A **digital circuit** is typically constructed from small electronic **circuits** called logic gates that can be used to create combinational logic. Each logic gate is **designed** to perform a function of boolean logic when acting on logic signals.

**Contents**

  [[hide](https://en.wikipedia.org/wiki/Digital_electronics)]

* [1History](https://en.wikipedia.org/wiki/Digital_electronics#History)
* [2Properties](https://en.wikipedia.org/wiki/Digital_electronics#Properties)
* [3Construction](https://en.wikipedia.org/wiki/Digital_electronics#Construction)
* [4Design](https://en.wikipedia.org/wiki/Digital_electronics#Design)
  + [4.1Structure of digital systems](https://en.wikipedia.org/wiki/Digital_electronics#Structure_of_digital_systems)
    - [4.1.1Representation](https://en.wikipedia.org/wiki/Digital_electronics#Representation)
    - [4.1.2Combinational vs. Sequential](https://en.wikipedia.org/wiki/Digital_electronics#Combinational_vs._Sequential)
    - [4.1.3Synchronous systems](https://en.wikipedia.org/wiki/Digital_electronics#Synchronous_systems)
    - [4.1.4Asynchronous systems](https://en.wikipedia.org/wiki/Digital_electronics#Asynchronous_systems)
    - [4.1.5Register transfer systems](https://en.wikipedia.org/wiki/Digital_electronics#Register_transfer_systems)
    - [4.1.6Computer design](https://en.wikipedia.org/wiki/Digital_electronics#Computer_design)
    - [4.1.7Computer architecture](https://en.wikipedia.org/wiki/Digital_electronics#Computer_architecture)
    - [4.1.8Design issues in digital circuits](https://en.wikipedia.org/wiki/Digital_electronics#Design_issues_in_digital_circuits)
  + [4.2Automated design tools](https://en.wikipedia.org/wiki/Digital_electronics#Automated_design_tools)
  + [4.3Design for testability](https://en.wikipedia.org/wiki/Digital_electronics#Design_for_testability)
  + [4.4Trade-offs](https://en.wikipedia.org/wiki/Digital_electronics#Trade-offs)
    - [4.4.1Cost](https://en.wikipedia.org/wiki/Digital_electronics#Cost)
    - [4.4.2Reliability](https://en.wikipedia.org/wiki/Digital_electronics#Reliability)
    - [4.4.3Fanout](https://en.wikipedia.org/wiki/Digital_electronics#Fanout)
    - [4.4.4Speed](https://en.wikipedia.org/wiki/Digital_electronics#Speed)
  + [4.5Logic families](https://en.wikipedia.org/wiki/Digital_electronics#Logic_families)
* [5Recent developments](https://en.wikipedia.org/wiki/Digital_electronics#Recent_developments)
  + - 1. HISTORY

The [binary number system](https://en.wikipedia.org/wiki/Binary_number_system) was refined by [Gottfried Wilhelm Leibniz](https://en.wikipedia.org/wiki/Gottfried_Wilhelm_Leibniz) (published in 1705) and he also established that by using the binary system, the principles of arithmetic and logic could be combined. Digital logic as we know it was the brain-child of George Boole, in the mid 19th century. Boole died young, but his ideas lived on. In an 1886 letter,[Charles Sanders Peirce](https://en.wikipedia.org/wiki/Charles_Sanders_Peirce" \o "Charles Sanders Peirce) described how logical operations could be carried out by electrical switching circuits.[[2]](https://en.wikipedia.org/wiki/Digital_electronics#cite_note-P2M-2) Eventually, [vacuum tubes](https://en.wikipedia.org/wiki/Vacuum_tube) replaced relays for logic operations. [Lee De Forest](https://en.wikipedia.org/wiki/Lee_De_Forest)'s modification, in 1907, of the [Fleming valve](https://en.wikipedia.org/wiki/Vacuum_tube) can be used as an AND logic gate. [Ludwig Wittgenstein](https://en.wikipedia.org/wiki/Ludwig_Wittgenstein) introduced a version of the 16-row [truth table](https://en.wikipedia.org/wiki/Truth_table) as proposition 5.101 of *[Tractatus Logico-Philosophicus](https://en.wikipedia.org/wiki/Tractatus_Logico-Philosophicus" \o "Tractatus Logico-Philosophicus)* (1921). [Walther Bothe](https://en.wikipedia.org/wiki/Walther_Bothe), inventor of the [coincidence circuit](https://en.wikipedia.org/wiki/Coincidence_circuit), got part of the 1954 [Nobel Prize](https://en.wikipedia.org/wiki/Nobel_Prize) in physics, for the first modern electronic AND gate in 1924.

