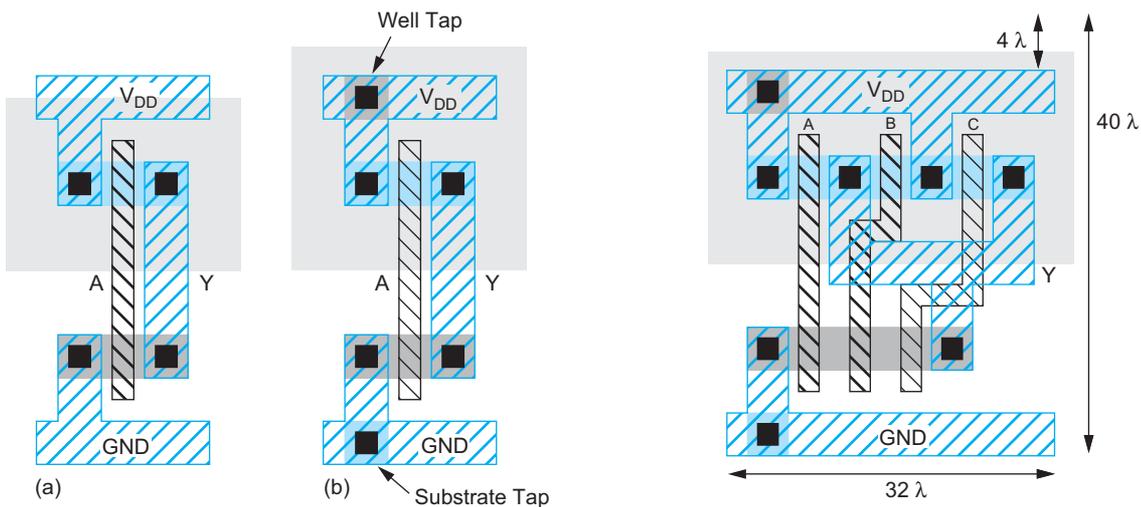


FIGURE 1.41 Inverter cell layout

FIGURE 1.42 3-input NAND standard cell gate layouts

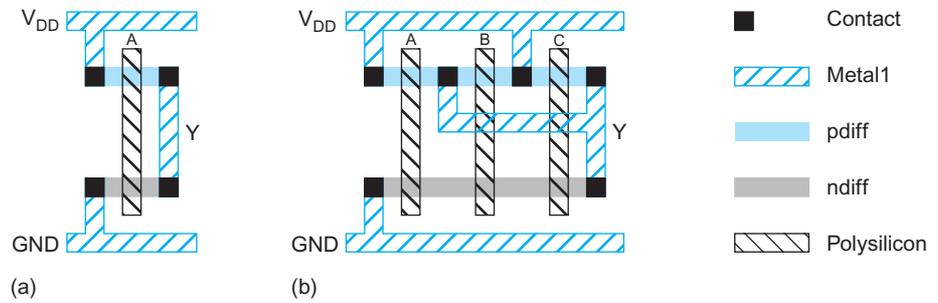


FIGURE 1.43 Stick diagrams of inverter and 3-input NAND gate. Color version on inside front cover.

to each gate. While this increases the size of the cell, it allows free access to all terminals on metal routing layers.

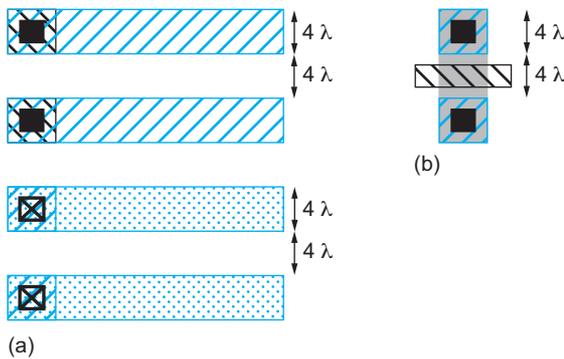


FIGURE 1.44 Pitch of routing tracks

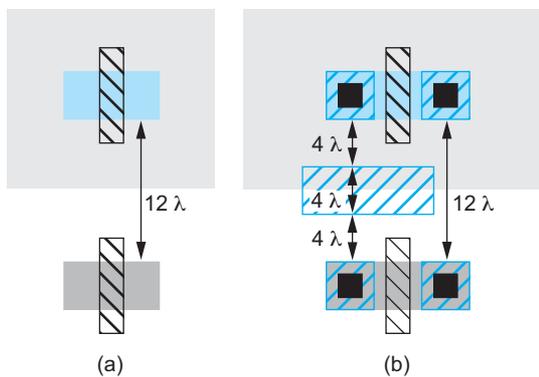


FIGURE 1.45 Spacing between nMOS and pMOS transistors

1.5.5 Stick Diagrams

Because layout is time-consuming, designers need fast ways to plan cells and estimate area before committing to a full layout. *Stick diagrams* are easy to draw because they do not need to be drawn to scale. Figure 1.43 and the inside front cover show stick diagrams for an inverter and a 3-input NAND gate. While this book uses stipple patterns, layout designers use dry-erase markers or colored pencils.

With practice, it is easy to estimate the area of a layout from the corresponding stick diagram even though the diagram is not to scale. Although schematics focus on transistors, layout area is usually determined by the metal wires. Transistors are merely widgets that fit under the wires. We define a *routing track* as enough space to place a wire and the required spacing to the next wire. If our wires have a width of 4λ and a spacing of 4λ to the next wire, the track *pitch* is 8λ , as shown in Figure 1.44(a). This pitch also leaves room for a transistor to be placed between the wires (Figure 1.44(b)). Therefore, it is reasonable to estimate the height and width of a cell by counting the number of metal tracks and multiplying by 8λ . A slight complication is the required spacing of 12λ between nMOS and pMOS transistors set by the well, as shown in Figure 1.45(a). This space can be occupied by an additional track of wire, shown in Figure 1.45(b). Therefore, an extra track must be allocated between nMOS and pMOS transistors regardless of whether wire is actually used in that track. Figure 1.46 shows how to count tracks to estimate the size of a 3-input NAND. There are four vertical wire tracks, multiplied by 8λ per track to give a cell width of 32λ . There are five horizontal tracks, giving a cell height of 40λ . Even though the horizontal tracks are not drawn to scale, they are still easy to count. Figure 1.42

shows that the actual NAND gate layout matches the dimensions predicted by the stick diagram. If transistors are wider than 4λ , the extra width must be factored into the area estimate. Of course, these estimates are oversimplifications of the complete design rules and a trial layout should be performed for truly critical cells.

Example 1.3

Sketch a stick diagram for a CMOS gate computing $Y = (A + B + C) \cdot D$ (see Figure 1.18) and estimate the cell width and height.

SOLUTION: Figure 1.47 shows a stick diagram. Counting horizontal and vertical pitches gives an estimated cell size of 40 by 48λ .

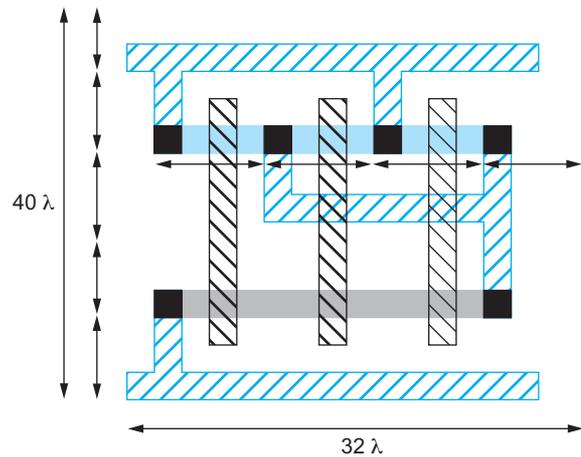


FIGURE 1.46 3-input NAND gate area estimation

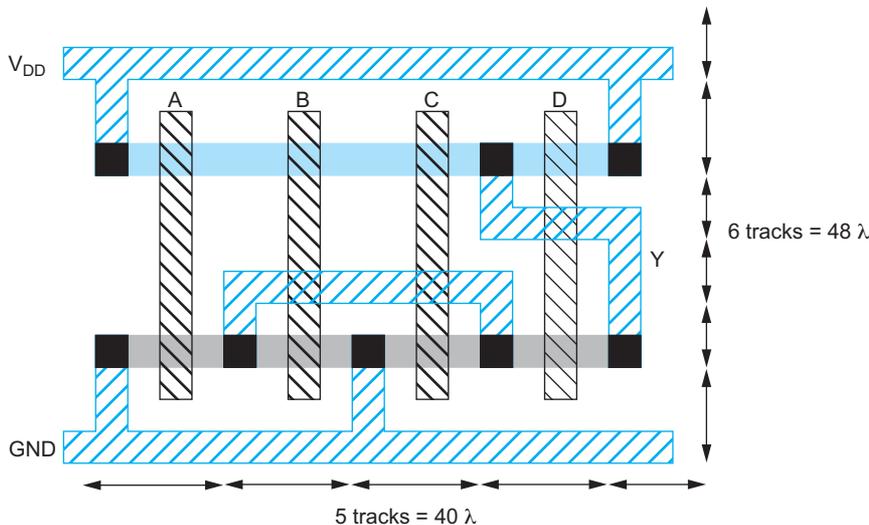


FIGURE 1.47 CMOS compound gate for function $Y = (A + B + C) \cdot D$

1.6 Design Partitioning

By this point, you know that MOS transistors behave as voltage-controlled switches. You know how to build logic gates out of transistors. And you know how transistors are fabricated and how to draw a layout that specifies how transistors should be placed and connected together. You know enough to start building your own simple chips.

The greatest challenge in modern VLSI design is not in designing the individual transistors but rather in managing system complexity. Modern *System-On-Chip* (SOC) designs combine memories, processors, high-speed I/O interfaces, and dedicated application-specific logic on a single chip. They use hundreds of millions or billions of transistors and cost tens of millions of dollars (or more) to design. The implementation

must be divided among large teams of engineers and each engineer must be highly productive. If the implementation is too rigidly partitioned, each block can be optimized without regard to its neighbors, leading to poor system results. Conversely, if every task is interdependent with every other task, design will progress too slowly. Design managers face the challenge of choosing a suitable trade-off between these extremes. There is no substitute for practical experience in making these choices, and talented engineers who have experience with multiple designs are very important to the success of a large project. Design proceeds through multiple levels of abstraction, hiding details until they become necessary. The practice of *structured design*, which is also used in large software projects, uses the principles of hierarchy, regularity, modularity, and locality to manage the complexity.

1.6.1 Design Abstractions

Digital VLSI design is often partitioned into five levels of abstractions: *architecture* design, *microarchitecture* design, *logic* design, *circuit* design, and *physical* design. Architecture describes the functions of the system. For example, the x86 microprocessor architecture specifies the instruction set, register set, and memory model. Microarchitecture describes how the architecture is partitioned into registers and functional units. The 80386, 80486, Pentium, Pentium II, Pentium III, Pentium 4, Core, Core 2, Atom, Cyrix MII, AMD Athlon, and Phenom are all microarchitectures offering different performance / transistor count / power trade-offs for the x86 architecture. Logic describes how functional units are constructed. For example, various logic designs for a 32-bit adder in the x86 integer unit include ripple carry, carry lookahead, and carry select. Circuit design describes how transistors are used to implement the logic. For example, a carry lookahead adder can use static CMOS circuits, domino circuits, or pass transistors. The circuits can be tailored to emphasize high performance or low power. Physical design describes the layout of the chip. Analog and RF VLSI design involves the same steps but with different layers of abstraction.

These elements are inherently interdependent and all influence each of the design objectives. For example, choices of microarchitecture and logic are strongly dependent on the number of transistors that can be placed on the chip, which depends on the physical design and process technology. Similarly, innovative circuit design that reduces a cache access from two cycles to one can influence which microarchitecture is most desirable. The choice of clock frequency depends on a complex interplay of microarchitecture and logic, circuit design, and physical design. Deeper pipelines allow higher frequencies but consume more power and lead to greater performance penalties when operations early in the pipeline are dependent on those late in the pipeline. Many functions have various logic and circuit designs trading speed for area, power, and design effort. Custom physical design allows more compact, faster circuits and lower manufacturing costs, but involves an enormous labor cost. Automatic layout with CAD systems reduces the labor and achieves faster times to market.

To deal with these interdependencies, microarchitecture, logic, circuit, and physical design must occur, at least in part, in parallel. Microarchitects depend on circuit and physical design studies to understand the cost of proposed microarchitectural features. Engineers are sometimes categorized as “short and fat” or “tall and skinny” (nothing personal, we assure you!). Tall, skinny engineers understand something about a broad range of topics. Short, fat engineers understand a large amount about a narrow field. Digital VLSI design favors the tall, skinny engineer who can evaluate how choices in one part of the system impact other parts of the system.

1.6.2 Structured Design

Hierarchy is a critical tool for managing complex designs. A large system can be partitioned hierarchically into multiple *cores*. Each core is built from various *units*. Each unit in turn is composed of multiple *functional blocks*.⁵ These blocks in turn are built from *cells*, which ultimately are constructed from transistors. The system can be more easily understood at the top level by viewing components as black boxes with well-defined interfaces and functions rather than looking at each individual transistor. Logic, circuit, and physical views of the design should share the same hierarchy for ease of verification. A design hierarchy can be viewed as a tree structure with the overall chip as the *root* and the primitive cells as *leaf*s.

Regularity aids the management of design complexity by designing the minimum number of different blocks. Once a block is designed and verified, it can be reused in many places. *Modularity* requires that the blocks have well-defined interfaces to avoid unanticipated interactions. *Locality* involves keeping information where it is used, physically and temporally. Structured design is discussed further in Section 14.2.

1.6.3 Behavioral, Structural, and Physical Domains

An alternative way of viewing design partitioning is shown with the Y-chart shown in Figure 1.48 [Gajski83, Kang03]. The radial lines on the Y-chart represent three distinct design domains: behavioral, structural, and physical. These domains can be used to describe the design of almost any artifact and thus form a general taxonomy for describing

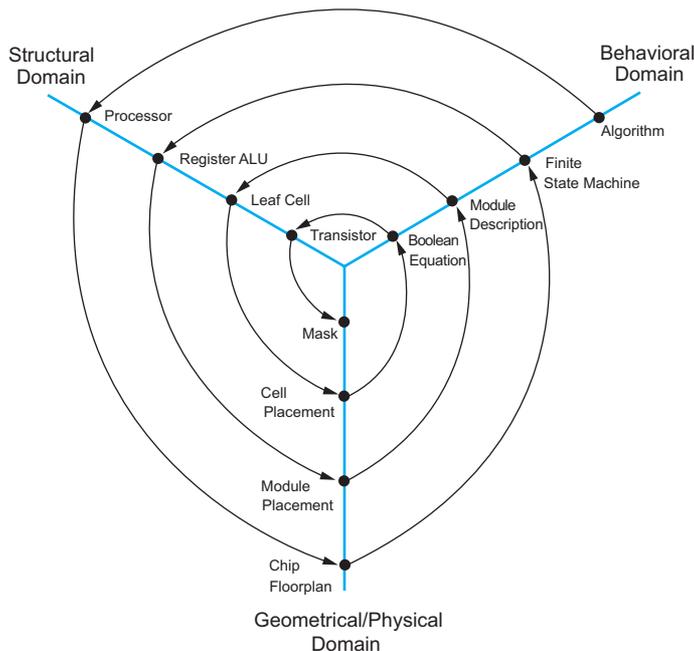


FIGURE 1.48 Y Diagram (Reproduced from [Kang03] with permission of The McGraw-Hill Companies.)

⁵Some designers refer to both units and functional blocks as *modules*.

the design process. Within each domain there are a number of levels of design abstraction that start at a very high level and descend eventually to the individual elements that need to be aggregated to yield the top level function (i.e., transistors in the case of chip design).

The behavioral domain describes what a particular system does. For instance, at the highest level we might specify a telephone touch-tone generator. This behavior can be successively refined to more precisely describe what needs to be done in order to build the tone generator (i.e., the frequencies desired, output levels, distortion allowed, etc.).

At each abstraction level, a corresponding structural description can be developed. The structural domain describes the interconnection of modules necessary to achieve a particular behavior. For instance, at the highest level, the touch-tone generator might consist of a keypad, a tone generator chip, an audio amplifier, a battery, and a speaker. Eventually at lower levels of abstraction, the individual gate and then transistor connections required to build the tone generator are described.

For each level of abstraction, the physical domain description explains how to physically construct that level of abstraction. At high levels, this might consist of an engineering drawing showing how to put together the keypad, tone generator chip, battery, and speaker in the associated housing. At the top chip level, this might consist of a floorplan, and at lower levels, the actual geometry of individual transistors.

The design process can be viewed as making transformations from one domain to another while maintaining the equivalency of the domains. Behavioral descriptions are transformed to structural descriptions, which in turn are transformed to physical descriptions. These transformations can be manual or automatic. In either case, it is normal design practice to verify the transformation of one domain to the other. This ensures that the design intent is carried across the domain boundaries. Hierarchically specifying each domain at successively detailed levels of abstraction allows us to design very large systems.

The reason for strictly describing the domains and levels of abstraction is to define a precise design process in which the final function of the system can be traced all the way back to the initial behavioral description. In an ideal flow, there should be no opportunity to produce an incorrect design. If anomalies arise, the design process is corrected so that those anomalies will not reoccur in the future. A designer should acquire a rigid discipline with respect to the design process, and be aware of each transformation and how and why it is failproof. Normally, these steps are fully automated in a modern design process, but it is important to be aware of the basis for these steps in order to debug them if they go astray.

The Y diagram can be used to illustrate each domain and the transformations between domains at varying levels of design abstraction. As the design process winds its way from the outer to inner rings, it proceeds from higher to lower levels of abstraction and hierarchy.

Most of the remainder of this chapter is a case study in the design of a simple micro-processor to illustrate the various aspects of VLSI design applied to a nontrivial system. We begin by describing the architecture and microarchitecture of the processor. We then consider logic design and discuss hardware description languages. The processor is built with static CMOS circuits, which we examined in Section 1.4; transistor-level design and netlist formats are discussed. We continue exploring the physical design of the processor including floorplanning and area estimation. Design verification is critically important and happens at each level of the hierarchy for each element of the design. Finally, the layout is converted into masks so the chip can be manufactured, packaged, and tested.

1.7 Example: A Simple MIPS Microprocessor

We consider an 8-bit subset of the MIPS microprocessor architecture [Patterson04, Harris07] because it is widely studied and is relatively simple, yet still large enough to illustrate hierarchical design. This section describes the architecture and the multicycle microarchitecture we will be implementing. If you are not familiar with computer architecture, you can regard the MIPS processor as a black box and skip to Section 1.8.

A set of laboratory exercises is available at www.cmosvlsi.com in which you can learn VLSI design by building the microprocessor yourself using a free open-source CAD tool called *Electric* or with commercial design tools from Cadence and Synopsys.

1.7.1 MIPS Architecture

The MIPS32 architecture is a simple 32-bit RISC architecture with relatively few idiosyncrasies. Our subset of the architecture uses 32-bit instruction encodings but only eight 8-bit general-purpose registers named \$0–\$7. We also use an 8-bit program counter (PC). Register \$0 is hardwired to contain the number 0. The instructions are ADD, SUB, AND, OR, SLT, ADDI, BEQ, J, LB, and SB.

The function and encoding of each instruction is given in Table 1.7. Each instruction is encoded using one of three templates: R, I, and J. R-type instructions (*register*-based) are used for arithmetic and specify two source registers and a destination register. I-type instructions are used when a 16-bit constant (also known as an *immediate*) and two registers must be specified. J-type instructions (*jumps*) dedicate most of the instruction word to a 26-bit jump destination. The format of each encoding is defined in Figure 1.49. The six most significant bits of all formats are the operation code (*op*). R-type instructions all share *op* = 000000 and use six more *funct* bits to differentiate the functions.

TABLE 1.7 MIPS instruction set (subset supported)

Instruction	Function	Encoding	op	funct
add \$1, \$2, \$3	addition: $\$1 \leftarrow \$2 + \$3$	R	000000	100000
sub \$1, \$2, \$3	subtraction: $\$1 \leftarrow \$2 - \$3$	R	000000	100010
and \$1, \$2, \$3	bitwise and: $\$1 \leftarrow \$2 \text{ and } \$3$	R	000000	100100
or \$1, \$2, \$3	bitwise or: $\$1 \leftarrow \$2 \text{ or } \$3$	R	000000	100101
slt \$1, \$2, \$3	set less than: $\$1 \leftarrow 1 \text{ if } \$2 < \$3$ $\$1 \leftarrow 0 \text{ otherwise}$	R	000000	101010
addi \$1, \$2, imm	add immediate: $\$1 \leftarrow \$2 + \text{imm}$	I	001000	n/a
beq \$1, \$2, imm	branch if equal: $\text{PC} \leftarrow \text{PC} + \text{imm} \times 4^a$	I	000100	n/a
j destination	jump: $\text{PC} \leftarrow \text{destination}^a$	J	000010	n/a
lb \$1, imm(\$2)	load byte: $\$1 \leftarrow \text{mem}[\$2 + \text{imm}]$	I	100000	n/a
sb \$1, imm(\$2)	store byte: $\text{mem}[\$2 + \text{imm}] \leftarrow \1	I	101000	n/a

a. Technically, MIPS addresses specify bytes. Instructions require a 4-byte word and must begin at addresses that are a multiple of four. To most effectively use instruction bits in the full 32-bit MIPS architecture, branch and jump constants are specified in words and must be multiplied by four (shifted left 2 bits) to be converted to byte addresses.

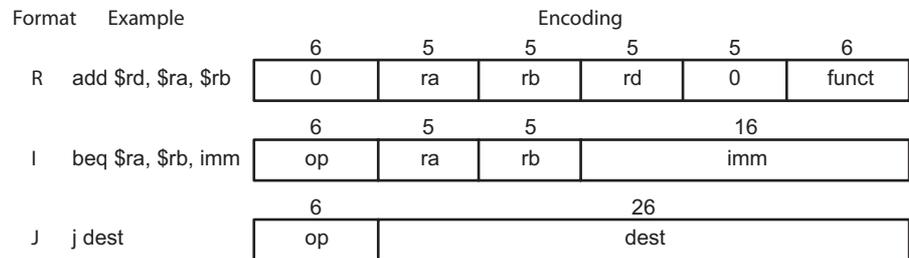


FIGURE 1.49 Instruction encoding formats

We can write programs for the MIPS processor in *assembly language*, where each line of the program contains one instruction such as `ADD` or `BEQ`. However, the MIPS hardware ultimately must read the program as a series of 32-bit numbers called *machine language*. An *assembler* automates the tedious process of translating from assembly language to machine language using the encodings defined in Table 1.7 and Figure 1.49. Writing nontrivial programs in assembly language is also tedious, so programmers usually work in a *high-level language* such as C or Java. A *compiler* translates a program from high-level language *source code* into the appropriate machine language *object code*.

Example 1.4

Figure 1.50 shows a simple C program that computes the n th Fibonacci number f_n defined recursively for $n > 0$ as $f_n = f_{n-1} + f_{n-2}$, $f_{-1} = -1$, $f_0 = 1$. Translate the program into MIPS assembly language and machine language.

SOLUTION: Figure 1.51 gives a commented assembly language program. Figure 1.52 translates the assembly language to machine language.

```
int fib(void)
{
    int n = 8;           /* compute nth Fibonacci number */
    int f1 = 1, f2 = -1; /* last two Fibonacci numbers */

    while (n != 0) {    /* count down to n = 0 */
        f1 = f1 + f2;
        f2 = f1 - f2;
        n = n - 1;
    }
    return f1;
}
```

FIGURE 1.50 C Code for Fibonacci program

1.7.2 Multicycle MIPS Microarchitecture

We will implement the multicycle MIPS microarchitecture given in Chapter 5 of [Patterson04] and Chapter 7 of [Harris07] modified to process 8-bit data. The microarchitecture is illustrated in Figure 1.53. Light lines indicate individual signals while heavy

```

# fib.asm
# Register usage: $3: n $4: f1 $5: f2
# return value written to address 255
fib:  addi $3, $0, 8      # initialize n=8
      addi $4, $0, 1      # initialize f1 = 1
      addi $5, $0, -1     # initialize f2 = -1
loop: beq $3, $0, end    # Done with loop if n = 0
      add $4, $4, $5      # f1 = f1 + f2
      sub $5, $4, $5      # f2 = f1 - f2
      addi $3, $3, -1     # n = n - 1
      j loop              # repeat until done
end:  sb $4, 255($0)     # store result in address 255

```

FIGURE 1.51 Assembly language code for Fibonacci program

Instruction	Binary Encoding	Hexadecimal Encoding
addi \$3, \$0, 8	001000 00000 00011 0000000000001000	20030008
addi \$4, \$0, 1	001000 00000 00100 0000000000000001	20040001
addi \$5, \$0, -1	001000 00000 00101 1111111111111111	2005ffff
beq \$3, \$0, end	000100 00011 00000 0000000000000100	10600004
add \$4, \$4, \$5	000000 00100 00101 00100 00000 100000	00852020
sub \$5, \$4, \$5	000000 00100 00101 00101 00000 100010	00852822
addi \$3, \$3, -1	001000 00011 00011 1111111111111111	2063ffff
j loop	000010 0000000000000000000000000011	08000003
sb \$4, 255(\$0)	101000 00000 00100 0000000011111111	a00400ff

FIGURE 1.52 Machine language code for Fibonacci program

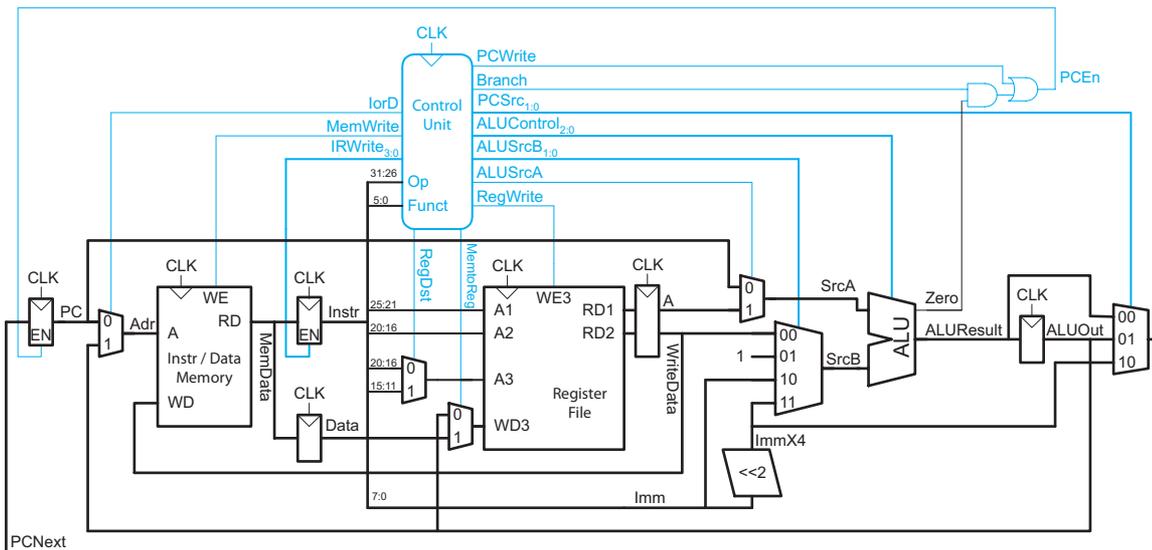


FIGURE 1.53 Multicycle MIPS microarchitecture. Adapted from [Patterson04] and [Harris07] with permission from Elsevier.

lines indicate busses. The control logic and signals are highlighted in blue while the datapath is shown in black. Control signals generally drive multiplexer select signals and register enables to tell the datapath how to execute an instruction.

Instruction execution generally flows from left to right. The program counter (PC) specifies the address of the instruction. The instruction is loaded 1 byte at a time over four cycles from an off-chip memory into the 32-bit instruction register (IR). The *Op* field (bits 31:26 of the instruction) is sent to the controller, which sequences the datapath through the correct operations to execute the instruction. For example, in an *ADD* instruction, the two source registers are read from the register file into temporary registers A and B. On the next cycle, the aludec unit commands the Arithmetic/Logic Unit (ALU) to add the inputs. The result is captured in the *ALUOut* register. On the third cycle, the result is written back to the appropriate destination register in the register file.

