[Template:Redirect](/wiki/Template:Redirect" \o "Template:Redirect) [Template:Pp-semi-indef](/wiki/Template:Pp-semi-indef) [Template:Pp-move-indef](/wiki/Template:Pp-move-indef) [Template:Infobox](/wiki/Template:Infobox)

A **computer** is a general purpose device that can be [programmed](/wiki/Computer_program) to carry out a set of [arithmetic](/wiki/Arithmetic) or [logical](/wiki/Boolean_algebra) operations automatically. Since a sequence of operations can be readily changed, the computer can solve more than one kind of problem.

Conventionally, a computer consists of at least one [processing element](/wiki/Processing_element), typically a [central processing unit](/wiki/Central_processing_unit) (CPU), and some form of [memory](/wiki/Memory_(computers)). The processing element carries out arithmetic and logic operations, and a sequencing and control unit can change the order of operations in response to stored [information](/wiki/Data). [Peripheral devices](/wiki/Peripheral) allow information to be retrieved from an external source, and the result of operations saved and retrieved.

## Contents

* 1 Etymology[[edit](/index.php?title=(none)&action=edit&section=1)]
* 2 History[[edit](/index.php?title=(none)&action=edit&section=2)]
  + 2.1 Pre-twentieth century[[edit](/index.php?title=(none)&action=edit&section=3)]
  + 2.2 First computing device[[edit](/index.php?title=(none)&action=edit&section=4)]
  + 2.3 Analog computers[[edit](/index.php?title=(none)&action=edit&section=5)]
  + 2.4 Digital Computers[[edit](/index.php?title=(none)&action=edit&section=6)]
    - 2.4.1 Electromechanical[[edit](/index.php?title=(none)&action=edit&section=7)]
    - 2.4.2 {{anchor|digital computer|digital}}Vacuum tubes and digital electronic circuits[[edit](/index.php?title=(none)&action=edit&section=8)]
    - 2.4.3 Transistors[[edit](/index.php?title=(none)&action=edit&section=12)]
    - 2.4.4 Integrated circuits[[edit](/index.php?title=(none)&action=edit&section=13)]
  + 2.5 Mobile computers become dominant[[edit](/index.php?title=(none)&action=edit&section=14)]
* 3 Programs[[edit](/index.php?title=(none)&action=edit&section=15)]
  + 3.1 Stored program architecture[[edit](/index.php?title=(none)&action=edit&section=16)]
  + 3.2 Machine code[[edit](/index.php?title=(none)&action=edit&section=17)]
  + 3.3 Programming language[[edit](/index.php?title=(none)&action=edit&section=18)]
    - 3.3.1 Low-level languages[[edit](/index.php?title=(none)&action=edit&section=19)]
    - 3.3.2 High-level languages/Third Generation Language[[edit](/index.php?title=(none)&action=edit&section=20)]
  + 3.4 Fourth Generation Languages[[edit](/index.php?title=(none)&action=edit&section=21)]
  + 3.5 Program design[[edit](/index.php?title=(none)&action=edit&section=22)]
  + 3.6 Bugs[[edit](/index.php?title=(none)&action=edit&section=23)]
* 4 Components[[edit](/index.php?title=(none)&action=edit&section=24)]
  + 4.1 Control unit[[edit](/index.php?title=(none)&action=edit&section=25)]
  + 4.2 Central processing unit (CPU)[[edit](/index.php?title=(none)&action=edit&section=26)]
  + 4.3 Arithmetic logic unit (ALU)[[edit](/index.php?title=(none)&action=edit&section=27)]
  + 4.4 Memory[[edit](/index.php?title=(none)&action=edit&section=28)]
  + 4.5 Input/output (I/O)[[edit](/index.php?title=(none)&action=edit&section=29)]
  + 4.6 Multitasking[[edit](/index.php?title=(none)&action=edit&section=30)]
  + 4.7 Multiprocessing[[edit](/index.php?title=(none)&action=edit&section=31)]
* 5 Networking and the Internet[[edit](/index.php?title=(none)&action=edit&section=32)]
  + 5.1 Computer architecture paradigms[[edit](/index.php?title=(none)&action=edit&section=33)]
* 6 Misconceptions[[edit](/index.php?title=(none)&action=edit&section=34)]
  + 6.1 Unconventional computing[[edit](/index.php?title=(none)&action=edit&section=35)]
* 7 Future[[edit](/index.php?title=(none)&action=edit&section=36)]
* 8 Further topics[[edit](/index.php?title=(none)&action=edit&section=37)]
  + 8.1 Artificial intelligence[[edit](/index.php?title=(none)&action=edit&section=38)]
* 9 Hardware[[edit](/index.php?title=(none)&action=edit&section=39)]
  + 9.1 History of computing hardware[[edit](/index.php?title=(none)&action=edit&section=40)]
  + 9.2 Other hardware topics[[edit](/index.php?title=(none)&action=edit&section=41)]
* 10 Software[[edit](/index.php?title=(none)&action=edit&section=42)]
* 11 Languages[[edit](/index.php?title=(none)&action=edit&section=43)]
  + 11.1 Firmware[[edit](/index.php?title=(none)&action=edit&section=44)]
* 12 Types of computers[[edit](/index.php?title=(none)&action=edit&section=45)]
  + 12.1 Based on uses[[edit](/index.php?title=(none)&action=edit&section=46)]
  + 12.2 Based on sizes[[edit](/index.php?title=(none)&action=edit&section=47)]
* 13 Input Devices[[edit](/index.php?title=(none)&action=edit&section=48)]
* 14 Output Devices[[edit](/index.php?title=(none)&action=edit&section=49)]
* 15 Professions and organizations[[edit](/index.php?title=(none)&action=edit&section=50)]
* 16 See also[[edit](/index.php?title=(none)&action=edit&section=51)]
* 17 Notes[[edit](/index.php?title=(none)&action=edit&section=52)]
* 18 References[[edit](/index.php?title=(none)&action=edit&section=53)]
* 19 External links[[edit](/index.php?title=(none)&action=edit&section=54)]

## Etymology[[edit](/index.php?title=(none)&action=edit&section=1)]

The first known use of the word "computer" was in 1613 in a book called *The Yong Mans Gleanings* by English writer Richard Braithwait: "I haue read the truest computer of Times, and the best Arithmetician that euer breathed, and he reduceth thy dayes into a short number." It referred to a person who carried out calculations, or computations. The word continued with the same meaning until the middle of the 20th century. From the end of the 19th century the word began to take on its more familiar meaning, a machine that carries out computations.<ref name=OEDComputer>[Template:Cite journal](/wiki/Template:Cite_journal)</ref>

## History[[edit](/index.php?title=(none)&action=edit&section=2)]

[Template:Main](/wiki/Template:Main)

### Pre-twentieth century[[edit](/index.php?title=(none)&action=edit&section=3)]

[thumb|100px|The](/wiki/File:Os_d'Ishango_IRSNB.JPG) [Ishango bone](/wiki/Ishango_bone) Devices have been used to aid computation for thousands of years, mostly using [one-to-one correspondence](/wiki/One-to-one_correspondence) with [fingers](/wiki/Finger_counting). The earliest counting device was probably a form of [tally stick](/wiki/Tally_stick). Later record keeping aids throughout the [Fertile Crescent](/wiki/Fertile_Crescent) included calculi (clay spheres, cones, etc.) which represented counts of items, probably livestock or grains, sealed in hollow unbaked clay containers.[[1]](#cite_note-1)[[2]](#cite_note-2) The use of [counting rods](/wiki/Counting_rods) is one example. [thumb|left|The Chinese](/wiki/File:Abacus_6.png) [Suanpan](/wiki/Suanpan) (算盘) (the number represented on this abacus is 6,302,715,408) The [abacus](/wiki/Abacus) was initially used for arithmetic tasks. The [Roman abacus](/wiki/Roman_abacus) was used in [Babylonia](/wiki/Babylonia) as early as 2400 BC. Since then, many other forms of reckoning boards or tables have been invented. In a medieval European [counting house](/wiki/Counting_house), a checkered cloth would be placed on a table, and markers moved around on it according to certain rules, as an aid to calculating sums of money.

[thumb|The ancient Greek-designed](/wiki/File:NAMA_Machine_d'Anticythère_1.jpg) [Antikythera mechanism](/wiki/Antikythera_mechanism), dating between 150 and 100 BC, is the world's oldest analog computer. The [Antikythera mechanism](/wiki/Antikythera_mechanism) is believed to be the earliest mechanical analog "computer", according to [Derek J. de Solla Price](/wiki/Derek_J._de_Solla_Price).[[3]](#cite_note-3) It was designed to calculate astronomical positions. It was discovered in 1901 in the [Antikythera wreck](/wiki/Antikythera_wreck) off the Greek island of [Antikythera](/wiki/Antikythera), between [Kythera](/wiki/Kythera) and [Crete](/wiki/Crete), and has been dated to *circa* 100 BC. Devices of a level of complexity comparable to that of the Antikythera mechanism would not reappear until a thousand years later.

Many mechanical aids to calculation and measurement were constructed for astronomical and navigation use. The [planisphere](/wiki/Planisphere) was a [star chart](/wiki/Star_chart) invented by Abū Rayhān al-Bīrūnī in the early 11th century.<ref name=Wiet>G. Wiet, V. Elisseeff, P. Wolff, J. Naudu (1975). *History of Mankind, Vol 3: The Great medieval Civilisations*, p. 649. George Allen & Unwin Ltd, [UNESCO](/wiki/UNESCO).</ref> The [astrolabe](/wiki/Astrolabe) was invented in the [Hellenistic world](/wiki/Hellenistic_civilization) in either the 1st or 2nd centuries BC and is often attributed to [Hipparchus](/wiki/Hipparchus). A combination of the planisphere and [dioptra](/wiki/Dioptra), the astrolabe was effectively an analog computer capable of working out several different kinds of problems in [spherical astronomy](/wiki/Spherical_astronomy). An astrolabe incorporating a mechanical [calendar](/wiki/Calendar) computer[[4]](#cite_note-4)[[5]](#cite_note-5) and [gear](/wiki/Gear)-wheels was invented by Abi Bakr of [Isfahan](/wiki/Isfahan), [Persia](/wiki/Persia) in 1235.[[6]](#cite_note-6) [Abū Rayhān al-Bīrūnī](/wiki/Abū_Rayhān_al-Bīrūnī) invented the first mechanical geared [lunisolar calendar](/wiki/Lunisolar_calendar) astrolabe,[[7]](#cite_note-7) an early fixed-[wired](/wiki/Wire) knowledge processing [machine](/wiki/Machine)<ref name=Oren>[Template:Cite journal](/wiki/Template:Cite_journal)</ref> with a [gear train](/wiki/Gear_train) and gear-wheels,[[8]](#cite_note-8) *circa* 1000 AD.

