Technologies for Building Processors and Memory

Year	Technology used in computers	Relative performance/unit cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit	900
1995	Very large-scale integrated circuit	2,400,000
2013	Ultra large-scale integrated circuit	250,000,000,000

FIGURE 5 Relative performance per unit cost of technologies used in computers over time. Source: Computer Museum, Boston, with 2013 extrapolated by the authors.

- A transistor is an on/off switch controlled by electricity.
- The integrated circuit (IC) combined dozens to hundreds of transistors into a single chip.
- Very Large-Scale Integrated (VLSI) circuit: A device containing hundreds of thousands to millions of transistors.
- The manufacture of a chip begins with silicon, a natural element that is a semiconductor. (Semiconductor is a substance that does not conduct electricity well.)

Technologies for Building Processors and Memory

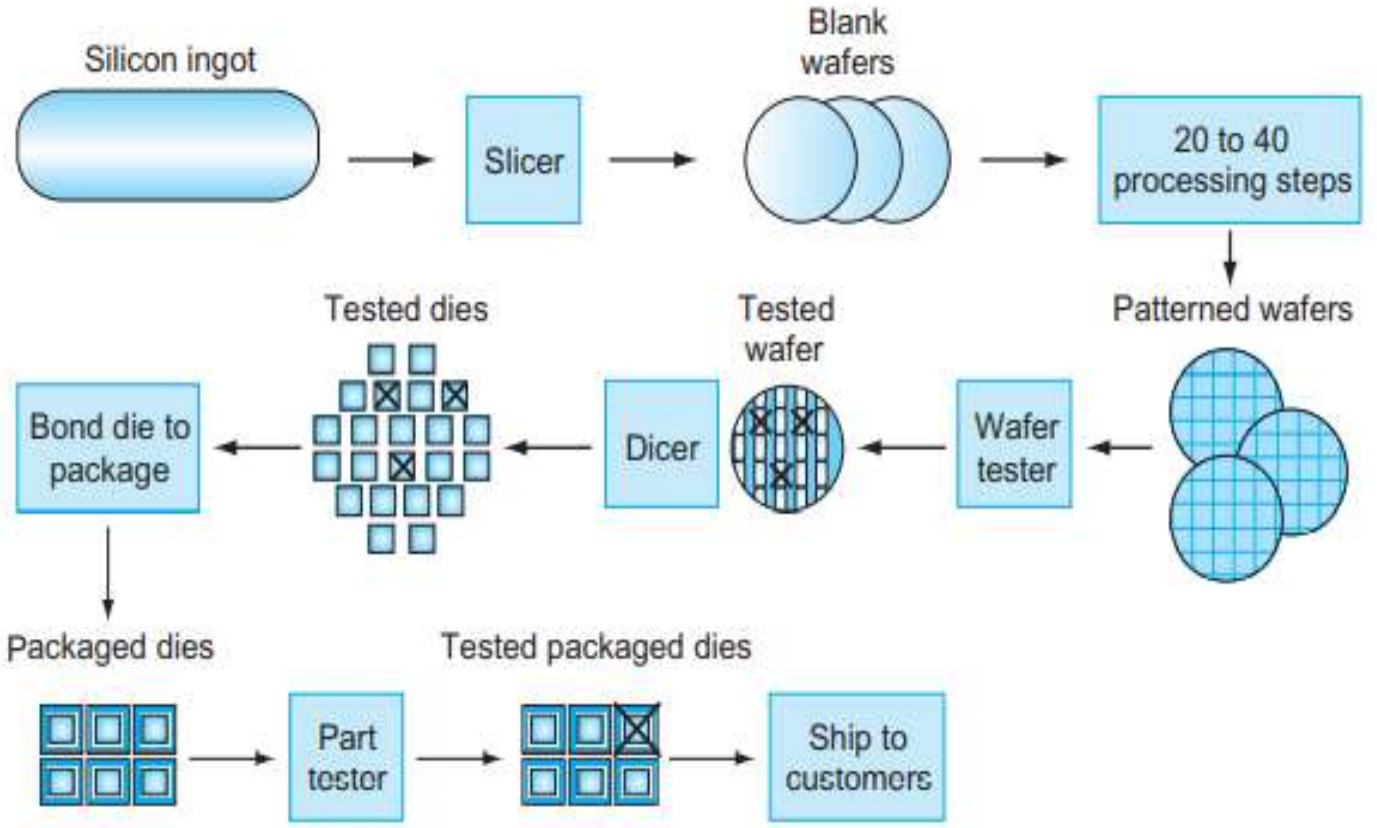

FIGURE 6 **The chip manufacturing process.** After being sliced from the silicon ingot, blank wafers are put through 20 to 40 steps to create patterned wafers These patterned wafers are then tested with a wafer tester, and a map of the good parts is made. Then, the wafers are diced into dies In this figure, one wafer produced 20 dies, of which 17 passed testing. (X means the die is bad.) The yield of good dies in this case was 17/20, or 85%. These good dies are then bonded into packages and tested one more time before shipping the packaged parts to customers. One bad packaged part was found in this final test.