The controller contains a finite state machine (FSM) that generates multiplexer select signals and register enables to sequence the datapath. A state transition diagram for the FSM is shown in Figure 1.54. As discussed, the first four states fetch the instruction from

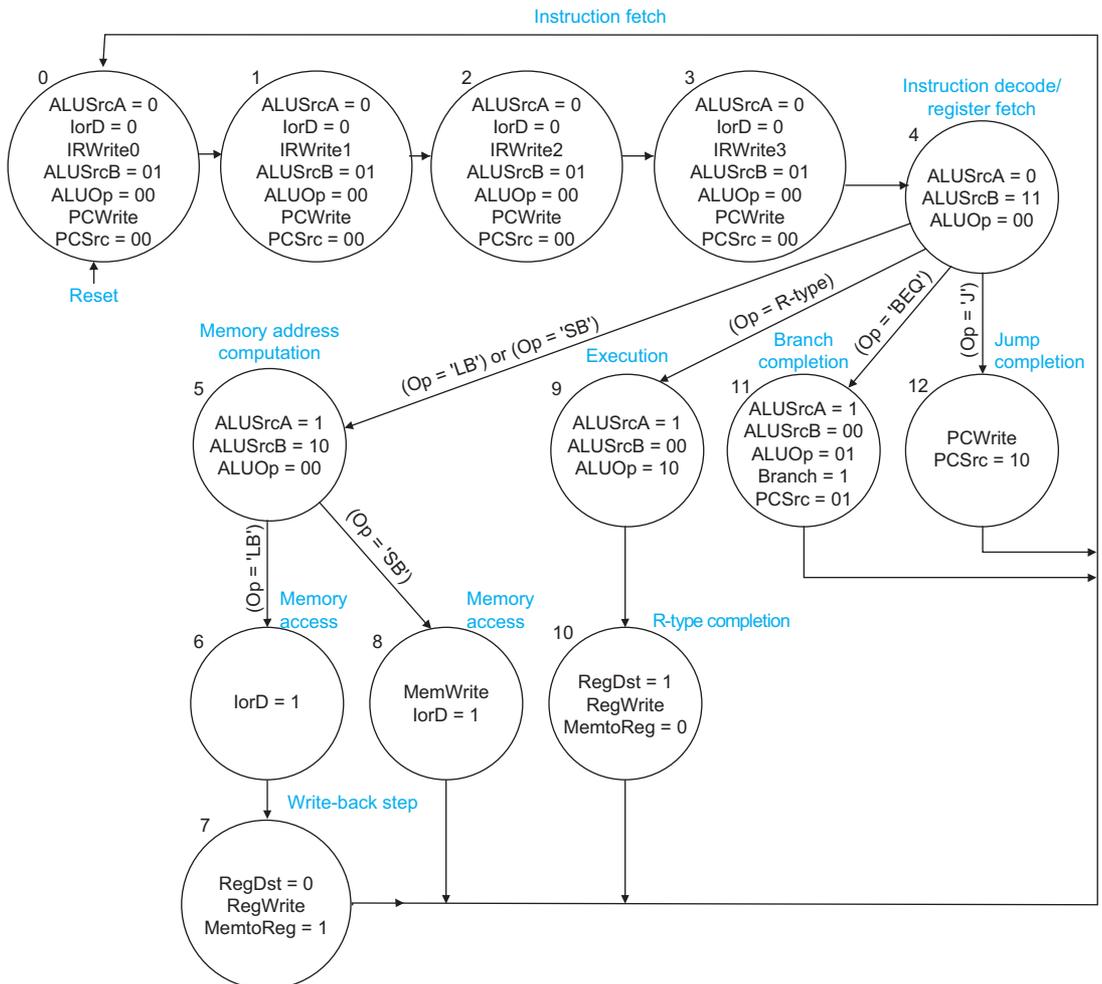


FIGURE 1.54 Multicycle MIPS control FSM (Adapted from [Patterson04] and [Harris07] with permission from Elsevier.)

memory. The FSM then is dispatched based on `Op` to execute the particular instruction. The FSM states for `ADDI` are missing and left as an exercise for the reader.

Observe that the FSM produces a 2-bit `ALUOp` output. The ALU decoder unit in the controller uses combinational logic to compute a 3-bit `ALUControl` signal from the `ALUOp` and `Func` fields, as specified in Table 1.8. `ALUControl` drives multiplexers in the ALU to select the appropriate computation.

TABLE 1.8 ALUControl determination

ALUOp	Func	ALUControl	Meaning
00	x	010	ADD
01	x	110	SUB
10	100000	010	ADD
10	100010	110	SUB
10	100100	000	AND
10	100101	001	OR
10	101010	111	SLT
11	x	x	undefined

Example 1.5

Referring to Figures 1.53 and 1.54, explain how the MIPS processor fetches and executes the `SUB` instruction.

SOLUTION: The first step is to fetch the 32-bit instruction. This takes four cycles because the instruction must come over an 8-bit memory interface. On each cycle, we want to fetch a byte from the address in memory specified by the program counter, then increment the program counter by one to point to the next byte.

The fetch is performed by states 0–3 of the FSM in Figure 1.54. Let us start with state 0. The program counter (PC) contains the address of the first byte of the instruction. The controller must select `IOrd = 0` so that the multiplexer sends this address to the memory. `MemRead` must also be asserted so the memory reads the byte onto the `MemData` bus. Finally, `IRWrite0` should be asserted to enable writing `memdata` into the least significant byte of the instruction register (IR).

Meanwhile, we need to increment the program counter. We can do this with the ALU by specifying PC as one input, 1 as the other input, and `ADD` as the operation. To select PC as the first input, `ALUSrcA = 0`. To select 1 as the other input, `ALUSrcB = 01`. To perform an addition, `ALUOp = 00`, according to Table 1.8. To write this result back into the program counter at the end of the cycle, `PCSrc = 00` and `PCEn = 1` (done by setting `PCWrite = 1`).

All of these control signals are indicated in state 0 of Figure 1.54. The other register enables are assumed to be 0 if not explicitly asserted and the other multiplexer selects are don't cares. The next three states are identical except that they write bytes 1, 2, and 3 of the IR, respectively.

The next step is to read the source registers, done in state 4. The two source registers are specified in bits 25:21 and 20:16 of the IR. The register file reads these registers and puts the values into the A and B registers. No control signals are necessary for `SUB` (although state 4 performs a branch address computation in case the instruction is `BEQ`).

The next step is to perform the subtraction. Based on the `Op` field (IR bits 31:26), the FSM jumps to state 9 because `SUB` is an R-type instruction. The two source registers are selected as input to the ALU by setting `ALUSrcA = 1` and `ALUSrcB = 00`. Choosing `ALUOp = 10` directs the ALU Control decoder to select the `ALUControl` signal as 110, subtraction. Other R-type instructions are executed identically except that the decoder receives a different `Funct` code (IR bits 5:0) and thus generates a different `ALUControl` signal. The result is placed in the `ALUOut` register.

Finally, the result must be written back to the register file in state 10. The data comes from the `ALUOut` register so `MemtoReg = 0`. The destination register is specified in bits 15:11 of the instruction so `RegDst = 1`. `RegWrite` must be asserted to perform the write. Then, the control FSM returns to state 0 to fetch the next instruction.

1.8 Logic Design

We begin the logic design by defining the top-level chip interface and block diagram. We then hierarchically decompose the units until we reach leaf cells. We specify the logic with a Hardware Description Language (HDL), which provides a higher level of abstraction than schematics or layout. This code is often called the Register Transfer Level (RTL) description.

1.8.1 Top-Level Interfaces

The top-level inputs and outputs are listed in Table 1.9. This example uses a two-phase clocking system to avoid hold-time problems. `Reset` initializes the PC to 0 and the control FSM to the start state.

TABLE 1.9 Top-level inputs and outputs

Inputs	Outputs
<code>ph1</code>	<code>MemWrite</code>
<code>ph2</code>	<code>Adr[7:0]</code>
<code>reset</code>	<code>WriteData[7:0]</code>
<code>MemData[7:0]</code>	

The remainder of the signals are used for an 8-bit memory interface (assuming the memory is located off chip). The processor sends an 8-bit address `Adr` and optionally asserts `MemWrite`. On a read cycle, the memory returns a value on the `MemData` lines while on a write cycle, the memory accepts input from `WriteData`. In many systems, `MemData` and `WriteData` can be combined onto a single bidirectional bus, but for this example we preserve the interface of Figure 1.53. Figure 1.55 shows a simple computer system built from the MIPS processor, external memory, reset switch, and clock generator.

1.8.2 Block Diagrams

The chip is partitioned into two top-level units: the controller and datapath, as shown in the block diagram in Figure 1.56. The controller comprises the control FSM, the ALU decoder, and the two gates used to compute `PCEn`. The ALU decoder consists of combina-

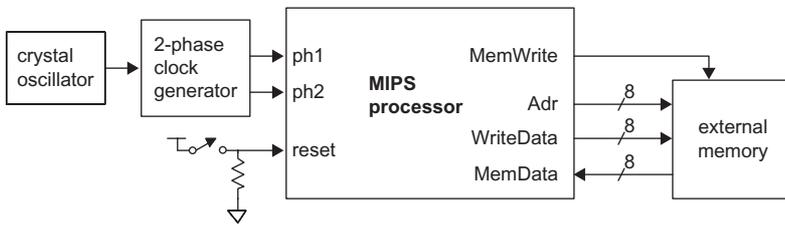


FIGURE 1.55 MIPS computer system

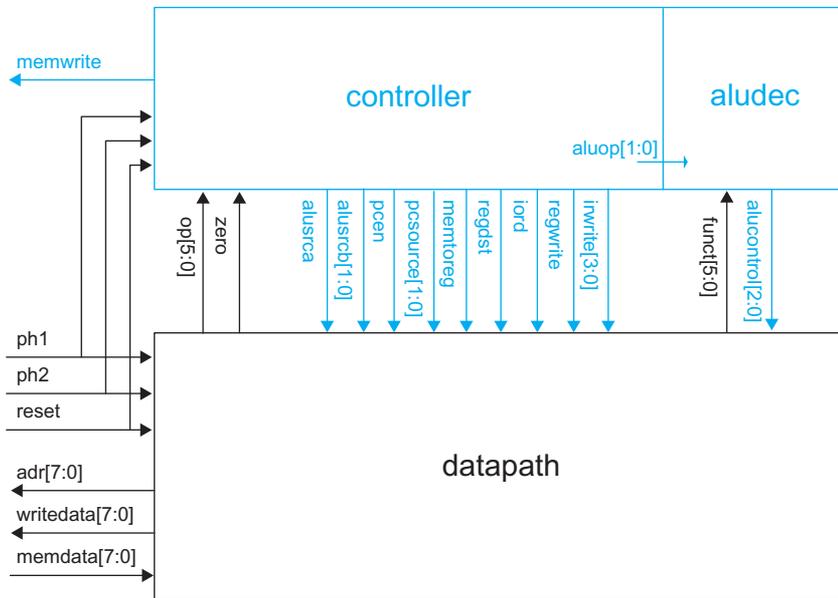


FIGURE 1.56 Top-level MIPS block diagram

tional logic to determine `ALUControl`. The 8-bit datapath contains the remainder of the chip. It can be viewed as a collection of wordslices or bitslices. A *wordslice* is a column containing an 8-bit flip-flop, adder, multiplexer, or other element. For example, Figure 1.57 shows a wordslice for an 8-bit 2:1 multiplexer. It contains eight individual 2:1 multiplexers, along with a *zipper* containing a buffer and inverter to drive the true and complementary select signals to all eight multiplexers.⁶ Factoring these drivers out into the zipper saves space as compared to putting inverters in each multiplexer. Alternatively, the datapath can be viewed as eight rows of *bitslices*. Each bitslice has one bit of each component, along with the horizontal wires connecting the bits together.

The chip partitioning is influenced by the intended physical design. The datapath contains most of the transistors and is very regular in structure. We can achieve high density with moderate design effort by handcrafting each wordslice or bitslice and tiling the

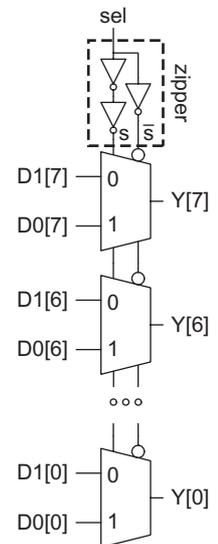


FIGURE 1.57 8-bit 2:1 multiplexer wordslice

⁶In this example, the zipper is shown at the top of the wordslice. In wider datapaths, the zipper is sometimes placed in the middle of the wordslice so that it drives shorter wires. The name comes from the way the layout resembles a plaid sweatshirt with a zipper down the middle.

circuits together. Building datapaths using wordslices is usually easier because certain structures, such as the zero detection circuit in the ALU, are not identical in each bitslice. However, thinking about bitslices is a valuable way to plan the wiring across the datapath. The controller has much less structure. It is tedious to translate an FSM into gates by hand, and in a new design, the controller is the most likely portion to have bugs and last-minute changes. Therefore, we will specify the controller more abstractly with a hardware description language and automatically generate it using synthesis and place & route tools or a programmable logic array (PLA).

1.8.3 Hierarchy

The best way to design complex systems is to decompose them into simpler pieces. Figure 1.58 shows part of the design hierarchy for the MIPS processor. The controller contains the `controller_pla` and `aludec`, which in turn is built from a library of standard cells such as NANDs, NORs, and inverters. The datapath is composed of 8-bit wordslices, each of which also is typically built from standard cells such as adders, register file bits, multiplexers, and flip-flops. Some of these cells are reused in multiple places.

The design hierarchy does not necessarily have to be identical in the logic, circuit, and physical designs. For example, in the logic view, a memory may be best treated as a black box, while in the circuit implementation, it may have a decoder, cell array, column multiplexers, and so forth. Different hierarchies complicate verification, however, because they must be *flattened* until the point that they agree. As a matter of practice, it is best to make logic, circuit, and physical design hierarchies agree as far as possible.

1.8.4 Hardware Description Languages

Designers need rapid feedback on whether a logic design is reasonable. Translating block diagrams and FSM state transition diagrams into circuit schematics is time-consuming and prone to error; before going through this entire process it is wise to know if the top-level design has major bugs that will require complete redesign. HDLs provide a way to specify the design at a higher level of abstraction to raise designer productivity. They were originally intended for documentation and simulation, but are now used to synthesize gates directly from the HDL.

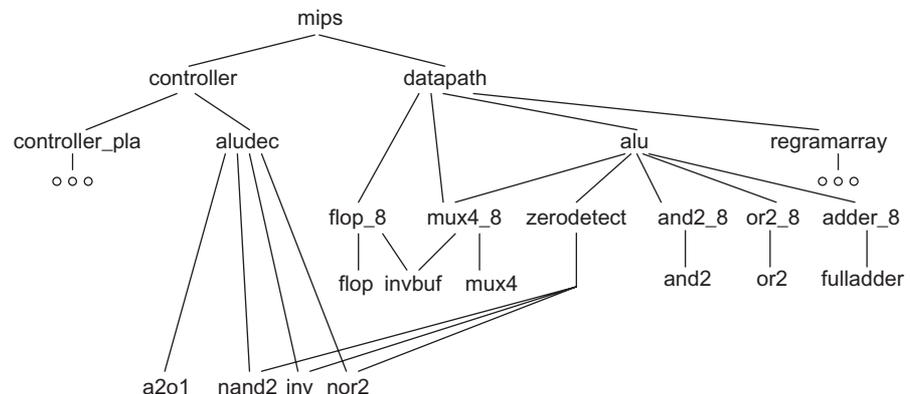


FIGURE 1.58 MIPS design hierarchy

The two most popular HDLs are *Verilog* and *VHDL*. Verilog was developed by Advanced Integrated Design Systems (later renamed Gateway Design Automation) in 1984 and became a *de facto* industry open standard by 1991. In 2005, the SystemVerilog extensions were standardized, and some of these features are used in this book. VHDL, which stands for VHSIC Hardware Description Language, where VHSIC in turn was a Department of Defense project on Very High Speed Integrated Circuits, was developed by committee under government sponsorship. As one might expect from their pedigrees, Verilog is less verbose and closer in syntax to C, while VHDL supports some abstractions useful for large team projects. Many Silicon Valley companies use Verilog while defense and telecommunications companies often use VHDL. Neither language offers a decisive advantage over the other so the industry is saddled with supporting both. Appendix A offers side-by-side tutorials on Verilog and VHDL. Examples in this book are given in Verilog for the sake of brevity.

When coding in an HDL, it is important to remember that you are specifying hardware that operates in parallel rather than software that executes in sequence. There are two general coding styles. *Structural* HDL specifies how a cell is composed of other cells or primitive gates and transistors. *Behavioral* HDL specifies what a cell does.

A *logic simulator* simulates HDL code; it can report whether results match expectations, and can display waveforms to help debug discrepancies. A *logic synthesis* tool is similar to a compiler for hardware: it maps HDL code onto a *library* of gates called *standard cells* to minimize area while meeting some timing constraints. Only a subset of HDL constructs are synthesizable; this subset is emphasized in the appendix. For example, file I/O commands used in testbenches are obviously not synthesizable. Logic synthesis generally produces circuits that are neither as dense nor as fast as those handcrafted by a skilled designer. Nevertheless, integrated circuit processes are now so advanced that synthesized circuits are good enough for the great majority of application-specific integrated circuits (ASICs) built today. Layout may be automatically generated using place & route tools.

Verilog and VHDL models for the MIPS processor are listed in Appendix A.12. In Verilog, each cell is called a *module*. The inputs and outputs are declared much as in a C program and bit widths are given for busses. Internal signals must also be declared in a way analogous to local variables. The processor is described hierarchically using structural Verilog at the upper levels and behavioral Verilog for the leaf cells. For example, the controller module shows how a finite state machine is specified in behavioral Verilog and the *aludec* module shows how complex combinational logic is specified. The datapath is specified structurally in terms of wordslices, which are in turn described behaviorally.

For the sake of illustration, the 8-bit adder wordslice could be described structurally as a ripple carry adder composed of eight cascaded full adders. The full adder could be expressed structurally as a sum and a carry subcircuit. In turn, the sum and carry subcircuits could be expressed behaviorally. The full adder block is shown in Figure 1.59 while the carry subcircuit is explored further in Section 1.9.

```

module adder(input  logic [7:0] a, b,
             input  logic    c,
             output logic [7:0] s,
             output logic    cout);

    wire [6:0] carry;

```

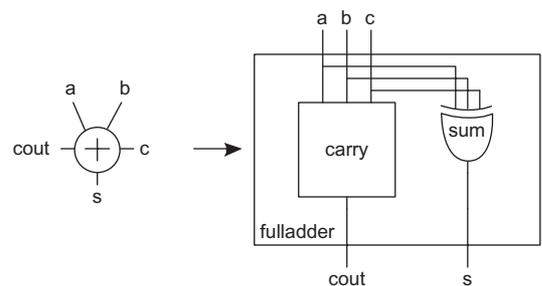


FIGURE 1.59 Full adder

```

fulladder fa0(a[0], b[0], c,          s[0], carry[0]);
fulladder fa1(a[1], b[1], carry[0], s[1], carry[1]);
fulladder fa2(a[2], b[2], carry[1], s[2], carry[2]);
...
fulladder fa7(a[7], b[7], carry[6], s[7], cout);
endmodule

module fulladder(input logic a, b, c,
                 output logic s, cout);

    sum s1(a, b, c, s);
    carry c1(a, b, c, cout);
endmodule

module carry(input logic a, b, c,
             output logic cout);

    assign cout = (a&b) | (a&c) | (b&c);
endmodule

```

1.9 Circuit Design

Circuit design is concerned with arranging transistors to perform a particular logic function. Given a circuit design, we can estimate the delay and power. The circuit can be represented as a schematic, or in textual form as a netlist. Common transistor level netlist formats include Verilog and SPICE. Verilog netlists are used for functional verification, while SPICE netlists have more detail necessary for delay and power simulations.

Because a transistor gate is a good insulator, it can be modeled as a capacitor, C . When the transistor is ON, some current I flows between source and drain. Both the current and capacitance are proportional to the transistor width.

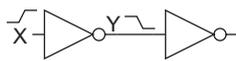
The delay of a logic gate is determined by the current that it can deliver and the capacitance that it is driving, as shown in Figure 1.60 for one inverter driving another inverter. The capacitance is charged or discharged according to the constitutive equation

$$I = C \frac{dV}{dt}$$

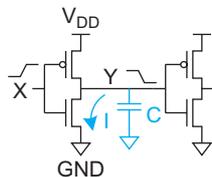
If an average current I is applied, the time t to switch between 0 and V_{DD} is

$$t = \frac{C}{I} V_{DD}$$

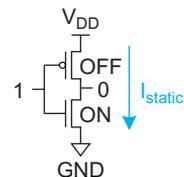
Hence, the delay increases with the load capacitance and decreases with the drive current. To make these calculations, we will have to delve below the switch-level model of a transistor. Chapter 2 develops more detailed models of transistors accounting for the current and capacitance. One of the goals of circuit design is to choose transistor widths to meet delay requirements. Methods for doing so are discussed in Chapter 4.



(a)



(b)



(c)

FIGURE 1.60 Circuit delay and power: (a) inverter pair, (b) transistor-level model showing capacitance and current during switching, (c) static leakage current during quiescent operation

Energy is required to charge and discharge the load capacitance. This is called dynamic power because it is consumed when the circuit is actively switching. The dynamic power consumed when a capacitor is charged and discharged at a frequency f is

$$P_{\text{dynamic}} = CV_{DD}^2 f$$

Even when the gate is not switching, it draws some static power. Because an OFF transistor is leaky, a small amount of current I_{static} flows between power and ground, resulting in a static power dissipation of

$$P_{\text{static}} = I_{\text{static}} V_{DD}$$

Chapter 5 examines power in more detail.

A particular logic function can be implemented in many ways. Should the function be built with ANDs, ORs, NANDs, or NORs? What should be the fan-in and fan-out of each gate? How wide should the transistors be on each gate? Each of these choices influences the capacitance and current and hence the speed and power of the circuit, as well as the area and cost.

As mentioned earlier, in many design methodologies, logic synthesis tools automatically make these choices, searching through the standard cells for the best implementation. For many applications, synthesis is good enough. When a system has critical requirements of high speed or low power or will be manufactured in large enough volume to justify the extra engineering, custom circuit design becomes important for critical portions of the chip.

Circuit designers often draw schematics at the transistor and/or gate level. For example, Figure 1.61 shows two alternative circuit designs for the carry circuit in a full adder. The gate-level design in Figure 1.61(a) requires 26 transistors and four stages of gate delays (recall that ANDs and ORs are built from NANDs and NORs followed by inverters). The transistor-level design in Figure 1.61(b) requires only 12 transistors and two stages of gate delays, illustrating the benefits of optimizing circuit designs to take advantage of CMOS technology.

These schematics are then *netlisted* for simulation and verification. One common netlist format is structural Verilog HDL. The gate-level design can be netlisted as follows:

```

module carry(input  logic a, b, c,
             output logic cout);

    logic x, y, z;

    and g1(x, a, b);
    and g2(y, a, c);
    and g3(z, b, c);
    or  g4(cout, x, y, z);
endmodule

```

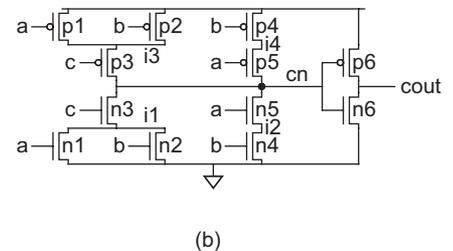
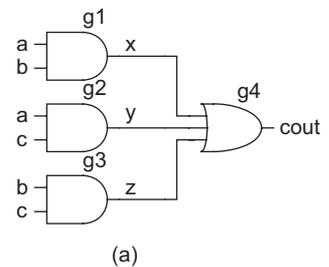


FIGURE 1.61 Carry subcircuit

This is a technology-independent structural description, because generic gates have been used and the actual gate implementations have not been specified. The transistor-level netlist follows:

```

module carry(input  logic a, b, c,
              output tri  cout);

    tri      i1, i2, i3, i4, cn;
    supply0 gnd;
    supply1 vdd;

    tranif1 n1(i1, gnd, a);
    tranif1 n2(i1, gnd, b);
    tranif1 n3(cn, i1, c);
    tranif1 n4(i2, gnd, b);
    tranif1 n5(cn, i2, a);
    tranif0 p1(i3, vdd, a);
    tranif0 p2(i3, vdd, b);
    tranif0 p3(cn, i3, c);
    tranif0 p4(i4, vdd, b);
    tranif0 p5(cn, i4, a);
    tranif1 n6(cout, gnd, cn);
    tranif0 p6(cout, vdd, cn);
endmodule

```

Transistors are expressed as

```

Transistor-type name(drain, source, gate);

```

`tranif1` corresponds to nMOS transistors that turn ON when the gate is 1 while `tranif0` corresponds to pMOS transistors that turn ON when the gate is 0. Appendix A.11 covers Verilog netlists in more detail.

With the description generated so far, we still do not have the information required to determine the speed or power consumption of the gate. We need to specify the size of the transistors and the stray capacitance. Because Verilog was designed as a switch-level and gate-level language, it is poorly suited to structural descriptions at this level of detail. Hence, we turn to another common structural language used by the circuit simulator SPICE. The specification of the transistor-level carry subcircuit at the circuit level might be represented as follows:

```

.SUBCKT CARRY A B C COUT VDD GND
MN1 I1 A GND GND NMOS W=2U L=0.6U AD=1.8P AS=3P
MN2 I1 B GND GND NMOS W=2U L=0.6U AD=1.8P AS=3P
MN3 CN C I1 GND NMOS W=2U L=0.6U AD=3P AS=3P
MN4 I2 B GND GND NMOS W=2U L=0.6U AD=0.9P AS=3P
MN5 CN A I2 GND NMOS W=2U L=0.6U AD=3P AS=0.9P
MP1 I3 A VDD VDD PMOS W=4U L=0.6U AD=3.6P AS=6P
MP2 I3 B VDD VDD PMOS W=4U L=0.6U AD=3.6P AS=6P
MP3 CN C I3 VDD PMOS W=4U L=0.6U AD=6P AS=6P

```

```

MP4 I4 B VDD VDD PMOS W=4U L=0.6U AD=1.8P AS=6P
MP5 CN A I4 VDD PMOS W=4U L=0.6U AD=6P AS=1.8P
MN6 COUT CN GND GND NMOS W=4U L=0.6U AD=6P AS=6P
MP6 COUT CN VDD VDD PMOS W=8U L=0.6U AD=12P AS=12P
CI1 I1 GND 6FF
CI3 I3 GND 9FF
CA A GND 12FF
CB B GND 12FF
CC C GND 6FF
CCN CN GND 12FF
CCOUT COUT GND 6FF
.ENDS

```

Transistors are specified by lines beginning with an M as follows:

```

Mname drain gate source body type W=width L=length
      AD=drain area AS=source area

```

Although MOS switches have been masquerading as three terminal devices (gate, source, and drain) until this point, they are in fact four terminal devices with the substrate or well forming the *body* terminal. The body connection was not listed in Verilog but is required for SPICE. The type specifies whether the transistor is a p-device or n-device. The width, length, and area parameters specify physical dimensions of the actual transistors. Units include U (micro, 10^{-6}), P (pico, 10^{-12}), and F (femto, 10^{-15}). Capacitors are specified by lines beginning with C as follows:

```

Cname node1 node2 value

```

In this description, the MOS model in SPICE calculates the parasitic capacitances inherent in the MOS transistor using the device dimensions specified. The extra capacitance statements in the above description designate additional routing capacitance not inherent to the device structure. This depends on the physical design of the gate. Long wires also contribute resistance, which increases delay. At the circuit level of structural specification, all connections are given that are necessary to fully characterize the carry gate in terms of speed, power, and connectivity. Chapter 8 describes SPICE models in more detail.