The [sector](/wiki/Sector_(instrument)), a calculating instrument used for solving problems in proportion, trigonometry, multiplication and division, and for various functions, such as squares and cube roots, was developed in the late 16th century and found application in gunnery, surveying and navigation.

The [planimeter](/wiki/Planimeter) was a manual instrument to calculate the area of a closed figure by tracing over it with a mechanical linkage.

[thumb|A slide rule](/wiki/Image:Sliderule_2005.png) The [slide rule](/wiki/Slide_rule) was invented around 1620–1630, shortly after the publication of the concept of the [logarithm](/wiki/Logarithm). It is a hand-operated analog computer for doing multiplication and division. As slide rule development progressed, added scales provided reciprocals, squares and square roots, cubes and cube roots, as well as [transcendental functions](/wiki/Transcendental_function) such as logarithms and exponentials, circular and hyperbolic trigonometry and other [functions](/wiki/Function_(mathematics)). Aviation is one of the few fields where slide rules are still in widespread use, particularly for solving time–distance problems in light aircraft. To save space and for ease of reading, these are typically circular devices rather than the classic linear slide rule shape. A popular example is the [E6B](/wiki/E6B).

In the 1770s [Pierre Jaquet-Droz](/wiki/Pierre_Jaquet-Droz), a Swiss [watchmaker](/wiki/Watchmaker), built a mechanical doll ([automata](/wiki/Automata)) that could write holding a quill pen. By switching the number and order of its internal wheels different letters, and hence different messages, could be produced. In effect, it could be mechanically "programmed" to read instructions. Along with two other complex machines, the doll is at the Musée d'Art et d'Histoire of [Neuchâtel](/wiki/Neuchâtel), [Switzerland](/wiki/Switzerland), and still operates.[[9]](#cite_note-9) The [tide-predicting machine](/wiki/Tide-predicting_machine) invented by [Sir William Thomson](/wiki/William_Thomson,_1st_Baron_Kelvin) in 1872 was of great utility to navigation in shallow waters. It used a system of pulleys and wires to automatically calculate predicted tide levels for a set period at a particular location.

The [differential analyser](/wiki/Differential_analyser), a mechanical analog computer designed to solve [differential equations](/wiki/Differential_equations) by [integration](/wiki/Integral), used wheel-and-disc mechanisms to perform the integration. In 1876 [Lord Kelvin](/wiki/James_Thomson_(engineer)) had already discussed the possible construction of such calculators, but he had been stymied by the limited output torque of the [ball-and-disk integrators](/wiki/Ball-and-disk_integrator).[[10]](#cite_note-10) In a differential analyzer, the output of one integrator drove the input of the next integrator, or a graphing output. The [torque amplifier](/wiki/Torque_amplifier) was the advance that allowed these machines to work. Starting in the 1920s, [Vannevar Bush](/wiki/Vannevar_Bush) and others developed mechanical differential analyzers.

### First computing device[[edit](/index.php?title=(none)&action=edit&section=4)]

[thumb|A portion of](/wiki/File:Difference_engine_plate_1853.jpg) [Babbage's](/wiki/Charles_Babbage) [Difference engine](/wiki/Difference_engine). [Charles Babbage](/wiki/Charles_Babbage), an English mechanical engineer and [polymath](/wiki/Polymath), originated the concept of a programmable computer. Considered the "[father of the computer](/wiki/Computer_pioneer)",[[11]](#cite_note-11) he conceptualized and invented the first [mechanical computer](/wiki/Mechanical_computer) in the early 19th century. After working on his revolutionary [difference engine](/wiki/Difference_engine), designed to aid in navigational calculations, in 1833 he realized that a much more general design, an [Analytical Engine](/wiki/Analytical_Engine), was possible. The input of programs and data was to be provided to the machine via [punched cards](/wiki/Punched_card), a method being used at the time to direct mechanical [looms](/wiki/Loom) such as the [Jacquard loom](/wiki/Jacquard_loom). For output, the machine would have a printer, a curve plotter and a bell. The machine would also be able to punch numbers onto cards to be read in later. The Engine incorporated an [arithmetic logic unit](/wiki/Arithmetic_logic_unit), [control flow](/wiki/Control_flow) in the form of [conditional branching](/wiki/Conditional_branching) and [loops](/wiki/Program_loop#Loops), and integrated [memory](/wiki/Computer_memory), making it the first design for a general-purpose computer that could be described in modern terms as [Turing-complete](/wiki/Turing-complete).[[12]](#cite_note-12)[[13]](#cite_note-13) The machine was about a century ahead of its time. All the parts for his machine had to be made by hand — this was a major problem for a device with thousands of parts. Eventually, the project was dissolved with the decision of the [British Government](/wiki/British_Government) to cease funding. Babbage's failure to complete the analytical engine can be chiefly attributed to difficulties not only of politics and financing, but also to his desire to develop an increasingly sophisticated computer and to move ahead faster than anyone else could follow. Nevertheless, his son, Henry Babbage, completed a simplified version of the analytical engine's computing unit (the *mill*) in 1888. He gave a successful demonstration of its use in computing tables in 1906.

### Analog computers[[edit](/index.php?title=(none)&action=edit&section=5)]

[thumb|left|200px|](/wiki/File:099-tpm3-sk.jpg)[Sir William Thomson's](/wiki/William_Thomson,_1st_Baron_Kelvin) third tide-predicting machine design, 1879–81 During the first half of the 20th century, many scientific [computing](/wiki/Computing) needs were met by increasingly sophisticated [analog computers](/wiki/Analog_computer), which used a direct mechanical or electrical model of the problem as a basis for [computation](/wiki/Computation). However, these were not programmable and generally lacked the versatility and accuracy of modern digital computers.[[14]](#cite_note-14) The first modern analog computer was a [tide-predicting machine](/wiki/Tide-predicting_machine), invented by [Sir William Thomson](/wiki/William_Thomson,_1st_Baron_Kelvin) in 1872. The [differential analyser](/wiki/Differential_analyser), a mechanical analog computer designed to solve differential equations by integration using wheel-and-disc mechanisms, was conceptualized in 1876 by [James Thomson](/wiki/James_Thomson_(engineer)), the brother of the more famous Lord Kelvin.[[10]](#cite_note-10) The art of mechanical analog computing reached its zenith with the [differential analyzer](/wiki/Differential_analyzer), built by H. L. Hazen and [Vannevar Bush](/wiki/Vannevar_Bush) at [MIT](/wiki/MIT) starting in 1927. This built on the mechanical integrators of [James Thomson](/wiki/James_Thomson_(engineer)) and the torque amplifiers invented by H. W. Nieman. A dozen of these devices were built before their obsolescence became obvious.

By the 1950s the success of digital electronic computers had spelled the end for most analog computing machines, but analog computers remain in use in some specialized applications such as education ([control systems](/wiki/Control_systems)) and aircraft ([slide rule](/wiki/Slide_rule)).

### Digital Computers[[edit](/index.php?title=(none)&action=edit&section=6)]

#### Electromechanical[[edit](/index.php?title=(none)&action=edit&section=7)]

By 1938 the [United States Navy](/wiki/United_States_Navy) had developed an electromechanical analog computer small enough to use aboard a [submarine](/wiki/Submarine). This was the [Torpedo Data Computer](/wiki/Torpedo_Data_Computer), which used trigonometry to solve the problem of firing a torpedo at a moving target. During [World War II](/wiki/World_War_II) similar devices were developed in other countries as well.

[thumb|left|200px|Replica of](/wiki/File:Z3_Deutsches_Museum.JPG) [Zuse's](/wiki/Konrad_Zuse) [Z3](/wiki/Z3_(computer)), the first fully automatic, digital (electromechanical) computer. Early digital computers were electromechanical; electric switches drove mechanical relays to perform the calculation. These devices had a low operating speed and were eventually superseded by much faster all-electric computers, originally using [vacuum tubes](/wiki/Vacuum_tube). The [Z2](/wiki/Z2_(computer)), created by German engineer [Konrad Zuse](/wiki/Konrad_Zuse) in 1939, was one of the earliest examples of an electromechanical relay computer.[[15]](#cite_note-15) In 1941, Zuse followed his earlier machine up with the [Z3](/wiki/Z3_(computer)), the world's first working [electromechanical](/wiki/Electromechanical) [programmable](/wiki/Computer_programming), fully automatic digital computer.[[16]](#cite_note-16)[[17]](#cite_note-17) The Z3 was built with 2000 [relays](/wiki/Relay), implementing a 22 [bit](/wiki/Bit) [word length](/wiki/Word_(data_type)) that operated at a [clock frequency](/wiki/Clock_frequency) of about 5–10 [Hz](/wiki/Hertz).[[18]](#cite_note-18) Program code was supplied on punched [film](/wiki/Celluloid) while data could be stored in 64 words of memory or supplied from the keyboard. It was quite similar to modern machines in some respects, pioneering numerous advances such as [floating point numbers](/wiki/Floating_point_number). Replacement of the hard-to-implement decimal system (used in [Charles Babbage's](/wiki/Charles_Babbage) earlier design) by the simpler [binary](/wiki/Binary_numeral_system) system meant that Zuse's machines were easier to build and potentially more reliable, given the technologies available at that time.[[19]](#cite_note-19) The Z3 was [Turing complete](/wiki/Turing_complete).[[20]](#cite_note-20)[[21]](#cite_note-21)

#### {{anchor|digital computer|digital}}Vacuum tubes and digital electronic circuits[[edit](/index.php?title=(none)&action=edit&section=8)]

