

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. Powerful *formal verification* tools are also becoming available that prove that a circuit performs the same Boolean function as the associated logic. These are very useful for handcrafted circuits, but can be slow and difficult to operate. Exotic circuits should be simulated thoroughly to ensure they perform the intended logic function and have adequate noise margins; circuit pitfalls are discussed throughout this book.

Layout vs. Schematic tools (LVS) check to make sure 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 latch-up, noise problems, or electromigration risks; such problems will also be discussed later in the book.

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; now 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 modern 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 130 nm process, the mask set may be in the vicinity of a million dollars. The MOSIS service in the United States [Pina02] and its 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.

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 vendors such as TSMC, UMC, and IBM.

Multiple chips are manufactured simultaneously on a single silicon wafer, typically 6"–12" in diameter. A bare wafer costs about \$1000–\$5000. 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, although shorter turnarounds are available for a substantial premium. Figure 1.70 shows a silicon wafer after processing.

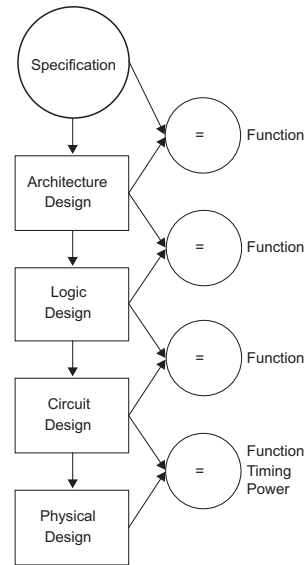
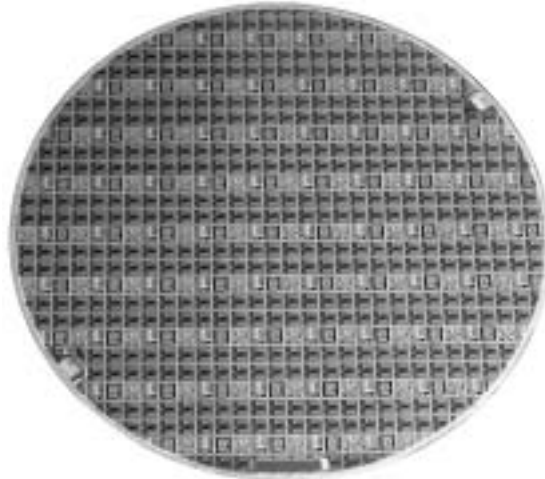
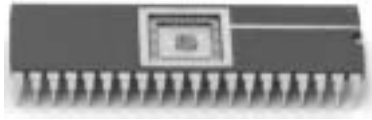


FIG 1.69 Design and verification sequence

**FIG 1.70** Processed 8-inch wafer**FIG 1.71** Chip in a 40-pin dual-inline package

Processed wafers are sliced into dice (chips) and packaged. Figure 1.71 shows a 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. More advanced packages offer different tradeoffs between cost, pin count, pin bandwidth, power handling, and reliability, as will be discussed in Section 12.2. 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.

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 9 is devoted to design verification and testing.

Summary

"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 developed the principles of designing a simple CMOS integrated circuit. MOS transistors can be viewed as electrically controlled switches. Complementary 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. The remainder of this book will expand on the material introduced in this chapter.

Exercises

- 1.1 Extrapolating the data from Figure 1.4, predict the transistor count of a microprocessor in 2010.
- 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 semilog scale. How many months pass between doubling of transistor counts?
- 1.3 Sketch a transistor-level schematic for a CMOS 4-input NOR gate.
- 1.4 Sketch a transistor-level schematic for a single-stage CMOS logic gate for each of the following functions:
 - a) $Y = \overline{ABC + D}$
 - b) $Y = \overline{(AB + C) \cdot D}$
 - c) $Y = \overline{AB + C \cdot (A + B)}$

