Listing of Operating Systems

<http://en.wikipedia.org/wiki/List_of_operating_systems>

Embedded system

From Wikipedia, the free encyclopedia

|  |  |
| --- | --- |
| Text document with red question mark.svg | This article includes a [list of references](http://en.wikipedia.org/wiki/Wikipedia%3ACiting_sources) or [external links](http://en.wikipedia.org/wiki/Wikipedia%3AExternal_links), but **its sources remain unclear because it has insufficient**[**inline citations**](http://en.wikipedia.org/wiki/Wikipedia%3ACiting_sources#Adding_the_citation). Please help to [improve](http://en.wikipedia.org/wiki/Wikipedia%3AWikiProject_Fact_and_Reference_Check) this article by introducing more precise citations [where appropriate](http://en.wikipedia.org/wiki/Wikipedia%3AWhen_to_cite). *(July 2008)* |





Picture of the internals of an [ADSL](http://en.wikipedia.org/wiki/ADSL) [modem](http://en.wikipedia.org/wiki/Modem)/[router](http://en.wikipedia.org/wiki/Router). A modern example of an embedded system. Labelled parts include a [microprocessor](http://en.wikipedia.org/wiki/Microprocessor) (4), [RAM](http://en.wikipedia.org/wiki/RAM) (6), and [flash memory](http://en.wikipedia.org/wiki/Flash_memory)(7).

An **embedded system** is a [computer system](http://en.wikipedia.org/wiki/Computer_system) designed to perform one or a few dedicated functions[[1]](http://en.wikipedia.org/wiki/Embedded_system%22%20%5Cl%20%22cite_note-barr-glossary-0)[[2]](http://en.wikipedia.org/wiki/Embedded_system#cite_note-1) often with [real-time computing](http://en.wikipedia.org/wiki/Real-time_computing)constraints. It is *embedded* as part of a complete device often including hardware and mechanical parts. By contrast, a general-purpose computer, such as a [personal computer](http://en.wikipedia.org/wiki/Personal_computer) (PC), is designed to be flexible and to meet a wide range of end-user needs. Embedded systems control many devices in common use today.[[3]](http://en.wikipedia.org/wiki/Embedded_system#cite_note-2)

Embedded systems are controlled by one or more main processing cores that are typically either [microcontrollers](http://en.wikipedia.org/wiki/Microcontroller) or [digital signal processors](http://en.wikipedia.org/wiki/Digital_signal_processor)(DSP).[[4]](http://en.wikipedia.org/wiki/Embedded_system#cite_note-microcontroller-3) The key characteristic, however, is being dedicated to handle a particular task, which may require very powerful processors. For example, [air traffic control](http://en.wikipedia.org/wiki/Air_traffic_control) systems may usefully be viewed as embedded, even though they involve [mainframe computers](http://en.wikipedia.org/wiki/Mainframe_computers) and dedicated regional and national networks between airports and radar sites. (Each radar probably includes one or more embedded systems of its own.)

Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from [economies of scale](http://en.wikipedia.org/wiki/Economies_of_scale).

Physically, embedded systems range from portable devices such as [digital watches](http://en.wikipedia.org/wiki/Digital_watch) and [MP3 players](http://en.wikipedia.org/wiki/Digital_audio_player), to large stationary installations like[traffic lights](http://en.wikipedia.org/wiki/Traffic_light), [factory controllers](http://en.wikipedia.org/wiki/Programmable_logic_controller), or the systems controlling [nuclear power plants](http://en.wikipedia.org/wiki/Nuclear_power_plant). Complexity varies from low, with a single [microcontroller](http://en.wikipedia.org/wiki/Microcontroller) chip, to very high with multiple units, peripherals and networks mounted inside a large [chassis](http://en.wikipedia.org/wiki/Chassis) or enclosure.

In general, "embedded system" is not a strictly definable term, as most systems have some element of extensibility or programmability. For example, [handheld computers](http://en.wikipedia.org/wiki/Handheld_computer) share some elements with embedded systems such as the operating systems and microprocessors which power them, but they allow different applications to be loaded and peripherals to be connected. Moreover, even systems which don't expose programmability as a primary feature generally need to support software updates. On a continuum from "general purpose" to "embedded", large application systems will have subcomponents at most points even if the system as a whole is "designed to perform one or a few dedicated functions", and is thus appropriate to call "embedded".

|  |
| --- |
| **Contents** [[hide](http://en.wikipedia.org/wiki/Embedded_system)]* [1 Variety of embedded systems](http://en.wikipedia.org/wiki/Embedded_system#Variety_of_embedded_systems)
* [2 History](http://en.wikipedia.org/wiki/Embedded_system#History)
* [3 Characteristics](http://en.wikipedia.org/wiki/Embedded_system#Characteristics)
	+ [3.1 User interface](http://en.wikipedia.org/wiki/Embedded_system#User_interface)
	+ [3.2 Processors in embedded systems](http://en.wikipedia.org/wiki/Embedded_system#Processors_in_embedded_systems)
		- [3.2.1 Ready made computer boards](http://en.wikipedia.org/wiki/Embedded_system#Ready_made_computer_boards)
		- [3.2.2 ASIC and FPGA solutions](http://en.wikipedia.org/wiki/Embedded_system#ASIC_and_FPGA_solutions)
	+ [3.3 Peripherals](http://en.wikipedia.org/wiki/Embedded_system#Peripherals)
	+ [3.4 Tools](http://en.wikipedia.org/wiki/Embedded_system#Tools)
	+ [3.5 Debugging](http://en.wikipedia.org/wiki/Embedded_system#Debugging)
	+ [3.6 Reliability](http://en.wikipedia.org/wiki/Embedded_system#Reliability)
	+ [3.7 High vs low volume](http://en.wikipedia.org/wiki/Embedded_system#High_vs_low_volume)
* [4 Embedded software architectures](http://en.wikipedia.org/wiki/Embedded_system#Embedded_software_architectures)
	+ [4.1 Simple control loop](http://en.wikipedia.org/wiki/Embedded_system#Simple_control_loop)
	+ [4.2 Interrupt controlled system](http://en.wikipedia.org/wiki/Embedded_system#Interrupt_controlled_system)
	+ [4.3 Cooperative multitasking](http://en.wikipedia.org/wiki/Embedded_system#Cooperative_multitasking)
	+ [4.4 Preemptive multitasking or multi-threading](http://en.wikipedia.org/wiki/Embedded_system#Preemptive_multitasking_or_multi-threading)
	+ [4.5 Microkernels and exokernels](http://en.wikipedia.org/wiki/Embedded_system#Microkernels_and_exokernels)
	+ [4.6 Monolithic kernels](http://en.wikipedia.org/wiki/Embedded_system#Monolithic_kernels)
	+ [4.7 Exotic custom operating systems](http://en.wikipedia.org/wiki/Embedded_system#Exotic_custom_operating_systems)
	+ [4.8 Additional software components](http://en.wikipedia.org/wiki/Embedded_system#Additional_software_components)
* [5 See also](http://en.wikipedia.org/wiki/Embedded_system#See_also)
* [6 References](http://en.wikipedia.org/wiki/Embedded_system#References)
* [7 External links](http://en.wikipedia.org/wiki/Embedded_system#External_links)
 |