Purely [electronic circuit](/wiki/Electronic_circuit) elements soon replaced their mechanical and electromechanical equivalents, at the same time that digital calculation replaced analog. The engineer [Tommy Flowers](/wiki/Tommy_Flowers), working at the [Post Office Research Station](/wiki/Post_Office_Research_Station) in [London](/wiki/London) in the 1930s, began to explore the possible use of electronics for the [telephone exchange](/wiki/Telephone_exchange). Experimental equipment that he built in 1934 went into operation 5 years later, converting a portion of the [telephone exchange](/wiki/Telephone_exchange) network into an electronic data processing system, using thousands of [vacuum tubes](/wiki/Vacuum_tube).[[14]](#cite_note-14) In the US, John Vincent Atanasoff and Clifford E. Berry of Iowa State University developed and tested the [Atanasoff–Berry Computer](/wiki/Atanasoff–Berry_Computer) (ABC) in 1942,[[22]](#cite_note-22) the first "automatic electronic digital computer".[[23]](#cite_note-23) This design was also all-electronic and used about 300 vacuum tubes, with capacitors fixed in a mechanically rotating drum for memory.<ref name=Copeland2006>[Template:Citation](/wiki/Template:Citation)</ref>

[thumbnail|right|](/wiki/File:Colossus.jpg)[Colossus](/wiki/Colossus_computer) was the first [electronic](/wiki/Electronics) [digital](/wiki/Digital_electronics) [programmable](/wiki/Computer_programming) computing device, and was used to break German ciphers during World War II. During World War II, the British at [Bletchley Park](/wiki/Bletchley_Park) achieved a number of successes at breaking encrypted German military communications. The German encryption machine, [Enigma](/wiki/Enigma_(machine)), was first attacked with the help of the electro-mechanical [bombes](/wiki/Bombe). To crack the more sophisticated German [Lorenz SZ 40/42](/wiki/Lorenz_SZ_40/42) machine, used for high-level Army communications, [Max Newman](/wiki/Max_Newman) and his colleagues commissioned Flowers to build the [Colossus](/wiki/Colossus_computer).[[24]](#cite_note-24) He spent eleven months from early February 1943 designing and building the first Colossus.[[25]](#cite_note-25) After a functional test in December 1943, Colossus was shipped to Bletchley Park, where it was delivered on 18 January 1944[[26]](#cite_note-26) Built by [Ferranti](/wiki/Ferranti), it was delivered to the [University of Manchester](/wiki/University_of_Manchester) in February 1951. At least seven of these later machines were delivered between 1953 and 1957, one of them to [Shell](/wiki/Royal_Dutch_Shell) labs in [Amsterdam](/wiki/Amsterdam).[[36]](#cite_note-36) In October 1947, the directors of British catering company [J. Lyons & Company](/wiki/J._Lyons_and_Co.) decided to take an active role in promoting the commercial development of computers. The [LEO I](/wiki/LEO_computer) computer became operational in April 1951[[37]](#cite_note-37) and ran the world's first regular routine office computer [job](/wiki/Job_(software)).

#### Transistors[[edit](/index.php?title=(none)&action=edit&section=12)]

[thumb|right|A](/wiki/File:Transistor-die-KSY34.jpg) [bipolar junction transistor](/wiki/Bipolar_junction_transistor) The bipolar [transistor](/wiki/Transistor) was invented in 1947. From 1955 onwards transistors replaced [vacuum tubes](/wiki/Vacuum_tube) in computer designs, giving rise to the "second generation" of computers. Compared to vacuum tubes, transistors have many advantages: they are smaller, and require less power than vacuum tubes, so give off less heat. Silicon junction transistors were much more reliable than vacuum tubes and had longer, indefinite, service life. Transistorized computers could contain tens of thousands of binary logic circuits in a relatively compact space.

At the [University of Manchester](/wiki/University_of_Manchester), a team under the leadership of [Tom Kilburn](/wiki/Tom_Kilburn) designed and built a machine using the newly developed [transistors](/wiki/Transistor) instead of valves.[[38]](#cite_note-38) Their first [transistorised computer](/wiki/Transistor_computer) and the first in the world, was [operational by 1953](/wiki/Manchester_computers#Transistor_Computer), and a second version was completed there in April 1955. However, the machine did make use of valves to generate its 125 kHz clock waveforms and in the circuitry to read and write on its magnetic [drum memory](/wiki/Drum_memory), so it was not the first completely transistorized computer. That distinction goes to the [Harwell CADET](/wiki/Harwell_CADET) of 1955,[[39]](#cite_note-39) built by the electronics division of the [Atomic Energy Research Establishment](/wiki/Atomic_Energy_Research_Establishment) at [Harwell](/wiki/Harwell).[[39]](#cite_note-39)[[40]](#cite_note-40)

#### Integrated circuits[[edit](/index.php?title=(none)&action=edit&section=13)]

The next great advance in computing power came with the advent of the [integrated circuit](/wiki/Integrated_circuit). The idea of the integrated circuit was first conceived by a radar scientist working for the [Royal Radar Establishment](/wiki/Royal_Radar_Establishment) of the [Ministry of Defence](/wiki/Ministry_of_Defence_(United_Kingdom)), [Geoffrey W.A. Dummer](/wiki/Geoffrey_Dummer). Dummer presented the first public description of an integrated circuit at the Symposium on Progress in Quality Electronic Components in [Washington, D.C.](/wiki/Washington,_D.C.) on 7 May 1952.[[41]](#cite_note-41) The first practical ICs were invented by [Jack Kilby](/wiki/Jack_Kilby) at [Texas Instruments](/wiki/Texas_Instruments) and [Robert Noyce](/wiki/Robert_Noyce) at [Fairchild Semiconductor](/wiki/Fairchild_Semiconductor).[[42]](#cite_note-42) Kilby recorded his initial ideas concerning the integrated circuit in July 1958, successfully demonstrating the first working integrated example on 12 September 1958.[[43]](#cite_note-43) In his patent application of 6 February 1959, Kilby described his new device as "a body of semiconductor material ... wherein all the components of the electronic circuit are completely integrated".[[44]](#cite_note-44)[[45]](#cite_note-45) Noyce also came up with his own idea of an integrated circuit half a year later than Kilby.[[46]](#cite_note-46) His chip solved many practical problems that Kilby's had not. Produced at Fairchild Semiconductor, it was made of [silicon](/wiki/Silicon), whereas Kilby's chip was made of [germanium](/wiki/Germanium).

This new development heralded an explosion in the commercial and personal use of computers and led to the invention of the [microprocessor](/wiki/Microprocessor). While the subject of exactly which device was the first microprocessor is contentious, partly due to lack of agreement on the exact definition of the term "microprocessor", it is largely undisputed that the first single-chip microprocessor was the Intel 4004,[[47]](#cite_note-47) designed and realized by [Ted Hoff](/wiki/Marcian_Hoff), [Federico Faggin](/wiki/Federico_Faggin), and Stanley Mazor at [Intel](/wiki/Intel).[[48]](#cite_note-48)

### Mobile computers become dominant[[edit](/index.php?title=(none)&action=edit&section=14)]

With the continued miniaturization of computing resources, and advancements in portable battery life, [portable computers](/wiki/Portable_computer) grew in popularity in the 2000s.[[49]](#cite_note-49) The same developments that spurred the growth of laptop computers and other portable computers allowed manufacturers to integrate computing resources into cellular phones. These so-called [smartphones](/wiki/Smartphone) and [tablets](/wiki/Tablet_computer) run on a variety of operating systems and have become the dominant computing device on the market, with manufacturers reporting having shipped an estimated 237 million devices in 2Q 2013.[[50]](#cite_note-50)

## Programs[[edit](/index.php?title=(none)&action=edit&section=15)]

The defining feature of modern computers which distinguishes them from all other machines is that they can be [programmed](/wiki/Computer_programming). That is to say that some type of [instructions](/wiki/Instruction_(computer_science)) (the [program](/wiki/Computer_program)) can be given to the computer, and it will process them. Modern computers based on the [von Neumann architecture](/wiki/Von_Neumann_architecture) often have machine code in the form of an [imperative programming language](/wiki/Imperative_programming_language).

In practical terms, a computer program may be just a few instructions or extend to many millions of instructions, as do the programs for [word processors](/wiki/Word_processor) and [web browsers](/wiki/Web_browser) for example. A typical modern computer can execute billions of instructions per second ([gigaflops](/wiki/FLOPS)) and rarely makes a mistake over many years of operation. Large computer programs consisting of several million instructions may take teams of [programmers](/wiki/Programmer) years to write, and due to the complexity of the task almost certainly contain errors.

### Stored program architecture[[edit](/index.php?title=(none)&action=edit&section=16)]

[Template:Main](/wiki/Template:Main) [thumb|right|Replica of the Small-Scale Experimental Machine (SSEM), the world's first](/wiki/File:SSEM_Manchester_museum.jpg) [stored-program computer](/wiki/Stored-program_computer), at the [Museum of Science and Industry](/wiki/Museum_of_Science_and_Industry_(Manchester)) in Manchester, England This section applies to most common [RAM machine](/wiki/RAM_machine)-based computers.

In most cases, computer instructions are simple: add one number to another, move some data from one location to another, send a message to some external device, etc. These instructions are read from the computer's [memory](/wiki/Computer_data_storage) and are generally carried out ([executed](/wiki/Execution_(computing))) in the order they were given. However, there are usually specialized instructions to tell the computer to jump ahead or backwards to some other place in the program and to carry on executing from there. These are called "jump" instructions (or [branches](/wiki/Branch_(computer_science))). Furthermore, jump instructions may be made to happen [conditionally](/wiki/Conditional_(programming)) so that different sequences of instructions may be used depending on the result of some previous calculation or some external event. Many computers directly support [subroutines](/wiki/Subroutine) by providing a type of jump that "remembers" the location it jumped from and another instruction to return to the instruction following that jump instruction.

Program execution might be likened to reading a book. While a person will normally read each word and line in sequence, they may at times jump back to an earlier place in the text or skip sections that are not of interest. Similarly, a computer may sometimes go back and repeat the instructions in some section of the program over and over again until some internal condition is met. This is called the [flow of control](/wiki/Control_flow) within the program and it is what allows the computer to perform tasks repeatedly without human intervention.