While working at [Texas Instruments](https://en.wikipedia.org/wiki/Texas_Instruments), [Jack Kilby](https://en.wikipedia.org/wiki/Jack_Kilby) recorded his initial ideas concerning the [integrated circuit](https://en.wikipedia.org/wiki/Integrated_circuit) in July 1958, successfully demonstrating the first working integrated example on 12 September 1958.[[6]](https://en.wikipedia.org/wiki/Digital_electronics#cite_note-TIJackBuilt-6) This new technique allowed for quick, low-cost fabrication of complex circuits by having a set of [electronic circuits](https://en.wikipedia.org/wiki/Electronic_circuit) on one small plate ("chip") of[semiconductor material](https://en.wikipedia.org/wiki/Semiconductor_material), normally [silicon](https://en.wikipedia.org/wiki/Silicon).

* + - 1. PROPERTIES

In a digital system, a more precise representation of a signal can be obtained by using more binary digits to represent it. While this requires more digital circuits to process the signals, each digit is handled by the same kind of hardware, resulting in an easily [scalable](https://en.wikipedia.org/wiki/Scalability) system. In an analog system, additional resolution requires fundamental improvements in the linearity and noise characteristics of each step of the [signal chain](https://en.wikipedia.org/wiki/Signal_chain_(signal_processing_chain)).

The noise-immunity of digital systems permits data to be stored and retrieved without degradation. In an analog system, noise from aging and wear degrade the information stored. In a digital system, as long as the total noise is below a certain level, the information can be recovered perfectly.

Even when more significant noise is present, the use of [redundancy](https://en.wikipedia.org/wiki/Redundancy_(information_theory)) permits the recovery of the original data provided too many errors do not occur.

Digital circuits are sometimes more expensive, especially in small quantities.

* + - 1. CONSTRUCTION
* A digital circuit is typically constructed from small electronic circuits called [logic gates](https://en.wikipedia.org/wiki/Logic_gate) that can be used to create [combinational logic](https://en.wikipedia.org/wiki/Combinational_logic). Each logic gate is designed to perform a function of [boolean logic](https://en.wikipedia.org/wiki/Boolean_logic" \o "Boolean logic) when acting on logic signals. A logic gate is generally created from one or more electrically controlled switches, usually [transistors](https://en.wikipedia.org/wiki/Transistors) but [thermionic valves](https://en.wikipedia.org/wiki/Thermionic_valves) have seen historic use. The output of a logic gate can, in turn, control or feed into more logic gates.
* [Integrated circuits](https://en.wikipedia.org/wiki/Integrated_circuit) consist of multiple transistors on one silicon chip, and are the least expensive way to make large number of interconnected logic gates. Integrated circuits are usually designed by engineers using [electronic design automation](https://en.wikipedia.org/wiki/Electronic_design_automation) software

4.DESIGN

Each logic symbol is represented by a different shape. The actual set of shapes was introduced in 1984 under IEEE/ANSI standard 91-1984. "The logic symbol given under this standard are being increasingly used now and have even started appearing in the literature published by manufacturers of digital integrated circuits.

### Structure of digital systems

Engineers use many methods to minimize logic functions, in order to reduce the circuit's complexity. When the complexity is less, the circuit also has fewer errors and less electronics, and is therefore less expensive.