1.10 Physical Design

1.10.1 Floorplanning

Physical design begins with a floorplan. The floorplan estimates the area of major units in the chip and defines their relative placements. The floorplan is essential to determine whether a proposed design will fit in the chip area budgeted and to estimate wiring lengths and wiring congestion. An initial floorplan should be prepared as soon as the logic is loosely defined. As usual, this process involves feedback. The floorplan will often suggest changes to the logic (and microarchitecture), which in turn changes the floorplan. For example, suppose microarchitects assume that a cache requires a 2-cycle access latency. If the floorplan shows that the data cache can be placed adjacent to the execution units in the

datapath, the cache access time might reduce to a single cycle. This could allow the microarchitects to reduce the cache capacity while providing the same performance. Once the cache shrinks, the floorplan must be reconsidered to take advantage of the newly available space near the datapath. As a complex design begins to stabilize, the floorplan is often hierarchically subdivided to describe the functional blocks within the units.

The challenge of floorplanning is estimating the size of each unit without proceeding through a detailed design of the chip (which would depend on the floorplan and wire lengths). This section assumes that good estimates have been made and describes what a floorplan looks like. The next sections describe each of the types of components that might be in a floorplan and suggests ways to estimate the component sizes.

Figure 1.62 shows the chip floorplan for the MIPS processor including the pad frame. The top-level blocks are the controller and datapath. A wiring channel is located between the two blocks to provide room to route control signals to the datapath. The datapath is further partitioned into wordslices. The *pad frame* includes 40 I/O pads, which are wired to the pins on the chip package. There are 29 pads used for signals; the remainder are V_{DD} and GND.

The floorplan is drawn to scale and annotated with dimensions. The chip is designed in a $0.6\ \mu\text{m}$ process on a $1.5 \times 1.5\ \text{mm}$ die so the die is $5000\ \lambda$ on a side. Each pad is $750\ \lambda \times$

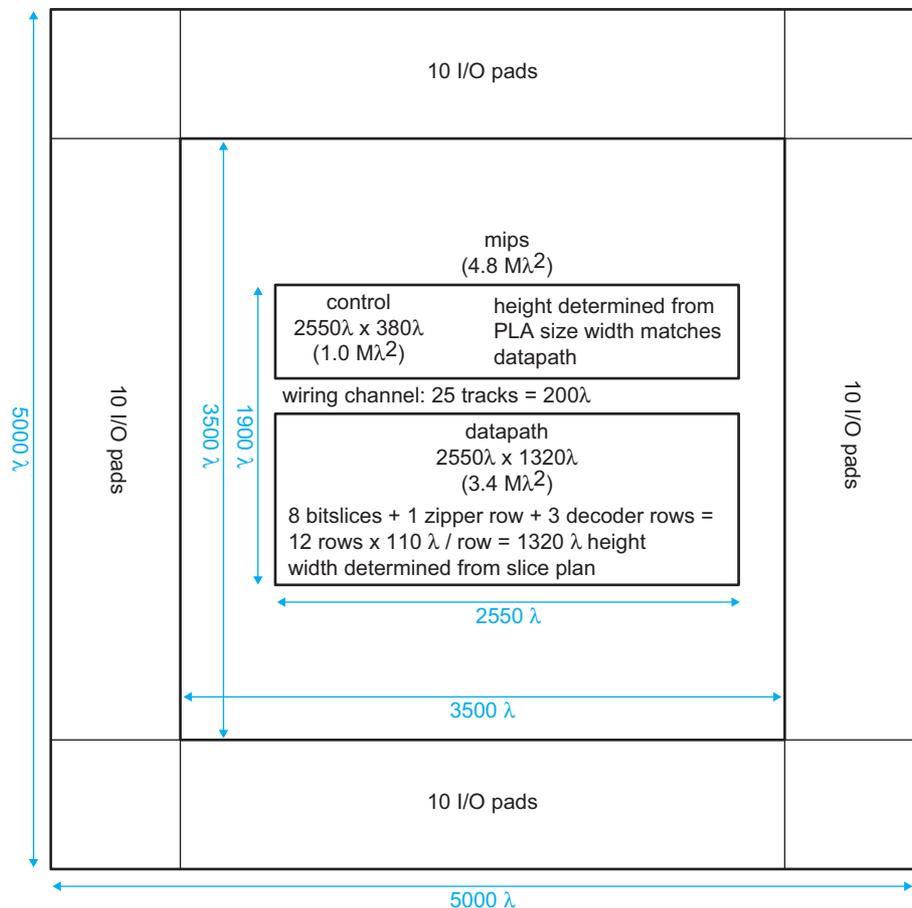


FIGURE 1.62 MIPS floorplan

350λ , so the maximum possible core area inside the pad frame is $3500 \lambda \times 3500 \lambda = 12.25 \text{ M}\lambda^2$. Due to the wiring channel, the actual core area of $4.8 \text{ M}\lambda^2$ is larger than the sum of the block areas. This design is said to be *pad-limited* because the I/O pads set the chip area. Most commercial chips are *core-limited* because the chip area is set by the logic excluding the pads. In general, blocks in a floorplan should be rectangular because it is difficult for a designer to stuff logic into an odd-shaped region (although some CAD tools do so just fine).

Figure 1.63 shows the actual chip layout. Notice the 40 I/O pads around the periphery. Just inside the pad frame are metal2 V_{DD} and GND rings, marked with + and -.

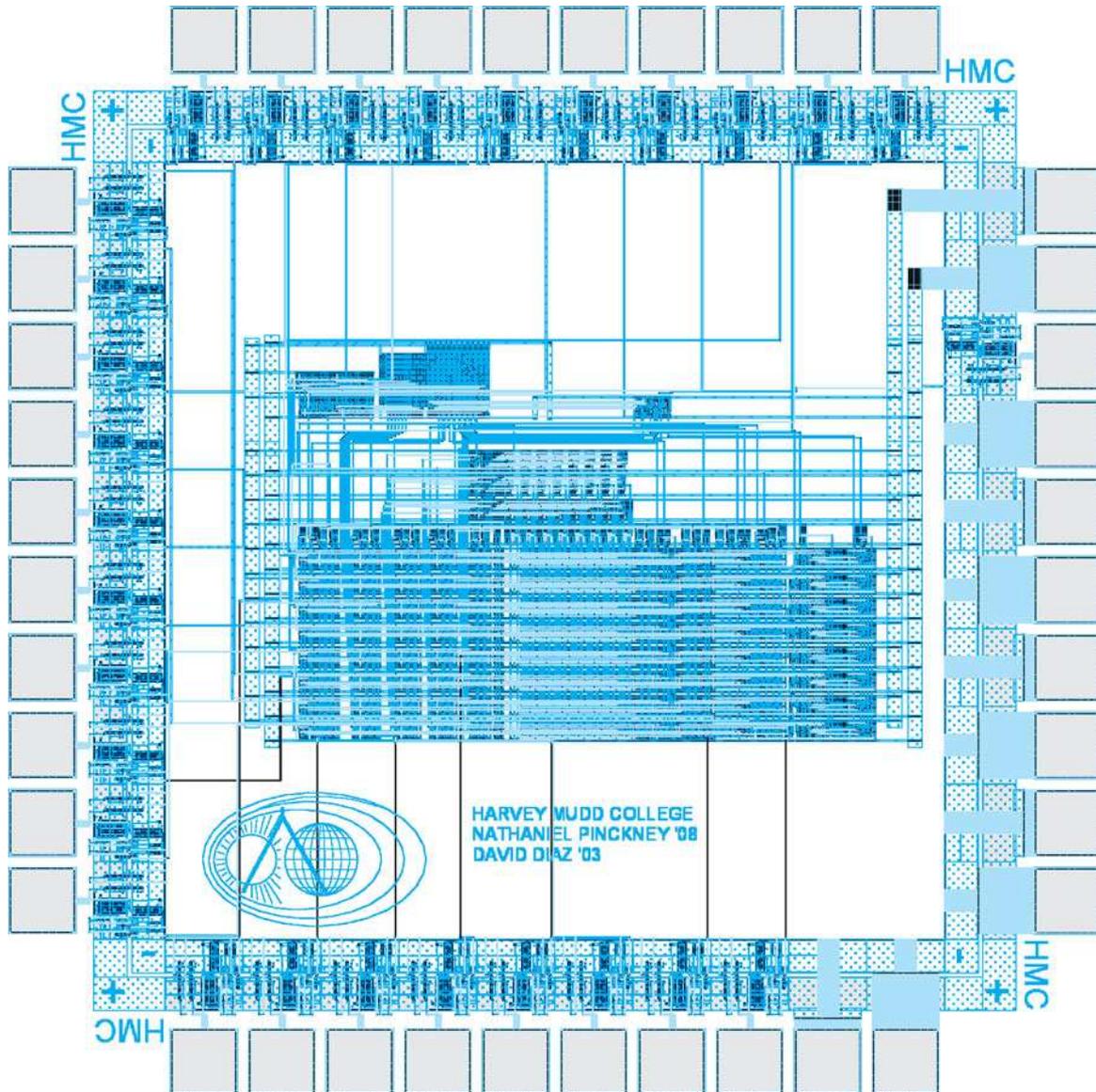


FIGURE 1.63 MIPS layout

On-chip structures can be categorized as *random logic*, *datapaths*, *arrays*, *analog*, and *input/output (I/O)*. Random logic, like the aludecoder, has little structure. Datapaths operate on multi-bit data words and perform roughly the same function on each bit so they consist of multiple N -bit wordslices. Arrays, like RAMs, ROMs, and PLAs, consist of identical cells repeated in two dimensions. Productivity is highest if layout can be reused or automatically generated. Datapaths and arrays are good VLSI building blocks because a single carefully crafted cell is reused in one or two dimensions. Automatic layout generators exist for memory arrays and random logic but are not as mature for datapaths. Therefore, many design methodologies ignore the potential structure of datapaths and instead lay them out with random logic tools except when performance or area are vital. Analog circuits still require careful design and simulation but tend to involve only small amounts of layout because they have relatively few transistors. I/O cells are also highly tuned to each fabrication process and are often supplied by the process vendor.

Random logic and datapaths are typically built from *standard cells* such as inverters, NAND gates, and flip-flops. Standard cells increase productivity because each cell only needs to be drawn and verified once. Often, a standard cell library is purchased from a third party vendor.

Another important decision during floorplanning is to choose the metal orientation. The MIPS floorplan uses horizontal metal1 wires, vertical metal2 wires, and horizontal metal3 wires. Alternating directions between each layer makes it easy to cross wires on different layers.

1.10.2 Standard Cells

A simple standard cell library is shown on the inside front cover. Power and ground run horizontally in metal1. These supply rails are 8λ wide (to carry more current) and are separated by 90λ center-to-center. The nMOS transistors are placed in the bottom 40λ of the cell and the pMOS transistors are placed in the top 50λ . Thus, cells can be connected by abutment with the supply rails and n-well matching up. Substrate and well contacts are placed under the supply rails. Inputs and outputs are provided in metal2, which runs vertically. Each cell is a multiple of 8λ in width so that it offers an integer number of metal2 tracks. Within the cell, poly is run vertically to form gates and diffusion and metal1 are run horizontally, though metal1 can also be run vertically to save space when it does not interfere with other connections.

Cells are tiled in rows. Each row is separated vertically by at least 110λ from the base of the previous row. In a 2-level metal process, horizontal metal1 wires are placed in *routing channels* between the rows. The number of wires that must be routed sets the height of the routing channels. Layout is often generated with automatic place & route tools. Figure 1.64 shows the controller layout generated by such a tool. Note that in this and subsequent layouts, the n-well around the pMOS transistors will usually not be shown.

When more layers of metal are available, routing takes place over the cells and routing channels may become unnecessary. For example, in a 3-level metal process, metal3 is run horizontally on a 10λ pitch. Thus, 11 horizontal tracks can run over each cell. If this is sufficient to accommodate all of the horizontal wires, the routing channels can be eliminated.

Automatic synthesis and place & route tools have become good enough to map entire designs onto standard cells. Figure 1.65 shows the entire 8-bit MIPS processor synthesized from the VHDL model given in Appendix A.12 onto a cell library in a 130 nm process with

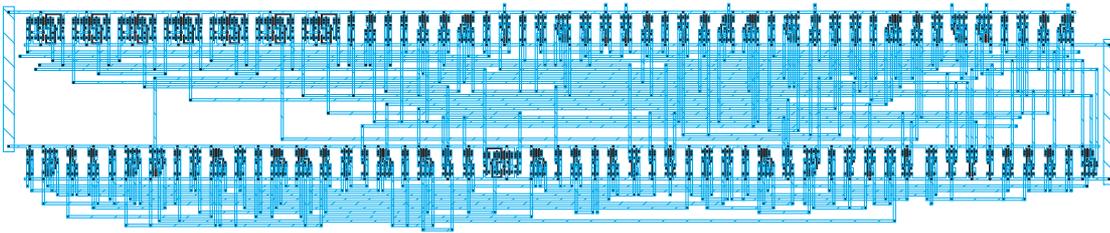


FIGURE 1.64 MIPS controller layout (synthesized)

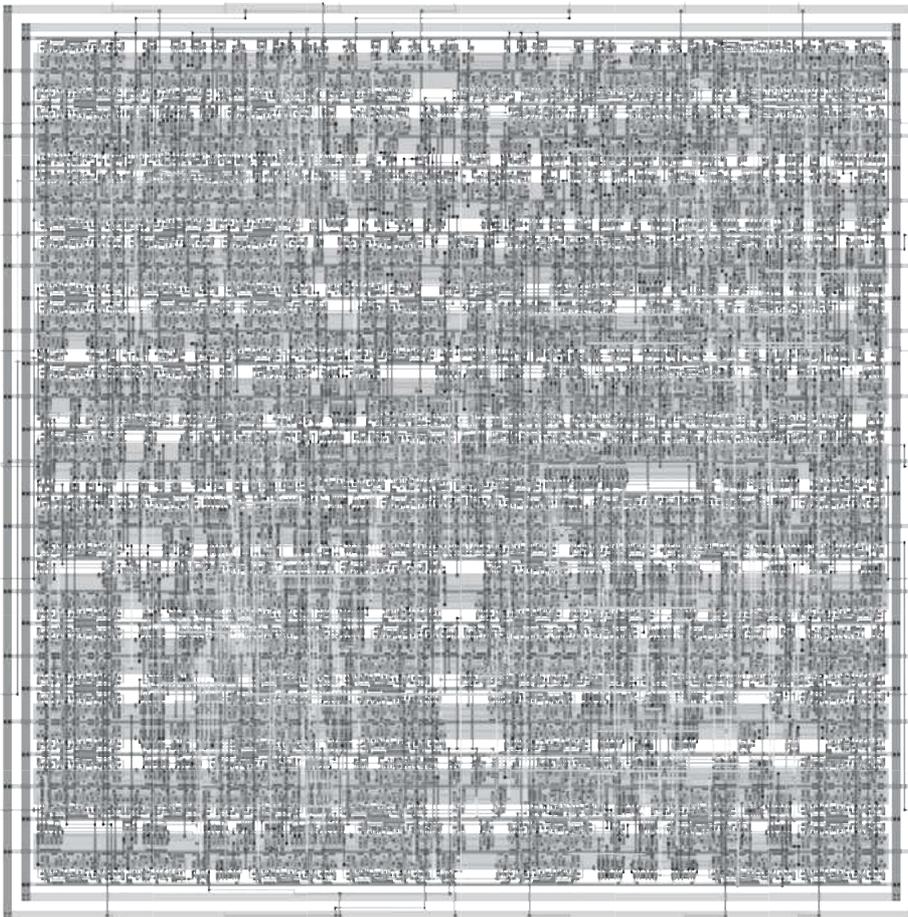


FIGURE 1.65 Synthesized MIPS processor

seven metal layers. Compared to Figure 1.63, the synthesized design shows little discernible structure except that 26 rows of standard cells can be identified beneath the wires. The area is approximately $4 M\lambda^2$. Synthesized designs tend to be somewhat slower than a good custom design, but they also take an order of magnitude less design effort.

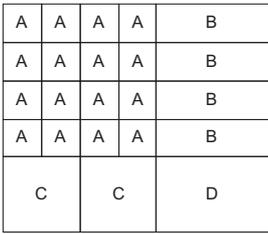


FIGURE 1.66 Pitch-matching of snap-together cells

1.10.3 Pitch Matching

The area of the controller in Figure 1.64 is dominated by the routing channels. When the logic is more regular, layout density can be improved by including the wires in cells that “snap together.” Snap-together cells require more design and layout effort but lead to smaller area and shorter (i.e., faster) wires. The key issue in designing snap-together cells is *pitch-matching*. Cells that connect must have the same size along the connecting edge. Figure 1.66 shows several pitch-matched cells. Reducing the size of cell *D* does not help the layout area. On the other hand, increasing the size of cell *D* also affects the area of *B* and/or *C*.

Figure 1.67 shows the MIPS datapath in more detail. The eight horizontal bitslices are clearly visible. The zipper at the top of the layout includes three rows for the decoder that is pitch-matched to the register file in the datapath. Vertical metal2 wires are used for control, including clocks, multiplexer selects, and register enables. Horizontal metal3 wires run over the tops of cells to carry data along a bitslice.

The width of the transistors in the cells and the number of wires that must run over the datapath determines the minimum height of the datapath cells. 60–100 λ are typical heights for relatively simple datapaths. The width of the cell depends on the cell contents.

1.10.4 Slice Plans

Figure 1.68 shows a *slice plan* of the datapath. The diagram illustrates the ordering of wordslices and the allocation of wiring tracks within each bitslice. Dots indicate that a bus passes over a cell and is also used in that cell. Each cell is annotated with its type and width (in number of tracks). For example, the program counter (pc) is an output of the PC flop and is also used as an input to the srcA and address multiplexers. The slice plan

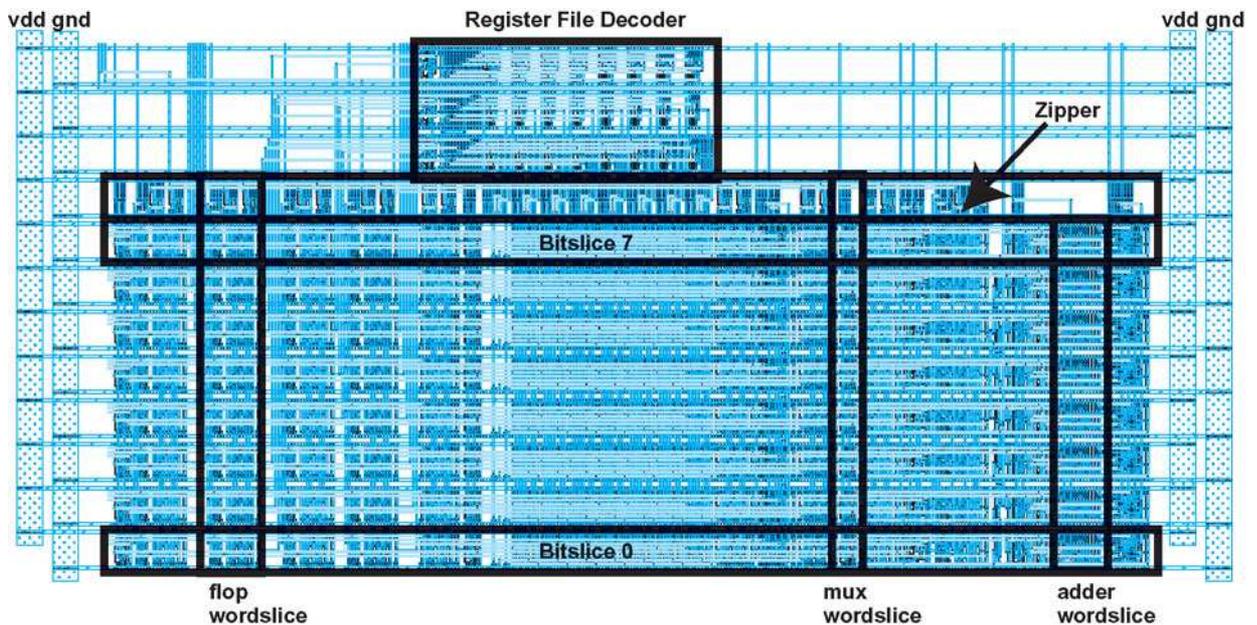


FIGURE 1.67 MIPS datapath layout

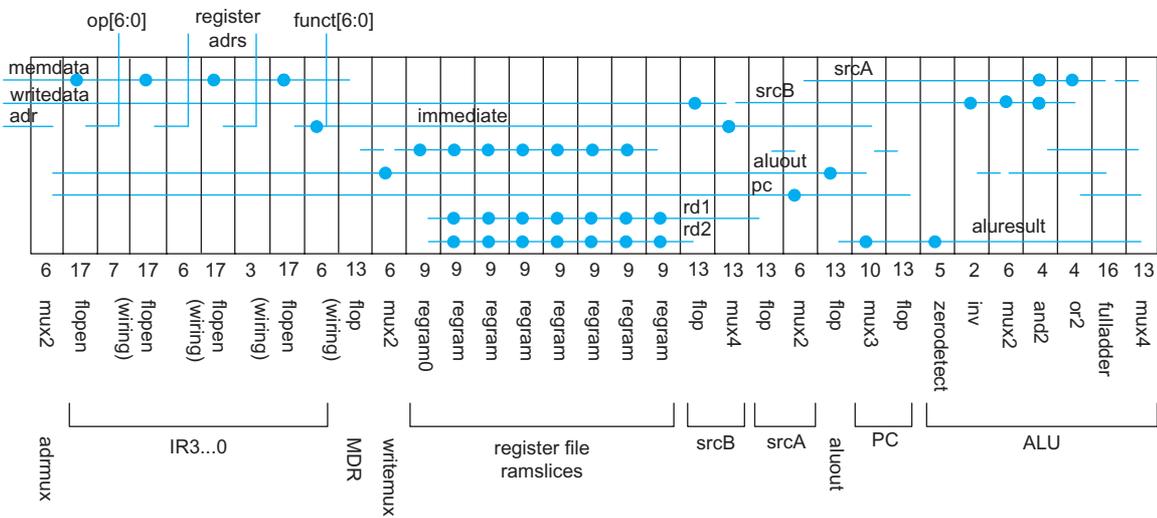


FIGURE 1.68 Datapath slice plan

makes it easy to calculate wire lengths and evaluate wiring congestion before laying out the datapath. In this case, it is evident that the greatest congestion takes place over the register file, where seven wiring tracks are required.

The slice plan is also critical for estimating area of datapaths. Each wordslice is annotated with its width, measured in tracks. This information can be obtained by looking at the cell library layouts. By adding up the widths of each element in the slice plan, we see that the datapath is 319 tracks wide, or 2552λ wide. There are eight bitslices in the 8-bit datapath. In addition, there is one more row for the zipper and three more for the three register file address decoders, giving a total of 12 rows. At a pitch of $110 \lambda / \text{row}$, the datapath is 1320λ tall. The address decoders only occupy a small fraction of their rows, leaving wasted empty space. In a denser design, the controller could share some of the unused area.

1.10.5 Arrays

Figure 1.69 shows a programmable logic array (PLA) used for the control FSM next state and output logic. A PLA can compute any function expressed in sum of products form. The structure on the left is called the AND plane and the structure on the right is the OR plane. PLAs are discussed further in Section 12.7.

This PLA layout uses 2 vertical tracks for each input and 3 for each output plus about 6 for overhead. It uses 1.5 horizontal tracks for each product or minterm, plus about 14 for overhead. Hence, the size of a PLA is easy to calculate. The total PLA area is $500 \lambda \times 350 \lambda$, plus another $336 \lambda \times 220 \lambda$ for the four external flip-flops needed in the control FSM. The height of the controller is dictated by the height of the PLA plus a few wiring tracks to route inputs and outputs. In comparison, the synthesized controller from Figure 1.64 has a size of $1500 \lambda \times 400 \lambda$ because the wiring tracks waste so much space.

1.10.6 Area Estimation

A good floorplan depends on reasonable area estimates, which may be difficult to make before logic is finalized. An experienced designer may be able to estimate block area by

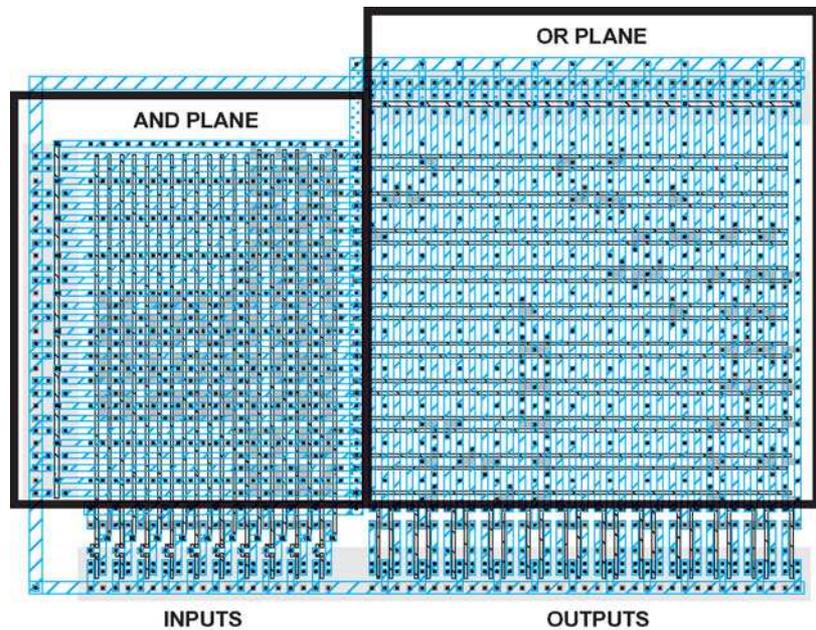


FIGURE 1.69 PLA for control FSM

comparison to the area of a comparable block drawn in the past. In the absence of data for such comparison, Table 1.10 lists some typical numbers. Be certain to account for large wiring channels at a pitch of 8λ / track. Larger transistors clearly occupy a greater area, so this may be factored into the area estimates as a function of W and L (width and length). For memories, don't forget about the decoders and other periphery circuits, which often take as much area as the memory bits themselves. Your mileage may vary, but datapaths and arrays typically achieve higher densities than standard cells.

TABLE 1.10 Typical layout densities

Element	Area
random logic (2-level metal process)	1000 – 1500 λ^2 / transistor
datapath	250 – 750 λ^2 / transistor or $6WL + 360 \lambda^2$ / transistor
SRAM	1000 λ^2 / bit
DRAM (in a DRAM process)	100 λ^2 / bit
ROM	100 λ^2 / bit

Given enough time, it is nearly always possible to shave a few lambda here or there from a design. However, such efforts are seldom a good investment unless an element is repeated so often that it accounts for a major fraction of the chip area or if floorplan errors have led to too little space for a block and the block must be shrunk before the chip can be completed. It is wise to make conservative area estimates in floorplans, especially if there is risk that more functionality may be added to a block.

Some cell library vendors specify typical routed standard cell layout densities in kgates / mm².⁷ Commonly, a gate is defined as a 3-input static CMOS NAND or NOR with six transistors. A 65 nm process ($\lambda \approx 0.03 \mu\text{m}$) with eight metal layers may achieve a density of 160–500 kgates / mm² for random logic. This corresponds to about 370–1160 λ^2 / transistor. Processes with many metal layers obtain high density because routing channels are not needed.

1.11 Design Verification

Integrated circuits are complicated enough that if anything can go wrong, it probably will. Design verification is essential to catching the errors before manufacturing and commonly accounts for half or more of the effort devoted to a chip.

As design representations become more detailed, verification time increases. It is not practical to simulate an entire chip in a circuit-level simulator such as SPICE for a large number of cycles to prove that the layout is correct. Instead, the design is usually tested for functionality at the architectural level with a model in a language such as C and at the logic level by simulating the HDL description. Then, the circuits are checked to ensure that they are a faithful representation of the logic and the layout is checked to ensure it is a faithful representation of the circuits, as shown in Figure 1.70. Circuits and layout must meet timing and power specifications as well.