Comparatively, a person using a pocket [calculator](/wiki/Calculator) can perform a basic arithmetic operation such as adding two numbers with just a few button presses. But to add together all of the numbers from 1 to 1,000 would take thousands of button presses and a lot of time, with a near certainty of making a mistake. On the other hand, a computer may be programmed to do this with just a few simple instructions. The following example is written in the [MIPS assembly language](/wiki/MIPS_instruction_set): [Template:Clear](/wiki/Template:Clear) <source lang="asm">

begin:

addi $8, $0, 0 # initialize sum to 0

addi $9, $0, 1 # set first number to add = 1

loop:

slti $10, $9, 1000 # check if the number is less than 1000

beq $10, $0, finish # if odd number is greater than n then exit

add $8, $8, $9 # update sum

addi $9, $9, 1 # get next number

j loop # repeat the summing process

finish:

add $2, $8, $0 # put sum in output register

</source>

Once told to run this program, the computer will perform the repetitive addition task without further human intervention. It will almost never make a mistake and a modern PC can complete the task in a fraction of a second.

### Machine code[[edit](/index.php?title=(none)&action=edit&section=17)]

In most computers, individual instructions are stored as [machine code](/wiki/Machine_code) with each instruction being given a unique number (its operation code or [opcode](/wiki/Opcode) for short). The command to add two numbers together would have one opcode; the command to multiply them would have a different opcode, and so on. The simplest computers are able to perform any of a handful of different instructions; the more complex computers have several hundred to choose from, each with a unique numerical code. Since the computer's memory is able to store numbers, it can also store the instruction codes. This leads to the important fact that entire programs (which are just lists of these instructions) can be represented as lists of numbers and can themselves be manipulated inside the computer in the same way as numeric data. The fundamental concept of storing programs in the computer's memory alongside the data they operate on is the crux of the von Neumann, or stored program[Template:Citation needed](/wiki/Template:Citation_needed), architecture. In some cases, a computer might store some or all of its program in memory that is kept separate from the data it operates on. This is called the [Harvard architecture](/wiki/Harvard_architecture) after the [Harvard Mark I](/wiki/Harvard_Mark_I) computer. Modern von Neumann computers display some traits of the Harvard architecture in their designs, such as in [CPU caches](/wiki/CPU_cache).

While it is possible to write computer programs as long lists of numbers ([machine language](/wiki/Machine_code)) and while this technique was used with many early computers,[[51]](#cite_note-51) it is extremely tedious and potentially error-prone to do so in practice, especially for complicated programs. Instead, each basic instruction can be given a short name that is indicative of its function and easy to remember – a [mnemonic](/wiki/Mnemonic) such as ADD, SUB, MULT or JUMP. These mnemonics are collectively known as a computer's [assembly language](/wiki/Assembly_language). Converting programs written in assembly language into something the computer can actually understand (machine language) is usually done by a computer program called an assembler.

[thumb|right|A 1970s](/wiki/File:FortranCardPROJ039.agr.jpg) [punched card](/wiki/Punched_card) containing one line from a [FORTRAN](/wiki/Fortran) program. The card reads: "Z(1) = Y + W(1)" and is labeled "PROJ039" for identification purposes.

### Programming language[[edit](/index.php?title=(none)&action=edit&section=18)]

[Template:Main](/wiki/Template:Main) Programming languages provide various ways of specifying programs for computers to run. Unlike [natural languages](/wiki/Natural_language), programming languages are designed to permit no ambiguity and to be concise. They are purely written languages and are often difficult to read aloud. They are generally either translated into [machine code](/wiki/Machine_code) by a [compiler](/wiki/Compiler) or an [assembler](/wiki/Assembler_(computer_programming)) before being run, or translated directly at run time by an [interpreter](/wiki/Interpreter_(computing)). Sometimes programs are executed by a hybrid method of the two techniques.

#### Low-level languages[[edit](/index.php?title=(none)&action=edit&section=19)]

[Template:Main](/wiki/Template:Main) Machine languages and the assembly languages that represent them (collectively termed *low-level programming languages*) tend to be unique to a particular type of computer. For instance, an [ARM architecture](/wiki/ARM_architecture) computer (such as may be found in a [PDA](/wiki/Personal_digital_assistant) or a [hand-held videogame](/wiki/Handheld_video_game)) cannot understand the machine language of an [Intel Pentium](/wiki/Pentium) or the [AMD Athlon 64](/wiki/Athlon_64) computer that might be in a [PC](/wiki/Personal_computer).[[52]](#cite_note-52)

#### High-level languages/Third Generation Language[[edit](/index.php?title=(none)&action=edit&section=20)]

[Template:Main](/wiki/Template:Main) Though considerably easier than in machine language, writing long programs in assembly language is often difficult and is also error prone. Therefore, most practical programs are written in more abstract [high-level programming languages](/wiki/High-level_programming_language) that are able to express the needs of the [programmer](/wiki/Programmer) more conveniently (and thereby help reduce programmer error). High level languages are usually "compiled" into machine language (or sometimes into assembly language and then into machine language) using another computer program called a [compiler](/wiki/Compiler).[[53]](#cite_note-53) High level languages are less related to the workings of the target computer than assembly language, and more related to the language and structure of the problem(s) to be solved by the final program. It is therefore often possible to use different compilers to translate the same high level language program into the machine language of many different types of computer. This is part of the means by which software like video games may be made available for different computer architectures such as personal computers and various [video game consoles](/wiki/Video_game_console).

### Fourth Generation Languages[[edit](/index.php?title=(none)&action=edit&section=21)]

These 4G languages are less procedural than 3G languages. The benefit of 4GL is that it provides ways to obtain information without requiring the direct help of a programmer. Example of 4GL is SQL.

### Program design[[edit](/index.php?title=(none)&action=edit&section=22)]

[Template:Unreferenced section](/wiki/Template:Unreferenced_section) Program design of small programs is relatively simple and involves the analysis of the problem, collection of inputs, using the programming constructs within languages, devising or using established procedures and algorithms, providing data for output devices and solutions to the problem as applicable. As problems become larger and more complex, features such as subprograms, modules, formal documentation, and new paradigms such as object-oriented programming are encountered. Large programs involving thousands of line of code and more require formal software methodologies. The task of developing large [software](/wiki/Computer_software) systems presents a significant intellectual challenge. Producing software with an acceptably high reliability within a predictable schedule and budget has historically been difficult; the academic and professional discipline of [software engineering](/wiki/Software_engineering) concentrates specifically on this challenge.

### Bugs[[edit](/index.php?title=(none)&action=edit&section=23)]

[Template:Main](/wiki/Template:Main) [thumb|The actual first computer bug, a moth found trapped on a relay of the Harvard Mark II computer](/wiki/File:H96566k.jpg) Errors in computer programs are called "[bugs](/wiki/Software_bug)". They may be benign and not affect the usefulness of the program, or have only subtle effects. But in some cases, they may cause the program or the entire system to "[hang](/wiki/Hang_(computing))", becoming unresponsive to input such as [mouse](/wiki/Mouse_(computing)) clicks or keystrokes, to completely fail, or to [crash](/wiki/Crash_(computing)). Otherwise benign bugs may sometimes be harnessed for malicious intent by an unscrupulous user writing an [exploit](/wiki/Exploit_(computer_security)), code designed to take advantage of a bug and disrupt a computer's proper execution. Bugs are usually not the fault of the computer. Since computers merely execute the instructions they are given, bugs are nearly always the result of programmer error or an oversight made in the program's design.[[54]](#cite_note-54) Admiral [Grace Hopper](/wiki/Grace_Hopper), an American computer scientist and developer of the first [compiler](/wiki/Compiler), is credited for having first used the term "bugs" in computing after a dead moth was found shorting a relay in the [Harvard Mark II](/wiki/Harvard_Mark_II) computer in September 1947.[[55]](#cite_note-55)

## Components[[edit](/index.php?title=(none)&action=edit&section=24)]

[Template:Main](/wiki/Template:Main)[thumb|Video demonstrating the standard components of a "slimline" computer](/wiki/File:Computer_Components.webm) A general purpose computer has four main components: the [arithmetic logic unit](/wiki/Arithmetic_logic_unit) (ALU), the [control unit](/wiki/Control_unit), the [memory](/wiki/Computer_data_storage), and the input and output devices (collectively termed I/O). These parts are interconnected by [buses](/wiki/Bus_(computing)), often made of groups of [wires](/wiki/Wire).

Inside each of these parts are thousands to trillions of small [electrical circuits](/wiki/Electrical_network) which can be turned off or on by means of an [electronic switch](/wiki/Transistor). Each circuit represents a [bit](/wiki/Bit) (binary digit) of information so that when the circuit is on it represents a "1", and when off it represents a "0" (in positive logic representation). The circuits are arranged in [logic gates](/wiki/Logic_gate) so that one or more of the circuits may control the state of one or more of the other circuits.

### Control unit[[edit](/index.php?title=(none)&action=edit&section=25)]

[Template:Main](/wiki/Template:Main) [thumb|right|Diagram showing how a particular](/wiki/File:Mips32_addi.svg) [MIPS architecture](/wiki/MIPS_architecture) instruction would be decoded by the control system

The control unit (often called a control system or central controller) manages the computer's various components; it reads and interprets (decodes) the program instructions, transforming them into control signals that activate other parts of the computer.[[56]](#cite_note-56) Control systems in advanced computers may change the order of execution of some instructions to improve performance.

A key component common to all CPUs is the [program counter](/wiki/Program_counter), a special memory cell (a [register](/wiki/Processor_register)) that keeps track of which location in memory the next instruction is to be read from.[[57]](#cite_note-57) The control system's function is as follows—note that this is a simplified description, and some of these steps may be performed concurrently or in a different order depending on the type of CPU:

1. Read the code for the next instruction from the cell indicated by the program counter.
2. Decode the numerical code for the instruction into a set of commands or signals for each of the other systems.
3. Increment the program counter so it points to the next instruction.
4. Read whatever data the instruction requires from cells in memory (or perhaps from an input device). The location of this required data is typically stored within the instruction code.
5. Provide the necessary data to an ALU or register.
6. If the instruction requires an ALU or specialized hardware to complete, instruct the hardware to perform the requested operation.
7. Write the result from the ALU back to a memory location or to a register or perhaps an output device.
8. Jump back to step (1).