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=1)]Variety of embedded systems





PC Engines' ALIX.1C [Mini-ITX](http://en.wikipedia.org/wiki/Mini-ITX) embedded board with an x86 [AMD Geode](http://en.wikipedia.org/wiki/Geode_%28processor%29) LX 800 together with[Compact Flash](http://en.wikipedia.org/wiki/Compact_Flash), [miniPCI](http://en.wikipedia.org/wiki/MiniPCI) and PCI slots, 44-pin [IDE](http://en.wikipedia.org/wiki/Integrated_Drive_Electronics)interface, audio, USB and 256MB RAM





An embedded RouterBoard 112 with[U.FL](http://en.wikipedia.org/wiki/U.FL)-[RSMA](http://en.wikipedia.org/wiki/SMA_connector) pigtail and R52 [miniPCI](http://en.wikipedia.org/wiki/MiniPCI) [Wi-Fi](http://en.wikipedia.org/wiki/Wi-Fi) card widely used by [wireless](http://en.wikipedia.org/wiki/Wireless) [Internet](http://en.wikipedia.org/wiki/Internet) service providers ([WISPs](http://en.wikipedia.org/wiki/Wireless_internet_service_provider)) in the [Czech Republic](http://en.wikipedia.org/wiki/Czech_Republic).

Embedded systems span all aspects of modern life and there are many examples of their use.

Telecommunications systems employ numerous embedded systems from [telephone switches](http://en.wikipedia.org/wiki/Telephone_switch) for the network to [mobile phones](http://en.wikipedia.org/wiki/Mobile_phone) at the end-user. Computer networking uses dedicated [routers](http://en.wikipedia.org/wiki/Router) and [network bridges](http://en.wikipedia.org/wiki/Network_bridge) to route data.

[Consumer electronics](http://en.wikipedia.org/wiki/Consumer_electronics) include [personal digital assistants](http://en.wikipedia.org/wiki/Personal_digital_assistant) (PDAs), [mp3 players](http://en.wikipedia.org/wiki/Mp3_player), mobile phones, [videogame consoles](http://en.wikipedia.org/wiki/Videogame_console), [digital cameras](http://en.wikipedia.org/wiki/Digital_camera), [DVD players](http://en.wikipedia.org/wiki/DVD_player),[GPS](http://en.wikipedia.org/wiki/Global_Positioning_System) receivers, and [printers](http://en.wikipedia.org/wiki/Computer_printer). Many household appliances, such as [microwave ovens](http://en.wikipedia.org/wiki/Microwave_oven), [washing machines](http://en.wikipedia.org/wiki/Washing_machine) and [dishwashers](http://en.wikipedia.org/wiki/Dishwashers), are including embedded systems to provide flexibility, efficiency and features. Advanced [HVAC](http://en.wikipedia.org/wiki/HVAC) systems use networked [thermostats](http://en.wikipedia.org/wiki/Thermostat) to more accurately and efficiently control temperature that can change by time of day and [season](http://en.wikipedia.org/wiki/Season). [Home automation](http://en.wikipedia.org/wiki/Home_automation) uses wired- and wireless-networking that can be used to control lights, climate, security, audio/visual, surveillance, etc., all of which use embedded devices for sensing and controlling.

Transportation systems from flight to automobiles increasingly use embedded systems. New airplanes contain advanced [avionics](http://en.wikipedia.org/wiki/Avionics) such as [inertial guidance systems](http://en.wikipedia.org/wiki/Inertial_guidance_system) and [GPS](http://en.wikipedia.org/wiki/Global_Positioning_System) receivers that also have considerable safety requirements. Various electric motors — [brushless DC motors](http://en.wikipedia.org/wiki/Brushless_DC_motor), [induction motors](http://en.wikipedia.org/wiki/Induction_motor) and [DC motors](http://en.wikipedia.org/wiki/DC_motor) — are using electric/electronic [motor controllers](http://en.wikipedia.org/wiki/Motor_controller). [Automobiles](http://en.wikipedia.org/wiki/Automobile), [electric vehicles](http://en.wikipedia.org/wiki/Electric_vehicle), and [hybrid vehicles](http://en.wikipedia.org/wiki/Hybrid_vehicle) are increasingly using embedded systems to maximize efficiency and reduce pollution. Other automotive safety systems include [anti-lock braking system](http://en.wikipedia.org/wiki/Anti-lock_braking_system) (ABS), [Electronic Stability Control](http://en.wikipedia.org/wiki/Electronic_Stability_Control) (ESC/ESP), [traction control](http://en.wikipedia.org/wiki/Traction_control_system) (TCS) and automatic [four-wheel drive](http://en.wikipedia.org/wiki/Four-wheel_drive).

[Medical equipment](http://en.wikipedia.org/wiki/Medical_equipment) is continuing to advance with more embedded systems for [vital signs](http://en.wikipedia.org/wiki/Vital_signs) monitoring, [electronic stethoscopes](http://en.wikipedia.org/wiki/Electronic_stethoscope) for amplifying sounds, and various [medical imaging](http://en.wikipedia.org/wiki/Medical_imaging) ([PET](http://en.wikipedia.org/wiki/Positron_emission_tomography), [SPECT](http://en.wikipedia.org/wiki/Single_photon_emission_computed_tomography), [CT](http://en.wikipedia.org/wiki/Computed_tomography), [MRI](http://en.wikipedia.org/wiki/Magnetic_resonance_imaging)) for non-invasive internal inspections.