Silicon crystal ingot: A rod composed of a silicon crystal that is between 8 and 12 inches in diameter and about 12 to 24 inches long.

Wafer: A slice from a silicon ingot no more than 0.1 inches thick, used to create chips.

[J. Hennessy and D. Patterson, "Computer Organization and Design: The Hardware/Software Interface", 5th Edition.]

Technologies for Building Processors and Memory

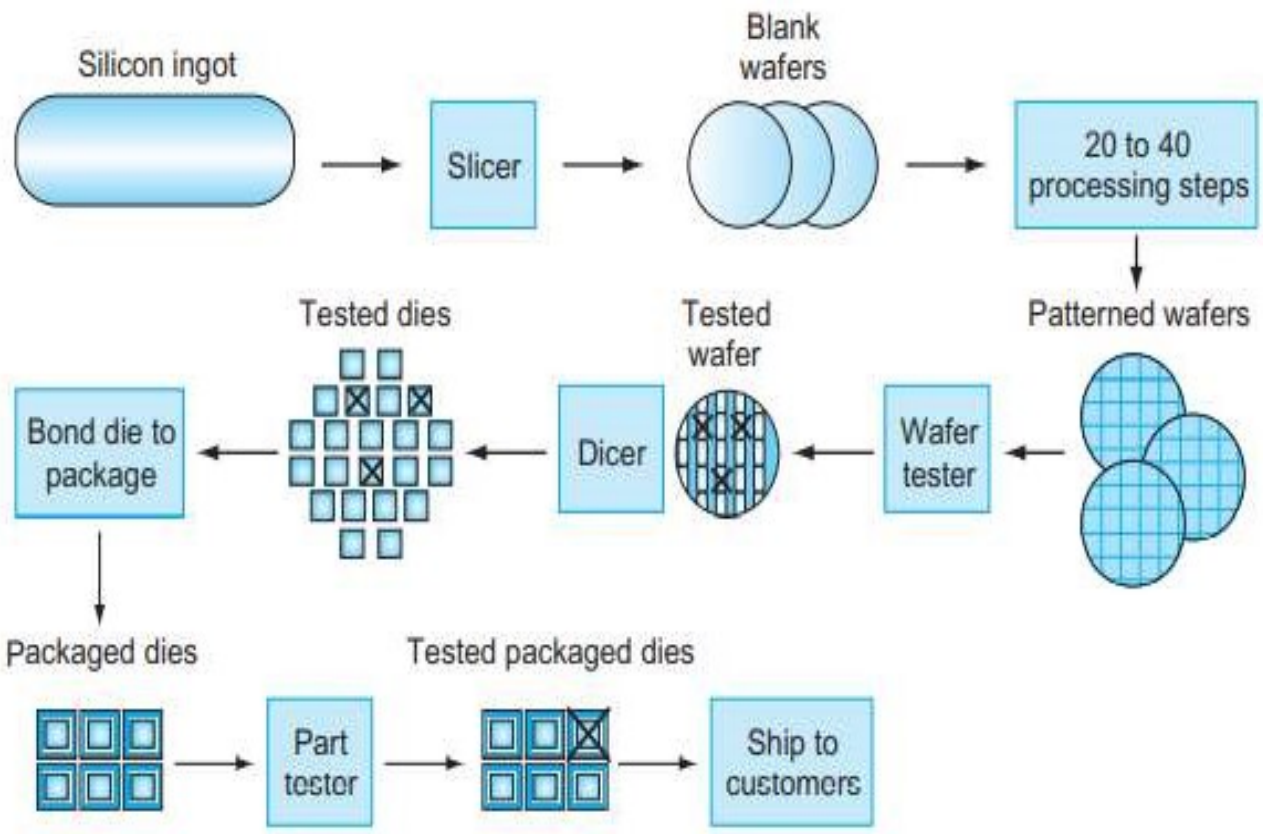

FIGURE 6 **The chip manufacturing process.** After being sliced from the silicon ingot, blank wafers are put through 20 to 40 steps to create patterned wafers. These patterned wafers are then tested with a wafer tester, and a map of the good parts is made. Then, the wafers are diced into dies. In this figure, one wafer produced 20 dies, of which 17 passed testing. (X means the die is bad.) The yield of good dies in this case was 17/20, or 85%. These good dies are then bonded into packages and tested one more time before shipping the packaged parts to customers. One bad packaged part was found in this final test.