The most widely used simplification is a minimization algorithm like the [Espresso heuristic logic minimizer](https://en.wikipedia.org/wiki/Espresso_heuristic_logic_minimizer) within a [CAD](https://en.wikipedia.org/wiki/Computer-aided_design) system, although historically, [binary decision diagrams](https://en.wikipedia.org/wiki/Binary_decision_diagrams), an automated [Quine–McCluskey algorithm](https://en.wikipedia.org/wiki/Quine%E2%80%93McCluskey_algorithm" \o "Quine–McCluskey algorithm), [truth tables](https://en.wikipedia.org/wiki/Truth_table), [Karnaugh maps](https://en.wikipedia.org/wiki/Karnaugh_map" \o "Karnaugh map), and [Boolean algebra](https://en.wikipedia.org/wiki/Boolean_algebra_(logic)) have been used.

#### Representation

Representations are crucial to an engineer's design of digital circuits. Some analysis methods only work with particular representations

 One of the easiest ways is to simply have a memory containing a truth table. The inputs are fed into the address of the memory, and the data outputs of the memory become the outputs.

For automated analysis, these representations have digital file formats that can be processed by computer programs. Most digital engineers are very careful to select computer programs ("tools") with compatible file formats.

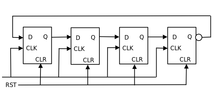
#### Combinational vs. Sequential

To choose representations, engineers consider types of digital systems. Most digital systems divide into "[combinational systems](https://en.wikipedia.org/wiki/Combinational_logic)" and "[sequential systems](https://en.wikipedia.org/wiki/Sequential_logic)." A combinational system always presents the same output when given the same inputs. It is basically a representation of a set of logic functions, as already discussed.

A sequential system is a combinational system with some of the outputs fed back as inputs. This makes the digital machine perform a "sequence" of operations. The simplest sequential system is probably a [flip flop](https://en.wikipedia.org/wiki/Flip-flop_(electronics)), a mechanism that represents a [binary](https://en.wikipedia.org/wiki/Binary_number_system) [digit](https://en.wikipedia.org/wiki/Numerical_digit) or "[bit](https://en.wikipedia.org/wiki/Bit)".

Sequential systems are often designed as [state machines](https://en.wikipedia.org/wiki/State_machine). Sequential systems divide into two further subcategories. ["Synchronous" sequential systems](https://en.wikipedia.org/wiki/Synchronous_system) change state all at once, when a "clock" signal changes state. ["Asynchronous" sequential systems](https://en.wikipedia.org/wiki/Asynchronous_system) propagate changes whenever inputs change

#### Synchronous systems

[](https://en.wikipedia.org/wiki/File:JohnsonCounter2.png)

A 4-bit ring counter using D-type flip flops is an example of synchronous logic. Each device is connected to the clock signal, and update together.

The usual way to implement a synchronous sequential state machine is to divide it into a piece of combinational logic and a set of flip flops called a "state register." Each time a clock signal ticks, the state register captures the feedback generated from the previous state of the combinational logic, and feeds it back as an unchanging input to the combinational part of the state machine. The fastest rate of the clock is set by the most time-consuming logic calculation in the combinational logic.

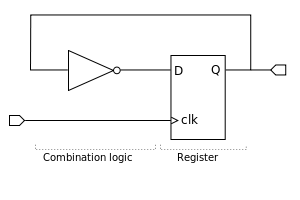
The state register is just a representation of a binary number. If the states in the state machine are numbered (easy to arrange), the logic function is some combinational logic that produces the number of the next state.

#### Asynchronous systems

Many systems need circuits that allow external unsynchronized signals to enter synchronous logic circuits. These are inherently asynchronous in their design and must be analyzed as such. Examples of widely used asynchronous circuits include synchronizer flip-flops, switch [debouncers](https://en.wikipedia.org/wiki/Debounce" \o "Debounce) and [arbiters](https://en.wikipedia.org/wiki/Arbiter_(electronics)).