Since the program counter is (conceptually) just another set of memory cells, it can be changed by calculations done in the ALU. Adding 100 to the program counter would cause the next instruction to be read from a place 100 locations further down the program. Instructions that modify the program counter are often known as "jumps" and allow for loops (instructions that are repeated by the computer) and often conditional instruction execution (both examples of [control flow](/wiki/Control_flow)).

The sequence of operations that the control unit goes through to process an instruction is in itself like a short computer program, and indeed, in some more complex CPU designs, there is another yet smaller computer called a [microsequencer](/wiki/Microsequencer), which runs a [microcode](/wiki/Microcode) program that causes all of these events to happen.

### Central processing unit (CPU)[[edit](/index.php?title=(none)&action=edit&section=26)]

The control unit, ALU, and registers are collectively known as a [central processing unit](/wiki/Central_processing_unit) (CPU). Early CPUs were composed of many separate components but since the mid-1970s CPUs have typically been constructed on a single [integrated circuit](/wiki/Integrated_circuit) called a [*microprocessor*](/wiki/Microprocessor).

### Arithmetic logic unit (ALU)[[edit](/index.php?title=(none)&action=edit&section=27)]

[Template:Main](/wiki/Template:Main)

The ALU is capable of performing two classes of operations: arithmetic and logic.[[58]](#cite_note-58) The set of arithmetic operations that a particular ALU supports may be limited to addition and subtraction, or might include multiplication, division, [trigonometry](/wiki/Trigonometry) functions such as sine, cosine, etc., and [square roots](/wiki/Square_root). Some can only operate on whole numbers ([integers](/wiki/Integer)) whilst others use [floating point](/wiki/Floating_point) to represent [real numbers](/wiki/Real_number), albeit with limited precision. However, any computer that is capable of performing just the simplest operations can be programmed to break down the more complex operations into simple steps that it can perform. Therefore, any computer can be programmed to perform any arithmetic operation—although it will take more time to do so if its ALU does not directly support the operation. An ALU may also compare numbers and return [boolean truth values](/wiki/Truth_value) (true or false) depending on whether one is equal to, greater than or less than the other ("is 64 greater than 65?").

Logic operations involve [Boolean logic](/wiki/Boolean_logic): [AND](/wiki/Logical_conjunction), [OR](/wiki/Logical_disjunction), [XOR](/wiki/Exclusive_or), and [NOT](/wiki/Negation). These can be useful for creating complicated [conditional statements](/wiki/Conditional_(programming)) and processing [boolean logic](/wiki/Boolean_logic).

[Superscalar](/wiki/Superscalar) computers may contain multiple ALUs, allowing them to process several instructions simultaneously.[[59]](#cite_note-59) [Graphics processors](/wiki/Graphics_processing_unit) and computers with [SIMD](/wiki/SIMD) and [MIMD](/wiki/MIMD) features often contain ALUs that can perform arithmetic on [vectors](/wiki/Euclidean_vector) and [matrices](/wiki/Matrix_(mathematics)).

### Memory[[edit](/index.php?title=(none)&action=edit&section=28)]

[Template:Main](/wiki/Template:Main) [thumb|right|](/wiki/File:Magnetic_core.jpg)[Magnetic core memory](/wiki/Magnetic_core_memory) was the computer memory of choice throughout the 1960s, until it was replaced by semiconductor memory.

A computer's memory can be viewed as a list of cells into which numbers can be placed or read. Each cell has a numbered "address" and can store a single number. The computer can be instructed to "put the number 123 into the cell numbered 1357" or to "add the number that is in cell 1357 to the number that is in cell 2468 and put the answer into cell 1595." The information stored in memory may represent practically anything. Letters, numbers, even computer instructions can be placed into memory with equal ease. Since the CPU does not differentiate between different types of information, it is the software's responsibility to give significance to what the memory sees as nothing but a series of numbers.

In almost all modern computers, each memory cell is set up to store [binary numbers](/wiki/Binary_numeral_system) in groups of eight bits (called a [byte](/wiki/Byte)). Each byte is able to represent 256 different numbers (28 = 256); either from 0 to 255 or −128 to +127. To store larger numbers, several consecutive bytes may be used (typically, two, four or eight). When negative numbers are required, they are usually stored in [two's complement](/wiki/Two's_complement) notation. Other arrangements are possible, but are usually not seen outside of specialized applications or historical contexts. A computer can store any kind of information in memory if it can be represented numerically. Modern computers have billions or even trillions of bytes of memory.

The CPU contains a special set of memory cells called [registers](/wiki/Processor_register) that can be read and written to much more rapidly than the main memory area. There are typically between two and one hundred registers depending on the type of CPU. Registers are used for the most frequently needed data items to avoid having to access main memory every time data is needed. As data is constantly being worked on, reducing the need to access main memory (which is often slow compared to the ALU and control units) greatly increases the computer's speed.

Computer main memory comes in two principal varieties:

* [random-access memory](/wiki/Random-access_memory) or RAM
* [read-only memory](/wiki/Read-only_memory) or ROM

RAM can be read and written to anytime the CPU commands it, but ROM is preloaded with data and software that never changes, therefore the CPU can only read from it. ROM is typically used to store the computer's initial start-up instructions. In general, the contents of RAM are erased when the power to the computer is turned off, but ROM retains its data indefinitely. In a PC, the ROM contains a specialized program called the [BIOS](/wiki/BIOS) that orchestrates loading the computer's [operating system](/wiki/Operating_system) from the hard disk drive into RAM whenever the computer is turned on or reset. In [embedded computers](/wiki/Embedded_system), which frequently do not have disk drives, all of the required software may be stored in ROM. Software stored in ROM is often called [firmware](/wiki/Firmware), because it is notionally more like hardware than software. [Flash memory](/wiki/Flash_memory) blurs the distinction between ROM and RAM, as it retains its data when turned off but is also rewritable. It is typically much slower than conventional ROM and RAM however, so its use is restricted to applications where high speed is unnecessary.[[60]](#cite_note-60) In more sophisticated computers there may be one or more RAM [cache memories](/wiki/CPU_cache), which are slower than registers but faster than main memory. Generally computers with this sort of cache are designed to move frequently needed data into the cache automatically, often without the need for any intervention on the programmer's part.

### Input/output (I/O)[[edit](/index.php?title=(none)&action=edit&section=29)]

[Template:Main](/wiki/Template:Main) [thumb|right|](/wiki/File:HDDspin.JPG)[Hard disk drives](/wiki/Hard_disk_drive) are common storage devices used with computers. I/O is the means by which a computer exchanges information with the outside world.[[61]](#cite_note-61) Devices that provide input or output to the computer are called [peripherals](/wiki/Peripheral).[[62]](#cite_note-62) On a typical personal computer, peripherals include input devices like the keyboard and [mouse](/wiki/Mouse_(computing)), and output devices such as the [display](/wiki/Computer_monitor) and [printer](/wiki/Printer_(computing)). [Hard disk drives](/wiki/Hard_disk_drive), [floppy disk drives](/wiki/Floppy_disk) and [optical disc drives](/wiki/Optical_disc_drive) serve as both input and output devices. [Computer networking](/wiki/Computer_networking) is another form of I/O.

I/O devices are often complex computers in their own right, with their own CPU and memory. A [graphics processing unit](/wiki/Graphics_processing_unit) might contain fifty or more tiny computers that perform the calculations necessary to display [3D graphics](/wiki/3D_computer_graphics).[Template:Citation needed](/wiki/Template:Citation_needed) Modern [desktop computers](/wiki/Desktop_computer) contain many smaller computers that assist the main CPU in performing I/O.

### Multitasking[[edit](/index.php?title=(none)&action=edit&section=30)]

[Template:Main](/wiki/Template:Main) While a computer may be viewed as running one gigantic program stored in its main memory, in some systems it is necessary to give the appearance of running several programs simultaneously. This is achieved by multitasking i.e. having the computer switch rapidly between running each program in turn.[[63]](#cite_note-63) One means by which this is done is with a special signal called an [interrupt](/wiki/Interrupt), which can periodically cause the computer to stop executing instructions where it was and do something else instead. By remembering where it was executing prior to the interrupt, the computer can return to that task later. If several programs are running "at the same time". then the interrupt generator might be causing several hundred interrupts per second, causing a program switch each time. Since modern computers typically execute instructions several orders of magnitude faster than human perception, it may appear that many programs are running at the same time even though only one is ever executing in any given instant. This method of multitasking is sometimes termed "time-sharing" since each program is allocated a "slice" of time in turn.[[64]](#cite_note-64) Before the era of cheap computers, the principal use for multitasking was to allow many people to share the same computer.

Seemingly, multitasking would cause a computer that is switching between several programs to run more slowly, in direct proportion to the number of programs it is running, but most programs spend much of their time waiting for slow input/output devices to complete their tasks. If a program is waiting for the user to click on the mouse or press a key on the keyboard, then it will not take a "time slice" until the event it is waiting for has occurred. This frees up time for other programs to execute so that many programs may be run simultaneously without unacceptable speed loss.

### Multiprocessing[[edit](/index.php?title=(none)&action=edit&section=31)]

[Template:Main](/wiki/Template:Main) [thumb|](/wiki/File:Cray_2_Arts_et_Metiers_dsc03940.jpg)[Cray](/wiki/Cray) designed many supercomputers that used multiprocessing heavily. Some computers are designed to distribute their work across several CPUs in a multiprocessing configuration, a technique once employed only in large and powerful machines such as [supercomputers](/wiki/Supercomputer), [mainframe computers](/wiki/Mainframe_computer) and [servers](/wiki/Server_(computing)). Multiprocessor and [multi-core](/wiki/Multi-core) (multiple CPUs on a single integrated circuit) personal and laptop computers are now widely available, and are being increasingly used in lower-end markets as a result.

Supercomputers in particular often have highly unique architectures that differ significantly from the basic stored-program architecture and from general purpose computers.[[65]](#cite_note-65) They often feature thousands of CPUs, customized high-speed interconnects, and specialized computing hardware. Such designs tend to be useful only for specialized tasks due to the large scale of program organization required to successfully utilize most of the available resources at once. Supercomputers usually see usage in large-scale [simulation](/wiki/Computer_simulation), [graphics rendering](/wiki/Rendering_(computer_graphics)), and [cryptography](/wiki/Cryptography) applications, as well as with other so-called "[embarrassingly parallel](/wiki/Embarrassingly_parallel)" tasks.