In addition to commonly described embedded systems based on small computers, a new class of miniature wireless devices called [motes](http://en.wikipedia.org/wiki/Motes) are quickly gaining popularity as the field of wireless sensor networking rises. Wireless sensor networking, [WSN](http://en.wikipedia.org/wiki/WSN), makes use of miniaturization made possible by advanced IC design to couple full wireless subsystems to sophisticated sensors, enabling people and companies to measure a myriad of things in the physical world and act on this information through IT monitoring and control systems. These motes are completely self contained, and will typically run off a battery source for many years before the batteries need to be changed or charged.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=2)]History

In the earliest years of computers in the 1940–50s, computers were sometimes dedicated to a single task, but were far too large and expensive for most kinds of tasks performed by embedded computers of today. Over time however, the concept of [programmable controllers](http://en.wikipedia.org/wiki/Programmable_controllers) evolved from traditional[electromechanical](http://en.wikipedia.org/wiki/Electromechanical) sequencers, via solid state devices, to the use of computer technology.

One of the first recognizably modern embedded systems was the [Apollo Guidance Computer](http://en.wikipedia.org/wiki/Apollo_Guidance_Computer), developed by [Charles Stark Draper](http://en.wikipedia.org/wiki/Charles_Stark_Draper) at the MIT Instrumentation Laboratory. At the project's inception, the Apollo guidance computer was considered the riskiest item in the Apollo project as it employed the then newly developed monolithic integrated circuits to reduce the size and weight. An early mass-produced embedded system was the Autonetics D-17 guidance computer for the [Minuteman missile](http://en.wikipedia.org/wiki/Minuteman_%28missile%29), released in 1961. It was built from [transistor](http://en.wikipedia.org/wiki/Transistor) [logic](http://en.wikipedia.org/wiki/Digital_circuit) and had a [hard disk](http://en.wikipedia.org/wiki/Hard_disk) for main memory. When the Minuteman II went into production in 1966, the D-17 was replaced with a new computer that was the first high-volume use of integrated circuits. This program alone reduced prices on quad [nand gate ICs](http://en.wikipedia.org/wiki/Sheffer_stroke#NAND_gate) from $1000/each to $3/each[*[citation needed](http://en.wikipedia.org/wiki/Wikipedia%3ACitation_needed%22%20%5Co%20%22Wikipedia%3ACitation%20needed)*], permitting their use in commercial products.

Since these early applications in the 1969s, embedded systems have come down in price and there has been a dramatic rise in processing power and functionality. The first [microprocessor](http://en.wikipedia.org/wiki/Microprocessor) for example, the [Intel 4004](http://en.wikipedia.org/wiki/Intel_4004), was designed for [calculators](http://en.wikipedia.org/wiki/Calculator) and other small systems but still required many external memory and support chips. In 1978 National Engineering Manufacturers Association released a "standard" for programmable microcontrollers, including almost any computer-based controllers, such as single board computers, numerical, and event-based controllers.

As the cost of microprocessors and microcontrollers fell it became feasible to replace expensive knob-based [analog](http://en.wikipedia.org/wiki/Analog_electronics) components such as [potentiometers](http://en.wikipedia.org/wiki/Potentiometer) and [variable capacitors](http://en.wikipedia.org/wiki/Variable_capacitor) with up/down buttons or knobs read out by a microprocessor even in some consumer products. By the mid-1980s, most of the common previously external system components had been integrated into the same chip as the processor and this modern form of the [microcontroller](http://en.wikipedia.org/wiki/Microcontroller) allowed an even more widespread use, which by the end of the decade were the norm rather than the exception for almost all electronics devices.

The integration of microcontrollers has further increased the applications for which embedded systems are used into areas where traditionally a computer would not have been considered. A general purpose and comparatively low-cost microcontroller may often be programmed to fulfill the same role as a large number of separate components. Although in this context an embedded system is usually more complex than a traditional solution, most of the complexity is contained within the microcontroller itself. Very few additional components may be needed and most of the design effort is in the software. The intangible nature of software makes it much easier to prototype and test new revisions compared with the design and construction of a new circuit not using an embedded processor.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=3)]Characteristics

[File:Gumstix-Overo-X-w coin.jpg](http://en.wikipedia.org/wiki/Wikipedia%3AUpload?wpDestFile=Gumstix-Overo-X-w_coin.jpg)

[Gumstix](http://en.wikipedia.org/wiki/Gumstix) Overo COM, a tiny, [OMAP](http://en.wikipedia.org/wiki/OMAP)-based embedded [computer-on-module](http://en.wikipedia.org/wiki/Computer-on-module) with [Wifi](http://en.wikipedia.org/wiki/Wifi) and [Bluetooth](http://en.wikipedia.org/wiki/Bluetooth).

1. Embedded systems are designed to do some specific task, rather than be a general-purpose computer for multiple tasks. Some also have [real-time](http://en.wikipedia.org/wiki/Real-time_computing)performance constraints that must be met, for reasons such as safety and usability; others may have low or no performance requirements, allowing the system hardware to be simplified to reduce costs.
2. Embedded systems are not always standalone devices. Many embedded systems consist of small, computerized parts within a larger device that serves a more general purpose. For example, the [Gibson Robot Guitar](http://en.wikipedia.org/wiki/Gibson_Robot_Guitar) features an embedded system for tuning the strings, but the overall purpose of the Robot Guitar is, of course, to play music.[[5]](http://en.wikipedia.org/wiki/Embedded_system#cite_note-4) Similarly, an embedded system in an [automobile](http://en.wikipedia.org/wiki/Automobile) provides a specific function as a subsystem of the car itself.
3. The program instructions written for embedded systems are referred to as [firmware](http://en.wikipedia.org/wiki/Firmware), and are stored in read-only memory or [Flash memory](http://en.wikipedia.org/wiki/Flash_memory) chips. They run with limited computer hardware resources: little memory, small or non-existent keyboard and/or screen.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=4)]**User interface**





Embedded system [text user interface](http://en.wikipedia.org/wiki/Text_user_interface)using MicroVGA

Embedded systems range from no user interface at all — dedicated only to one task — to complex [graphical user interfaces](http://en.wikipedia.org/wiki/Desktop_operating_system#Graphical_user_interfaces) that resemble modern computer desktop operating systems. Simple embedded devices use [buttons](http://en.wikipedia.org/wiki/Push-button), [LEDs](http://en.wikipedia.org/wiki/LED), graphic or character [LCDs](http://en.wikipedia.org/wiki/LCD) (for example popular [HD44780 LCD](http://en.wikipedia.org/wiki/HD44780_Character_LCD)) with a simple[menu system](http://en.wikipedia.org/wiki/Menu_%28computing%29).