Asynchronous logic components can be hard to design because all possible states, in all possible timings must be considered. The usual method is to construct a table of the minimum and maximum time that each such state can exist, and then adjust the circuit to minimize the number of such states. Then the designer must force the circuit to periodically wait for all of its parts to enter a compatible state (this is called "self-resynchronization"). Without such careful design, it is easy to accidentally produce asynchronous logic that is "unstable," that is, real electronics will have unpredictable results because of the cumulative delays caused by small variations in the values of the electronic components.

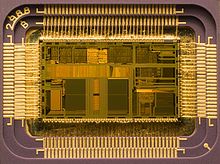
#### Register transfer systems

[](https://en.wikipedia.org/wiki/File:Register_transfer_level_-_example_toggler.svg)Example of a simple circuit with a toggling output. The inverter forms the combinational logic in this circuit, and the register holds the state.

Many digital systems are data flow machines. These are usually designed using synchronous [register transfer logic](https://en.wikipedia.org/wiki/Register_transfer_level), using[hardware description languages](https://en.wikipedia.org/wiki/Hardware_description_language) such as [VHDL](https://en.wikipedia.org/wiki/VHDL) or [Verilog](https://en.wikipedia.org/wiki/Verilog" \o "Verilog).

In register transfer logic, binary [numbers](https://en.wikipedia.org/wiki/Number) are stored in groups of flip flops called [registers](https://en.wikipedia.org/wiki/Processor_register). The outputs of each register are a bundle of wires called a "[bus](https://en.wikipedia.org/wiki/Computer_bus)" that carries that number to other calculations. A calculation is simply a piece of combinational logic. Each calculation also has an output bus, and these may be connected to the inputs of several registers. Sometimes a register will have a [multiplexer](https://en.wikipedia.org/wiki/Multiplexer) on its input, so that it can store a number from any one of several buses. Alternatively, the outputs of several items may be connected to a bus through [buffers](https://en.wikipedia.org/wiki/3-state) that can turn off the output of all of the devices except one. A sequential state machine controls when each register accepts new data from its input.

#### Computer design

[](https://en.wikipedia.org/wiki/File:80486dx2-large.jpg)

Intel 80486DX2 [microprocessor](https://en.wikipedia.org/wiki/Microprocessor)

The most general-purpose register-transfer logic machine is a [computer](https://en.wikipedia.org/wiki/Computer). This is basically an [automatic](https://en.wikipedia.org/wiki/Automaton) binary [abacus](https://en.wikipedia.org/wiki/Abacus). The [control unit](https://en.wikipedia.org/wiki/Control_unit) of a computer is usually designed as a [microprogram](https://en.wikipedia.org/wiki/Microprogram" \o "Microprogram) run by a [microsequencer](https://en.wikipedia.org/wiki/Microsequencer" \o "Microsequencer). A microprogram is much like a player-piano roll. Each table entry or "word" of the microprogram commands the state of every bit that controls the computer. The sequencer then counts, and the count addresses the memory or combinational logic machine that contains the microprogram. The bits from the microprogram control the[arithmetic logic unit](https://en.wikipedia.org/wiki/Arithmetic_logic_unit), [memory](https://en.wikipedia.org/wiki/Memory) and other parts of the computer, including the microsequencer itself.A "specialized computer" is usually a conventional computer with special-purpose control logic or microprogram.

In this way, the complex task of designing the controls of a computer is reduced to a simpler task of programming a collection of much simpler logic machines.

Almost all computers are synchronous. However, true asynchronous computers have also been designed. One example is the Aspida [DLX](https://en.wikipedia.org/wiki/DLX" \o "DLX)core.[[10]](https://en.wikipedia.org/wiki/Digital_electronics#cite_note-aspida-10) Another was offered by [ARM Holdings](https://en.wikipedia.org/wiki/ARM_Holdings).