A *testbench* is used to verify that the logic is correct. The testbench instantiates the logic under test. It reads a file of inputs and expected outputs called *test vectors*, applies them to the module under test, and logs mismatches. Appendix A.12 provides an example of a testbench for verifying the MIPS processor logic.

A number of techniques are available for circuit verification. If the logic is synthesized onto a cell library, the postsynthesis gate-level netlist can be expressed in an HDL again and simulated using the same test vectors. Alternatively, a transistor-level netlist can be simulated against the test vector, although this can result in tricky race conditions for sequential circuits. Powerful *formal verification* tools are also available to check that a circuit performs the same Boolean function as the associated logic. Exotic circuits should be simulated thoroughly to ensure that they perform the intended logic function and have adequate noise margins; circuit pitfalls are discussed throughout this book.

Layout vs. Schematic tools (LVS) check that transistors in a layout are connected in the same way as in the circuit schematic. *Design rule checkers* (DRC) verify that the layout satisfies design rules. *Electrical rule checkers* (ERC) scan for other potential problems such as noise or premature wearout; such problems will also be discussed later in the book.

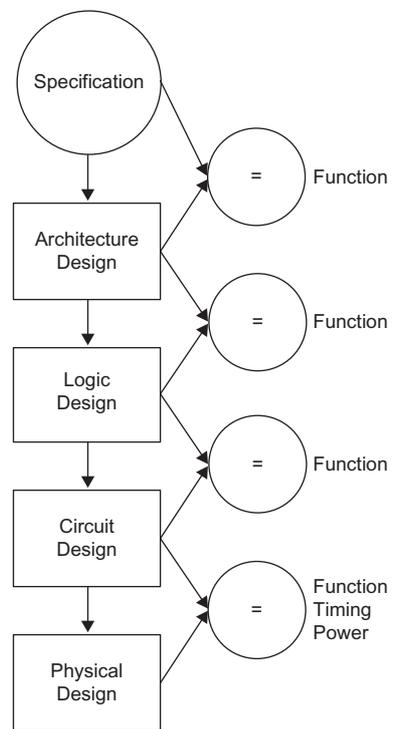


FIGURE 1.70 Design and verification sequence

⁷1 kgate = 1000 gates.

1.12 Fabrication, Packaging, and Testing

Once a chip design is complete, it is taped out for manufacturing. *Tapeout* gets its name from the old practice of writing a specification of masks to magnetic tape; today, the mask descriptions are usually sent to the manufacturer electronically. Two common formats for mask descriptions are the Caltech Interchange Format (CIF) [Mead80] (mainly used in academia) and the Calma GDS II Stream Format (GDS) [Calma84] (used in industry).

Masks are made by etching a pattern of chrome on glass with an electron beam. A set of masks for a nanometer process can be very expensive. For example, masks for a large chip in a 180 nm process may cost on the order of a quarter of a million dollars. In a 65 nm process, the mask set costs about \$3 million. The MOSIS service in the United States and its EURO PRACTICE and VDEC counterparts in Europe and Japan make a single set of masks covering multiple small designs from academia and industry to amortize the cost across many customers. With a university discount, the cost for a run of 40 small chips on a multi-project wafer can run about \$10,000 in a 130 nm process down to \$2000 in a 0.6 μm process. MOSIS offers certain grants to cover fabrication of class project chips.

Integrated circuit fabrication plants (fabs) now cost billions of dollars and become obsolete in a few years. Some large companies still own their own fabs, but an increasing number of fabless semiconductor companies contract out manufacturing to foundries such as TSMC, UMC, and IBM.

Multiple chips are manufactured simultaneously on a single silicon wafer, typically 150–300 mm (6"–12") in diameter. Fabrication requires many deposition, masking, etching, and implant steps. Most fabrication plants are optimized for wafer throughput rather than latency, leading to turnaround times of up to 10 weeks. Figure 1.71 shows an engineer in a *clean room* holding a completed 300 mm wafer. Clean rooms are filtered to eliminate most dust and other particles that could damage a partially processed wafer. The engineer is wearing a “bunny suit” to avoid contaminating the clean room. Figure 1.72 is a



FIGURE 1.71 Engineer holding processed 12-inch wafer (Photograph courtesy of the Intel Corporation.)

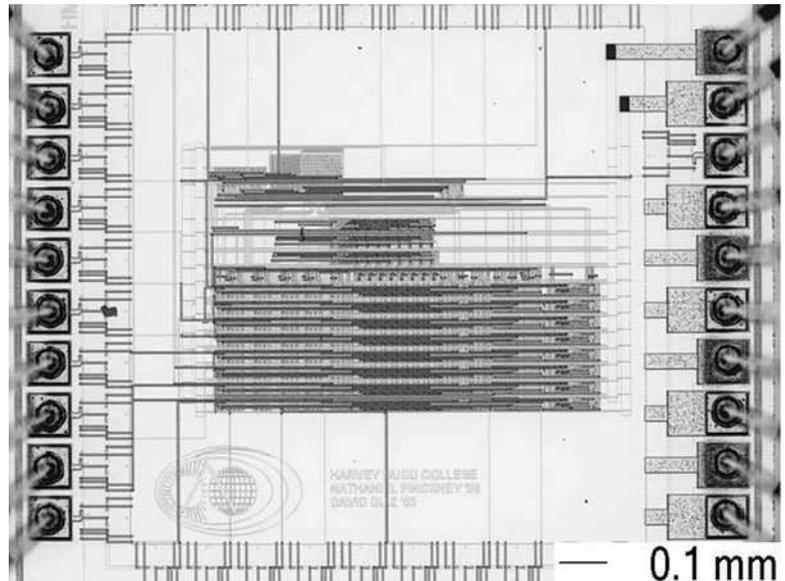


FIGURE 1.72 MIPS processor photomicrograph (only part of pad frame shown)

photomicrograph (a photograph taken under a microscope) of the 8-bit MIPS processor.

Processed wafers are sliced into dice (chips) and packaged. Figure 1.73 shows the 1.5×1.5 mm chip in a 40-pin *dual-inline package* (DIP). This *wire-bonded* package uses thin gold wires to connect the pads on the die to the lead frame in the center cavity of the package. These wires are visible on the pads in Figure 1.72. More advanced packages offer different trade-offs between cost, pin count, pin bandwidth, power handling, and reliability, as will be discussed in Section 13.2. Flip-chip technology places small solder balls directly onto the die, eliminating the bond wire inductance and allowing contacts over the entire chip area rather than just at the periphery.

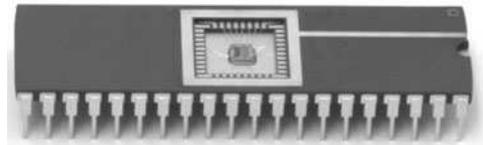


FIGURE 1.73 Chip in a 40-pin dual-inline package

Even tiny defects in a wafer or dust particles can cause a chip to fail. Chips are tested before being sold. Testers capable of handling high-speed chips cost millions of dollars, so many chips use built-in self-test features to reduce the tester time required. Chapter 15 is devoted to design verification and testing.

Summary and a Look Ahead

“If the automobile had followed the same development cycle as the computer, a Rolls-Royce would today cost \$100, get one million miles to the gallon, and explode once a year . . .”

—Robert X. Cringely

CMOS technology, driven by Moore’s Law, has come to dominate the semiconductor industry. This chapter examined the principles of designing a simple CMOS integrated circuit. MOS transistors can be viewed as electrically controlled switches. Static CMOS gates are built from pull-down networks of nMOS transistors and pull-up networks of pMOS transistors. Transistors and wires are fabricated on silicon wafers using a series of deposition, lithography, and etch steps. These steps are defined by a set of masks drawn as a chip layout. Design rules specify minimum width and spacing between elements in the layout. The chip design process can be divided into architecture, logic, circuit, and physical design. The performance, area, and power of the chip are influenced by interrelated decisions made at each level. Design verification plays an important role in constructing such complex systems; the reliability requirements for hardware are much greater than those typically imposed on software.

Primary design objectives include reliability, performance, power, and cost. Any chip should, with high probability, operate reliably for its intended lifetime. For example, the chip must be designed so that it does not overheat or break down from excessive voltage. Performance is influenced by many factors including clock speed and parallelism. CMOS transistors dissipate power every time they switch, so the dynamic power consumption is related to the number and size of transistors and the rate at which they switch. At feature

sizes below 180 nm, transistors also leak a significant amount of current even when they should be OFF. Thus, chips now draw static power even when they are idle. One of the central challenges of VLSI design is making good trade-offs between performance and power for a particular application. The cost of a chip includes nonrecurring engineering (NRE) expenses for the design and masks, along with per-chip manufacturing costs related to the size of the chip. In processes with smaller feature sizes, the per-unit cost goes down because more transistors can be packed into a given area, but the NRE increases. The latest manufacturing processes are only cost-effective for chips that will sell in huge volumes. Nevertheless, plenty of interesting markets exist for chips in mature, inexpensive manufacturing processes.

To quantify how a chip meets these objectives, we must develop and analyze more complete models. The remainder of this book will expand on the material introduced in this chapter. Of course, transistors are not simply switches. Chapter 2 examines the current and capacitance of transistors, which are essential for estimating delay and power. A more detailed description of CMOS processing technology and layout rules is presented in Chapter 3. The next four chapters address the fundamental concerns of circuit designers. The models from Chapter 2 are too detailed to apply by hand to large systems, yet not detailed enough to fully capture the complexity of modern transistors. Chapter 4 develops simplified models to estimate the delay of circuits. If modern chips were designed to squeeze out the ultimate possible performance without regard to power, they would burn up. Thus, it is essential to estimate and trade off the power consumption against performance. Moreover, low power consumption is crucial to mobile battery-operated systems. Power is considered in Chapter 5. Wires are as important as transistors in their contribution to overall performance and power, and are discussed in Chapter 6. Chapter 7 addresses design of robust circuits with a high yield and low failure rate.

Simulation is discussed in Chapter 8 and is used to obtain more accurate performance and power predictions as well as to verify the correctness of circuits and logic. Chapter 9 considers combinational circuit design. A whole kit of circuit families are available with different trade-offs in speed, power, complexity, and robustness. Chapter 10 continues with sequential circuit design, including clocking and latching techniques.

The next three chapters delve into CMOS subsystems. Chapter 11 catalogs designs for a host of datapath subsystems including adders, shifters, multipliers, and counters. Chapter 12 similarly describes memory subsystems including SRAMs, DRAMs, CAMs, ROMs, and PLAs. Chapter 13 addresses special-purpose subsystems including power distribution, clocking, and I/O.

The final chapters address practicalities of CMOS system design. Chapter 14 focuses on a range of current design methods, identifying the issues peculiar to CMOS. Testing, design-for-test, and debugging techniques are discussed in Chapter 15. Hardware description languages (HDLs) are used in the design of nearly all digital integrated circuits today. Appendix A provides side-by-side tutorials for 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. To keep the length of this book under control, some optional topics have been published on the Internet rather than in print. These sections can be found at www.cmosvlsi.com and are labeled with a “Web Enhanced” icon. A companion text, *Digital VLSI Chip Design with Cadence and Synopsys CAD Tools* [Brunvand09], covers practical details of using the leading industrial CAD tools to build chips.

Exercises

- 1.1 Extrapolating the data from Figure 1.4, predict the transistor count of a microprocessor in 2016.
- 1.2 Search the Web for transistor counts of Intel's more recent microprocessors. Make a graph of transistor count vs. year of introduction from the Pentium Processor in 1993 to the present on a semilogarithmic scale. How many months pass between doubling of transistor counts?
- 1.3 As the cost of a transistor drops from a microbuck ($\$10^{-6}$) toward a nanobuck, what opportunities can you imagine to change the world with integrated circuits?
- 1.4 Read a biography or history about a major event in the development of integrated circuits. For example, see *Crystal Fire* by Lillian Hoddeson, *Microchip* by Jeffrey Zygmunt, or *The Pentium Chronicles* by Robert Colwell. Pick a team or individual that made a major contribution to the field. In your opinion, what were the characteristics that led to success? What traits of the team management would you seek to emulate or avoid in your own professional life?
- 1.5 Sketch a transistor-level schematic for a CMOS 4-input NOR gate.
- 1.6 Sketch a transistor-level schematic for a compound CMOS logic gate for each of the following functions:
 - a) $Y = \overline{ABC + D}$
 - b) $Y = \overline{(\overline{AB} + C) \cdot D}$
 - c) $Y = \overline{AB} + C \cdot (A + B)$
- 1.7 Use a combination of CMOS gates (represented by their symbols) to generate the following functions from A , B , and C .
 - a) $Y = A$ (buffer)
 - b) $Y = \overline{AB} + \overline{A}B$ (XOR)
 - c) $Y = \overline{A}\overline{B} + AB$ (XNOR)
 - d) $Y = AB + BC + AC$ (majority)
- 1.8 Sketch a transistor-level schematic of a CMOS 3-input XOR gate. You may assume you have both true and complementary versions of the inputs available.
- 1.9 Sketch transistor-level schematics for the following logic functions. You may assume you have both true and complementary versions of the inputs available.
 - a) A 2:4 decoder defined by

$$Y_0 = \overline{A_0} \cdot \overline{A_1}$$

$$Y_1 = A_0 \cdot \overline{A_1}$$

$$Y_2 = \overline{A_0} \cdot A_1$$

$$Y_3 = A_0 \cdot A_1$$
 - b) A 3:2 priority encoder defined by

$$Y_0 = \overline{A_0} \cdot (A_1 + \overline{A_2})$$

$$Y_1 = \overline{A_0} \cdot \overline{A_1}$$

- 1.10 Sketch a stick diagram for a CMOS 4-input NOR gate from Exercise 1.5.
- 1.11 Estimate the area of your 4-input NOR gate from Exercise 1.10.
- 1.12 Using a CAD tool of your choice, layout a 4-input NOR gate. How does its size compare to the prediction from Exercise 1.11?
- 1.13 Figure 1.74 shows a stick diagram of a 2-input NAND gate. Sketch a side view (cross-section) of the gate from X to X' .
- 1.14 Figure 1.75 gives a stick diagram for a level-sensitive latch. Estimate the area of the latch.
- 1.15 Draw a transistor-level schematic for the latch of Figure 1.75. How does the schematic differ from Figure 1.31(b)?
- 1.16 Consider the design of a CMOS compound OR-AND-INVERT (OAI21) gate computing $F = (\overline{A + B}) \cdot \overline{C}$.
- sketch a transistor-level schematic
 - sketch a stick diagram
 - estimate the area from the stick diagram
 - layout your gate with a CAD tool using unit-sized transistors
 - compare the layout size to the estimated area

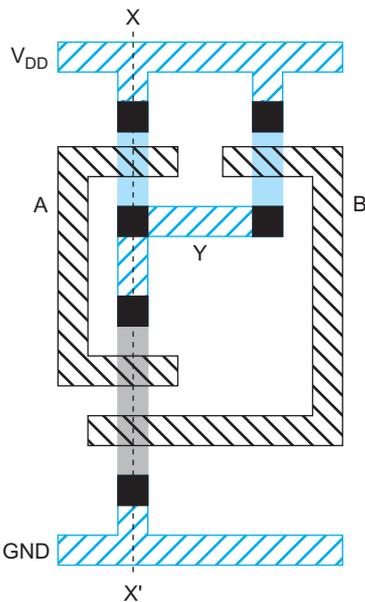


FIGURE 1.74 2-input NAND gate stick diagram

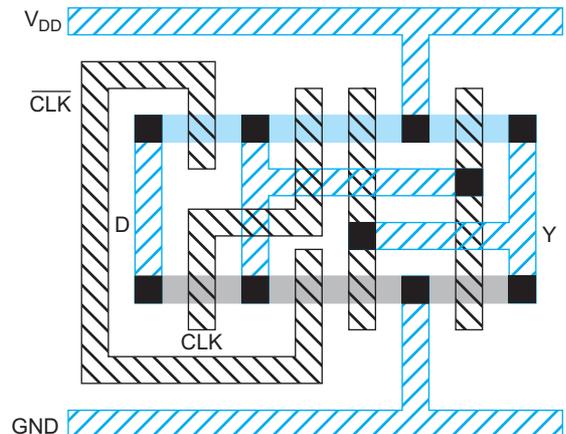


FIGURE 1.75 Level-sensitive latch stick diagram

- 1.17 Consider the design of a CMOS compound OR-OR-AND-INVERT (OAI22) gate computing $F = \overline{(A + B)} \cdot (C + \overline{D})$.
- sketch a transistor-level schematic
 - sketch a stick diagram
 - estimate the area from the stick diagram
 - layout your gate with a CAD tool using unit-sized transistors
 - compare the layout size to the estimated area
- 1.18 A 3-input majority gate returns a true output if at least two of the inputs are true. A minority gate is its complement. Design a 3-input CMOS minority gate using a single stage of logic.
- sketch a transistor-level schematic
 - sketch a stick diagram
 - estimate the area from the stick diagram
- 1.19 Design a 3-input minority gate using CMOS NANDs, NORs, and inverters. How many transistors are required? How does this compare to a design from Exercise 1.18(a)?
- 1.20 A carry lookahead adder computes $G = G_3 + P_3(G_2 + P_2(G_1 + P_1G_0))$. Consider designing a compound gate to compute \overline{G} .
- sketch a transistor-level schematic
 - sketch a stick diagram
 - estimate the area from the stick diagram
- 1.21 www.cmosvlsi.com has a series of four labs in which you can learn VLSI design by completing the multicycle MIPS processor described in this chapter. The labs use the open-source Electric CAD tool or commercial tools from Cadence and Synopsys. They cover the following:
- leaf cells: schematic entry, layout, icons, simulation, DRC, ERC, LVS; hierarchical design
 - datapath design: wordslices, ALU assembly, datapath routing
 - control design: random logic or PLAs
 - chip assembly, pad frame, global routing, full-chip verification, tapeout

This page intentionally left blank

MOS Transistor Theory

2

2.1 Introduction

In Chapter 1, the Metal-Oxide-Semiconductor (MOS) transistor was introduced in terms of its operation as an ideal switch. As we saw in Section 1.9, the performance and power of a chip depend on the current and capacitance of the transistors and wires. In this chapter, we will examine the characteristics of MOS transistors in more detail; Chapter 6 addresses wires.

Figure 2.1 shows some of the symbols that are commonly used for MOS transistors. The three-terminal symbols in Figure 2.1(a) are used in the great majority of schematics. If the body (substrate or well) connection needs to be shown, the four-terminal symbols in Figure 2.1(b) will be used. Figure 2.1(c) shows an example of other symbols that may be encountered in the literature.

The MOS transistor is a *majority-carrier* device in which the current in a conducting channel between the source and drain is controlled by a voltage applied to the gate. In an nMOS transistor, the majority carriers are electrons; in a pMOS transistor, the majority carriers are holes. The behavior of MOS transistors can be understood by first examining an isolated MOS structure with a gate and body but no source or drain. Figure 2.2 shows a simple MOS structure. The top layer of the structure is a good conductor called the *gate*. Early transistors used metal gates. Transistor gates soon changed to use polysilicon, i.e., silicon formed from many small crystals, although metal gates are making a resurgence at 65 nm and beyond, as will be seen in Section 3.4.1.3. The middle layer is a very thin insulating film of SiO_2 called the *gate oxide*. The bottom layer is the doped silicon body. The figure shows a p-type body in which the carriers are holes. The body is grounded and a voltage is applied to the gate. The gate oxide is a good insulator so almost zero current flows from the gate to the body.¹

In Figure 2.2(a), a negative voltage is applied to the gate, so there is negative charge on the gate. The mobile positively charged holes are attracted to the region beneath the gate. This is called the *accumulation* mode. In Figure 2.2(b), a small positive voltage is applied to the gate, resulting in some positive charge on the gate. The holes in the body are repelled from the region directly beneath the gate, resulting in a *depletion* region forming below the gate. In Figure 2.2(c), a higher positive potential exceeding a critical threshold voltage V_t is applied, attracting more positive charge to the gate. The holes are repelled further and some free electrons in the body are attracted to the region beneath the gate. This conductive layer of electrons in the p-type body is called the *inversion* layer. The threshold

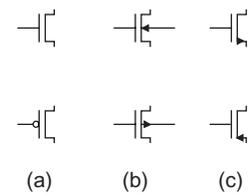


FIGURE 2.1
MOS transistor symbols

¹Gate oxides are now only a handful of atomic layers thick and carriers sometimes tunnel through the oxide, creating a current through the gate. This effect is explored in Section 2.4.4.2.

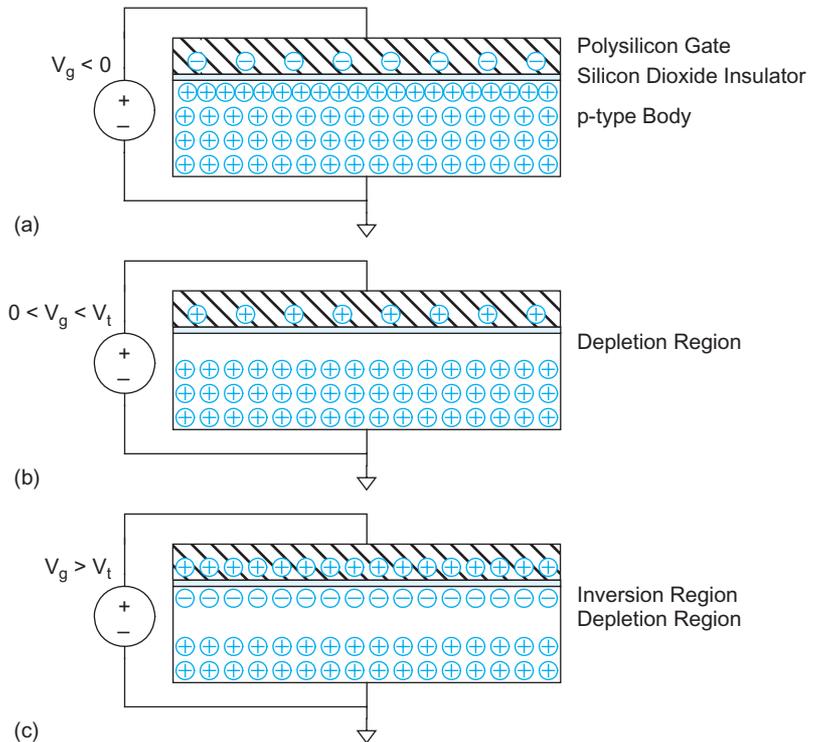


FIGURE 2.2 MOS structure demonstrating (a) accumulation, (b) depletion, and (c) inversion

voltage depends on the number of dopants in the body and the thickness t_{ox} of the oxide. It is usually positive, as shown in this example, but can be engineered to be negative.

Figure 2.3 shows an nMOS transistor. The transistor consists of the MOS stack between two n-type regions called the *source* and *drain*. In Figure 2.3(a), the gate-to-source voltage V_{gs} is less than the threshold voltage. The source and drain have free electrons. The body has free holes but no free electrons. Suppose the source is grounded. The junctions between the body and the source or drain are zero-biased or reverse-biased, so little or no current flows. We say the transistor is OFF, and this mode of operation is called *cutoff*. It is often convenient to approximate the current through an OFF transistor as zero, especially in comparison to the current through an ON transistor. Remember, however, that small amounts of current leaking through OFF transistors can become significant, especially when multiplied by millions or billions of transistors on a chip. In Figure 2.3(b), the gate voltage is greater than the threshold voltage. Now an inversion region of electrons (majority carriers) called the *channel* connects the source and drain, creating a conductive path and turning the transistor ON. The number of carriers and the conductivity increases with the gate voltage. The potential difference between drain and source is $V_{ds} = V_{gs} - V_{gd}$. If $V_{ds} = 0$ (i.e., $V_{gs} = V_{gd}$), there is no electric field tending to push current from drain to source.

When a small positive potential V_{ds} is applied to the drain (Figure 2.3(c)), current I_{ds} flows through the channel from drain to source.² This mode of operation is termed *linear*,

²The terminology of source and drain might initially seem backward. Recall that the current in an nMOS transistor is carried by moving electrons with a negative charge. Therefore, positive current from drain to source corresponds to electrons flowing from their source to their drain.

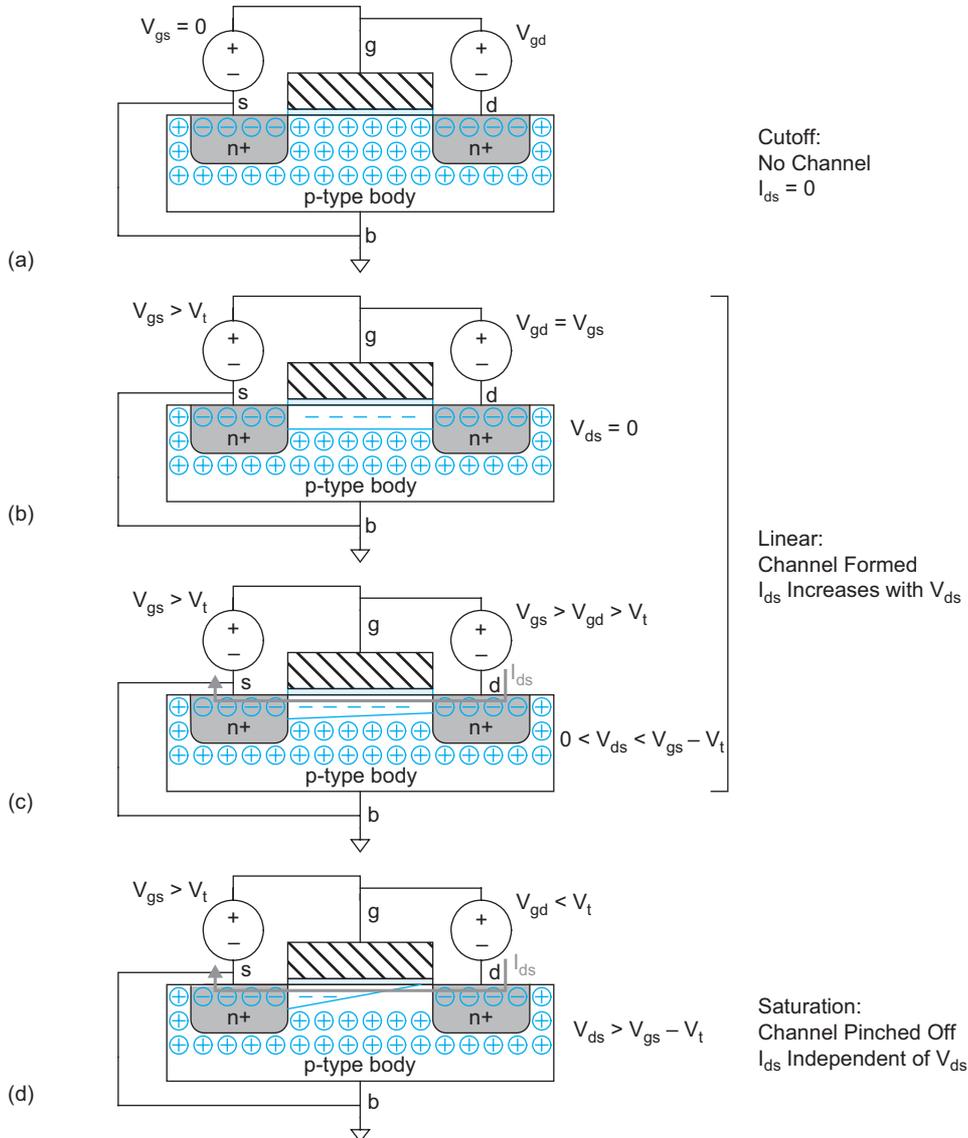


FIGURE 2.3 nMOS transistor demonstrating cutoff, linear, and saturation regions of operation

resistive, triode, nonsaturated, or unsaturated; the current increases with both the drain voltage and gate voltage. If V_{ds} becomes sufficiently large that $V_{gd} < V_t$, the channel is no longer inverted near the drain and becomes *pinched off* (Figure 2.3(d)). However, conduction is still brought about by the drift of electrons under the influence of the positive drain voltage. As electrons reach the end of the channel, they are injected into the depletion region near the drain and accelerated toward the drain. Above this drain voltage the current I_{ds} is controlled only by the gate voltage and ceases to be influenced by the drain. This mode is called *saturation*.