A more sophisticated devices use graphical screen with [touch](http://en.wikipedia.org/wiki/Touch_screen) sensing or screen-edge buttons provide flexibility while minimizing space used: the meaning of the buttons can change with the screen, and selection involves the natural behavior of pointing at what's desired. [Handheld systems](http://en.wikipedia.org/wiki/Mobile_device) often have a screen with a "joystick button" for a pointing device.

Some systems provide user interface remotely with the help of serial (e.g. [RS-232](http://en.wikipedia.org/wiki/RS-232), [USB](http://en.wikipedia.org/wiki/USB), [I²C](http://en.wikipedia.org/wiki/I%C2%B2C), etc.) or network (e.g. [Ethernet](http://en.wikipedia.org/wiki/Ethernet)) connection. In spite of installed client software and cables are needed this approach usually gives a lot of advantages: extends the capabilities of embedded system, avoids the cost of a display, simplifies [BSP](http://en.wikipedia.org/wiki/Board_support_package), allows to build rich user interface on PC.

|  |  |
| --- | --- |
| Ambox question.svg | The text in this march 2010 may be **incoherent or very hard to understand**, and should be reworded if the intended meaning can be determined. The [talk page](http://en.wikipedia.org/wiki/Talk%3AEmbedded_system#march_2010) may have details. *(March 2010)* |

One of the well established models in this direction is the combination of [embedded web server](http://en.wikipedia.org/wiki/Embedded_HTTP_server) running on embedded device like [IP camera](http://en.wikipedia.org/wiki/IP_camera) or [network routers](http://en.wikipedia.org/wiki/Router) and user interface shown in [web browser](http://en.wikipedia.org/wiki/Web_browser)of PC connected with ethernet or wi-fi to embedded device.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=5)]**Processors in embedded systems**

Embedded processors can be broken into two broad categories: ordinary microprocessors (μP) and microcontrollers (μC), which have many more peripherals on chip, reducing cost and size. Contrasting to the personal computer and server markets, a fairly large number of basic [CPU architectures](http://en.wikipedia.org/wiki/CPU_architecture) are used; there are [Von Neumann](http://en.wikipedia.org/wiki/Von_Neumann_architecture) as well as various degrees of [Harvard architectures](http://en.wikipedia.org/wiki/Harvard_architecture), [RISC](http://en.wikipedia.org/wiki/RISC)as well as non-RISC and [VLIW](http://en.wikipedia.org/wiki/VLIW); word lengths vary from 4-bit to 64-bits and beyond (mainly in [DSP](http://en.wikipedia.org/wiki/Digital_signal_processor) processors) although the most typical remain 8/16-bit. Most architectures come in a large number of different variants and shapes, many of which are also manufactured by several different companies.

A long but still not exhaustive list of common architectures are: [65816](http://en.wikipedia.org/wiki/65816), [65C02](http://en.wikipedia.org/wiki/65C02), [68HC08](http://en.wikipedia.org/wiki/68HC08), [68HC11](http://en.wikipedia.org/wiki/68HC11), [68k](http://en.wikipedia.org/wiki/68k), [8051](http://en.wikipedia.org/wiki/Intel_8051), [ARM](http://en.wikipedia.org/wiki/ARM_architecture), [AVR](http://en.wikipedia.org/wiki/Atmel_AVR), [AVR32](http://en.wikipedia.org/wiki/AVR32), [Blackfin](http://en.wikipedia.org/wiki/Blackfin), [C167](http://en.wikipedia.org/wiki/C167_family), [Coldfire](http://en.wikipedia.org/wiki/Coldfire), [COP8](http://en.wikipedia.org/wiki/COP8), [Cortus APS3](http://en.wikipedia.org/w/index.php?title=Cortus_APS3&action=edit&redlink=1), [eZ8](http://en.wikipedia.org/wiki/Zilog_Z8), [eZ80](http://en.wikipedia.org/wiki/EZ80), [FR-V](http://en.wikipedia.org/wiki/FR-V), [H8](http://en.wikipedia.org/wiki/Renesas_H8),[HT48](http://en.wikipedia.org/wiki/HT48FXX_Flash_I/O_type_series), [M16C](http://en.wikipedia.org/wiki/M16C), [M32C](http://en.wikipedia.org/wiki/M32C), [MIPS](http://en.wikipedia.org/wiki/MIPS_architecture), [MSP430](http://en.wikipedia.org/wiki/MSP430), [PIC](http://en.wikipedia.org/wiki/PIC_microcontroller), [PowerPC](http://en.wikipedia.org/wiki/PowerPC), [R8C](http://en.wikipedia.org/wiki/R8C), [SHARC](http://en.wikipedia.org/wiki/Super_Harvard_Architecture_Single-Chip_Computer), [SPARC](http://en.wikipedia.org/wiki/SPARC), [ST6](http://en.wikipedia.org/wiki/ST6), [SuperH](http://en.wikipedia.org/wiki/SuperH), [TLCS-47](http://en.wikipedia.org/wiki/TLCS-47), [TLCS-870](http://en.wikipedia.org/wiki/TLCS-870), [TLCS-900](http://en.wikipedia.org/wiki/TLCS-900), [Tricore](http://en.wikipedia.org/wiki/Tricore), [V850](http://en.wikipedia.org/wiki/V850), [x86](http://en.wikipedia.org/wiki/X86_architecture), [XE8000](http://en.wikipedia.org/wiki/XE8000), [Z80](http://en.wikipedia.org/wiki/Z80), [AsAP](http://en.wikipedia.org/wiki/Asynchronous_array_of_simple_processors) etc.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=6)]**Ready made computer boards**