## Networking and the Internet[[edit](/index.php?title=(none)&action=edit&section=32)]

[Template:Main](/wiki/Template:Main) [thumb|left|Visualization of a portion of the](/wiki/File:Internet_map_1024.jpg) [routes](/wiki/Routing) on the Internet Computers have been used to coordinate information between multiple locations since the 1950s. The U.S. military's [SAGE](/wiki/Semi_Automatic_Ground_Environment) system was the first large-scale example of such a system, which led to a number of special-purpose commercial systems such as [Sabre](/wiki/Sabre_(computer_system)).[[66]](#cite_note-66) In the 1970s, computer engineers at research institutions throughout the United States began to link their computers together using telecommunications technology. The effort was funded by ARPA (now [DARPA](/wiki/DARPA)), and the [computer network](/wiki/Computer_network) that resulted was called the [ARPANET](/wiki/ARPANET).[[67]](#cite_note-67) The technologies that made the Arpanet possible spread and evolved.

In time, the network spread beyond academic and military institutions and became known as the Internet. The emergence of networking involved a redefinition of the nature and boundaries of the computer. Computer operating systems and applications were modified to include the ability to define and access the resources of other computers on the network, such as peripheral devices, stored information, and the like, as extensions of the resources of an individual computer. Initially these facilities were available primarily to people working in high-tech environments, but in the 1990s the spread of applications like e-mail and the [World Wide Web](/wiki/World_Wide_Web), combined with the development of cheap, fast networking technologies like [Ethernet](/wiki/Ethernet) and [ADSL](/wiki/Asymmetric_digital_subscriber_line) saw computer networking become almost ubiquitous. In fact, the number of computers that are networked is growing phenomenally. A very large proportion of personal computers regularly connect to the Internet to communicate and receive information. "Wireless" networking, often utilizing mobile phone networks, has meant networking is becoming increasingly ubiquitous even in mobile computing environments. [Template:Clear](/wiki/Template:Clear)

### Computer architecture paradigms[[edit](/index.php?title=(none)&action=edit&section=33)]

There are many types of [computer architectures](/wiki/Computer_architecture):

* [Quantum computer](/wiki/Quantum_computer) vs. [Chemical computer](/wiki/Chemical_computer)
* [Scalar processor](/wiki/Scalar_processor) vs. [Vector processor](/wiki/Vector_processor)
* [Non-Uniform Memory Access](/wiki/Non-Uniform_Memory_Access) (NUMA) computers
* [Register machine](/wiki/Register_machine) vs. [Stack machine](/wiki/Stack_machine)
* [Harvard architecture](/wiki/Harvard_architecture) vs. [von Neumann architecture](/wiki/Von_Neumann_architecture)
* [Cellular architecture](/wiki/Cellular_architecture)

Of all these [abstract machines](/wiki/Abstract_machine), a quantum computer holds the most promise for revolutionizing computing.[[68]](#cite_note-68) [Logic gates](/wiki/Logic_gates) are a common abstraction which can apply to most of the above [digital](/wiki/Digital_data) or [analog](/wiki/Analog_signal) paradigms.

The ability to store and execute lists of instructions called [programs](/wiki/Computer_program) makes computers extremely versatile, distinguishing them from [calculators](/wiki/Calculator). The [Church–Turing thesis](/wiki/Church–Turing_thesis) is a mathematical statement of this versatility: any computer with a [minimum capability (being Turing-complete)](/wiki/Turing-complete) is, in principle, capable of performing the same tasks that any other computer can perform. Therefore, any type of computer ([netbook](/wiki/Netbook), [supercomputer](/wiki/Supercomputer), [cellular automaton](/wiki/Cellular_automaton), etc.) is able to perform the same computational tasks, given enough time and storage capacity.

## Misconceptions[[edit](/index.php?title=(none)&action=edit&section=34)]

[Template:Main](/wiki/Template:Main) [thumb|Women as *computers* in NACA High Speed Flight Station "Computer Room"](/wiki/File:Human_computers_-_Dryden.jpg) A computer does not need to be [electronic](/wiki/Electronics), nor even have a [processor](/wiki/Central_processing_unit), nor [RAM](/wiki/Random-access_memory), nor even a [hard disk](/wiki/Hard_disk). While popular usage of the word "computer" is synonymous with a personal electronic computer, the modern[[69]](#cite_note-69) definition of a computer is literally: "*A device that computes*, especially a programmable [usually] electronic machine that performs high-speed mathematical or logical operations or that assembles, stores, correlates, or otherwise processes information."[[70]](#cite_note-70) Any device which *processes information* qualifies as a computer, especially if the processing is purposeful.[Template:Citation needed](/wiki/Template:Citation_needed)

### Unconventional computing[[edit](/index.php?title=(none)&action=edit&section=35)]

[Template:Main](/wiki/Template:Main) Historically, computers evolved from [mechanical computers](/wiki/Mechanical_computer) and eventually from [vacuum tubes](/wiki/Vacuum_tube) to [transistors](/wiki/Transistor). However, conceptually computational systems as [flexible](/wiki/Turing_complete) as a personal computer can be built out of almost anything. For example, a computer can be made out of billiard balls ([billiard ball computer](/wiki/Billiard_ball_computer)); an often quoted example.[Template:Citation needed](/wiki/Template:Citation_needed) More realistically, modern computers are made out of [transistors](/wiki/Transistors) made of [photolithographed](/wiki/Photolithography) [semiconductors](/wiki/Semiconductors).

## Future[[edit](/index.php?title=(none)&action=edit&section=36)]

There is active research to make computers out of many promising new types of technology, such as [optical computers](/wiki/Optical_computing), [DNA computers](/wiki/DNA_computing), [neural computers](/wiki/Wetware_computer), and [quantum computers](/wiki/Quantum_computing). Most computers are universal, and are able to calculate any [computable function](/wiki/Computable_function), and are limited only by their memory capacity and operating speed. However different designs of computers can give very different performance for particular problems; for example quantum computers can potentially break some modern encryption algorithms (by [quantum factoring](/wiki/Shor's_algorithm)) very quickly.

## Further topics[[edit](/index.php?title=(none)&action=edit&section=37)]

* [Glossary of computers](/wiki/Glossary_of_computers)

### Artificial intelligence[[edit](/index.php?title=(none)&action=edit&section=38)]

A computer will solve problems in exactly the way it is programmed to, without regard to efficiency, alternative solutions, possible shortcuts, or possible errors in the code. Computer programs that learn and adapt are part of the emerging field of [artificial intelligence](/wiki/Artificial_intelligence) and [machine learning](/wiki/Machine_learning).

## Hardware[[edit](/index.php?title=(none)&action=edit&section=39)]

[Template:Main](/wiki/Template:Main) The term *hardware* covers all of those parts of a computer that are tangible objects. Circuits, displays, power supplies, cables, keyboards, printers and mice are all hardware.

### History of computing hardware[[edit](/index.php?title=(none)&action=edit&section=40)]

[Template:Main](/wiki/Template:Main)

|  |  |  |
| --- | --- | --- |
| First generation (mechanical/electromechanical) | Calculators | [Pascal's calculator](/wiki/Pascal's_calculator), [Arithmometer](/wiki/Arithmometer), [Difference engine](/wiki/Difference_engine), [Quevedo's analytical machines](/wiki/Leonardo_Torres_y_Quevedo#Analytical_machines) |
| Programmable devices | [Jacquard loom](/wiki/Jacquard_loom), [Analytical engine](/wiki/Analytical_engine), [IBM ASCC/Harvard Mark I](/wiki/Harvard_Mark_I), [Harvard Mark II](/wiki/Harvard_Mark_II), [IBM SSEC](/wiki/IBM_SSEC), [Z1](/wiki/Z1_(computer)), [Z2](/wiki/Z2_(computer)), [Z3](/wiki/Z3_(computer)) |
| Second generation (vacuum tubes) | Calculators | [Atanasoff–Berry Computer](/wiki/Atanasoff–Berry_Computer), [IBM 604](/wiki/IBM_604), [UNIVAC 60](/wiki/Remington_Rand_409), [UNIVAC 120](/wiki/Remington_Rand_409) |
| [Programmable devices](/wiki/List_of_vacuum_tube_computers) | [Colossus](/wiki/Colossus_computer), [ENIAC](/wiki/ENIAC), [Manchester Small-Scale Experimental Machine](/wiki/Manchester_Small-Scale_Experimental_Machine), [EDSAC](/wiki/Electronic_Delay_Storage_Automatic_Calculator), [Manchester Mark 1](/wiki/Manchester_Mark_1), [Ferranti Pegasus](/wiki/Ferranti_Pegasus), [Ferranti Mercury](/wiki/Ferranti_Mercury), [CSIRAC](/wiki/CSIRAC), [EDVAC](/wiki/EDVAC), [UNIVAC I](/wiki/UNIVAC_I), [IBM 701](/wiki/IBM_701), [IBM 702](/wiki/IBM_702), [IBM 650](/wiki/IBM_650), [Z22](/wiki/Z22_(computer)) |
| Third generation (discrete transistors and SSI, MSI, LSI [integrated circuits](/wiki/Integrated_circuit)) | [Mainframes](/wiki/Mainframe_computer) | [IBM 7090](/wiki/IBM_7090), [IBM 7080](/wiki/IBM_7080), [IBM System/360](/wiki/IBM_System/360), [BUNCH](/wiki/BUNCH) |
| [Minicomputer](/wiki/Minicomputer) | [HP 2116A](/wiki/HP_2100), [IBM System/32](/wiki/IBM_System/32), [IBM System/36](/wiki/IBM_System/36), [LINC](/wiki/LINC), [PDP-8](/wiki/PDP-8), [PDP-11](/wiki/PDP-11) |
| Fourth generation (VLSI integrated circuits) | Minicomputer | [VAX](/wiki/VAX), [IBM System i](/wiki/IBM_System_i) |
| [4-bit](/wiki/4-bit) microcomputer | [Intel 4004](/wiki/Intel_4004), [Intel 4040](/wiki/Intel_4040) |
| [8-bit](/wiki/8-bit) microcomputer | [Intel 8008](/wiki/Intel_8008), [Intel 8080](/wiki/Intel_8080), [Motorola 6800](/wiki/Motorola_6800), [Motorola 6809](/wiki/Motorola_6809), [MOS Technology 6502](/wiki/MOS_Technology_6502), [Zilog Z80](/wiki/Zilog_Z80) |
| [16-bit](/wiki/16-bit) microcomputer | [Intel 8088](/wiki/Intel_8088), [Zilog Z8000](/wiki/Zilog_Z8000), [WDC 65816/65802](/wiki/WDC_65816/65802) |
| [32-bit](/wiki/32-bit) microcomputer | [Intel 80386](/wiki/Intel_80386), [Pentium](/wiki/Pentium), [Motorola 68000](/wiki/Motorola_68000), [ARM](/wiki/ARMv7) |
| [64-bit](/wiki/64-bit) microcomputer[[71]](#cite_note-71) | [Alpha](/wiki/DEC_Alpha), [MIPS](/wiki/MIPS_architecture), [PA-RISC](/wiki/PA-RISC), [PowerPC](/wiki/PowerPC), [SPARC](/wiki/SPARC), [x86-64](/wiki/X86-64), [ARMv8-A](/wiki/ARMv8-A) |
| [Embedded computer](/wiki/Embedded_system) | [Intel 8048](/wiki/Intel_8048), [Intel 8051](/wiki/Intel_8051) |
| Personal computer | [Desktop computer](/wiki/Desktop_computer), [Home computer](/wiki/Home_computer), [Laptop computer](/wiki/Laptop), [Personal digital assistant](/wiki/Personal_digital_assistant) (PDA), [Portable computer](/wiki/Portable_computer), [Tablet PC](/wiki/Tablet_computer), [Wearable computer](/wiki/Wearable_computer) |
| Theoretical/experimental | [Quantum computer](/wiki/Quantum_computer), [Chemical computer](/wiki/Chemical_computer), [DNA computing](/wiki/DNA_computing), [Optical computer](/wiki/Photonic_computing), [Spintronics based computer](/wiki/Spintronics) |  |