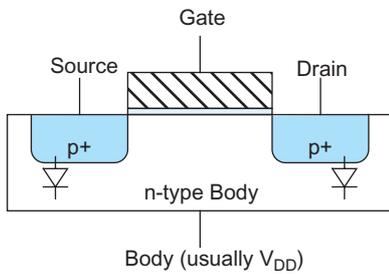


FIGURE 2.4 pMOS transistor

In summary, the nMOS transistor has three modes of operation. If $V_{gs} < V_t$, the transistor is cutoff (OFF). If $V_{gs} > V_t$, the transistor turns ON. If V_{ds} is small, the transistor acts as a linear resistor in which the current flow is proportional to V_{ds} . If $V_{gs} > V_t$ and V_{ds} is large, the transistor acts as a current source in which the current flow becomes independent of V_{ds} .

The pMOS transistor in Figure 2.4 operates in just the opposite fashion. The n-type body is tied to a high potential so the junctions with the p-type source and drain are normally reverse-biased. When the gate is also at a high potential, no current flows between drain and source. When the gate voltage is lowered by a threshold V_t , holes are attracted to form a p-type channel immediately beneath the gate, allowing current to flow between drain and source.

The threshold voltages of the two types of transistors are not necessarily equal, so we use the terms V_{tn} and V_{tp} to distinguish the nMOS and pMOS thresholds.

Although MOS transistors are symmetrical, by convention we say that majority carriers flow from their source to their drain. Because electrons are negatively charged, the source of an nMOS transistor is the more negative of the two terminals. Holes are positively charged so the source of a pMOS transistor is the more positive of the two terminals. In static CMOS gates, the source is the terminal closer to the supply rail and the drain is the terminal closer to the output.

We begin in Section 2.2 by deriving an ideal model relating current and voltage (I-V) for a transistor. The delay of MOS circuits is determined by the time required for this current to charge or discharge the capacitance of the circuits. Section 2.3 investigates transistor capacitances. The gate of an MOS transistor is inherently a good capacitor with a thin dielectric; indeed, its capacitance is responsible for attracting carriers to the channel and thus for the operation of the device. The p-n junctions from source or drain to the body contribute additional *parasitic* capacitance. The capacitance of wires interconnecting the transistors is also important and will be explored in Section 6.2.2.

This idealized I-V model provides a general qualitative understanding of transistor behavior but is of limited quantitative value. On the one hand, it neglects too many effects that are important in transistors with short channel lengths L . Therefore, the model is not sufficient to calculate current accurately. Circuit simulators based on SPICE [Nagel75] use models such as BSIM that capture transistor behavior quite thoroughly but require entire books to fully describe [Cheng99]. Chapter 8 discusses simulation with SPICE. The most important effects seen in these simulations that impact digital circuit designers are examined in Section 2.4. On the other hand, the idealized I-V model is still too complicated to use in back-of-the-envelope calculations tuning the performance of large circuits. Therefore, we will develop even simpler models for performance estimation in Chapter 4.

Section 2.5 wraps up this chapter by applying the I-V models to understand the DC transfer characteristics of CMOS gates and pass transistors.

2.2 Long-Channel I-V Characteristics

As stated previously, MOS transistors have three regions of operation:

- Cutoff or subthreshold region
- Linear region
- Saturation region

Let us derive a model [Shockley52, Cobbold70, Sah64] relating the current and voltage (I-V) for an nMOS transistor in each of these regions. The model assumes that the channel length is long enough that the lateral electric field (the field between source and drain) is relatively low, which is no longer the case in nanometer devices. This model is variously known as the *long-channel*, *ideal*, *first-order*, or *Shockley* model. Subsequent sections will refine the model to reflect high fields, leakage, and other nonidealities.

The long-channel model assumes that the current through an OFF transistor is 0. When a transistor turns ON ($V_{gs} > V_t$), the gate attracts carriers (electrons) to form a channel. The electrons drift from source to drain at a rate proportional to the electric field between these regions. Thus, we can compute currents if we know the amount of charge in the channel and the rate at which it moves. We know that the charge on each plate of a capacitor is $Q = CV$. Thus, the charge in the channel Q_{channel} is

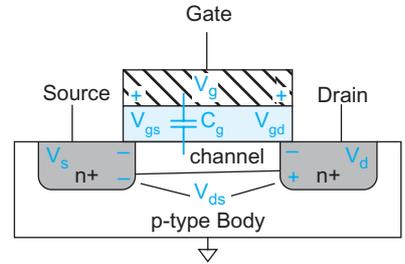
$$Q_{\text{channel}} = C_g (V_{gs} - V_t) \quad (2.1)$$

where C_g is the capacitance of the gate to the channel and $V_{gs} - V_t$ is the amount of voltage attracting charge to the channel beyond the minimum required to invert from p to n. The gate voltage is referenced to the channel, which is not grounded. If the source is at V_s and the drain is at V_d , the average is $V_c = (V_s + V_d)/2 = V_s + V_{ds}/2$. Therefore, the mean difference between the gate and channel potentials V_{gc} is $V_g - V_c = V_{gs} - V_{ds}/2$, as shown in Figure 2.5.

We can model the gate as a parallel plate capacitor with capacitance proportional to area over thickness. If the gate has length L and width W and the oxide thickness is t_{ox} , as shown in Figure 2.6, the capacitance is

$$C_g = k_{\text{ox}} \epsilon_0 \frac{WL}{t_{\text{ox}}} = \epsilon_{\text{ox}} \frac{WL}{t_{\text{ox}}} = C_{\text{ox}} WL \quad (2.2)$$

where ϵ_0 is the permittivity of free space, 8.85×10^{-14} F/cm, and the permittivity of SiO_2 is $k_{\text{ox}} = 3.9$ times as great. Often, the $\epsilon_{\text{ox}}/t_{\text{ox}}$ term is called C_{ox} , the capacitance per unit area of the gate oxide.



Average gate to channel potential:

$$V_{gc} = (V_{gs} + V_{gd})/2 = V_{gs} - V_{ds}/2$$

FIGURE 2.5 Average gate to channel voltage

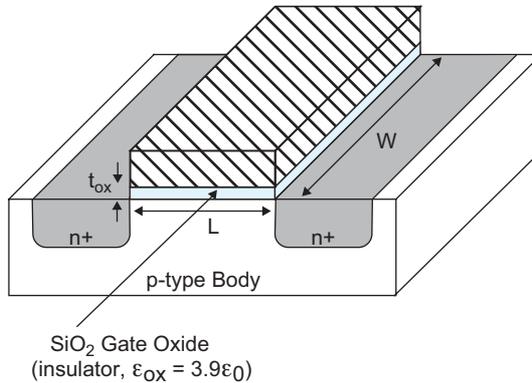


FIGURE 2.6 Transistor dimensions

Some nanometer processes use a different gate dielectric with a higher dielectric constant. In these processes, we call t_{ox} the *equivalent oxide thickness* (EOT), the thickness of a layer of SiO₂ that has the same C_{ox} . In this case, t_{ox} is thinner than the actual dielectric.

Each carrier in the channel is accelerated to an average velocity, v , proportional to the lateral electric field, i.e., the field between source and drain. The constant of proportionality μ is called the *mobility*.

$$v = \mu E \quad (2.3)$$

A typical value of μ for electrons in an nMOS transistor with low electric fields is 500–700 cm²/V·s. However, most transistors today operate at far higher fields where the mobility is severely curtailed (see Section 2.4.1).

The electric field E is the voltage difference between drain and source V_{ds} divided by the channel length

$$E = \frac{V_{ds}}{L} \quad (2.4)$$

The time required for carriers to cross the channel is the channel length divided by the carrier velocity: L/v . Therefore, the current between source and drain is the total amount of charge in the channel divided by the time required to cross

$$\begin{aligned} I_{ds} &= \frac{Q_{\text{channel}}}{L/v} \\ &= \mu C_{\text{ox}} \frac{W}{L} (V_{gs} - V_t - V_{ds}/2) V_{ds} \\ &= \beta (V_{GT} - V_{ds}/2) V_{ds} \end{aligned} \quad (2.5)$$

where

$$\beta = \mu C_{\text{ox}} \frac{W}{L}; \quad V_{GT} = V_{gs} - V_t \quad (2.6)$$

The term $V_{gs} - V_t$ arises so often that it is convenient to abbreviate it as V_{GT} . EQ (2.5) describes the linear region of operation, for $V_{gs} > V_t$, but V_{ds} relatively small. It is called *linear* or *resistive* because when $V_{ds} \ll V_{GT}$, I_{ds} increases almost linearly with V_{ds} , just like an ideal resistor. The geometry and technology-dependent parameters are sometimes merged into a single factor β . Do not confuse this use of β with the same symbol used for the ratio of collector-to-base current in a bipolar transistor. Some texts [Gray01] lump the technology-dependent parameters alone into a constant called “ k prime.”³

$$k' = \mu C_{\text{ox}} \quad (2.7)$$

If $V_{ds} > V_{\text{dsat}} \equiv V_{GT}$, the channel is no longer inverted in the vicinity of the drain; we say it is pinched off. Beyond this point, called the *drain saturation voltage*, increasing the drain voltage has no further effect on current. Substituting $V_{ds} = V_{\text{dsat}}$ at this point of maximum current into EQ (2.5), we find an expression for the saturation current that is independent of V_{ds} .

$$I_{ds} = \frac{\beta}{2} V_{GT}^2 \quad (2.8)$$

³Other sources (e.g., MOSIS) define $k' = \frac{\mu C_{\text{ox}}}{2}$; check the definition before using quoted data.

This expression is valid for $V_{gs} > V_t$ and $V_{ds} > V_{dsat}$. Thus, long-channel MOS transistors are said to exhibit *square-law behavior* in saturation.

Two key figures of merit for a transistor are I_{on} and I_{off} . I_{on} (also called I_{dsat}) is the ON current, I_{ds} , when $V_{gs} = V_{ds} = V_{DD}$. I_{off} is the OFF current when $V_{gs} = 0$ and $V_{ds} = V_{DD}$. According to the long-channel model, $I_{off} = 0$ and

$$I_{on} = \frac{\beta}{2}(V_{DD} - V_t)^2 \quad (2.9)$$

EQ.(2.10) summarizes the current in the three regions:

$$I_{ds} = \begin{cases} 0 & V_{gs} < V_t & \text{Cutoff} \\ \beta(V_{GT} - V_{ds}/2)V_{ds} & V_{ds} < V_{dsat} & \text{Linear} \\ \frac{\beta}{2}V_{GT}^2 & V_{ds} > V_{dsat} & \text{Saturation} \end{cases} \quad (2.10)$$

Example 2.1

Consider an nMOS transistor in a 65 nm process with a minimum drawn channel length of 50 nm ($\lambda = 25$ nm). Let $W/L = 4/2 \lambda$ (i.e., $0.1/0.05 \mu\text{m}$). In this process, the gate oxide thickness is 10.5 \AA . Estimate the high-field mobility of electrons to be $80 \text{ cm}^2/\text{V}\cdot\text{s}$ at 70°C . The threshold voltage is 0.3 V . Plot I_{ds} vs. V_{ds} for $V_{gs} = 0, 0.2, 0.4, 0.6, 0.8,$ and 1.0 V using the long-channel model.

SOLUTION: We first calculate β .

$$\beta = \mu C_{ox} \frac{W}{L} = \left(80 \frac{\text{cm}^2}{\text{V}\cdot\text{s}}\right) \left(\frac{3.9 \times 8.85 \times 10^{-14} \frac{\text{F}}{\text{cm}}}{10.5 \times 10^{-8} \text{cm}}\right) \left(\frac{W}{L}\right) = 262 \frac{W}{L} \frac{\text{A}}{\text{V}^2} \quad (2.11)$$

Figure 2.7(a) shows the I-V characteristics for the transistor. According to the first-order model, the current is zero for gate voltages below V_t . For higher gate voltages, current increases linearly with V_{ds} for small V_{ds} . As V_{ds} reaches the saturation point $V_{dsat} = V_{GT}$, current rolls off and eventually becomes independent of V_{ds} when the transistor is saturated. We will later see that the Shockley model overestimates current at high voltage because it does not account for mobility degradation and velocity saturation caused by the high electric fields.

pMOS transistors behave in the same way, but with the signs of all voltages and currents reversed. The I-V characteristics are in the third quadrant, as shown in Figure 2.7(b). To keep notation simple in this text, we will disregard the signs and just remember that the current flows from source to drain in a pMOS transistor. The mobility of holes in silicon is typically lower than that of electrons. This means that pMOS transistors provide less current than nMOS transistors of comparable size and hence are slower. The symbols μ_n and μ_p are used to distinguish mobility of electrons and of holes in nMOS and pMOS transistors, respectively. The *mobility ratio* μ_n/μ_p is typically 2–3; we will generally use 2 for examples in this book. The pMOS transistor has the same geometry as the nMOS in Figure 2.7(a), but with $\mu_p = 40 \text{ cm}^2/\text{V}\cdot\text{s}$ and $V_{tp} = -0.3 \text{ V}$. Similarly, $\beta_n, \beta_p, k'_n,$ and k'_p are sometimes used to distinguish nMOS and pMOS I-V characteristics.

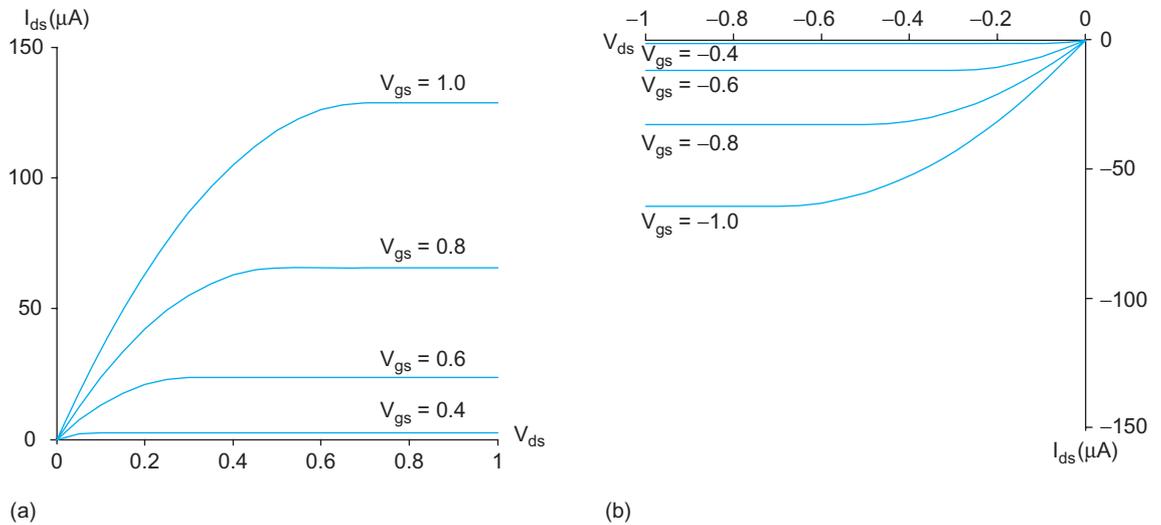


FIGURE 2.7 I-V characteristics of ideal $4/2 \lambda$ (a) nMOS and (b) pMOS transistors

2.3 C-V Characteristics

Each terminal of an MOS transistor has capacitance to the other terminals. In general, these capacitances are nonlinear and voltage dependent (C-V); however, they can be approximated as simple capacitors when their behavior is averaged across the switching voltages of a logic gate. This section first presents simple models of each capacitance suitable for estimating delay and power consumption of transistors. It then explores more detailed models used for circuit simulation. The more detailed models may be skipped on a first reading.

2.3.1 Simple MOS Capacitance Models

The gate of an MOS transistor is a good capacitor. Indeed, its capacitance is necessary to attract charge to invert the channel, so high gate capacitance is required to obtain high I_{ds} . As seen in Section 2.2, the gate capacitor can be viewed as a parallel plate capacitor with the gate on top and channel on bottom with the thin oxide dielectric between. Therefore, the capacitance is

$$C_g = C_{ox}WL \quad (2.12)$$

The bottom plate of the capacitor is the channel, which is not one of the transistor's terminals. When the transistor is on, the channel extends from the source (and reaches the drain if the transistor is unsaturated, or stops short in saturation). Thus, we often approximate the gate capacitance as terminating at the source and call the capacitance C_{gs} .

Most transistors used in logic are of minimum manufacturable length because this results in greatest speed and lowest dynamic power consumption.⁴ Thus, taking this mini-

⁴Some designs use slightly longer than minimum transistors that have higher thresholds because of the short-channel effect (see Sections 2.4.3.3 and 5.3.3). This avoids the cost of an extra mask step for high- V_t transistors. The change in channel length is small (~ 5 – 10%), so the change in gate capacitance is minor.

mum L as a constant for a particular process, we can define

$$C_g = C_{\text{permicron}} \times W \quad (2.13)$$

where

$$C_{\text{permicron}} = C_{\text{ox}} L = \frac{\epsilon_{\text{ox}}}{t_{\text{ox}}} L \quad (2.14)$$

Notice that if we develop a more advanced manufacturing process in which both the channel length and oxide thickness are reduced by the same factor, $C_{\text{permicron}}$ remains unchanged. This relationship is handy for quick calculations but not exact; $C_{\text{permicron}}$ has fallen from about 2 fF/ μm in old processes to about 1 fF/ μm at the 90 and 65 nm nodes. Table 8.5 lists gate capacitance for a variety of processes.

In addition to the gate, the source and drain also have capacitances. These capacitances are not fundamental to operation of the devices, but do impact circuit performance and hence are called *parasitic* capacitors. The source and drain capacitances arise from the p-n junctions between the source or drain diffusion and the body and hence are also called *diffusion*⁵ capacitance C_{sb} and C_{db} . A *depletion region* with no free carriers forms along the junction. The depletion region acts as an insulator between the conducting p- and n-type regions, creating capacitance across the junction. The capacitance of these junctions depends on the area and perimeter of the source and drain diffusion, the depth of the diffusion, the doping levels, and the voltage. As diffusion has both high capacitance and high resistance, it is generally made as small as possible in the layout. Three types of diffusion regions are frequently seen, illustrated by the two series transistors in Figure 2.8. In Figure

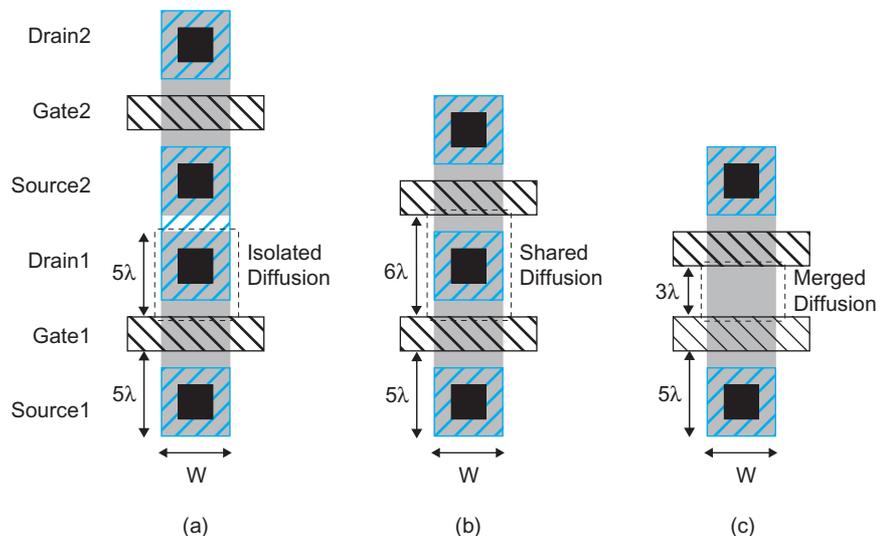


FIGURE 2.8 Diffusion region geometries

⁵Device engineers more properly call this *depletion* capacitance, but the term *diffusion* capacitance is widely used by circuit designers.

2.8(a), each source and drain has its own *isolated* region of contacted diffusion. In Figure 2.8(b), the drain of the bottom transistor and source of the top transistor form a *shared* contacted diffusion region. In Figure 2.8(c), the source and drain are *merged* into an uncontacted region. The average capacitance of each of these types of regions can be calculated or measured from simulation as a transistor switches between V_{DD} and GND. Table 8.5 also lists the capacitance for each scenario for a variety of processes.

For the purposes of hand estimation, you can observe that the diffusion capacitance C_{sb} and C_{db} of contacted source and drain regions is comparable to the gate capacitance (e.g., 1–2 fF/ μm of gate width). The diffusion capacitance of the uncontacted source or drain is somewhat less because the area is smaller but the difference is usually unimportant for hand calculations. These values of $C_g = C_{sb} = C_{db} \approx 1\text{fF}/\mu\text{m}$ will be used in examples throughout the text, but you should obtain the appropriate data for your process using methods to be discussed in Section 8.4.



2.3.2 Detailed MOS Gate Capacitance Model

The MOS gate sits above the channel and may partially overlap the source and drain diffusion areas. Therefore, the gate capacitance has two components: the intrinsic capacitance C_{gc} (over the channel) and the overlap capacitances C_{gol} (to the source and drain).

The intrinsic capacitance was approximated as a simple parallel plate in EQ. (2.12) with capacitance $C_0 = WLC_{ox}$. However, the bottom plate of the capacitor depends on the mode of operation of the transistor. The intrinsic capacitance has three components representing the different terminals connected to the bottom plate: C_{gb} (gate-to-body), C_{gs} (gate-to-source), and C_{gd} (gate-to-drain). Figure 2.9(a) plots capacitance vs. V_{gs} in the cut-off region and for small V_{ds} , while 2.9(b) plots capacitance vs. V_{ds} in the linear and saturation regions [Dally98].

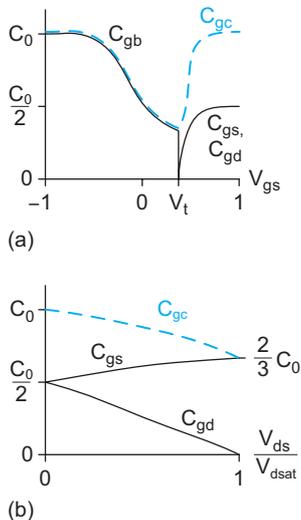


FIGURE 2.9 Intrinsic gate capacitance $C_{gc} = C_{gs} + C_{gd} + C_{gb}$ as a function of (a) V_{gs} and (b) V_{ds}

1. *Cutoff.* When the transistor is OFF ($V_{gs} < V_t$), the channel is not inverted and charge on the gate is matched with opposite charge from the body. This is called C_{gb} , the gate-to-body capacitance. For negative V_{gs} , the transistor is in accumulation and $C_{gb} = C_0$. As V_{gs} increases but remains below a threshold, a depletion region forms at the surface. This effectively moves the bottom plate downward from the oxide, reducing the capacitance, as shown in Figure 2.9(a).
2. *Linear.* When $V_{gs} > V_t$, the channel inverts and again serves as a good conductive bottom plate. However, the channel is connected to the source and drain, rather than the body, so C_{gb} drops to 0. At low values of V_{ds} , the channel charge is roughly shared between source and drain, so $C_{gs} = C_{gd} = C_0/2$. As V_{ds} increases, the region near the drain becomes less inverted, so a greater fraction of the capacitance is attributed to the source and a smaller fraction to the drain, as shown in Figure 2.9(b).
3. *Saturation.* At $V_{ds} > V_{dsat}$, the transistor saturates and the channel pinches off. At this point, all the intrinsic capacitance is to the source, as shown in Figure 2.9(b). Because of pinchoff, the capacitance in saturation reduces to $C_{gs} = 2/3 C_0$ for an ideal transistor [Gray01].

The behavior in these three regions can be approximated as shown in Table 2.1.

TABLE 2.1 Approximation for intrinsic MOS gate capacitance

Parameter	Cutoff	Linear	Saturation
C_{gb}	$\leq C_0$	0	0
C_{gs}	0	$C_0/2$	$2/3 C_0$
C_{gd}	0	$C_0/2$	0
$C_g = C_{gs} + C_{gd} + C_{gb}$	C_0	C_0	$2/3 C_0$

The gate overlaps the source and drain in a real device and also has fringing fields terminating on the source and drain. This leads to additional overlap capacitances, as shown in Figure 2.10. These capacitances are proportional to the width of the transistor. Typical values are $C_{gsol} = C_{gdol} = 0.2 - 0.4 \text{ fF}/\mu\text{m}$. They should be added to the intrinsic gate capacitance to find the total.

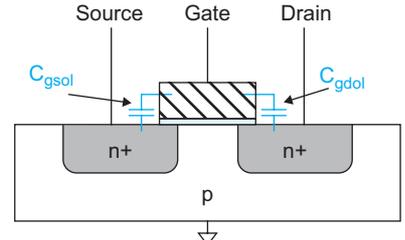


FIGURE 2.10 Overlap capacitance

$$C_{gsol(overlap)} = C_{gsol}W$$

$$C_{gdol(overlap)} = C_{gdol}W \tag{2.15}$$

It is convenient to view the gate capacitance as a single-terminal capacitor attached to the gate (with the other side not switching). Because the source and drain actually form second terminals, the effective gate capacitance varies with the switching activity of the source and drain. Figure 2.11 shows the effective gate capacitance in a $0.35 \mu\text{m}$ process for seven different combinations of source and drain behavior [Bailey98].

More accurate modeling of the gate capacitance may be achieved by using a charge-based model [Cheng99]. For the purpose of delay calculation of digital circuits, we usually approximate $C_g = C_{gs} + C_{gd} + C_{gb} \approx C_0 + 2C_{gsol}W$ or use an effective capacitance extracted

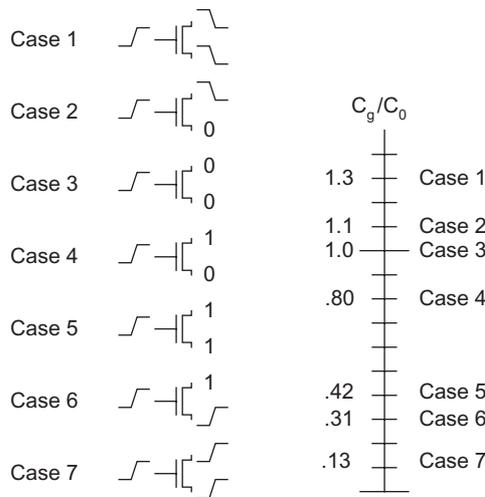


FIGURE 2.11 Data-dependent gate capacitance

from simulation [Nose00b]. It is important to remember that this model significantly overestimates the capacitance of transistors operating just below threshold.



2.3.3 Detailed MOS Diffusion Capacitance Model

As mentioned in Section 2.3.1, the p–n junction between the source diffusion and the body contributes parasitic capacitance across the depletion region. The capacitance depends on both the *area* AS and *sidewall perimeter* PS of the source diffusion region. The geometry is illustrated in Figure 2.12. The area is $AS = WD$. The perimeter is $PS = 2W + 2D$. Of this perimeter, W abuts the channel and the remaining $W + 2D$ does not.

The total source parasitic capacitance is

$$C_{sb} = AS \times C_{jbs} + PS \times C_{jbsw} \quad (2.16)$$

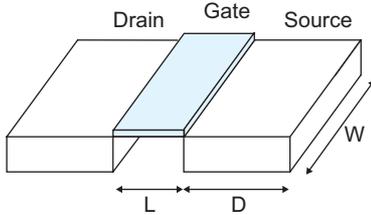


FIGURE 2.12 Diffusion region geometry

where C_{jbs} (the capacitance of the junction between the body and the bottom of the source) has units of capacitance/area and C_{jbsw} (the capacitance of the junction between the body and the side walls of the source) has units of capacitance/length.