- 1.5 Use a combination of CMOS gates (represented by their symbols) to generate the following functions from A, B, and C.
- $Y = A$ (buffer)
 - $Y = \overline{AB} + \overline{AB}$ (XOR)
 - $Y = \overline{AB} + AB$ (XNOR)
 - $Y = AB + BC + AC$ (majority)
- 1.6 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.
- 1.7 Sketch transistor-level schematics for the following logic functions. You may assume you have both true and complementary versions of the inputs available.
- A 2:4 decoder defined by

$$Y0 = \overline{A0} \cdot \overline{A1}$$

$$Y1 = \overline{A0} \cdot A1$$

$$Y2 = A0 \cdot \overline{A1}$$

$$Y3 = A0 \cdot A1$$
 - A 3:2 priority encoder defined by

$$Y0 = \overline{A0} \cdot (\overline{A1} + \overline{A2})$$

$$Y1 = \overline{A0} \cdot A1$$
- 1.8 Sketch a stick diagram for a CMOS 4-input NOR gate from Exercise 1.3.
- 1.9 Estimate the area of your 4-input NOR gate from Exercise 1.8.
- 1.10 Using a CAD tool of your choice, layout a 4-input NOR gate. How does its size compare to the prediction from Exercise 1.9?
- 1.11 Figure 1.72 shows a stick diagram of a 2-input NAND gate. Sketch a side view (cross-section) of the gate from X to X'.
- 1.12 Figure 1.73 gives a stick diagram for a level-sensitive latch. Estimate the area of the latch.
- 1.13 Draw a transistor-level schematic for the latch of Figure 1.73. How does the schematic differ from Figure 1.30(b)?
- 1.14 Consider the design of a CMOS compound OR-AND-INVERT (OAI) gate computing $F = (A + B) \cdot C$.
- sketch a transistor-level schematic
 - sketch a stick diagram
 - estimate the area from the stick diagram
 - layout your gate with a CAD tool
 - compare the layout size to the estimated area

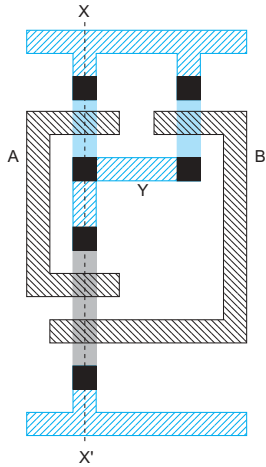


FIG 1.72 2-input NAND gate stick diagram

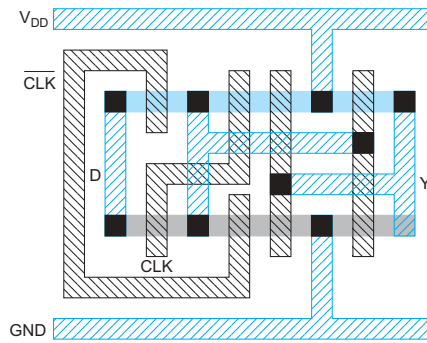


FIG 1.73 Level-sensitive latch stick diagram

- 1.15 Consider the design of a CMOS compound OR-OR-AND-INVERT (OOAI) gate computing $F = (A + B) \cdot (C + D)$.
- sketch a transistor-level schematic
 - sketch a stick diagram
 - estimate the area from the stick diagram
 - layout your gate with a CAD tool
 - compare the layout size to the estimated area
- 1.16 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



- 1.17 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 1.16(a)?
- 1.18 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 \bar{G} .
- sketch a transistor-level schematic
 - sketch a stick diagram
 - estimate the area from the stick diagram
- 1.19 The course Web page has a series of five 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. They cover:
- leaf cells: schematic entry, layout, icons, simulation, DRC, ERC, LVS; hierarchical design
 - complex leaf cell design and verification: full adder
 - hierarchical design: ALU assembly, datapath routing, and simulation
 - control design: Verilog, synthesis, place & route
 - chip assembly, pad frame, full-chip verification, tapeout