### Other hardware topics[[edit](/index.php?title=(none)&action=edit&section=41)]

|  |  |  |
| --- | --- | --- |
| [Peripheral device](/wiki/Peripheral) ([input/output](/wiki/Input/output)) | Input | [Mouse](/wiki/Mouse_(computing)), [keyboard](/wiki/Keyboard_(computing)), [joystick](/wiki/Joystick), [image scanner](/wiki/Image_scanner), [webcam](/wiki/Webcam), [graphics tablet](/wiki/Graphics_tablet), [microphone](/wiki/Microphone) |
| Output | [Monitor](/wiki/Computer_monitor), [printer](/wiki/Printer_(computing)), [loudspeaker](/wiki/Computer_speaker) |
| Both | [Floppy disk drive](/wiki/Floppy_disk), [hard disk drive](/wiki/Hard_disk_drive), [optical disc](/wiki/Optical_disc) drive, [teleprinter](/wiki/Teleprinter) |
| [Computer buses](/wiki/Bus_(computing)) | Short range | [RS-232](/wiki/RS-232), [SCSI](/wiki/SCSI), [PCI](/wiki/Conventional_PCI), [USB](/wiki/Universal_Serial_Bus) |
| Long range ([computer networking](/wiki/Computer_networking)) | [Ethernet](/wiki/Ethernet), [ATM](/wiki/Asynchronous_Transfer_Mode), [FDDI](/wiki/Fiber_distributed_data_interface) |

## Software[[edit](/index.php?title=(none)&action=edit&section=42)]

[Template:Main](/wiki/Template:Main) *Software* refers to parts of the computer which do not have a material form, such as programs, data, protocols, etc. When software is stored in hardware that cannot easily be modified (such as [BIOS](/wiki/BIOS) [ROM](/wiki/Read-only_memory) in an [IBM PC compatible](/wiki/IBM_PC_compatible)), it is sometimes called "firmware".

|  |  |  |
| --- | --- | --- |
| [Operating system](/wiki/Operating_system) /System Software | [Unix](/wiki/Unix) and [BSD](/wiki/Berkeley_Software_Distribution) | [UNIX System V](/wiki/UNIX_System_V), [IBM AIX](/wiki/IBM_AIX), [HP-UX](/wiki/HP-UX), [Solaris](/wiki/Solaris_(operating_system)) ([SunOS](/wiki/SunOS)), [IRIX](/wiki/IRIX), [List of BSD operating systems](/wiki/List_of_BSD_operating_systems) |
| [GNU](/wiki/GNU)/[Linux](/wiki/Linux) | [List of Linux distributions](/wiki/List_of_Linux_distributions), [Comparison of Linux distributions](/wiki/Comparison_of_Linux_distributions) |
| [Microsoft Windows](/wiki/Microsoft_Windows) | [Windows 95](/wiki/Windows_95), [Windows 98](/wiki/Windows_98), [Windows NT](/wiki/Windows_NT), [Windows 2000](/wiki/Windows_2000), [Windows Me](/wiki/Windows_Me), [Windows XP](/wiki/Windows_XP), [Windows Vista](/wiki/Windows_Vista), [Windows 7](/wiki/Windows_7), [Windows 8](/wiki/Windows_8), [Windows 10](/wiki/Windows_10) |
| [DOS](/wiki/DOS) | [86-DOS](/wiki/86-DOS) (QDOS), [IBM PC DOS](/wiki/IBM_PC_DOS), [MS-DOS](/wiki/MS-DOS), [DR-DOS](/wiki/DR-DOS), [FreeDOS](/wiki/FreeDOS) |
| [Mac OS](/wiki/Mac_OS) | [Mac OS classic](/wiki/Mac_OS), [Mac OS X](/wiki/Mac_OS_X) |
| [Embedded](/wiki/Embedded_operating_system) and [real-time](/wiki/Real-time_operating_system) | [List of embedded operating systems](/wiki/List_of_operating_systems#Embedded) |
| Experimental | [Amoeba](/wiki/Amoeba_distributed_operating_system), [Oberon](/wiki/Oberon_(operating_system))/[Bluebottle](/wiki/Bluebottle_OS), [Plan 9 from Bell Labs](/wiki/Plan_9_from_Bell_Labs) |
| [Library](/wiki/Library_(computing)) | [Multimedia](/wiki/Multimedia) | [DirectX](/wiki/DirectX), [OpenGL](/wiki/OpenGL), [OpenAL](/wiki/OpenAL), [Vulkan (API)](/wiki/Vulkan_(API)) |
| Programming library | [C standard library](/wiki/C_standard_library), [Standard Template Library](/wiki/Standard_Template_Library) |
| [Data](/wiki/Data_(computing)) | [Protocol](/wiki/Protocol_(computing)) | [TCP/IP](/wiki/Internet_Protocol_Suite), [Kermit](/wiki/Kermit_(protocol)), [FTP](/wiki/File_Transfer_Protocol), [HTTP](/wiki/Hypertext_Transfer_Protocol), [SMTP](/wiki/Simple_Mail_Transfer_Protocol) |
| [File format](/wiki/File_format) | [HTML](/wiki/HTML), [XML](/wiki/XML), [JPEG](/wiki/JPEG), [MPEG](/wiki/Moving_Picture_Experts_Group), [PNG](/wiki/Portable_Network_Graphics) |
| [User interface](/wiki/User_interface) | [Graphical user interface](/wiki/Graphical_user_interface) ([WIMP](/wiki/WIMP_(computing))) | [Microsoft Windows](/wiki/Microsoft_Windows), [GNOME](/wiki/GNOME), [KDE](/wiki/KDE), [QNX Photon](/wiki/QNX), [CDE](/wiki/Common_Desktop_Environment), [GEM](/wiki/Graphical_Environment_Manager), [Aqua](/wiki/Aqua_(user_interface)) |
| [Text-based user interface](/wiki/Text-based_(computing)) | [Command-line interface](/wiki/Command-line_interface), [Text user interface](/wiki/Text_user_interface) |
| [Application](/wiki/Application_software) Software | [Office suite](/wiki/Office_suite) | [Word processing](/wiki/Word_processing), [Desktop publishing](/wiki/Desktop_publishing), [Presentation program](/wiki/Presentation_program), [Database management system](/wiki/Database_management_system), Scheduling & Time management, [Spreadsheet](/wiki/Spreadsheet), [Accounting software](/wiki/Accounting_software) |
| Internet Access | [Browser](/wiki/Web_browser), [E-mail client](/wiki/E-mail_client), [Web server](/wiki/Web_server), [Mail transfer agent](/wiki/Mail_transfer_agent), [Instant messaging](/wiki/Instant_messaging) |
| Design and manufacturing | [Computer-aided design](/wiki/Computer-aided_design), [Computer-aided manufacturing](/wiki/Computer-aided_manufacturing), Plant management, Robotic manufacturing, Supply chain management |
| [Graphics](/wiki/Computer_graphics) | [Raster graphics editor](/wiki/Raster_graphics_editor), [Vector graphics editor](/wiki/Vector_graphics_editor), [3D modeler](/wiki/3D_computer_graphics_software), [Animation editor](/wiki/Computer_animation), [3D computer graphics](/wiki/3D_computer_graphics), [Video editing](/wiki/Video_editing), [Image processing](/wiki/Image_processing) |
| [Audio](/wiki/Digital_audio) | [Digital audio editor](/wiki/Digital_audio_editor), [Audio playback](/wiki/Audio_player_(software)), [Mixing](/wiki/Audio_mixing), [Audio synthesis](/wiki/Software_synthesizer), [Computer music](/wiki/Computer_music) |
| [Software engineering](/wiki/Software_engineering) | [Compiler](/wiki/Compiler), [Assembler](/wiki/Assembler_(computer_programming)), [Interpreter](/wiki/Interpreter_(computing)), [Debugger](/wiki/Debugger), [Text editor](/wiki/Text_editor), [Integrated development environment](/wiki/Integrated_development_environment), [Software performance analysis](/wiki/Software_performance_analysis), [Revision control](/wiki/Revision_control), [Software configuration management](/wiki/Software_configuration_management) |
| Educational | [Edutainment](/wiki/Edutainment), [Educational game](/wiki/Educational_game), [Serious game](/wiki/Serious_game), [Flight simulator](/wiki/Flight_simulator) |
| [Games](/wiki/Video_game) | [Strategy](/wiki/Strategy_game), [Arcade](/wiki/Arcade_game), [Puzzle](/wiki/Puzzle_video_game), [Simulation](/wiki/Simulation_video_game), [First-person shooter](/wiki/First-person_shooter), [Platform](/wiki/Platform_game), [Massively multiplayer](/wiki/Massively_multiplayer_online_game), [Interactive fiction](/wiki/Interactive_fiction) |
| Misc | [Artificial intelligence](/wiki/Artificial_intelligence), [Antivirus software](/wiki/Antivirus_software), [Malware scanner](/wiki/Malware_scanner), [Installer](/wiki/Installation_(computer_programs))/[Package management systems](/wiki/Package_management_system), [File manager](/wiki/File_manager) |