[PC/104](http://en.wikipedia.org/wiki/PC/104) and PC/104+ are examples of standards for *ready made* computer boards intended for small, low-volume embedded and ruggedized systems, mostly x86-based. These often use [DOS](http://en.wikipedia.org/wiki/DOS), [Linux](http://en.wikipedia.org/wiki/Linux),[NetBSD](http://en.wikipedia.org/wiki/NetBSD), or an embedded [real-time operating system](http://en.wikipedia.org/wiki/Real-time_operating_system) such as [MicroC/OS-II](http://en.wikipedia.org/wiki/MicroC/OS-II), [QNX](http://en.wikipedia.org/wiki/QNX) or [VxWorks](http://en.wikipedia.org/wiki/VxWorks). Sometimes these boards use non-x86 processors.

In certain applications, where small size or power efficiency are not primary concerns, the components used may be compatible with those used in general purpose x86 personal computers. Boards such as the VIA [EPIA](http://en.wikipedia.org/wiki/EPIA) range help to bridge the gap by being PC-compatible but highly integrated, physically smaller or have other attributes making them attractive to embedded engineers. The advantage of this approach is that low-cost commodity components may be used along with the same software development tools used for general software development. Systems built in this way are still regarded as embedded since they are integrated into larger devices and fulfill a single role. Examples of devices that may adopt this approach are [ATMs](http://en.wikipedia.org/wiki/Automated_teller_machines) and [arcade machines](http://en.wikipedia.org/wiki/Arcade_machines), which contain code specific to the application.

However, most ready-made embedded systems boards are not PC-centered and do not use the ISA or PCI busses. When a [System-on-a-chip](http://en.wikipedia.org/wiki/System-on-a-chip) processor is involved, there may be little benefit to having a standarized bus connecting discrete compontents, and the environment for both hardware and software tools may be very different.

One common design style uses a small system module, perhaps the size of a business card, holding high density [BGA](http://en.wikipedia.org/wiki/Ball_grid_array) chips such as an [ARM](http://en.wikipedia.org/wiki/ARM_architecture)-based [System-on-a-chip](http://en.wikipedia.org/wiki/System-on-a-chip) processor and peripherals, external [flash memory](http://en.wikipedia.org/wiki/Flash_memory) for storage, and [DRAM](http://en.wikipedia.org/wiki/Dynamic_random_access_memory) for runtime memory. The module vendor will usually provide boot software and make sure there is a selection of operating systems, usually including[Linux](http://en.wikipedia.org/wiki/Linux) and some real time choices. These modules can be manufactured in high volume, by organizations familiar with their specialized testing issues, and combined with much lower volume custom mainboards with application-specific external peripherals. [Gumstix](http://en.wikipedia.org/wiki/Gumstix) product lines are a Linux-centric example of this model.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=7)]**ASIC and FPGA solutions**

A common configuration for very-high-volume embedded systems is the [system on a chip](http://en.wikipedia.org/wiki/System_on_a_chip) (SoC) which contains a complete system consisting of multiple processors, multipliers, caches and interfaces on a single chip. SoCs can be implemented as an [application-specific integrated circuit](http://en.wikipedia.org/wiki/Application-specific_integrated_circuit) (ASIC) or using a [field-programmable gate array](http://en.wikipedia.org/wiki/Field-programmable_gate_array) (FPGA).

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=8)]**Peripherals**

Embedded Systems talk with the outside world via [peripherals](http://en.wikipedia.org/wiki/Peripheral), such as:

* Serial Communication Interfaces (SCI): [RS-232](http://en.wikipedia.org/wiki/RS-232), [RS-422](http://en.wikipedia.org/wiki/RS-422), [RS-485](http://en.wikipedia.org/wiki/RS-485) etc.
* Synchronous Serial Communication Interface: [I2C](http://en.wikipedia.org/wiki/I2C), [SPI](http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus), SSC and ESSI (Enhanced Synchronous Serial Interface)
* [Universal Serial Bus](http://en.wikipedia.org/wiki/Universal_Serial_Bus) (USB)
* Multi Media Cards (SD Cards, Compact Flash etc.)
* Networks: [Ethernet](http://en.wikipedia.org/wiki/Ethernet), [Controller Area Network](http://en.wikipedia.org/wiki/Controller_Area_Network), [LonWorks](http://en.wikipedia.org/wiki/LonWorks), etc.
* Timers: [PLL](http://en.wikipedia.org/wiki/PLL)(s), Capture/Compare and [Time Processing Units](http://en.wikipedia.org/wiki/Time_Processing_Unit)
* Discrete IO: aka [General Purpose Input/Output](http://en.wikipedia.org/wiki/General_Purpose_Input/Output) (GPIO)
* Analog to Digital/Digital to Analog ([ADC](http://en.wikipedia.org/wiki/Analog-to-digital_converter)/[DAC](http://en.wikipedia.org/wiki/Digital-to-analog_converter))
* Debugging: [JTAG](http://en.wikipedia.org/wiki/JTAG), [ISP](http://en.wikipedia.org/wiki/In-System_Programming), [ICSP](http://en.wikipedia.org/wiki/In_Circuit_Serial_Programming_%28ICSP%29), [BDM](http://en.wikipedia.org/wiki/Background_Debug_Mode_interface) Port, BITP DP9 port ...