A single microscopic flaw in the wafer itself or in one of the dozens of patterning steps can result in that area of the wafer failing. These defects make it virtually impossible to manufacture a perfect wafer.

The simplest way to cope with imperfection is to place many independent components on a single wafer. The patterned wafer is then chopped up, or diced, into these components, called dies and more informally known as chips.

Yield: The percentage of good dies from the total number of dies on the wafer.

[J. Hennessy and D. Patterson, "Computer Organization and Design: The Hardware/Software Interface", 5th Edition.]

Technologies for Building Processors and Memory

The cost of an integrated circuit can be expressed in three simple equations:

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Dies per wafer} \times \text{yield}}$$

$$\text{Dies per wafer} \approx \frac{\text{Wafer area}}{\text{Die area}}$$

$$\text{Yield} = \frac{1}{(1 + (\text{Defects per area} \times \text{Die area}/2))^2}$$

- The final equation is based on empirical observations of yields at integrated circuit factories, with the exponent related to the number of critical processing steps.
- Depending on the defect rate and the size of the die and wafer, costs are generally not linear in the die area

Performance

- Assessing the performance of computers can be quite challenging.
- Understanding how best to measure performance and the limitations of performance measurements is important in selecting a computer.

Performance

- Response time: Also called execution time. The total time required for the computer to complete a task, including disk accesses, memory accesses, I/O activities, operating system overhead, CPU execution time, and so on.
- As an individual computer user, we are interested in reducing response time—the time between the start and completion of a task.
- Datacenter managers are often interested in increasing throughput or bandwidth—the total amount of work done in a given time.
- Throughput: Also called bandwidth. Another measure of performance, it is the number of tasks completed per unit time.

Performance

To maximize performance, we want to minimize response time or execution time for some task. Thus, we can relate performance and execution time for a computer X:

$$\text{Performance}_X = \frac{1}{\text{Execution time}_X}$$

This means that for two computers X and Y, if the performance of X is greater than the performance of Y, we have

$$\text{Performance}_X > \text{Performance}_Y$$

$$\frac{1}{\text{Execution time}_X} > \frac{1}{\text{Execution time}_Y}$$

$$\text{Execution time}_Y > \text{Execution time}_X$$

That is, the execution time on Y is longer than that on X, if X is faster than Y.

Performance

In discussing a computer design, we often want to relate the performance of two different computers quantitatively. "X is n times faster than Y"—or equivalently "X is n times as fast as Y"—to mean

$$\frac{\text{Performance}_X}{\text{Performance}_Y} = n$$

If X is n times as fast as Y, then the execution time on Y is n times as long as it is on X

$$\frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution time}_Y}{\text{Execution time}_X} = n$$

Measuring Performance

Time is the measure of computer performance:

- The computer that performs the same amount of work in the least time is the fastest.
- Program execution time is measured in seconds per program.
- Time can be defined in different ways, depending on what we count.
- The most straightforward definition of time is called wall clock time, response time or elapsed time.
- These terms mean the total time to complete a task, including disk accesses, memory accesses, input/output (I/O) activities, operating system overhead—everything.

Measuring Performance

- CPU execution time or simply CPU time is the time the CPU spends computing for the task and does not include time spent waiting for I/O or running other programs(a processor may work on several programs simultaneously).
- CPU time can be further divided into the CPU time spent in the program, called user CPU time, and the CPU time spent in the operating system performing tasks on behalf of the program, called system CPU time.

  CPU time = User CPU time + System CPU time

- We will use the term system performance to refer to elapsed time on an unloaded system and CPU performance to refer to user CPU time.

Understanding Program Performance

- To improve the performance of a program, one must have a clear definition of what performance metric matters and then proceed to look for performance bottlenecks by measuring program execution and looking for the likely bottlenecks.
- Eg: the user may care about throughput, response time, or a complex combination of the two (e.g., maximum throughput with a worst-case response time)
- All computers are constructed using a clock that determines when events take place in the hardware. These discrete time intervals are called clock cycles (or ticks, clock ticks, clock periods, clocks, cycles).
- Designers refer to the length of a clock period both as the time for a complete clock cycle (e.g., 250 picoseconds, or 250 ps) and as the clock rate (e.g., 4 gigahertz, or 4 GHz), which is the inverse of the clock period. [The clock rate is the frequency at which a CPU is running]

CPU Performance and Its Factors

- A simple formula relates the most basic metrics (clock cycles and clock cycle time) to CPU time

$$\text{CPU execution time for a program} = \text{CPU clock cycles for a program} \times \text{Clock cycle time}$$

- Clock rate and clock cycle time are inverses,

$$\text{CPU execution time for a program} = \frac{\text{CPU clock cycles for a program}}{\text{Clock rate}}$$

- This formula makes it clear that the hardware designer can improve performance by reducing the number of clock cycles required for a program or the length of the clock cycle.