## Languages[[edit](/index.php?title=(none)&action=edit&section=43)]

There are thousands of different programming languages—some intended to be general purpose, others useful only for highly specialized applications.

|  |  |
| --- | --- |
| [**Programming languages**](/wiki/Programming_language) | |
| Lists of programming languages | [Timeline of programming languages](/wiki/Timeline_of_programming_languages), [List of programming languages by category](/wiki/List_of_programming_languages_by_category), [Generational list of programming languages](/wiki/Generational_list_of_programming_languages), [List of programming languages](/wiki/List_of_programming_languages), [Non-English-based programming languages](/wiki/Non-English-based_programming_languages) |
| Commonly used [assembly languages](/wiki/Assembly_language) | [ARM](/wiki/ARM_architecture), [MIPS](/wiki/MIPS_architecture), [x86](/wiki/X86_assembly_language) |
| Commonly used [high-level programming languages](/wiki/High-level_programming_language) | [Ada](/wiki/Ada_(programming_language)), [BASIC](/wiki/BASIC), [C](/wiki/C_(programming_language)), [C++](/wiki/C++), [C#](/wiki/C_Sharp_(programming_language)), [COBOL](/wiki/COBOL), [Fortran](/wiki/Fortran), [PL/1](/wiki/PL/1), [REXX](/wiki/REXX), [Java](/wiki/Java_(programming_language)), [Lisp](/wiki/Lisp_(programming_language)), [Pascal](/wiki/Pascal_(programming_language)), [Object Pascal](/wiki/Object_Pascal) |
| Commonly used [scripting languages](/wiki/Scripting_language) | [Bourne script](/wiki/Bourne_shell), [JavaScript](/wiki/JavaScript), [Python](/wiki/Python_(programming_language)), [Ruby](/wiki/Ruby_(programming_language)), [PHP](/wiki/PHP), [Perl](/wiki/Perl) |

### Firmware[[edit](/index.php?title=(none)&action=edit&section=44)]

Firmware is the technology which has the combination of both hardware and software such as BIOS chip inside a computer. This chip (hardware) is located on the motherboard and has the BIOS set up (software) stored in it.

## Types of computers[[edit](/index.php?title=(none)&action=edit&section=45)]

Computers are typically classified based on their uses:

### Based on uses[[edit](/index.php?title=(none)&action=edit&section=46)]

* [Analog computer](/wiki/Analog_computer)
* Digital computer
* [Hybrid computer](/wiki/Hybrid_computer)

### Based on sizes[[edit](/index.php?title=(none)&action=edit&section=47)]

* [Micro computer](/wiki/Microcomputer)
* [Personal computer](/wiki/Personal_computer)
* Mini Computer
* Mainframe computer
* [Super computer](/wiki/Supercomputer)

## Input Devices[[edit](/index.php?title=(none)&action=edit&section=48)]

When unprocessed data is sent to the computer with the help of input devices, the data is processed and sent to output devices. The input devices may be hand-operated or automated. The act of processing is mainly regulated by the CPU. Some examples of hand-operated input devices are:

* [Overlay keyboard](/wiki/Overlay_keyboard)
* [Trackball](/wiki/Trackball)
* [Joystick](/wiki/Joystick)
* [Digital camera](/wiki/Digital_camera)
* [Microphone](/wiki/Microphone)
* [Touchscreen](/wiki/Touchscreen)
* [Digital video](/wiki/Digital_video)
* [Image scanner](/wiki/Image_scanner)
* [Graphics tablet](/wiki/Graphics_tablet)
* [Computer keyboard](/wiki/Computer_keyboard)
* [Mouse](/wiki/Mouse_(computing))

## Output Devices[[edit](/index.php?title=(none)&action=edit&section=49)]

The means through which computer gives output are known as output devices. Some examples of output devices are:

* [Computer monitor](/wiki/Computer_monitor)
* [Printer](/wiki/Printer_(computing))
* [Projector](/wiki/Projector)
* [Sound card](/wiki/Sound_card)
* [PC speaker](/wiki/PC_speaker)
* [Video card](/wiki/Video_card)

## Professions and organizations[[edit](/index.php?title=(none)&action=edit&section=50)]

As the use of computers has spread throughout society, there are an increasing number of careers involving computers.

|  |  |
| --- | --- |
| [**Computer-related professions**](/wiki/Category:Computer_occupations) | |
| Hardware-related | [Electrical engineering](/wiki/Electrical_engineering), [Electronic engineering](/wiki/Electronic_engineering), [Computer engineering](/wiki/Computer_engineering), [Telecommunications engineering](/wiki/Telecommunications_engineering), [Optical engineering](/wiki/Optical_engineering), [Nanoengineering](/wiki/Nanoengineering) |
| Software-related | [Computer science](/wiki/Computer_science), [Computer engineering](/wiki/Computer_engineering), [Desktop publishing](/wiki/Desktop_publishing), [Human–computer interaction](/wiki/Human–computer_interaction), [Information technology](/wiki/Information_technology), [Information systems](/wiki/Information_systems_(discipline)), [Computational science](/wiki/Computational_science), [Software engineering](/wiki/Software_engineering), [Video game industry](/wiki/Video_game_industry), [Web design](/wiki/Web_design) |

The need for computers to work well together and to be able to exchange information has spawned the need for many standards organizations, clubs and societies of both a formal and informal nature.

|  |  |
| --- | --- |
| [**Organizations**](/wiki/Category:Computer-related_organizations) | |
| Standards groups | [ANSI](/wiki/American_National_Standards_Institute), [IEC](/wiki/International_Electrotechnical_Commission), [IEEE](/wiki/Institute_of_Electrical_and_Electronics_Engineers), [IETF](/wiki/Internet_Engineering_Task_Force), [ISO](/wiki/International_Organization_for_Standardization), [W3C](/wiki/World_Wide_Web_Consortium) |
| Professional societies | [ACM](/wiki/Association_for_Computing_Machinery), [AIS](/wiki/Association_for_Information_Systems), [IET](/wiki/Institution_of_Engineering_and_Technology), [IFIP](/wiki/International_Federation_for_Information_Processing), [BCS](/wiki/British_Computer_Society) |
| [Free](/wiki/Free_software)/[open source software](/wiki/Open_source_software) groups | [Free Software Foundation](/wiki/Free_Software_Foundation), [Mozilla Foundation](/wiki/Mozilla_Foundation), [Apache Software Foundation](/wiki/Apache_Software_Foundation) |

## See also[[edit](/index.php?title=(none)&action=edit&section=51)]

[Template:Portal](/wiki/Template:Portal) [Template:Div col](/wiki/Template:Div_col)

* [Computability theory](/wiki/Computability_theory)
* [Computer insecurity](/wiki/Computer_insecurity)
* [Computer security](/wiki/Computer_security)
* [List of computer term etymologies](/wiki/List_of_computer_term_etymologies)
* [List of fictional computers](/wiki/List_of_fictional_computers)
* [Pulse computation](/wiki/Pulse_computation)
* [TOP500](/wiki/TOP500) (list of most powerful computers)
* [Glossary of computer hardware terms](/wiki/Glossary_of_computer_hardware_terms)

[Template:Div col end](/wiki/Template:Div_col_end)

## Notes[[edit](/index.php?title=(none)&action=edit&section=52)]

[Template:Reflist](/wiki/Template:Reflist)

## References[[edit](/index.php?title=(none)&action=edit&section=53)]

[Template:Refbegin](/wiki/Template:Refbegin)

* Fuegi, J. and Francis, J. "Lovelace & Babbage and the creation of the 1843 'notes'". *IEEE Annals of the History of Computing* 25 No. 4 (October–December 2003): [Digital Object Identifier](http://dx.doi.org/10.1109%2FMAHC.2003.1253887)[Template:Dead link](/wiki/Template:Dead_link)[Template:Cbignore](/wiki/Template:Cbignore)
* [Template:Note label](/wiki/Template:Note_label) [Template:Cite journal](/wiki/Template:Cite_journal)
* [Template:Note label](/wiki/Template:Note_label) [Template:Cite web](/wiki/Template:Cite_web)
* [Template:Note label](/wiki/Template:Note_label) [Template:Cite journal](/wiki/Template:Cite_journal)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite journal](/wiki/Template:Cite_journal)
* [Template:Cite journal](/wiki/Template:Cite_journal)
* [Template:Cite web](/wiki/Template:Cite_web)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite book](/wiki/Template:Cite_book)
* [Template:Cite web](/wiki/Template:Cite_web)

[Template:Refend](/wiki/Template:Refend)

## External links[[edit](/index.php?title=(none)&action=edit&section=54)]

* [Warhol & The Computer](http://www.computerhistory.org/atchm/warhol-the-computer/)

[Template:Sister-inline](/wiki/Template:Sister-inline)

[Template:Use dmy dates](/wiki/Template:Use_dmy_dates)

[Template:Authority control](/wiki/Template:Authority_control) [Template:Digital systems](/wiki/Template:Digital_systems)

[Category:Computers](/wiki/Category:Computers) [Category:Articles containing video clips](/wiki/Category:Articles_containing_video_clips) [Category:Articles with example code](/wiki/Category:Articles_with_example_code)