### Automated design tools

To save costly engineering effort, much of the effort of designing large logic machines has been automated. The computer programs are called "[electronic design automation](https://en.wikipedia.org/wiki/Electronic_design_automation)tools" or just "EDA."

Simple truth table-style descriptions of logic are often optimized with EDA that automatically produces reduced systems of logic gates or smaller lookup tables that still produce the desired outputs. The most common example of this kind of software is the [Espresso heuristic logic minimizer](https://en.wikipedia.org/wiki/Minilog).

Most practical algorithms for optimizing large logic systems use [algebraic manipulations](https://en.wikipedia.org/wiki/Quine%E2%80%93McCluskey_algorithm) or [binary decision diagrams](https://en.wikipedia.org/wiki/Binary_decision_diagram), and there are promising experiments with [genetic algorithms](https://en.wikipedia.org/wiki/Genetic_algorithm)and [annealing optimizations](https://en.wikipedia.org/wiki/Simulated_annealing).

To automate costly engineering processes, some EDA can take [state tables](https://en.wikipedia.org/wiki/State_table) that describe [state machines](https://en.wikipedia.org/wiki/State_machine) and automatically produce a truth table or a [function table](https://en.wikipedia.org/wiki/Function_table) for the[combinational logic](https://en.wikipedia.org/wiki/Combinational_logic) of a state machine. The state table is a piece of text that lists each state, together with the conditions controlling the transitions between them and the belonging output signals.

### Design for testability

There are several reasons for testing a logic circuit. When the circuit is first developed, it is necessary to verify that the design circuit meets the required functional and timing specifications. When multiple copies of a correctly designed circuit are being manufactured, it is essential to test each copy to ensure that the manufacturing process has not introduced any flaws.

Fortunately, large logic machines are almost always designed as assemblies of smaller logic machines. To save time, the smaller sub-machines are isolated by permanently installed "design for test" circuitry, and are tested independently.

One common test scheme known as "scan design" moves test bits serially (one after another) from external test equipment through one or more serial [shift registers](https://en.wikipedia.org/wiki/Shift_register) known as "scan chains". Serial scans have only one or two wires to carry the data, and minimize the physical size and expense of the infrequently used test logic.

#### Cost

The cost of a logic gate is crucial, primarily because very many gates are needed to build a computer or other advanced digital system and because the more gates can be used, the more capable and/or fast the machine can be. Since the majority of a digital computer is simply an interconnected network of logic gates, the overall cost of building a computer correlates strongly with the price per logic gate. In the 1930s, the earliest digital logic systems were constructed from telephone relays because these were inexpensive and relatively reliable. After that, engineers always used the cheapest available electronic switches that could still fulfill the requirements.

#### Reliability[[edit](https://en.wikipedia.org/w/index.php?title=Digital_electronics&action=edit&section=18" \o "Edit section: Reliability)]

The "reliability" of a logic gate describes its mean time between failure (MTBF). Digital machines often have millions of logic gates. Also, most digital machines are "optimized" to reduce their cost. The result is that often, the failure of a single logic gate will cause a digital machine to stop working. It is possible to design machines to be more reliable by using redundant logic which will not malfunction as a result of the failure of any single gate (or even any two, three, or four gates), but this necessarily entails using more components, which raises the financial cost and also usually increases the weight of the machine and may increase the power it consumes.

Modern transistorized integrated circuit logic gates have MTBFs greater than 82 billion hours (8.2×1010) hours,[[13]](https://en.wikipedia.org/wiki/Digital_electronics" \l "cite_note-13) and need them because they have so many logic gates.

#### Fanout

Fanout describes how many logic inputs can be controlled by a single logic output without exceeding the electrical current ratings of the gate outputs.[[14]](https://en.wikipedia.org/wiki/Digital_electronics#cite_note-14) The minimum practical fanout is about five. Modern electronic logic gates using [CMOS](https://en.wikipedia.org/wiki/CMOS) transistors for switches have fanouts near fifty, and can sometimes go much higher.