Because the depletion region thickness depends on the bias conditions, these parasitics are nonlinear. The area junction capacitance term is [Gray01]

$$C_{jbs} = C_J \left(1 + \frac{V_{sb}}{\psi_0} \right)^{-M_J} \quad (2.17)$$

C_J is the junction capacitance at zero bias and is highly process-dependent. M_J is the *junction grading coefficient*, typically in the range of 0.5 to 0.33 depending on the abruptness of the diffusion junction. ψ_0 is the *built-in potential* that depends on doping levels.

$$\psi_0 = v_T \ln \frac{N_A N_D}{n_i^2} \quad (2.18)$$

v_T is the *thermal voltage* from thermodynamics, not to be confused with the threshold voltage V_t . It has a value equal to kT/q (26 mV at room temperature), where $k = 1.380 \times 10^{-23}$ J/K is Boltzmann's constant, T is absolute temperature (300 K at room temperature), and $q = 1.602 \times 10^{-19}$ C is the charge of an electron. N_A and N_D are the doping levels of the body and source diffusion region. n_i is the intrinsic carrier concentration in undoped silicon and has a value of 1.45×10^{10} cm⁻³ at 300 K.

The sidewall capacitance term is of a similar form but uses different coefficients.

$$C_{jbsw} = C_{JSW} \left(1 + \frac{V_{sb}}{\psi_{SW}} \right)^{-M_{JSW}} \quad (2.19)$$

In processes below about 0.35 μm that employ shallow trench isolation surrounding transistors with an SiO₂ insulator (see Section 3.2.6), the sidewall capacitance along the non-conductive trench tends to be minimal, while the sidewall facing the channel is more significant. In some SPICE models, the capacitance of this sidewall abutting the gate and channel is specified with another set of parameters:

$$C_{jbswg} = C_{JSWG} \left(1 + \frac{V_{sb}}{\psi_{SWG}} \right)^{-M_{JSWG}} \quad (2.20)$$

Section 8.3.4 discusses SPICE perimeter capacitance models further.

The drain diffusion has a similar parasitic capacitance dependent on AD , PD , and V_{db} . Equivalent relationships hold for pMOS transistors, but doping levels differ. As the capacitances are voltage-dependent, the most useful information to digital designers is the value averaged across a switching transition. This is the C_{sb} or C_{db} value that was presented in Section 2.3.1.

Example 2.2

Calculate the diffusion parasitic C_{db} of the drain of a unit-sized contacted nMOS transistor in a 65 nm process when the drain is at 0 V and again at $V_{DD} = 1.0$ V. Assume the substrate is grounded. The diffusion region conforms to the design rules from Figure 2.8 with $\lambda = 25$ nm. The transistor characteristics are $CJ = 1.2$ fF/ μm^2 , $MJ = 0.33$, $CJSW = 0.1$ fF/ μm , $CJSWG = 0.36$ fF/ μm , $MJSW = MJSWG = 0.10$, and $\psi_0 = 0.7$ V at room temperature.

SOLUTION: From Figure 2.8, we find a unit-size diffusion contact is $4 \times 5 \lambda$, or $0.1 \times 0.125 \mu\text{m}$. The area is $0.0125 \mu\text{m}^2$ and perimeter is $0.35 \mu\text{m}$ plus $0.1 \mu\text{m}$ along the channel. At zero bias, $C_{jbd} = 1.2$ fF/ μm^2 , $C_{jbdsw} = 0.1$ fF/ μm , and $C_{jbdswg} = 0.36$ fF/ μm . Hence, the total capacitance is

$$C_{db}(0 \text{ V}) = \left(0.0125 \mu\text{m}^2\right) \left(1.2 \frac{\text{fF}}{\mu\text{m}^2}\right) + \left(0.35 \mu\text{m}\right) \left(0.1 \frac{\text{fF}}{\mu\text{m}}\right) + \left(0.1 \mu\text{m}\right) \left(0.36 \frac{\text{fF}}{\mu\text{m}}\right) = 0.086 \text{ fF} \quad (2.21)$$

At a drain voltage of V_{DD} , the capacitance reduces to

$$C_{db}(1 \text{ V}) = \left(0.0125 \mu\text{m}^2\right) \left(1.2 \frac{\text{fF}}{\mu\text{m}^2}\right) \left(1 + \frac{1.0}{0.7}\right)^{-0.33} + \left[\left(0.35 \mu\text{m}\right) \left(0.1 \frac{\text{fF}}{\mu\text{m}}\right) + \left(0.1 \mu\text{m}\right) \left(0.36 \frac{\text{fF}}{\mu\text{m}}\right)\right] \left(1 + \frac{1.0}{0.7}\right)^{-0.10} = 0.076 \text{ fF} \quad (2.22)$$

For the purpose of manual performance estimation, this nonlinear capacitance is too much effort. An effective capacitance averaged over the switching range is quite satisfactory for digital applications. In this example, the effective drain capacitance would be approximated as the average of the two extremes, 0.081 fF.

Diffusion regions were historically used for short wires called *runners* in processes with only one or two metal levels. Diffusion capacitance and resistance are large enough that such practice is now discouraged; diffusion regions should be kept as small as possible on nodes that switch.

In summary, an MOS transistor can be viewed as a four-terminal device with capacitances between each terminal pair, as shown in Figure 2.13. The gate capacitance includes an intrinsic component (to the body, source and drain, or source alone, depending on operating regime) and overlap terms with the source and drain. The source and drain have parasitic diffusion capacitance to the body.

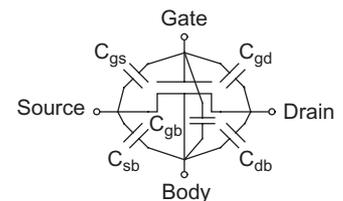


FIGURE 2.13 Capacitance of an MOS transistor

2.4 Nonideal I-V Effects

The long-channel I-V model of EQ (2.10) neglects many effects that are important to devices with channel lengths below 1 micron. This section summarizes the effects of greatest significance to designers, then models each one in more depth.

Figure 2.14 compares the simulated I-V characteristics of a 1-micron wide nMOS transistor in a 65 nm process to the ideal characteristics computed in Section 2.2. The saturation current increases less than quadratically with increasing V_{gs} . This is caused by two effects: velocity saturation and mobility degradation. At high lateral field strengths (V_{ds}/L), carrier velocity ceases to increase linearly with field strength. This is called *velocity saturation* and results in lower I_{ds} than expected at high V_{ds} . At high vertical field strengths (V_{gs}/t_{ox}), the carriers scatter off the oxide interface more often, slowing their progress. This *mobility degradation* effect also leads to less current than expected at high V_{gs} . The saturation current of the nonideal transistor increases somewhat with V_{ds} . This is caused by *channel length modulation*, in which higher V_{ds} increases the size of the depletion region around the drain and thus effectively shortens the channel.

The threshold voltage indicates the gate voltage necessary to invert the channel and is primarily determined by the oxide thickness and channel doping levels. However, other fields in the transistor have some effect on the channel, effectively modifying the threshold voltage. Increasing the potential between the source and body raises the threshold through the *body effect*. Increasing the drain voltage lowers the threshold through *drain-induced barrier lowering*. Increasing the channel length raises the threshold through the *short channel effect*.

Several sources of leakage result in current flow in nominally OFF transistors. When $V_{gs} < V_t$, the current drops off exponentially rather than abruptly becoming zero. This is called *subthreshold conduction*. The current into the gate I_g is ideally 0. However, as the

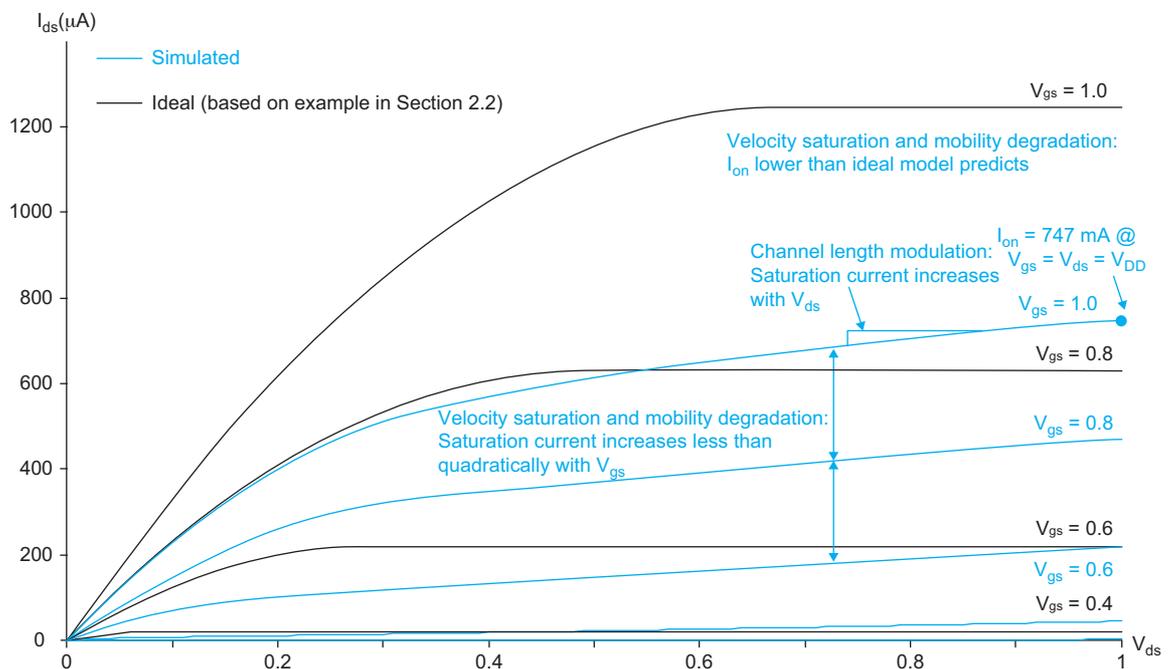


FIGURE 2.14 Simulated and ideal I-V characteristics

thickness of gate oxides reduces to only a small number of atomic layers, electrons *tunnel* through the gate, causing some *gate leakage* current. The source and drain diffusions are typically reverse-biased diodes and also experience *junction leakage* into the substrate or well.

Both mobility and threshold voltage decrease with rising temperature. The mobility effect tends to dominate for strongly ON transistors, resulting in lower I_{ds} at high temperature. The threshold effect is most important for OFF transistors, resulting in higher leakage current at high temperature. In summary, MOS characteristics degrade with temperature.

It is useful to have a qualitative understanding of nonideal effects to predict their impact on circuit behavior and to be able to anticipate how devices will change in future process generations. However, the effects lead to complicated I-V characteristics that are hard to directly apply in hand calculations. Instead, the effects are built into good transistor models and simulated with SPICE or similar software.

2.4.1 Mobility Degradation and Velocity Saturation

Recall from EQ(2.3) that carrier drift velocity, and hence current, is proportional to the lateral electric field $E_{lat} = V_{ds}/L$ between source and drain. The constant of proportionality is called the carrier mobility, μ . The long-channel model assumed that carrier mobility is independent of the applied fields. This is a good approximation for low fields, but breaks down when strong lateral or vertical fields are applied.

As an analogy, imagine that you have been working all night in the VLSI lab and decide to run down and across the courtyard to the coffee cart.⁶ The number of hours you have been up is analogous to the lateral electric field. The longer you have been up, the faster you want to reach coffee: Your speed equals your fatigue times your mobility. There is a strong wind blowing in the courtyard, analogous to the vertical electric field. This wind buffets you against the wall, slowing your progress. In the same way, a high voltage at the gate of the transistor attracts the carriers to the edge of the channel, causing collisions with the oxide interface that slow the carriers. This is called mobility degradation. Moreover, freshman physics is just letting out of the lecture hall. Occasionally, you bounce off a confused freshman, fall down, and have to get up and start running again. This is analogous to carriers scattering off the silicon lattice (technically called collisions with optical phonons). The faster you try to go, the more often you collide. Beyond a certain level of fatigue, you reach a maximum average speed. In the same way, carriers approach a maximum velocity v_{sat} when high fields are applied. This phenomenon is called velocity saturation.⁷

Mobility degradation can be modeled by replacing μ with a smaller μ_{eff} that is a function of V_{gs} . A universal model [Chen96, Chen97] that matches experimental data from multiple processes reasonably well is

$$\mu_{eff-n} = \frac{540 \frac{\text{cm}^2}{\text{V} \cdot \text{s}}}{1 + \left(\frac{V_{gs} + V_t}{0.54 \frac{\text{V}}{\text{nm}} t_{ox}} \right)^{1.85}} \quad \mu_{eff-p} = \frac{185 \frac{\text{cm}^2}{\text{V} \cdot \text{s}}}{1 + \frac{|V_{gs} + 1.5V_t|}{0.338 \frac{\text{V}}{\text{nm}} t_{ox}}} \quad (2.23)$$

⁶This practice has been observed empirically, but is not recommended. Productivity decreases with fatigue. Beyond a certain point of exhaustion, the net work accomplished per hour becomes negative because so many mistakes are made.

⁷Do not confuse the *saturation* region of transistor operation (where $V_{ds} > V_{gs} - V_t$) with *velocity saturation* (where $E_{lat} = V_{ds}/L$ approaches E_c). In this text, the word “saturation” alone refers to the operating region while “velocity saturation” refers to the limiting of carrier velocity at high field.

Example 2.3

Compute the effective mobilities for nMOS and pMOS transistors when they are fully ON. Use the physical parameters from Example 2.1.

SOLUTION: Use $V_{gs} = 1.0$ for ON transistors, remembering that we are treating voltages as positive in a pMOS transistor. Substituting $V_t = 0.3$ V and $t_{ox} = 1.05$ nm into EQ (2.23) gives:

$$\mu_{\text{eff-n}}(V_{gs} = 1.0) = 96 \text{ cm}^2/\text{V}, \mu_{\text{eff-p}}(V_{gs} = 1.0) = 36 \text{ cm}^2/\text{V}$$

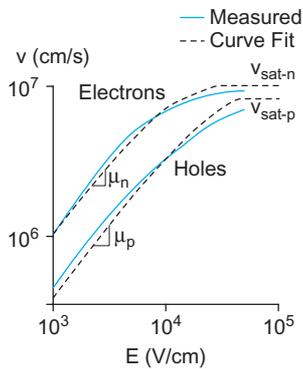


FIGURE 2.15 Carrier velocity vs. electric field at 300 K, adapted from [Jacoboni77]. Velocity saturates at high fields.

Figure 2.15 shows measured data for carrier velocity as a function of the electric field, E , between the drain and source. At low fields, the velocity increases linearly with the field. The slope is the mobility, μ_{eff} . At fields above a critical level, E_c , the velocity levels out at v_{sat} , which is approximately 10^7 cm/s for electrons and 8×10^6 cm/s for holes [Muller03]. As shown in the figure, the velocity can be approximated reasonably well with the following expression [Toh88, Takeuchi94]:

$$v = \begin{cases} \frac{\mu_{\text{eff}} E}{1 + \frac{E}{E_c}} & E < E_c \\ v_{\text{sat}} & E \geq E_c \end{cases} \quad (2.24)$$

where, by continuity, the *critical electric field* is

$$E_c = \frac{2v_{\text{sat}}}{\mu_{\text{eff}}} \quad (2.25)$$

The *critical voltage* V_c is the drain-source voltage at which the critical effective field is reached: $V_c = E_c L$.

Example 2.4

Find the critical voltage for fully ON nMOS and pMOS transistors using the effective mobilities from Example 2.3.

SOLUTION: Using EQ (2.25)

$$V_{c-n} = \frac{2 \left(10^7 \frac{\text{cm}}{\text{s}} \right)}{96 \frac{\text{cm}^2}{\text{V} \cdot \text{s}}} \left(5 \times 10^{-6} \text{ cm} \right) = 1.04 \text{ V}$$

$$V_{c-p} = \frac{2 \left(8 \times 10^6 \frac{\text{cm}}{\text{s}} \right)}{36 \frac{\text{cm}^2}{\text{V} \cdot \text{s}}} \left(5 \times 10^{-6} \text{ cm} \right) = 2.22 \text{ V}$$

The nMOS transistor is velocity saturated in normal operation because V_{c-n} is comparable to V_{DD} . The pMOS transistor has lower mobility and thus is not as badly velocity saturated.

Using a derivation similar to that of Section 2.2 with the new carrier velocity expression in EQ (2.24) gives modified equations for linear and saturation currents [Sodini84].

$$I_{ds} = \begin{cases} \frac{\mu_{\text{eff}}}{V_{ds}} C_{\text{ox}} \frac{W}{L} (V_{GT} - V_{ds}/2) V_{ds} & V_{ds} < V_{\text{dsat}} & \text{Linear} \\ 1 + \frac{V_{ds}}{V_c} & & \\ C_{\text{ox}} W (V_{GT} - V_{\text{dsat}}) v_{\text{sat}} & V_{ds} > V_{\text{dsat}} & \text{Saturation} \end{cases} \quad (2.26)$$

Note that μ_{eff} is a decreasing function of V_{gs} , because of mobility degradation. Observe that the current in the linear regime is the same as in EQ(2.5) except that the mobility term is reduced by a factor related to V_{ds} . At sufficiently high lateral fields, the current saturates at some value dependent on the maximum carrier velocity. Equating the two parts of EQ(2.26) at $V_{ds} = V_{\text{dsat}}$ lets us solve for the saturation voltage

$$V_{\text{dsat}} = \frac{V_{GT} V_c}{V_{GT} + V_c} \quad (2.27)$$

Noting that EQ(2.27) is in the same form as a parallel resistor equation, we see that V_{dsat} is less than the smaller of V_{GT} and V_c . Finally, substituting EQ(2.27) into EQ(2.26) gives a simplified expression for saturation current accounting for velocity saturations:

$$I_{\text{dsat}} = W C_{\text{ox}} v_{\text{sat}} \frac{V_{GT}^2}{V_{GT} + V_c} \quad V_{ds} > V_{\text{dsat}} \quad (2.28)$$

If $V_{GT} \ll V_c$, velocity saturation effects are negligible and EQ(2.28) reduces to the square-law model. This is also called the *long-channel regime*. But if $V_{GT} \gg V_c$, EQ(2.28) approaches the velocity-saturated limit

$$I_{\text{dsat}} \approx W C_{\text{ox}} v_{\text{sat}} V_{GT} \quad V_{ds} > V_c \quad (2.29)$$

Observe that the drain current is quadratically dependent on voltage in the long-channel regime and linearly dependent when fully velocity saturated. For moderate supply voltages, transistors operate in a region where the velocity neither increases linearly with field, nor is completely saturated. The *α -power law model* given in EQ(2.30) provides a simple approximation to capture this behavior [Sakurai90]. α is called the *velocity saturation index* and is determined by curve fitting measured I-V data. Transistors with long channels or low V_{DD} display quadratic I-V characteristics in saturation and are modeled with $\alpha = 2$. As transistors become more velocity saturated, increasing V_{gs} has less effect on current and α decreases, reaching 1 for transistors that are completely velocity saturated. For simplicity, the model uses a straight line in the linear region. Overall, the model is based on three parameters that can be determined empirically from a curve fit of I-V characteristics: α , βP_c , and P_v .

$$I_{ds} = \begin{cases} 0 & V_{gs} < V_t & \text{Cutoff} \\ I_{\text{dsat}} \frac{V_{ds}}{V_{\text{dsat}}} & V_{ds} < V_{\text{dsat}} & \text{Linear} \\ I_{\text{dsat}} & V_{ds} > V_{\text{dsat}} & \text{Saturation} \end{cases} \quad (2.30)$$

where

$$\begin{aligned} I_{\text{dsat}} &= P_c \frac{\beta}{2} V_{GT}^\alpha \\ V_{\text{dsat}} &= P_v V_{GT}^{\alpha/2} \end{aligned} \quad (2.31)$$

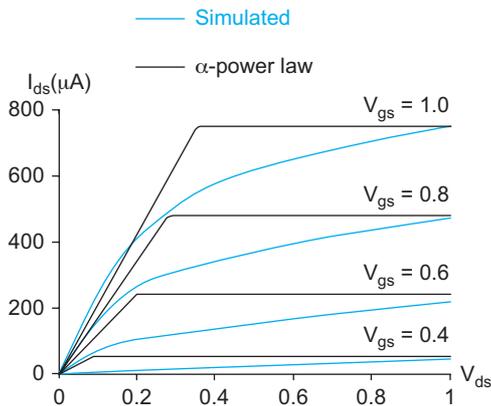


FIGURE 2.16 Comparison of α -power law model with simulated transistor behavior

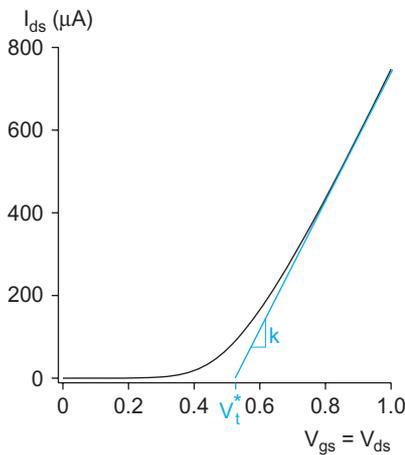


FIGURE 2.17 I_{ds} vs. V_{gs} in saturation, showing good linear fit at high V_{gs}

Figure 2.16 compares the α -power law model against simulated results, using $\alpha = 1.3$. The fit is poor at low V_{ds} , but the current at $V_{ds} = V_{DD}$ matches simulation fairly well across the full range of V_{gs} .

The low-field mobility of holes is much lower than that of electrons, so pMOS transistors experience less velocity saturation than nMOS for a given V_{DD} . This shows up as a larger value of α for pMOS than for nMOS transistors.

These models become too complicated to give much insight for hand calculations. A simpler approach is to observe, in velocity-saturated transistors, I_{ds} grows linearly rather than quadratically with V_{gs} when the transistor is strongly ON. Figure 2.17 plots I_{ds} vs. V_{gs} (holding $V_{ds} = V_{gs}$). This is equivalent to plotting I_{on} vs. V_{DD} . For V_{gs} significantly above V_t , I_{ds} fits a straight line quite well. Thus, we can approximate the ON current as

$$I_{ds} = k(V_{gs} - V_t^*) \tag{2.32}$$

where V_t^* is the x-intercept.

2.4.2 Channel Length Modulation

Ideally, I_{ds} is independent of V_{ds} for a transistor in saturation, making the transistor a perfect current source. As discussed in Section 2.3.3, the p-n junction between the drain and body forms a depletion region with a width L_d that increases with V_{db} , as shown in Figure 2.18. The depletion region effectively shortens the channel length to

$$L_{eff} = L - L_d \tag{2.33}$$

To avoid introducing the body voltage into our calculations, assume the source voltage is close to the body voltage so $V_{db} \approx V_{ds}$. Hence, increasing V_{ds} decreases the effective channel length. Shorter channel length results in higher current; thus, I_{ds} increases with V_{ds} in saturation, as shown in Figure 2.18. This can be crudely modeled by multiplying EQ (2.10) by a factor of $(1 + V_{ds} / V_A)$, where V_A is called the *Early voltage* [Gray01]. In the saturation region, we find

$$I_{ds} = \frac{\beta}{2} V_{GT}^2 \left(1 + \frac{V_{ds}}{V_A} \right) \tag{2.34}$$

As channel length gets shorter, the effect of the channel length modulation becomes relatively more important. Hence, V_A is proportional to channel length. This channel length modulation model is a gross oversimplification of nonlinear behavior and is more useful for conceptual understanding than for accurate device modeling.

Channel length modulation is very important to analog designers because it reduces the gain of amplifiers. It is generally unimportant for qualitatively understanding the behavior of digital circuits.

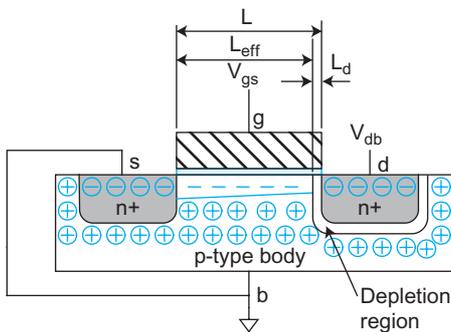


FIGURE 2.18 Depletion region shortens effective channel length

2.4.3 Threshold Voltage Effects

So far, we have treated the threshold voltage as a constant. However, V_t increases with the source voltage, decreases with the body voltage, decreases with the drain voltage, and increases with channel length [Roy03]. This section models each of these effects.

2.4.3.1 Body Effect Until now, we have considered a transistor to be a three-terminal device with gate, source, and drain. However, the body is an implicit fourth terminal. When a voltage V_{sb} is applied between the source and body, it increases the amount of charge required to invert the channel, hence, it increases the threshold voltage. The threshold voltage can be modeled as

$$V_t = V_{t0} + \gamma \left(\sqrt{\phi_s + V_{sb}} - \sqrt{\phi_s} \right) \quad (2.35)$$

where V_{t0} is the threshold voltage when the source is at the body potential, ϕ_s is the *surface potential* at threshold (see a device physics text such as [Tsividis99] for further discussion of surface potential), and γ is the *body effect coefficient*, typically in the range 0.4 to 1 V^{1/2}. In turn, these depend on the doping level in the channel, N_A . The body effect further degrades the performance of pass transistors trying to pass the weak value (e.g., nMOS transistors passing a ‘1’), as we will examine in Section 2.5.4. Section 5.3.4 will describe how a body bias can intentionally be applied to alter the threshold voltage, permitting trade-offs between performance and subthreshold leakage current.

$$\phi_s = 2v_T \ln \frac{N_A}{n_i} \quad (2.36)$$

$$\gamma = \frac{t_{\text{ox}}}{\epsilon_{\text{ox}}} \sqrt{2q\epsilon_{\text{si}}N_A} = \frac{\sqrt{2q\epsilon_{\text{si}}N_A}}{C_{\text{ox}}} \quad (2.37)$$

For small voltages applied to the source or body, EQ.(2.35) can be linearized to

$$V_t = V_{t0} + k_\gamma V_{sb} \quad (2.38)$$

where

$$k_\gamma = \frac{\gamma}{2\sqrt{\phi_s}} = \frac{\sqrt{\frac{q\epsilon_{\text{si}}N_A}{v_T \ln \frac{N_A}{n_i}}}}{2C_{\text{ox}}} \quad (2.39)$$

Example 2.5

Consider the nMOS transistor in a 65 nm process with a nominal threshold voltage of 0.3 V and a doping level of $8 \times 10^{17} \text{ cm}^{-3}$. The body is tied to ground with a substrate contact. How much does the threshold change at room temperature if the source is at 0.6 V instead of 0?

SOLUTION: At room temperature, the thermal voltage $v_T = kT/q = 26 \text{ mV}$ and $n_i = 1.45 \times 10^{10} \text{ cm}^{-3}$. The threshold increases by 0.04 V.

$$\phi_s = 2(0.026 \text{ V}) \ln \frac{8 \times 10^{17} \text{ cm}^{-3}}{1.45 \times 10^{10} \text{ cm}^{-3}} = 0.93 \text{ V}$$

$$\gamma = \frac{10.5 \times 10^{-8} \text{ cm}}{3.9 \times 8.85 \times 10^{-14} \frac{\text{F}}{\text{cm}}} \sqrt{2(1.6 \times 10^{-19} \text{ C}) \left(11.7 \times 8.85 \times 10^{-14} \frac{\text{F}}{\text{cm}} \right) (8 \times 10^{17} \text{ cm}^{-3})} = 0.16 \quad (2.40)$$

$$V_t = 0.3 + \gamma (\sqrt{\phi_s + 0.6 \text{ V}} - \sqrt{\phi_s}) = 0.34 \text{ V}$$

2.4.3.2 Drain-Induced Barrier Lowering The drain voltage V_{ds} creates an electric field that affects the threshold voltage. This *drain-induced barrier lowering* (DIBL) effect is especially pronounced in short-channel transistors. It can be modeled as

$$V_t = V_{t0} - \eta V_{ds} \quad (2.41)$$

where η is the DIBL coefficient, typically on the order of 0.1 (often expressed as 100 mV/V).