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=9)]**Tools**

As for other software, embedded system designers use [compilers](http://en.wikipedia.org/wiki/Compiler), [assemblers](http://en.wikipedia.org/wiki/Assembly_language#Assembler), and [debuggers](http://en.wikipedia.org/wiki/Debugger) to develop embedded system software. However, they may also use some more specific tools:

* In circuit debuggers or emulators (see next section).
* Utilities to add a checksum or [CRC](http://en.wikipedia.org/wiki/Cyclic_redundancy_check) to a program, so the embedded system can check if the program is valid.
* For systems using [digital signal processing](http://en.wikipedia.org/wiki/Digital_signal_processing), developers may use a math workbench such as [Scilab](http://en.wikipedia.org/wiki/Scilab) / [Scicos](http://en.wikipedia.org/wiki/Scicos), [MATLAB](http://en.wikipedia.org/wiki/MATLAB) / [Simulink](http://en.wikipedia.org/wiki/Simulink), [EICASLAB](http://en.wikipedia.org/wiki/EICASLAB), [MathCad](http://en.wikipedia.org/wiki/MathCad) [Mathematica](http://en.wikipedia.org/wiki/Mathematica),or [FlowStone DSP](http://en.wikipedia.org/w/index.php?title=FlowStone_DSP&action=edit&redlink=1) to simulate the mathematics. They might also use libraries for both the host and target which eliminates developing DSP routines as done in [DSPnano RTOS](http://en.wikipedia.org/wiki/DSPnano_RTOS) and [Unison Operating System](http://en.wikipedia.org/wiki/Unison_Operating_System).
* Custom compilers and linkers may be used to improve optimisation for the particular hardware.
* An embedded system may have its own special language or design tool, or add enhancements to an existing language such as [Forth](http://en.wikipedia.org/wiki/Forth_%28programming_language%29) or [Basic](http://en.wikipedia.org/wiki/BASIC_Stamp).
* Another alternative is to add a [real-time operating system](http://en.wikipedia.org/wiki/Real-time_operating_system) or [embedded operating system](http://en.wikipedia.org/wiki/Embedded_operating_system), which may have DSP capabilities like [DSPnano RTOS](http://en.wikipedia.org/wiki/DSPnano_RTOS).

Software tools can come from several sources:

* Software companies that specialize in the embedded market
* Ported from the [GNU](http://en.wikipedia.org/wiki/GNU) software development tools
* Sometimes, development tools for a personal computer can be used if the embedded processor is a close relative to a common PC processor

As the complexity of embedded systems grows, higher level tools and operating systems are migrating into machinery where it makes sense. For example, [cellphones](http://en.wikipedia.org/wiki/Cellphone), [personal digital assistants](http://en.wikipedia.org/wiki/Personal_digital_assistant) and other consumer computers often need significant software that is purchased or provided by a person other than the manufacturer of the electronics. In these systems, an open programming environment such as [Linux](http://en.wikipedia.org/wiki/Linux), [NetBSD](http://en.wikipedia.org/wiki/NetBSD), [OSGi](http://en.wikipedia.org/wiki/OSGi) or [Embedded Java](http://en.wikipedia.org/wiki/Embedded_Java) is required so that the third-party software provider can sell to a large market.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=10)]**Debugging**

Embedded [debugging](http://en.wikipedia.org/wiki/Debugging) may be performed at different levels, depending on the facilities available. From simplest to most sophisticated they can be roughly grouped into the following areas:

* Interactive resident debugging, using the simple shell provided by the embedded operating system (e.g. Forth and Basic)
* External debugging using logging or serial port output to trace operation using either a monitor in flash or using a debug server like the [Remedy Debugger](http://en.wikipedia.org/wiki/Remedy_Debugger) which even works for heterogeneous[multicore](http://en.wikipedia.org/wiki/Multi-core_processor) systems.
* An in-circuit debugger (ICD), a hardware device that connects to the microprocessor via a [JTAG](http://en.wikipedia.org/wiki/JTAG) or [Nexus](http://en.wikipedia.org/wiki/Nexus_%28standard%29) interface. This allows the operation of the microprocessor to be controlled externally, but is typically restricted to specific debugging capabilities in the processor.
* An [in-circuit emulator](http://en.wikipedia.org/wiki/In-circuit_emulator) replaces the microprocessor with a simulated equivalent, providing full control over all aspects of the microprocessor.
* A complete [emulator](http://en.wikipedia.org/wiki/Emulator) provides a simulation of all aspects of the hardware, allowing all of it to be controlled and modified, and allowing debugging on a normal PC.

Unless restricted to external debugging, the programmer can typically load and run software through the tools, view the code running in the processor, and start or stop its operation. The view of the code may be as [assembly code](http://en.wikipedia.org/wiki/Assembly_code) or [source-code](http://en.wikipedia.org/wiki/Source-code).

Because an embedded system is often composed of a wide variety of elements, the debugging strategy may vary. For instance, debugging a software- (and microprocessor-) centric embedded system is different from debugging an embedded system where most of the processing is performed by peripherals (DSP, FPGA, co-processor). An increasing number of embedded systems today use more than one single processor core. A common problem with multi-core development is the proper synchronization of software execution. In such a case, the embedded system design may wish to check the data traffic on the busses between the processor cores, which requires very low-level debugging, at signal/bus level, with a [logic analyzer](http://en.wikipedia.org/wiki/Logic_analyzer), for instance.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=11)]**Reliability**

Embedded systems often reside in machines that are expected to run continuously for years without errors, and in some cases recover by themselves if an error occurs. Therefore the software is usually developed and tested more carefully than that for personal computers, and unreliable mechanical moving parts such as disk drives, switches or buttons are avoided.

Specific reliability issues may include:

1. The system cannot safely be shut down for repair, or it is too inaccessible to repair. Examples include space systems, undersea cables, navigational beacons, bore-hole systems, and automobiles.
2. The system must be kept running for safety reasons. "Limp modes" are less tolerable. Often backups are selected by an operator. Examples include aircraft navigation, reactor control systems, safety-critical chemical factory controls, train signals, engines on single-engine aircraft.
3. The system will lose large amounts of money when shut down: Telephone switches, factory controls, bridge and elevator controls, funds transfer and market making, automated sales and service.