#### Speed

The "switching speed" describes how many times per second an inverter (an electronic representation of a "logical not" function) can change from true to false and back. Faster logic can accomplish more operations in less time. Digital logic first became useful when switching speeds got above fifty [hertz](https://en.wikipedia.org/wiki/Hertz), because that was faster than a team of humans operating mechanical calculators. Modern electronic digital logic routinely switches at five [gigahertz](https://en.wikipedia.org/wiki/Gigahertz) (5×109 hertz), and some laboratory systems switch at more than a [terahertz](https://en.wikipedia.org/wiki/Terahertz)(1×1012 hertz).

### Logic families

Design started with [relays](https://en.wikipedia.org/wiki/Relay). Relay logic was relatively inexpensive and reliable, but slow. Occasionally a mechanical failure would occur. Fanouts were typically about ten, limited by the resistance of the coils and arcing on the contacts from high voltages.

Later, [vacuum tubes](https://en.wikipedia.org/wiki/Vacuum_tube) were used. These were very fast, but generated heat, and were unreliable because the filaments would burn out. Fanouts were typically five to seven, limited by the heating from the tubes' current. In the 1950s, special "computer tubes" were developed with filaments that omitted volatile elements like silicon. These ran for hundreds of thousands of hours.

The first [semiconductor](https://en.wikipedia.org/wiki/Semiconductor) logic family was [resistor–transistor logic](https://en.wikipedia.org/wiki/Resistor%E2%80%93transistor_logic). This was a thousand times more reliable than tubes, ran cooler, and used less power, but had a very low [fan-in](https://en.wikipedia.org/wiki/Fan-in)of three. [Diode–transistor logic](https://en.wikipedia.org/wiki/Diode%E2%80%93transistor_logic) improved the fanout up to about seven, and reduced the power. Some DTL designs used two power-supplies with alternating layers of NPN and PNP transistors to increase the fanout.

[Transistor–transistor logic](https://en.wikipedia.org/wiki/Transistor%E2%80%93transistor_logic) (TTL) was a great improvement over these. In early devices, fanout improved to ten, and later variations reliably achieved twenty. TTL was also fast, with some variations achieving switching times as low as twenty nanoseconds. TTL is still used in some designs.

[Emitter coupled logic](https://en.wikipedia.org/wiki/Emitter_coupled_logic) is very fast but uses a lot of power. It was extensively used for high-performance computers made up of many medium-scale components (such as the [Illiac IV](https://en.wikipedia.org/wiki/Illiac_IV" \o "Illiac IV)).

By far, the most common digital integrated circuits built today use [CMOS logic](https://en.wikipedia.org/wiki/CMOS), which is fast, offers high circuit density and low-power per gate. This is used even in large, fast computers, such as the [IBM System z](https://en.wikipedia.org/wiki/IBM_System_z).

## Recent developments

In 2009, researchers discovered that [memristors](https://en.wikipedia.org/wiki/Memristor" \o "Memristor) can implement a boolean state storage (similar to a [flip flop](https://en.wikipedia.org/wiki/Flip-flop_(electronics)), [implication](https://en.wikipedia.org/wiki/Logical_consequence) and logical [inversion](https://en.wikipedia.org/wiki/Inverse_(logic))), providing a complete logic family with very small amounts of space and power, using familiar CMOS semiconductor processes.[[15]](https://en.wikipedia.org/wiki/Digital_electronics#cite_note-15)

The discovery of [superconductivity](https://en.wikipedia.org/wiki/Superconductivity) has enabled the development of [rapid single flux quantum](https://en.wikipedia.org/wiki/Rapid_single_flux_quantum) (RSFQ) circuit technology, which uses [Josephson junctions](https://en.wikipedia.org/wiki/Josephson_junction) instead of transistors. Most recently, attempts are being made to construct purely [optical computing](https://en.wikipedia.org/wiki/Optical_computing) systems capable of processing digital information using [nonlinear](https://en.wikipedia.org/wiki/Nonlinear) optical elements.