Drain-induced barrier lowering causes I_{ds} to increase with V_{ds} in saturation, in much the same way as channel length modulation does. This effect can be lumped into a smaller Early voltage V_A used in EQ.(2.34). Again, this is a bane for analog design but insignificant for most digital circuits. More significantly, DIBL increases subthreshold leakage at high V_{ds} , as we will discuss in Section 2.4.4.

2.4.3.3 Short Channel Effect The threshold voltage typically increases with channel length. This phenomenon is especially pronounced for small L where the source and drain depletion regions extend into a significant portion of the channel, and hence is called the *short channel effect*⁸ or V_t rolloff [Tsividis99, Cheng99]. In some processes, a *reverse short channel effect* causes V_t to decrease with length.

There is also a *narrow channel effect* in which V_t varies with channel width; this effect tends to be less significant because the minimum width is greater than the minimum length.

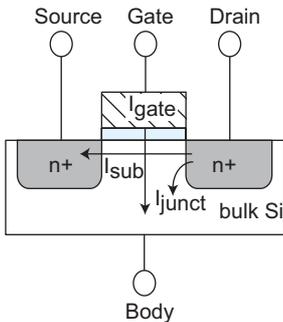


FIGURE 2.19
Leakage current paths

2.4.4 Leakage

Even when transistors are nominally OFF, they leak small amounts of current. Leakage mechanisms include subthreshold conduction between source and drain, gate leakage from the gate to body, and junction leakage from source to body and drain to body, as illustrated in Figure 2.19 [Roy03, Narendra06]. Subthreshold conduction is caused by thermal emission of carriers over the potential barrier set by the threshold. Gate leakage is a quantum-mechanical effect caused by tunneling through the extremely thin gate dielectric. Junction leakage is caused by current through the p-n junction between the source/drain diffusions and the body.

⁸The term *short-channel effect* is overused in the CMOS literature. Sometimes, it refers to any behavior outside the long-channel models. Other times, it refers to a range of behaviors including DIBL that are most significant for very short channel lengths [Muller03]. In this text, we restrict the term to describe the sensitivity of threshold voltage to channel length.

In processes with feature sizes above 180 nm, leakage was typically insignificant except in very low power applications. In 90 and 65 nm processes, threshold voltage has reduced to the point that subthreshold leakage reaches levels of 1s to 10s of nA per transistor, which is significant when multiplied by millions or billions of transistors on a chip. In 45 nm processes, oxide thickness reduces to the point that gate leakage becomes comparable to subthreshold leakage unless high-k gate dielectrics are employed. Overall, leakage has become an important design consideration in nanometer processes.

2.4.4.1 Subthreshold Leakage The long-channel transistor I-V model assumes current only flows from source to drain when $V_{gs} > V_t$. In real transistors, current does not abruptly cut off below threshold, but rather drops off exponentially, as seen in Figure 2.20. When the gate voltage is high, the transistor is strongly ON. When the gate falls below V_t , the exponential decline in current appears as a straight line on the logarithmic scale. This regime of $V_{gs} < V_t$ is called *weak inversion*. The *subthreshold leakage current* increases significantly with V_{ds} because of drain-induced barrier lowering (see Section 2.4.3.2). There is a lower limit on I_{ds} set by drain junction leakage that is exacerbated by the negative gate voltage (see Section 2.4.4.3).

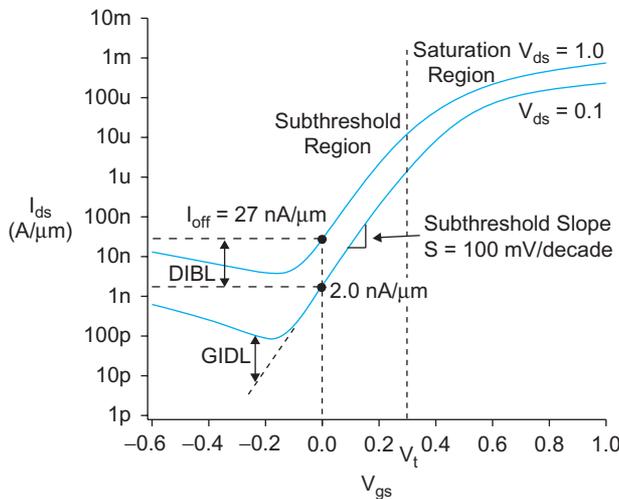


FIGURE 2.20 I-V characteristics of a 65 nm nMOS transistor at 70 °C on a log scale

Subthreshold leakage current is described by EQ (2.42). I_{ds0} is the current at threshold and is dependent on process and device geometry. It is typically extracted from simulation but can also be calculated from EQ (2.43); the $e^{1.8}$ term was found empirically [Sheu87]. n is a process-dependent term affected by the depletion region characteristics and is typically in the range of 1.3–1.7 for CMOS processes. The final term indicates that leakage is 0 if $V_{ds} = 0$, but increases to its full value when V_{ds} is a few multiples of the thermal voltage v_T (e.g., when $V_{ds} > 50$ mV). More significantly, drain-induced barrier lowering effectively reduces the threshold voltage, as indicated by the ηV_{ds} term. This can increase leakage by an order of magnitude for $V_{ds} = V_{DD}$ as compared to small V_{ds} . The body effect also modulates V_t when $V_{sb} \approx 0$.

$$I_{ds} = I_{ds0} e^{\frac{V_{gs} - V_t0 + \eta V_{ds} - k_p V_{sb}}{n v_T}} \left(1 - e^{-\frac{V_{ds}}{v_T}} \right) \quad (2.42)$$

$$I_{ds0} = \beta v_T^2 e^{1.8} \quad (2.43)$$

Subthreshold conduction is used to advantage in very low-power circuits, as will be explored in Section 9.6. It afflicts dynamic circuits and DRAMs, which depend on the storage of charge on a capacitor. Conduction through an OFF transistor discharges the capacitor unless it is periodically refreshed or a trickle of current is available to counter the leakage. Leakage also contributes to power dissipation in idle circuits. Subthreshold leakage increases exponentially as V_t decreases or as temperature rises, so it is a major problem for chips using low supply and threshold voltages and for chips operating at high temperature.

As shown in Figure 2.20, subthreshold current fits a straight line on a semilog plot. The inverse of the slope of this line is called the *subthreshold slope*, S

$$S = \left[\frac{d(\log_{10} I_{ds})}{dV_{gs}} \right]^{-1} = n v_T \ln 10 \quad (2.44)$$

The subthreshold slope indicates how much the gate voltage must drop to decrease the leakage current by an order of magnitude. A typical value is 100 mV/decade at room temperature. EQ.(2.42) can be rewritten using the subthreshold slope as

$$I_{ds} = I_{\text{off}} 10^{\frac{V_{gs} + \eta(V_{ds} - V_{dd}) - k_V V_{sb}}{s}} \left(1 - e^{-\frac{V_{ds}}{v_T}} \right) \quad (2.45)$$

where I_{off} is the subthreshold current at $V_{gs} = 0$ and $V_{ds} = V_{DD}$.

Example 2.6

What is the minimum threshold voltage for which the leakage current through an OFF transistor ($V_{gs} = 0$) is 10^3 times less than that of a transistor that is barely ON ($V_{gs} = V_t$) at room temperature if $n = 1.5$? One of the advantages of silicon-on-insulator (SOI) processes is that they have smaller n (see Section 9.5). What threshold is required for SOI if $n = 1.3$?

SOLUTION: $v_T = 26$ mV at room temperature. Assume $V_{ds} \gg v_T$ so leakage is significant. We solve

$$I_{ds}(V_{gs} = 0) = 10^{-3} I_{ds0} = I_{ds0} e^{\frac{-V_t}{n v_T}} \quad (2.46)$$

$$V_t = -n v_T \ln 10^{-3} = 270 \text{ mV}$$

In the CMOS process, leakage rolls off by a factor of 10 for every 90 mV V_{gs} falls below threshold. This is often quoted as a subthreshold slope of $S = 90$ mV/decade. In the SOI process, the subthreshold slope S is 78 mV/decade, so a threshold of only 234 mV is required.

2.4.4.2 Gate Leakage According to quantum mechanics, the electron cloud surrounding an atom has a probabilistic spatial distribution. For gate oxides thinner than 15–20 Å,

there is a nonzero probability that an electron in the gate will find itself on the wrong side of the oxide, where it will get whisked away through the channel. This effect of carriers crossing a thin barrier is called tunneling, and results in leakage current through the gate.

Two physical mechanisms for gate tunneling are called *Fowler-Nordheim (FN) tunneling* and *direct tunneling*. FN tunneling is most important at high voltage and moderate oxide thickness and is used to program EEPROM memories (see Section 12.4). Direct tunneling is most important at lower voltage with thin oxides and is the dominant leakage component.

The direct gate tunneling current can be estimated as [Chandrakasan01]

$$I_{\text{gate}} = WA \left(\frac{V_{DD}}{t_{\text{ox}}} \right)^2 e^{-B \frac{t_{\text{ox}}}{V_{DD}}} \quad (2.47)$$

where A and B are technology constants.

Transistors need high C_{ox} to deliver good ON current, driving the decrease in oxide thickness. Tunneling current drops exponentially with the oxide thickness and has only recently become significant. Figure 2.21 plots gate leakage current density (current/area) J_G against voltage for various oxide thicknesses. Gate leakage increases by a factor of 2.7 or more per angstrom reduction in thickness [Rohrer05]. Large tunneling currents impact not only dynamic nodes but also quiescent power consumption and thus limits equivalent oxide thicknesses t_{ox} to at least 10.5 Å to keep gate leakage below 100 A/cm². To keep these dimensions in perspective, recall that each atomic layer of SiO₂ is about 3 Å, so such gate oxides are a handful of atomic layers thick. Section 3.4.1.3 describes innovations in gate insulators with higher dielectric constants that offer good C_{ox} while reducing tunneling.

Tunneling current can be an order of magnitude higher for nMOS than pMOS transistors with SiO₂ gate dielectrics because the electrons tunnel from the conduction band while the holes tunnel from the valence band and see a higher barrier [Hamzaoglu02]. Different dielectrics may have different tunneling properties.

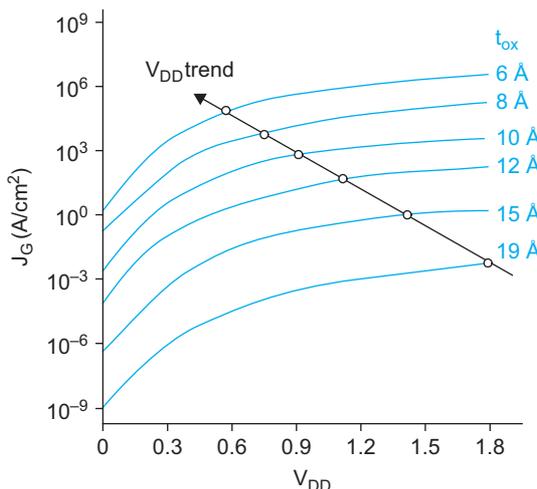


FIGURE 2.21 Gate leakage current from [Song01]

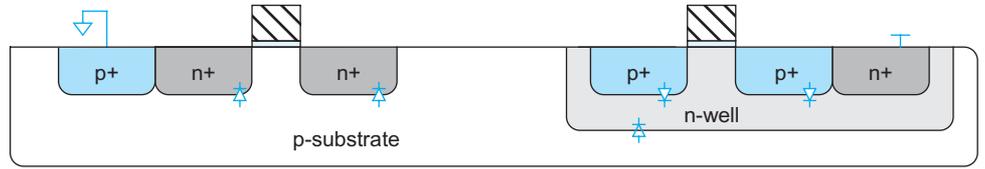


FIGURE 2.22 Substrate to diffusion diodes in CMOS circuits

2.4.4.3 Junction Leakage The p-n junctions between diffusion and the substrate or well form diodes, as shown in Figure 2.22. The well-to-substrate junction is another diode. The substrate and well are tied to GND or V_{DD} to ensure these diodes do not become forward biased in normal operation. However, reverse-biased diodes still conduct a small amount of current I_D .⁹

$$I_D = I_S \left(e^{\frac{V_D}{v_T}} - 1 \right) \quad (2.48)$$

where I_S depends on doping levels and on the area and perimeter of the diffusion region and V_D is the diode voltage (e.g., $-V_{sb}$ or $-V_{db}$). When a junction is reverse biased by significantly more than the thermal voltage, the leakage is just $-I_S$, generally in the 0.1–0.01 fA/ μm^2 range, which is negligible compared to other leakage mechanisms.

More significantly, heavily doped drains are subject to *band-to-band tunneling* (BTBT) and *gate-induced drain leakage* (GIDL).

BTBT occurs across the junction between the source or drain and the body when the junction is reverse-biased. It is a function of the reverse bias and the doping levels. High halo doping used to increase V_t to alleviate subthreshold leakage instead causes BTBT to grow. The leakage is exacerbated by *trap-assisted tunneling* (TAT) when defects in the silicon lattice called traps reduce the distance that a carrier must tunnel. Most of the leakage occurs along the sidewall closest to the channel where the doping is highest. It can be modeled as

$$I_{BTBT} = WX_j A \frac{E_j}{E_g^{0.5}} V_{dd} e^{-B \frac{E_g^{1.5}}{E_j}} \quad (2.49)$$

where X_j is the junction depth of the diffusion, E_g is the bandgap voltage, and A and B are technology constants [Mukhopadhyay05]. The electric field along the junction at a reverse bias of V_{DD} is

$$E_j = \sqrt{\frac{2qN_{halo}N_{sd}}{\epsilon(N_{halo} + N_{sd})}} \left(V_{DD} + v_T \ln \frac{N_{halo}N_{sd}}{n_i^2} \right) \quad (2.50)$$

GIDL occurs where the gate partially overlaps the drain. This effect is most pronounced when the drain is at a high voltage and the gate is at a low voltage. GIDL current is proportional to gate-drain overlap area and hence to transistor width. It is a strong function of the electric field and hence increases rapidly with the drain-to-gate voltage. How-

⁹Beware that I_D and I_S stand for the diode current and diode reverse-biased saturation currents, respectively. The D and S are not related to drain or source.

ever, it is normally insignificant at $|V_{gd}| \leq V_{DD}$ [Mukhopadhyay05], only coming into play when the gate is driven outside the rails in an attempt to cut off subthreshold leakage.

2.4.5 Temperature Dependence

Transistor characteristics are influenced by temperature [Cobbold66, Vadasz66, Tsvidis99, Gutierrez01]. Carrier mobility decreases with temperature. An approximate relation is

$$\mu(T) = \mu(T_r) \left(\frac{T}{T_r} \right)^{-k_\mu} \quad (2.51)$$

where T is the absolute temperature, T_r is room temperature, and k_μ is a fitting parameter with a typical value of about 1.5. v_{sat} also decreases with temperature, dropping by about 20% from 300 to 400 K.

The magnitude of the threshold voltage decreases nearly linearly with temperature and may be approximated by

$$V_t(T) = V_t(T_r) - k_{vt}(T - T_r) \quad (2.52)$$

where k_{vt} is typically about 1–2 mV/K.

I_{on} at high V_{DD} decreases with temperature. Subthreshold leakage increases exponentially with temperature. BTBT increases slowly with temperature, and gate leakage is almost independent of temperature.

The combined temperature effects are shown in Figure 2.23. At high V_{gs} , the current has a *negative temperature coefficient*, i.e., it decreases with temperature. At low V_{gs} , the current has a positive temperature coefficient. Thus, OFF current increases with temperature. ON current I_{dsat} normally decreases with temperature, as shown in Figure 2.24, so circuit performance is worst at high temperature. However, for systems operating at low V_{DD} (typically < 0.7–1.1 V), I_{dsat} increases with temperature [Kumar06].

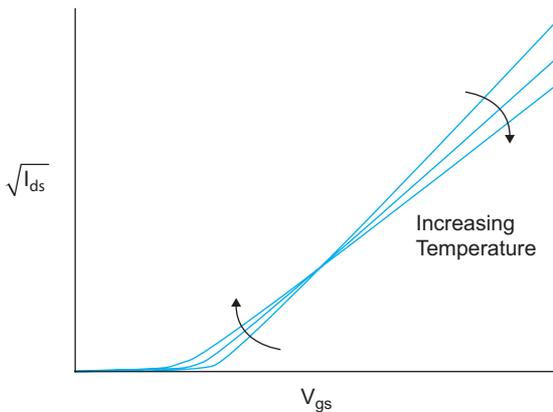


FIGURE 2.23 I-V characteristics of nMOS transistor in saturation at various temperatures

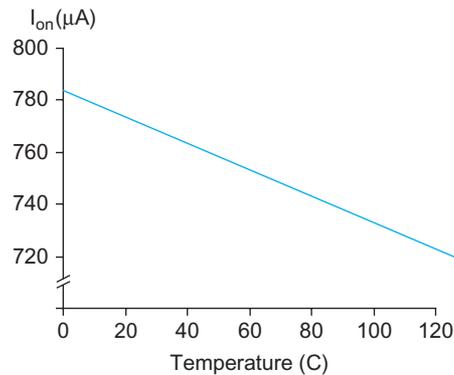


FIGURE 2.24 I_{dsat} vs. temperature

Conversely, circuit performance can be improved by cooling. Most systems use natural convection or fans in conjunction with heat sinks, but water cooling, thin-film refrigerators, or even liquid nitrogen can increase performance if the expense is justified. There are many advantages of operating at low temperature [Keyes70, Sun87]. Subthreshold leakage is exponentially dependent on temperature, so lower threshold voltages can be used. Velocity saturation occurs at higher fields, providing more current. As mobility is also higher, these fields are reached at a lower power supply, saving power. Depletion regions become wider, resulting in less junction capacitance.

Two popular lab tools for determining temperature dependence in circuits are a can of freeze spray and a heat gun. The former can be used to momentarily “freeze” a chip to see whether performance alters and the other, of course, can be used to heat up a chip. Often, these tests are done to quickly determine whether a chip is prone to temperature effects. Be careful—sometimes the sudden temperature change can fracture chips or their packages.

2.4.6 Geometry Dependence

The layout designer draws transistors with width and length W_{drawn} and L_{drawn} . The actual gate dimensions may differ by some factors X_W and X_L . For example, the manufacturer may create masks with narrower polysilicon or may overetch the polysilicon to provide shorter channels (negative X_L) without changing the overall design rules or metal pitch. Moreover, the source and drain tend to diffuse laterally under the gate by L_D , producing a shorter effective channel length that the carriers must traverse between source and drain. Similarly, W_D accounts for other effects that shrink the transistor width. Putting these factors together, we can compute effective transistor lengths and widths that should be used in place of L and W in the current and capacitance equations given elsewhere in the book. The factors of two come from lateral diffusion on both sides of the channel.

$$\begin{aligned} L_{\text{eff}} &= L_{\text{drawn}} + X_L - 2L_D \\ W_{\text{eff}} &= W_{\text{drawn}} + X_W - 2W_D \end{aligned} \quad (2.53)$$

Therefore, a transistor drawn twice as long may have an effective length that is more than twice as great. Similarly, two transistors differing in drawn widths by a factor of two may differ in saturation current by more than a factor of two. Threshold voltages also vary with transistor dimensions because of the short and narrow channel effects.

Combining threshold changes, effective channel lengths, channel length modulation, and velocity saturation effects, I_{dsat} does not scale exactly as $1/L$. In general, when currents must be precisely matched (e.g., in sense amplifiers or A/D converters), it is best to use the same width and length for each device. Current ratios can be produced by tying several identical transistors in parallel.

In processes below $0.25 \mu\text{m}$, the effective length of the transistor also depends significantly on the orientation of the transistor. Moreover, the amount of nearby polysilicon also affects etch rates during manufacturing and thus channel length. Transistors that must match well should have the same orientation. Dummy polysilicon wires can be placed nearby to improve etch uniformity.

2.4.7 Summary

Although the physics of nanometer-scale devices is complicated, the impact of nonideal I-V behavior is fairly easy to understand from the designer’s viewpoint.

Threshold drops Pass transistors suffer a threshold drop when passing the wrong value: nMOS transistors only pull up to $V_{DD} - V_{tm}$, while pMOS transistors only pull down to $|V_{tp}|$. The magnitude of the threshold drop is increased by the body effect. Therefore, pass transistors do not operate very well in nanometer processes where the threshold voltage is a significant fraction of the supply voltage. Fully complementary transmission gates should be used where both 0s and 1s must be passed well.

Leakage current Ideally, static CMOS gates draw zero current and dissipate zero power when idle. Real gates draw some leakage current. The most important source at this time is subthreshold leakage between source and drain of a transistor that should be cut off. The subthreshold current of an OFF transistor decreases by an order of magnitude for every 60–100 mV that V_{gs} is below V_t . Threshold voltages have been decreasing, so subthreshold leakage has been increasing dramatically. Some processes offer multiple choices of V_t : low- V_t devices are used for high performance in critical circuits, while high- V_t devices are used for low leakage elsewhere.

The transistor gate is a good insulator. However, significant tunneling current flows through very thin gates. This has limited the scaling of gate oxide and led to new high-k gate dielectrics.

Leakage current causes CMOS gates to consume power when idle. It also limits the amount of time that data is retained in dynamic logic, latches, and memory cells. In nanometer processes, dynamic logic and latches require some sort of feedback to prevent data loss from leakage. Leakage increases at high temperature.

V_{DD} Velocity saturation and mobility degradation result in less current than expected at high voltage. This means that there is no point in trying to use a high V_{DD} to achieve fast transistors, so V_{DD} has been decreasing with process generation to reduce power consumption. Moreover, the very short channels and thin gate oxides would be damaged by high V_{DD} .

Delay Transistors in series drop part of the voltage across each transistor and thus experience smaller fields and less velocity saturation than single transistors. Therefore, series transistors tend to be a bit faster than a simple model would predict. For example, two nMOS transistors in series deliver more than half the current of a single nMOS transistor of the same width. This effect is more pronounced for nMOS transistors than pMOS transistors because nMOS transistors have higher mobility to begin with and thus are more velocity saturated.

Matching If two transistors should behave identically, both should have the same dimensions and orientation and be interdigitated if possible.

2.5 DC Transfer Characteristics

Digital circuits are merely analog circuits used over a special portion of their range. The DC transfer characteristics of a circuit relate the output voltage to the input voltage, assuming the input changes slowly enough that capacitances have plenty of time to charge or discharge. Specific ranges of input and output voltages are defined as valid 0 and 1 logic levels. This section explores the DC transfer characteristics of CMOS gates and pass transistors.

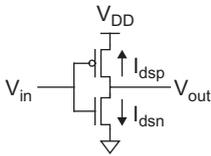


FIGURE 2.25
A CMOS inverter

2.5.1 Static CMOS Inverter DC Characteristics

Let us derive the DC transfer function (V_{out} vs. V_{in}) for the static CMOS inverter shown in Figure 2.25. We begin with Table 2.2, which outlines various regions of operation for the n- and p-transistors. In this table, V_{tn} is the threshold voltage of the n-channel device, and V_{tp} is the threshold voltage of the p-channel device. Note that V_{tp} is negative. The equations are given both in terms of V_{gs}/V_{ds} and V_{in}/V_{out} . As the source of the nMOS transistor is grounded, $V_{gsn} = V_{in}$ and $V_{dsn} = V_{out}$. As the source of the pMOS transistor is tied to V_{DD} , $V_{gsp} = V_{in} - V_{DD}$ and $V_{dsp} = V_{out} - V_{DD}$.

TABLE 2.2 Relationships between voltages for the three regions of operation of a CMOS inverter

	Cutoff	Linear	Saturated
nMOS	$V_{gsn} < V_{tn}$	$V_{gsn} > V_{tn}$	$V_{gsn} > V_{tn}$
	$V_{in} < V_{tn}$	$V_{in} > V_{tn}$	$V_{in} > V_{tn}$
		$V_{dsn} < V_{gsn} - V_{tn}$	$V_{dsn} > V_{gsn} - V_{tn}$
		$V_{out} < V_{in} - V_{tn}$	$V_{out} > V_{in} - V_{tn}$
pMOS	$V_{gsp} > V_{tp}$	$V_{gsp} < V_{tp}$	$V_{gsp} < V_{tp}$
	$V_{in} > V_{tp} + V_{DD}$	$V_{in} < V_{tp} + V_{DD}$	$V_{in} < V_{tp} + V_{DD}$
		$V_{dsp} > V_{gsp} - V_{tp}$	$V_{dsp} < V_{gsp} - V_{tp}$
		$V_{out} > V_{in} - V_{tp}$	$V_{out} < V_{in} - V_{tp}$

The objective is to find the variation in output voltage (V_{out}) as a function of the input voltage (V_{in}). This may be done graphically, analytically (see Exercise 2.16), or through simulation [Carr72]. Given V_{in} , we must find V_{out} subject to the constraint that $I_{dsn} = |I_{dsp}|$. For simplicity, we assume $V_{tp} = -V_{tn}$ and that the pMOS transistor is 2–3 times as wide as the nMOS transistor so $\beta_n = \beta_p$. We relax this assumption in Section 2.5.2.

We commence with the graphical representation of the simple algebraic equations described by EQ.(2.10) for the two transistors shown in Figure 2.26(a). The plot shows I_{dsn} and I_{dsp} in terms of V_{dsn} and V_{dsp} for various values of V_{gsn} and V_{gsp} . Figure 2.26(b) shows the same plot of I_{dsn} and $|I_{dsp}|$ now in terms of V_{out} for various values of V_{in} . The possible operating points of the inverter, marked with dots, are the values of V_{out} where $I_{dsn} = |I_{dsp}|$ for a given value of V_{in} . These operating points are plotted on V_{out} vs. V_{in} axes in Figure 2.26(c) to show the inverter DC transfer characteristics. The supply current $I_{DD} = I_{dsn} = |I_{dsp}|$ is also plotted against V_{in} in Figure 2.26(d) showing that both transistors are momentarily ON as V_{in} passes through voltages between GND and V_{DD} , resulting in a pulse of current drawn from the power supply.

The operation of the CMOS inverter can be divided into five regions indicated on Figure 2.26(c). The state of each transistor in each region is shown in Table 2.3. In region *A*, the nMOS transistor is OFF so the pMOS transistor pulls the output to V_{DD} . In region *B*, the nMOS transistor starts to turn ON, pulling the output down. In region *C*, both transistors are in saturation. Notice that ideal transistors are only in region *C* for $V_{in} = V_{DD}/2$ and that the slope of the transfer curve in this example is $-\infty$ in this region, corresponding to infinite gain. Real transistors have finite output resistances on account of channel length modulation, described in Section 2.4.2, and thus have finite slopes over a broader region *C*. In region *D*, the pMOS transistor is partially ON and in region *E*, it is completely

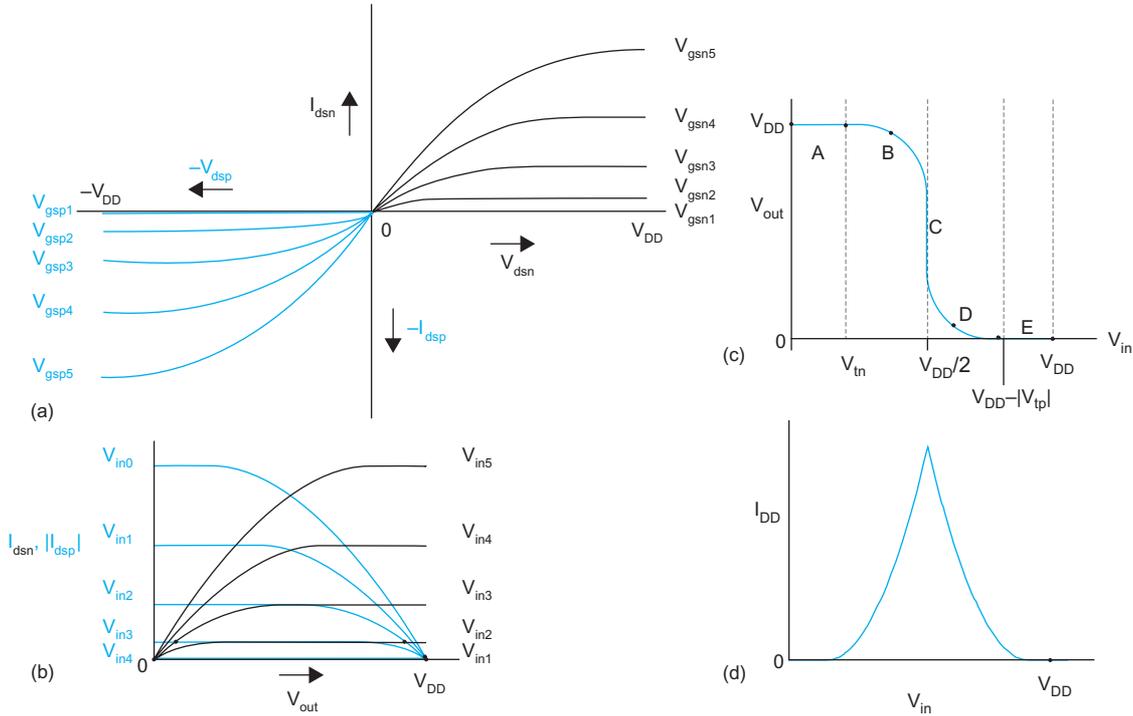


FIGURE 2.26 Graphical derivation of CMOS inverter DC characteristic

OFF, leaving the nMOS transistor to pull the output down to GND. Also notice that the inverter’s current consumption is ideally zero, neglecting leakage, when the input is within a threshold voltage of the V_{DD} or GND rails. This feature is important for low-power operation.