A variety of techniques are used, sometimes in combination, to recover from errors—both software bugs such as memory leaks, and also [soft errors](http://en.wikipedia.org/wiki/Soft_error) in the hardware:

* [watchdog timer](http://en.wikipedia.org/wiki/Watchdog_timer) that resets the computer unless the software periodically notifies the watchdog
* subsystems with redundant spares that can be switched over to
* software "limp modes" that provide partial function
* Designing with a [Trusted Computing Base](http://en.wikipedia.org/wiki/Trusted_Computing_Base) (TCB) architecture[[6]](http://en.wikipedia.org/wiki/Embedded_system#cite_note-5) ensures a highly secure & reliable system environment
* An [Embedded Hypervisor](http://en.wikipedia.org/wiki/Embedded_Hypervisor) is able to provide secure encapsulation for any subsystem component, so that a compromised software component cannot interfere with other subsystems, or privileged-level system software. This encapsulation keeps faults from propagating from one subsystem to another, improving reliability. This may also allow a subsystem to be automatically shut down and restarted on fault detection.
* [Immunity Aware Programming](http://en.wikipedia.org/wiki/Immunity_Aware_Programming)

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=12)]**High vs low volume**

For high volume systems such as [portable music players](http://en.wikipedia.org/wiki/Digital_audio_player) or [mobile phones](http://en.wikipedia.org/wiki/Mobile_phone), minimizing cost is usually the primary design consideration. Engineers typically select hardware that is just “good enough” to implement the necessary functions.

For low-volume or prototype embedded systems, general purpose computers may be adapted by limiting the programs or by replacing the operating system with a [real-time operating system](http://en.wikipedia.org/wiki/Real-time_operating_system).

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=13)]Embedded software architectures

There are several different types of software architecture in common use.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=14)]**Simple control loop**

In this design, the software simply has a loop. The loop calls subroutines, each of which manages a part of the hardware or software.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=15)]**Interrupt controlled system**

Some embedded systems are predominantly interrupt controlled. This means that tasks performed by the system are triggered by different kinds of events. An interrupt could be generated for example by a timer in a predefined frequency, or by a serial port controller receiving a byte.

These kinds of systems are used if event handlers need low latency and the event handlers are short and simple.

Usually these kinds of systems run a simple task in a main loop also, but this task is not very sensitive to unexpected delays.

Sometimes the interrupt handler will add longer tasks to a queue structure. Later, after the interrupt handler has finished, these tasks are executed by the main loop. This method brings the system close to a multitasking kernel with discrete processes.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=16)]**Cooperative multitasking**

A [nonpreemptive multitasking](http://en.wikipedia.org/wiki/Nonpreemptive_multitasking) system is very similar to the simple control loop scheme, except that the loop is hidden in an [API](http://en.wikipedia.org/wiki/Application_programming_interface). The programmer defines a series of tasks, and each task gets its own environment to “run” in. When a task is idle, it calls an idle routine, usually called “pause”, “wait”, “yield”, “nop” (stands for *no operation*), etc.

The advantages and disadvantages are very similar to the control loop, except that adding new software is easier, by simply writing a new task, or adding to the queue-interpreter.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=17)]**Preemptive multitasking or multi-threading**

In this type of system, a low-level piece of code switches between tasks or threads based on a timer (connected to an interrupt). This is the level at which the system is generally considered to have an "operating system" kernel. Depending on how much functionality is required, it introduces more or less of the complexities of managing multiple tasks running conceptually in parallel.

As any code can potentially damage the data of another task (except in larger systems using an [MMU](http://en.wikipedia.org/wiki/Memory_management_unit)) programs must be carefully designed and tested, and access to shared data must be controlled by some synchronization strategy, such as [message queues](http://en.wikipedia.org/wiki/Message_queue), [semaphores](http://en.wikipedia.org/wiki/Semaphore_%28programming%29) or a [non-blocking synchronization](http://en.wikipedia.org/wiki/Non-blocking_synchronization) scheme.

Because of these complexities, it is common for organizations to buy a [real-time operating system](http://en.wikipedia.org/wiki/Real-time_operating_system), allowing the application programmers to concentrate on device functionality rather than operating system services, at least for large systems; smaller systems often cannot afford the overhead associated with a *generic* real time system, due to limitations regarding memory size, performance, and/or battery life.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=18)]**Microkernels and exokernels**

A [microkernel](http://en.wikipedia.org/wiki/Microkernel) is a logical step up from a real-time OS. The usual arrangement is that the operating system kernel allocates memory and switches the CPU to different threads of execution. User mode processes implement major functions such as file systems, network interfaces, etc.

In general, microkernels succeed when the task switching and intertask communication is fast, and fail when they are slow.

[Exokernels](http://en.wikipedia.org/wiki/Exokernel) communicate efficiently by normal subroutine calls. The hardware, and all the software in the system are available to, and extensible by application programmers.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=19)]**Monolithic kernels**

In this case, a relatively large kernel with sophisticated capabilities is adapted to suit an embedded environment. This gives programmers an environment similar to a desktop operating system like[Linux](http://en.wikipedia.org/wiki/Linux) or [Microsoft Windows](http://en.wikipedia.org/wiki/Microsoft_Windows), and is therefore very productive for development; on the downside, it requires considerably more hardware resources, is often more expensive, and because of the complexity of these kernels can be less predictable and reliable.

Common examples of embedded monolithic kernels are [Embedded Linux](http://en.wikipedia.org/wiki/Embedded_Linux) and [Windows CE](http://en.wikipedia.org/wiki/Windows_CE).

Despite the increased cost in hardware, this type of embedded system is increasing in popularity, especially on the more powerful embedded devices such as [Wireless Routers](http://en.wikipedia.org/wiki/Routers) and [GPS Navigation Systems](http://en.wikipedia.org/wiki/Automotive_navigation_system). Here are some of the reasons:

* Ports to common embedded chip sets are available.
* They permit re-use of publicly available code for [Device Drivers](http://en.wikipedia.org/wiki/Device_driver), [Web Servers](http://en.wikipedia.org/wiki/Web_servers), [Firewalls](http://en.wikipedia.org/wiki/Firewall_%28networking%29), and other code.
* Development systems can start out with broad feature-sets, and then the distribution can be configured to exclude unneeded functionality, and save the expense of the memory that it would consume.
* Many engineers believe that running application code in user mode is more reliable, easier to debug and that therefore the development process is easier and the code more portable.
* Many embedded systems lack the tight real time requirements of a control system. Although a system such as Embedded Linux may be fast enough in order to respond to many other applications.
* Features requiring faster response than can be guaranteed can often be placed in [hardware](http://en.wikipedia.org/wiki/Programmable_logic_device).
* Many RTOS systems have a per-unit cost. When used on a product that is or will become a commodity, that cost is significant.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=20)]**Exotic custom operating systems**

A small fraction of embedded systems require safe, timely, reliable or efficient behavior unobtainable with the one of the above architectures. In this case an organization builds a system to suit. In some cases, the system may be partitioned into a "mechanism controller" using special techniques, and a "display controller" with a conventional operating system. A communication system passes data between the two.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=21)]**Additional software components**

In addition to the core operating system, many embedded systems have additional upper-layer software components. These components consist of networking protocol stacks like [CAN](http://en.wikipedia.org/wiki/Controller%E2%80%93area_network), [TCP/IP](http://en.wikipedia.org/wiki/TCP/IP), [FTP](http://en.wikipedia.org/wiki/FTP),[HTTP](http://en.wikipedia.org/wiki/HTTP), and [HTTPS](http://en.wikipedia.org/wiki/HTTPS), and also included storage capabilities like [FAT](http://en.wikipedia.org/wiki/File_Allocation_Table) and flash memory management systems. If the embedded devices has audio and video capabilities, then the appropriate drivers and codecs will be present in the system. In the case of the monolithic kernels, many of these software layers are included. In the RTOS category, the availability of the additional software components depends upon the commercial offering.

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=22)]See also

* [Communications server](http://en.wikipedia.org/wiki/Communications_server)
* [Cyber-physical system](http://en.wikipedia.org/wiki/Cyber-physical_system)
* [DSP](http://en.wikipedia.org/wiki/Digital_signal_processing)
* [Electronic Control Unit](http://en.wikipedia.org/wiki/Electronic_Control_Unit)
* [Embedded Hypervisor](http://en.wikipedia.org/wiki/Embedded_Hypervisor)
* [Embedded operating systems](http://en.wikipedia.org/wiki/Embedded_operating_system)
* [Embedded software](http://en.wikipedia.org/wiki/Embedded_software)
* [Firmware](http://en.wikipedia.org/wiki/Firmware)
* [Information appliance](http://en.wikipedia.org/wiki/Information_appliance)
* [Microprocessor](http://en.wikipedia.org/wiki/Microprocessor)
* [Microcontroller](http://en.wikipedia.org/wiki/Microcontroller)
* [Programming languages](http://en.wikipedia.org/wiki/Programming_language)
* [Real-time operating system](http://en.wikipedia.org/wiki/Real-time_operating_system)
* [Software engineering](http://en.wikipedia.org/wiki/Software_engineering)
* [System on a chip](http://en.wikipedia.org/wiki/System_on_a_chip)
* [System on module](http://en.wikipedia.org/wiki/System_on_module)
* [Ubiquitous computing](http://en.wikipedia.org/wiki/Ubiquitous_computing)

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=23)]References

1. [**^**](http://en.wikipedia.org/wiki/Embedded_system#cite_ref-barr-glossary_0-0) [Michael Barr](http://en.wikipedia.org/wiki/Michael_Barr_%28software_engineer%29). ["Embedded Systems Glossary"](http://www.netrino.com/Embedded-Systems/Glossary). *Netrino Technical Library*. Retrieved 2007-04-21.
2. [**^**](http://en.wikipedia.org/wiki/Embedded_system#cite_ref-1) Heath, Steve (2003). [*Embedded systems design*](http://books.google.com/books?id=BjNZXwH7HlkC&pg=PA2). EDN series for design engineers (2 ed.). Newnes. p. 2. [ISBN](http://en.wikipedia.org/wiki/International_Standard_Book_Number) [9780750655460](http://en.wikipedia.org/wiki/Special%3ABookSources/9780750655460). "An embedded system is a [microprocessor](http://en.wikipedia.org/wiki/Microprocessor) based system that is built to control a function or a range of functions."
3. [**^**](http://en.wikipedia.org/wiki/Embedded_system#cite_ref-2) Michael Barr; Anthony J. Massa (2006). ["Introduction"](http://books.google.com/books?id=nPZaPJrw_L0C&pg=PA1). *Programming embedded systems: with C and GNU development tools*. O'Reilly. pp. 1–2. [ISBN](http://en.wikipedia.org/wiki/International_Standard_Book_Number) [9780596009830](http://en.wikipedia.org/wiki/Special%3ABookSources/9780596009830).
4. [**^**](http://en.wikipedia.org/wiki/Embedded_system#cite_ref-microcontroller_3-0) [Giovino, Bill](http://en.wikipedia.org/w/index.php?title=Bill_Giovino_(Industry_Analyst)&action=edit&redlink=1). ["Microcontroller.com - Embedded Systems supersite"](http://www.microcontroller.com/).
5. [**^**](http://en.wikipedia.org/wiki/Embedded_system#cite_ref-4) [Embedded.com - Under the Hood: Robot Guitar embeds autotuning](http://www.embedded.com/underthehood/207401418) By David Carey, TechOnline EE Times (04/22/08, 11:10:00 AM EDT)Embedded Systems Design - Embedded.com
6. [**^**](http://en.wikipedia.org/wiki/Embedded_system#cite_ref-5) [Your System is secure? Prove it!](http://www.usenix.org/publications/login/2007-12/pdfs/heiser.pdf), Gernot Heiser, December 2007, Vol. 2 No. 6 Page 35-38, ;login: The USENIX Magazine

[[edit](http://en.wikipedia.org/w/index.php?title=Embedded_system&action=edit&section=24)]External links

|  |  |
| --- | --- |
| Search Wikimedia Commons | Wikimedia Commons has media related to: [***Embedded system***](http://commons.wikimedia.org/wiki/Category%3AEmbedded_system) |

|  |  |
| --- | --- |
| Search Wikibooks | Wikibooks has a book on the topic of[***Embedded Systems***](http://en.wikibooks.org/wiki/Embedded_Systems) |

|  |  |
| --- | --- |
| Search Wikiversity | Wikiversity has learning materials about [***Embedded System Engineering***](http://en.wikiversity.org/wiki/Embedded_System_Engineering) |

* [Designing Embedded Hardware](http://www.oreilly.com/catalog/dbhardware2/) John Catsoulis, O'Reilly, May 2005, [ISBN 0-596-00755-8](http://en.wikipedia.org/wiki/Special%3ABookSources/0596007558).