TABLE 2.3 Summary of CMOS inverter operation

Region	Condition	p-device	n-device	Output
A	$0 \leq V_{in} < V_{tn}$	linear	cutoff	$V_{out} = V_{DD}$
B	$V_{tn} \leq V_{in} < V_{DD}/2$	linear	saturated	$V_{out} > V_{DD}/2$
C	$V_{in} = V_{DD}/2$	saturated	saturated	V_{out} drops sharply
D	$V_{DD}/2 < V_{in} \leq V_{DD} - V_{tp} $	saturated	linear	$V_{out} < V_{DD}/2$
E	$V_{in} > V_{DD} - V_{tp} $	cutoff	linear	$V_{out} = 0$

Figure 2.27 shows simulation results of an inverter from a 65 nm process. The pMOS transistor is twice as wide as the nMOS transistor to achieve approximately equal betas. Simulation matches the simple models reasonably well, although the transition is not quite as steep because transistors are not ideal current sources in saturation.

The crossover point where $V_{inv} = V_{in} = V_{out}$ is called the *input threshold*. Because both mobility and the magnitude of the threshold voltage decrease with temperature for nMOS and pMOS transistors, the input threshold of the gate is only weakly sensitive to temperature.

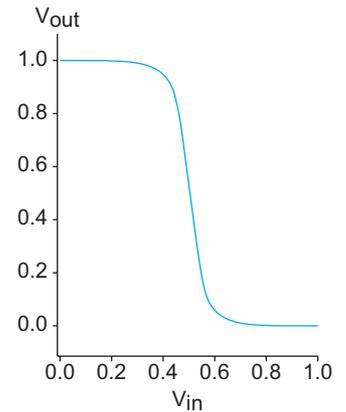


FIGURE 2.27 Simulated CMOS inverter DC characteristic

2.5.2 Beta Ratio Effects

We have seen that for $\beta_p = \beta_n$, the inverter threshold voltage V_{inv} is $V_{DD}/2$. This may be desirable because it maximizes noise margins (see Section 2.5.3) and allows a capacitive load to charge and discharge in equal times by providing equal current source and sink capabilities (see Section 4.2). Inverters with different beta ratios $r = \beta_p/\beta_n$ are called *skewed* inverters [Sutherland99]. If $r > 1$, the inverter is *HI-skewed*. If $r < 1$, the inverter is *LO-skewed*. If $r = 1$, the inverter has normal skew or is *unskewed*.

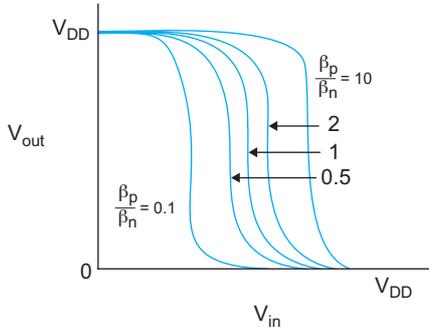


FIGURE 2.28 Transfer characteristics of skewed inverters

A HI-skew inverter has a stronger pMOS transistor. Therefore, if the input is $V_{DD}/2$, we would expect the output will be greater than $V_{DD}/2$. In other words, the input threshold must be higher than for an unskewed inverter. Similarly, a LO-skew inverter has a weaker pMOS transistor and thus a lower switching threshold.

Figure 2.28 explores the impact of skewing the beta ratio on the DC transfer characteristics. As the beta ratio is changed, the switching threshold moves. However, the output voltage transition remains sharp. Gates are usually skewed by adjusting the widths of transistors while maintaining minimum length for speed.

The inverter threshold can also be computed analytically. If the long-channel models of EQ.(2.10) for saturated transistors are valid:

$$I_{dn} = \frac{\beta_n}{2} (V_{inv} - V_{tn})^2 \quad (2.54)$$

$$I_{dp} = \frac{\beta_p}{2} (V_{inv} - V_{DD} - V_{tp})^2$$

By setting the currents to be equal and opposite, we can solve for V_{inv} as a function of r :

$$V_{inv} = \frac{V_{DD} + V_{tp} + V_{tn} \sqrt{\frac{1}{r}}}{1 + \sqrt{\frac{1}{r}}} \quad (2.55)$$

In the limit that the transistors are fully velocity saturated, EQ.(2.29) shows

$$\begin{aligned} I_{dn} &= W_n C_{ox} v_{sat-n} (V_{inv} - V_{tn}) \\ I_{dp} &= W_p C_{ox} v_{sat-p} (V_{inv} - V_{DD} - V_{tp}) \end{aligned} \quad (2.56)$$

Redefining $r = W_p v_{sat-p} / W_n v_{sat-n}$, we can again find the inverter threshold

$$V_{inv} = \frac{V_{DD} + V_{tp} + V_{tn} \frac{1}{r}}{1 + \frac{1}{r}} \quad (2.57)$$

In either case, if $V_{tn} = -V_{tp}$ and $r = 1$, $V_{inv} = V_{DD}/2$ as expected. However, velocity saturated inverters are more sensitive to skewing because their DC transfer characteristics are not as sharp.

DC transfer characteristics of other static CMOS gates can be understood by collapsing the gates into an equivalent inverter. Series transistors can be viewed as a single transistor of greater length. If only one of several parallel transistors is ON, the other

transistors can be ignored. If several parallel transistors are ON, the collection can be viewed as a single transistor of greater width.

2.5.3 Noise Margin

Noise margin is closely related to the DC voltage characteristics [Wakerly00]. This parameter allows you to determine the allowable noise voltage on the input of a gate so that the output will not be corrupted. The specification most commonly used to describe noise margin (or *noise immunity*) uses two parameters: the *LOW* noise margin, NM_L , and the *HIGH* noise margin, NM_H . With reference to Figure 2.29, NM_L is defined as the difference in maximum LOW input voltage recognized by the receiving gate and the maximum LOW output voltage produced by the driving gate.

$$NM_L = V_{IL} - V_{OL} \quad (2.58)$$

The value of NM_H is the difference between the minimum HIGH output voltage of the driving gate and the minimum HIGH input voltage recognized by the receiving gate. Thus,

$$NM_H = V_{OH} - V_{IH} \quad (2.59)$$

where

- V_{IH} = minimum HIGH input voltage
- V_{IL} = maximum LOW input voltage
- V_{OH} = minimum HIGH output voltage
- V_{OL} = maximum LOW output voltage

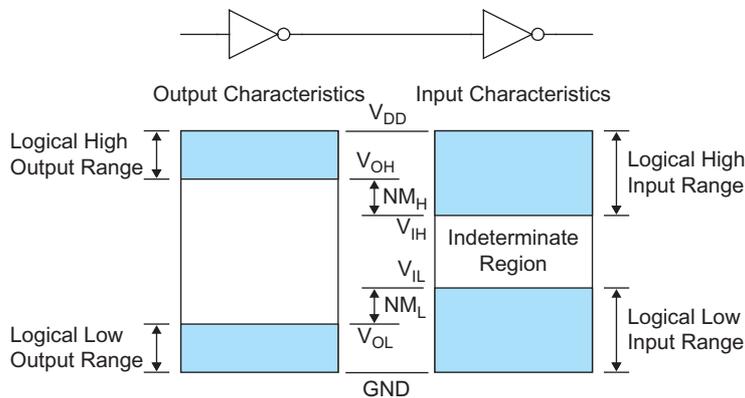


FIGURE 2.29 Noise margin definitions

Inputs between V_{IL} and V_{IH} are said to be in the *indeterminate region* or *forbidden zone* and do not represent legal digital logic levels. Therefore, it is generally desirable to have V_{IH} as close as possible to V_{IL} and for this value to be midway in the “logic swing,” V_{OL} to V_{OH} . This implies that the transfer characteristic should switch abruptly; that is, there should be high gain in the transition region. For the purpose of calculating noise margins, the transfer characteristic of the inverter and the definition of voltage levels V_{IL} , V_{OL} , V_{IH} , and V_{OH} are shown in Figure 2.30. Logic levels are defined at the unity gain point where

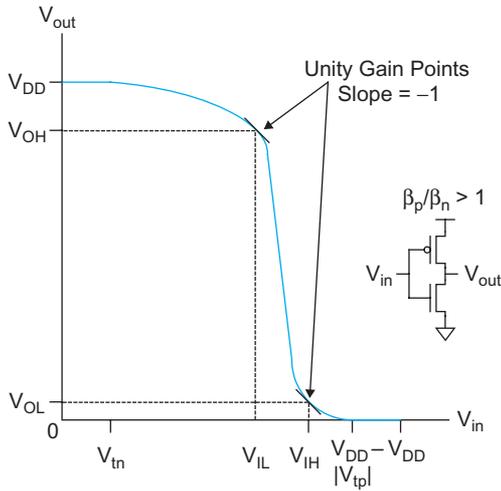


FIGURE 2.30 CMOS inverter noise margins

the slope is -1 . This gives a conservative bound on the worst case static noise margin [Hill68, Lohstroh83, Shepard99]. For the inverter shown, the NM_L is $0.46 V_{DD}$ while the NM_H is $0.13 V_{DD}$. Note that the output is slightly degraded when the input is at its worst legal value; this is called *noise feedthrough* or *propagated noise*. The exercises at the end of the chapter examine graphical and analytical approaches of finding the logic levels and noise margins.

If either NM_L or NM_H for a gate are too small, the gate may be disturbed by noise that occurs on the inputs. An unskewed gate has equal noise margins, which maximizes immunity to arbitrary noise sources. If a gate sees more noise in the high or low input state, the gate can be skewed to improve that noise margin at the expense of the other. Note that if $|V_{tp}| = V_{tn}$, then NM_H and NM_L increase as threshold voltages are increased.

Quite often, noise margins are compromised to improve speed. Circuit examples in Chapter 9 will illustrate this trade-off. Noise sources tend to scale with the supply voltage, so noise margins are best given as a fraction of the supply voltage. A noise margin of $0.4 V$ is quite comfortable in a $1.8 V$ process, but marginal in a $5 V$ process.

DC analysis gives us the *static noise margins* specifying the level of noise that a gate may see for an indefinite duration. Larger noise pulses may be acceptable if they are brief; these are described by *dynamic noise margins* specified by a maximum amplitude as a function of the duration [Lohstroh79, Somasekhar00]. Unfortunately, there is no simple amplitude-duration product that conveniently specifies dynamic noise margins.

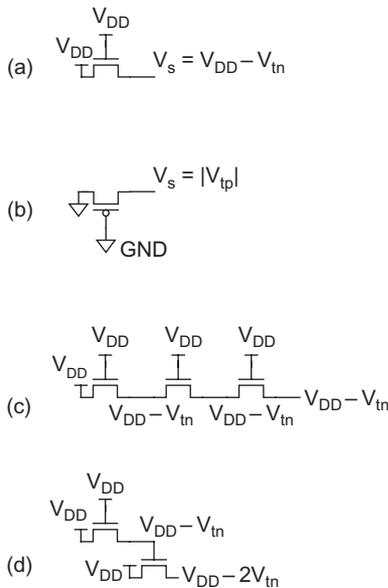


FIGURE 2.31 Pass transistor threshold drops

2.5.4 Pass Transistor DC Characteristics

Recall from Section 1.4.6 that nMOS transistors pass ‘0’s well but 1s poorly. We are now ready to better define “poorly.” Figure 2.31(a) shows an nMOS transistor with the gate and drain tied to V_{DD} . Imagine that the source is initially at $V_s = 0$. $V_{gs} > V_{tn}$, so the transistor is ON and current flows. If the voltage on the source rises to $V_s = V_{DD} - V_{tn}$, V_{gs} falls to V_{tn} and the transistor cuts itself OFF. Therefore, nMOS transistors attempting to pass a 1 never pull the source above $V_{DD} - V_{tn}$.¹⁰ This loss is sometimes called a *threshold drop*.

Moreover, when the source of the nMOS transistor rises, V_{sb} becomes nonzero. As described in Section 2.4.3.1, this nonzero source to body potential introduces the body effect that increases the threshold voltage. Using the data from the example in that section, a pass transistor driven with $V_{DD} = 1 V$ would produce an output of only $0.65 V$, potentially violating the noise margins of the next stage.

Similarly, pMOS transistors pass 1s well but 0s poorly. If the pMOS source drops below $|V_{tp}|$, the transistor cuts off. Hence, pMOS transistors only pull down to within a threshold above GND, as shown in Figure 2.31(b).

¹⁰Technically, the output can rise higher very slowly by means of subthreshold leakage.

As the source can rise to within a threshold voltage of the gate, the output of several transistors in series is no more degraded than that of a single transistor (Figure 2.31(c)). However, if a degraded output drives the gate of another transistor, the second transistor can produce an even further degraded output (Figure 2.31(d)).

If we attempt to use a transistor as a switch, the threshold drop degrades the output voltage. In old processes where the power supply voltage was high and V_t was a small fraction of V_{DD} , the drop was tolerable. In modern processes where V_t is closer to 1/3 of V_{DD} , the threshold drop can produce an invalid or marginal logic level at the output. To solve this problem, CMOS switches are generally built using transmission gates.

Recall from Section 1.4.6 that a transmission gate consists of an nMOS transistor and a pMOS transistor in parallel with gates controlled by complementary signals. When the transmission gate is ON, at least one of the two transistors is ON for any output voltage and hence, the transmission gate passes both 0s and 1s well. The transmission gate is a fundamental and ubiquitous component in MOS logic. It finds use as a multiplexing element, a logic structure, a latch element, and an analog switch. The transmission gate acts as a voltage-controlled switch connecting the input and the output.

2.6 Pitfalls and Fallacies

This section lists a number of pitfalls and fallacies that can deceive the novice (or experienced) designer.

Blindly trusting one's models

Models should be viewed as only approximations to reality, not reality itself, and used within their limitations. In particular, simple models like the Shockley or RC models aren't even close to accurate fits for the I-V characteristics of a modern transistor. They are valuable for the insight they give on trends (i.e., making a transistor wider increases its gate capacitance and decreases its ON resistance), not for the absolute values they predict. Cutting-edge projects often target processes that are still under development, so these models should only be viewed as speculative. Finally, processes may not be fully characterized over all operating regimes; for example, don't assume that your models are accurate in the subthreshold region unless your vendor tells you so. Having said this, modern SPICE models do an extremely good job of predicting performance well into the GHz range for well-characterized processes and models when using proper design practices (such as accounting for temperature, voltage, and process variation).

Using excessively complicated models for manual calculations

Because models cannot be perfectly accurate, there is little value in using excessively complicated models, particularly for hand calculations. Simpler models give more insight on key trade-offs and more rapid feedback during design. Moreover, RC models calibrated against simulated data for a fabrication process can estimate delay just as accurately as elaborate models based on a large number of physical parameters but not calibrated to the process.

Assuming a transistor with twice the drawn length has exactly half the current

To first order, current is proportional to W/L . In modern transistors, the effective transistor length is usually shorter than the drawn length, so doubling the drawn length reduces current by more than a factor of two. Moreover, the threshold voltage tends to increase for longer transistors, resulting in less current. Therefore, it is a poor strategy to try to ratio currents by ratioing transistor lengths.

Assuming two transistors in series deliver exactly half the current of a single transistor

To first order, this would be true. However, each series transistor sees a smaller electric field across the channel and hence are each less velocity saturated. Therefore, two series transistors in a nanometer process will deliver more than half the current of a single transistor. This is more pronounced for nMOS than pMOS transistors because of the higher mobility and the higher degree of velocity saturation of electrons than holes at a given field. Hence, NAND gates perform better than first order estimates might predict.

Ignoring leakage

In contemporary processes, subthreshold and gate leakage can be quite significant. Leakage is exacerbated by high temperature and by random process variations. Undriven nodes will not retain their state for long; they will leak to some new voltage. Leakage power can account for a large fraction of total power, especially in battery-operated devices that are idle most of the time.

Using nMOS pass transistors

nMOS pass transistors only pull up to $V_{DD} - V_t$. This voltage may fall below V_{IH} of a receiver, especially as V_{DD} decreases. For example, one author worked with a scan latch containing an nMOS pass transistor that operated correctly in a 250 nm process at 2.5 V. When the latch was ported to a 180 nm process at 1.8 V, the scan chain stopped working. The problem was traced to the pass transistor and the scan chain was made operational in the lab by raising V_{DD} to 2 V. A better solution is to use transmission gates in place of pass transistors.

Summary

In summary, we have seen that MOS transistors are four-terminal devices with a gate, source, drain, and body. In normal operation, the body is tied to GND or V_{DD} so the transistor can be modeled as a three-terminal device. The transistor behaves as a voltage-controlled switch. An nMOS switch is OFF (no path from source to drain) when the gate voltage is below some threshold V_t . The switch turns ON, forming a channel connecting source to drain, when the gate voltage rises above V_t . This chapter has developed more elaborate models to predict the amount of current that flows when the transistor is ON. The transistor operates in three modes depending on the terminal voltages:

- $V_{gs} < V_t$ Cutoff $I_{ds} \approx 0$
- $V_{gs} > V_t$, $V_{ds} < V_{dsat}$ Linear I_{ds} increases with V_{ds} (like a resistor)
- $V_{gs} > V_t$, $V_{ds} > V_{dsat}$ Saturation I_{ds} constant (like a current source)

In a long-channel transistor, the saturation current depends on V_{GT}^2 . pMOS transistors are similar to nMOS transistors, but have the signs reversed and deliver about half the current because of lower mobility.

In a real transistor, the I-V characteristics are more complicated. Modern transistors are extraordinarily small and thus experience enormous electric fields even at low voltage. The high fields cause velocity saturation and mobility degradation that lead to less current than you might otherwise expect. This can be modeled as a saturation current dependent on V_{GT}^α , where the velocity saturation index α is less than 2. Moreover, the saturation current does increase slightly with V_{ds} because of channel length modulation. Although simple hand calculations are no longer accurate, the general shape does not change very much and the transistor characteristics can still be derived using graphical or simulation methods.

Even when the gate voltage is low, the transistor is not completely OFF. Subthreshold current through the channel drops off exponentially for $V_{gs} < V_t$, but is nonnegligible for transistors with low thresholds. Junction leakage currents flow through the reverse-biased p–n junctions. Tunneling current flows through the insulating gate when the oxide becomes thin enough.

We can derive the DC transfer characteristics and noise margins of logic gates using either analytical expressions or a graphical load line analysis or simulation. Static CMOS gates have excellent noise margins.

Unlike ideal switches, MOS transistors pass some voltage levels better than others. An nMOS transistor passes 0s well, but only pulls up to $V_{DD} - V_{tn}$ when passing 1s. The pMOS passes 1s well, but only pulls down to $|V_{tp}|$ when passing 0s. This threshold drop is exacerbated by the body effect, which increases the threshold voltage when the source is at a different potential than the body.

There are too many parameters in a modern BSIM model for a designer to deal with intuitively. Instead, CMOS transistors are usually characterized by the following basic figures of merit:

- V_{DD} Target supply voltage
- $L_{\text{gate}} / L_{\text{poly}}$ Effective channel length (< feature size)
- t_{ox} Effective oxide thickness (a.k.a. EOT)
- I_{dsat} I_{ds} @ $V_{gs} = V_{ds} = V_{DD}$
- I_{off} I_{ds} @ $V_{gs} = 0, V_{ds} = V_{DD}$
- I_g Gate leakage @ $V_{gs} = V_{DD}$

[Muller03] and [Tsividis99] offer comprehensive treatments of device physics at a more advanced level. [Gray01] describes MOSFET models in more detail from the analog designer's point of view.

Exercises

- 2.1 Consider an nMOS transistor in a $0.6 \mu\text{m}$ process with $W/L = 4/2 \lambda$ (i.e., $1.2/0.6 \mu\text{m}$). In this process, the gate oxide thickness is 100 \AA and the mobility of electrons is $350 \text{ cm}^2/\text{V} \cdot \text{s}$. The threshold voltage is 0.7 V . Plot I_{ds} vs. V_{ds} for $V_{gs} = 0, 1, 2, 3, 4,$ and 5 V .
- 2.2 Show that the current through two transistors in series is equal to the current through a single transistor of twice the length if the transistors are well described by the Shockley model. Specifically, show that $I_{DS1} = I_{DS2}$ in Figure 2.32 when the transistors are in their linear region: $V_{DS} < V_{DD} - V_t$, $V_{DD} > V_t$ (this is also true in saturation). *Hint:* Express the currents of the series transistors in terms of V_1 and solve for V_1 .
- 2.3 In Exercise 2.2, the body effect was ignored. If the body effect is considered, will I_{DS2} be equal to, greater than, or less than I_{DS1} ? Explain.

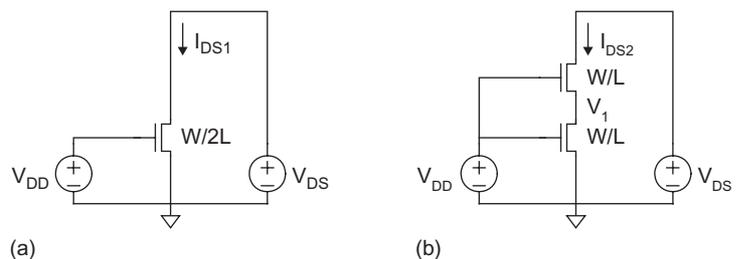


FIGURE 2.32 Current in series transistors

- 2.4 A 90 nm long transistor has a gate oxide thickness of 16 Å. What is its gate capacitance per micron of width?
- 2.5 Calculate the diffusion parasitic C_{db} of the drain of a unit-sized contacted nMOS transistor in a 0.6 μm process when the drain is at 0 and at $V_{DD} = 5$ V. Assume the substrate is grounded. The transistor characteristics are $CJ = 0.42$ fF/μm², $MJ = 0.44$, $CJSW = 0.33$ fF/μm, $MJSW = 0.12$, and $\psi_0 = 0.98$ V at room temperature.
- 2.6 Prove EQ.(2.27).
- 2.7 Consider the nMOS transistor in a 0.6 μm process with gate oxide thickness of 100 Å. The doping level is $N_A = 2 \times 10^{17}$ cm⁻³ and the nominal threshold voltage is 0.7 V. The body is tied to ground with a substrate contact. How much does the threshold change at room temperature if the source is at 4 V instead of 0?
- 2.8 Does the body effect of a process limit the number of transistors that can be placed in series in a CMOS gate at low frequencies?
- 2.9 Sometimes the substrate is connected to a voltage called the substrate bias to alter the threshold of the nMOS transistors. If the threshold of an nMOS transistor is to be raised, should a positive or negative substrate bias be used?
- 2.10 An nMOS transistor has a threshold voltage of 0.4 V and a supply voltage of $V_{DD} = 1.2$ V. A circuit designer is evaluating a proposal to reduce V_t by 100 mV to obtain faster transistors.
- By what factor would the saturation current increase (at $V_{gs} = V_{ds} = V_{DD}$) if the transistor were ideal?
 - By what factor would the subthreshold leakage current increase at room temperature at $V_{gs} = 0$? Assume $n = 1.4$.
 - By what factor would the subthreshold leakage current increase at 120 °C? Assume the threshold voltage is independent of temperature.
- 2.11 Find the subthreshold leakage current of an inverter at room temperature if the input $A = 0$. Let $\beta_n = 2\beta_p = 1$ mA/V², $n = 1.0$, and $|V_t| = 0.4$ V. Assume the body effect and DIBL coefficients are $\gamma = \eta = 0$.
- 2.12 Repeat Exercise 2.11 for a NAND gate built from unit transistors with inputs $A = B = 0$. Show that the subthreshold leakage current through the series transistors is half that of the inverter if $n = 1$.
- 2.13 Repeat Exercises 2.11 and 2.12 when $\eta = 0.04$ and $V_{DD} = 1.8$ V, as in the case of a more realistic transistor. γ has a secondary effect, so assume that it is 0. Did the leakage currents go up or down in each case? Is the leakage through the series transistors more than half, exactly half, or less than half of that through the inverter?
- 2.14 Peter Pitfall is offering to license to you his patented noninverting buffer circuit shown in Figure 2.33. Graphically derive the transfer characteristics for this buffer. Assume $\beta_n = \beta_p = \beta$ and $V_{tn} = |V_{tp}| = V_t$. Why is it a bad circuit idea?

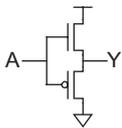


FIGURE 2.33
Noninverting buffer

- 2.15 A novel inverter has the transfer characteristics shown in Figure 2.34. What are the values of V_{LL} , V_{IH} , V_{OL} , and V_{OH} that give best noise margins? What are these high and low noise margins?
- 2.16 Section 2.5.1 graphically determined the transfer characteristics of a static CMOS inverter. Derive analytic expressions for V_{out} as a function of V_{in} for regions B and D of the transfer function. Let $|V_{tp}| = V_{tn}$ and $\beta_p = \beta_n$.
- 2.17 Using the results from Exercise 2.16, calculate the noise margin for a CMOS inverter operating at 1.0 V with $V_{tn} = |V_{tp}| = 0.35$ V, $\beta_p = \beta_n$.
- 2.18 Repeat Exercise 2.16 if the thresholds and betas of the two transistors are not necessarily equal. Also solve for the value of V_{in} for region C where both transistors are saturated.
- 2.19 Using the results from Exercise 2.18, calculate the noise margin for a CMOS inverter operating at 1.0 V with $V_{tn} = |V_{tp}| = 0.35$ V, $\beta_p = 0.5\beta_n$.
- 2.20 Give an expression for the output voltage for the pass transistor networks shown in Figure 2.35. Neglect the body effect.

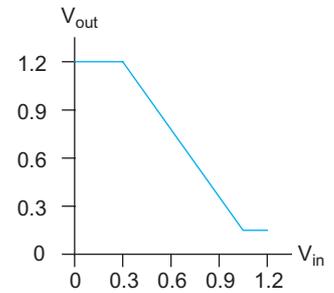


FIGURE 2.34 Transfer characteristics

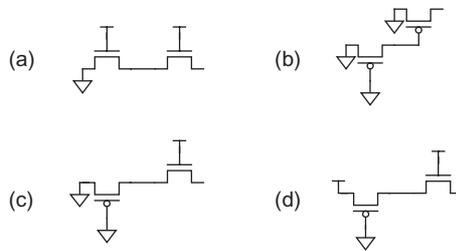


FIGURE 2.35 Pass transistor networks

- 2.21 Suppose $V_{DD} = 1.2$ V and $V_t = 0.4$ V. Determine V_{out} in Figure 2.36 for the following. Neglect the body effect.
 - a) $V_{in} = 0$ V
 - b) $V_{in} = 0.6$ V
 - c) $V_{in} = 0.9$ V
 - d) $V_{in} = 1.2$ V.

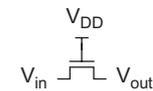


FIGURE 2.36 Single pass transistor

This page intentionally left blank