Embedded Operating system

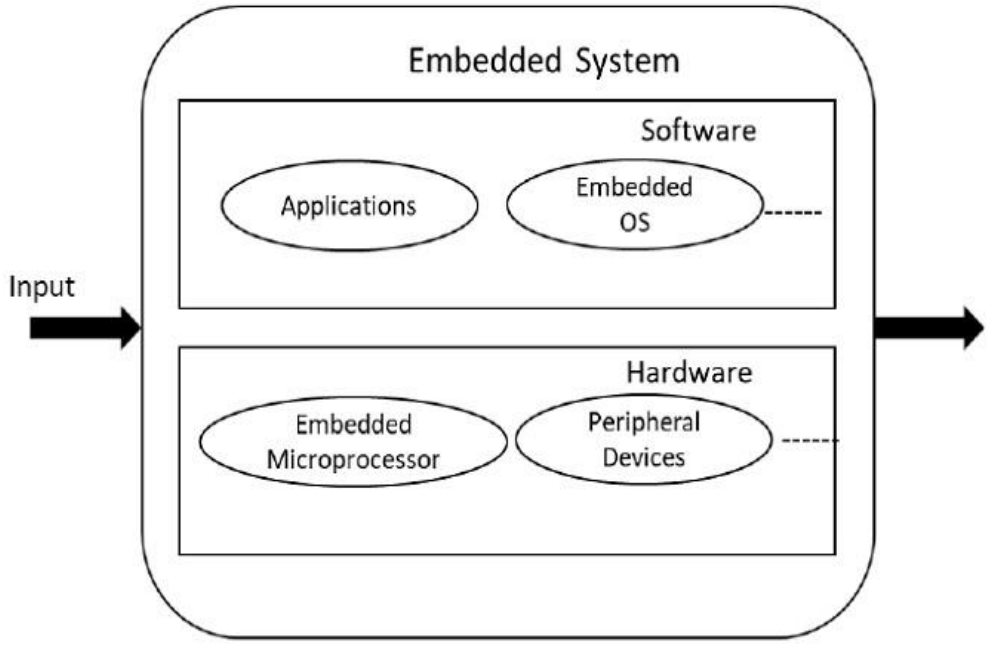
All Embedded Systems are task specific. They mostly do a particular task on loop/repeatedly for their entire lifetime. These systems are designed to execute their task within a particular time interval, and thus they have to be fast enough to be up to their time limit.

They have little or no user interface like a fully automatic washing machine does its task fully once its programmed is set and stops after its work is finished with almost no user interface.

They are built to achieve a particularly good efficiency level. They are very small in size operating system, need little power

These systems can't at all be upgraded or updated. Thus, they must be really high on efficiency and reliability as they can't be updated.

This operating system is shown below –



Characteristics of Embedded Operating System

There are various characteristics of an embedded operating system. Some of them are as follows:

1. It provides real-time operations.
2. Direct use of interrupts
3. Input/Output device flexibility
4. Reactive operation
5. Streamlined protection mechanisms
6. Configurability

Popular Embedded Operating Systems

There are various popular embedded operating systems. Some of them are as follows:

eCos

It stands for **'Embedded Configurable Operating System'**, and all of its components provide a wide range of configuration options. The eCos operating system may support a wide range of popular embedded CPUs.

mbed OS

It is a free and open-source embedded operating system that offers a systematic and comprehensive environment for intelligent hardware development.

VxWorks

Wind River Company firstly introduced it in **1983**. It is supported with task synchronization, memory efficiency management, and other features.

µC/OS-II:

It is introduced based on the **µC/OS** principle. **µC/OS-II** may handle **64** tasks and provide various functionalities such as interrupt services, task scheduling, memory management, synchronization, and time management.

FreeRTOS

It is a lightweight operating system that supports the priority scheduling algorithm. It provides various functionalities like memory management, message queue, task management, semaphore, time management, etc.

QNX

QNX was created in **1980** and is a commercial embedded real operating system that requires the POSIX specification to compile.

µ Clinux

It stands for **'Micro-Control Linux'**, and it is the latest version of embedded Linux. It is capable of grab all features of the Linux operating system.

Embedded Operating System Uses

The embedded operating system is commonly used in various areas, including car navigation systems, multimedia players, airplane navigation systems, and medical equipment.

Car navigation system

The car navigation system is a small computer system with a touch screen that enables the driver to navigate numerous menus such as audio playback, radio, GPS and route mapping, fuel level, hands-free calls, and tire pressure monitoring systems. All of these tasks are performed by the computer to improve the driving experience.

Parking Metering

Smart city parking meters use the embedded system to manage the user input and track time and costs. Depending on the design, these devices contain a variety of built-in functions. For example, some sensors detect vehicle entry and exit, while others require the driver to enter the parking space or vehicle license. A user interface offers the driver options, including defining the expected return time and paying appropriately.

Medical Equipment

Medical equipment automatically monitors bio constants, administers drugs. If the bio constants exceed or fall below a threshold value, it alerts the staff. As a result, it may help doctors treat the patients, monitor health issues, and save their lives.

The navigation system of a plane

The navigation system of a plane is a good instance of a real-time operating system. The main computer of an airplane is connected to most control systems such as the wing, engine, pressure controls, and safety. As a result, it is specifically built to work inside a plane and help with takeoff, landing, and emergency operations.

Advantages and disadvantages of Embedded Operating System

There are various advantages and disadvantages of an embedded operating system. Some of them are as follows:

Advantages

There are various advantages of an embedded operating system. Some of them are as follows:

1. It is small in size and faster to load.
2. It is low cost.
3. It is easy to manage.
4. It provides better stability.
5. It provides higher reliability.
6. It provides some interconnections.
7. It has low power consumption.
8. It helps to increase the product quality.

Disadvantages

There are various disadvantages of an embedded operating system. Some of them are as follows:

1. It isn't easy to maintain.
2. The troubleshooting is harder.

3. It has limited resources for memory.
4. It isn't easy to take a back of embedded files.
5. You can't change, improve, or upgrade an embedded system once it's been developed.
6. If any problem occurs, you need to reset the setting.
7. Its hardware is limited.

Classification of Embedded Systems

The classification of embedded systems can be done on the basis of their functionality and performance as given below.

Types of Embedded Systems Based on Functionality

Based on their functionality, embedded systems can be categorized as follows −

Mobile Embedded Systems

Mobile embedded systems are small-sized and portable embedded systems. These embedded systems are commonly used in smartphones, laptops, computers, digital cameras, smart watches, etc.

Real-Time Embedded Systems

Real-time embedded systems are designed to produce outputs in a definite time interval. These systems perform time-critical functions and are widely used in medical systems, industrial automation, traffic control systems, etc.

Networked Embedded Systems

Networked embedded systems are designed to uses in network connected systems such as security systems, point-of-sale systems, remote monitoring systems, etc.

Standalone Embedded Systems

Standalone embedded systems are designed to function independently without a host system or computer. Examples of this type of embedded systems include digital watches, MP3 players, calculators, etc.

Types of Embedded Systems Based on Performance Requirements

Based on their performance requirements, embedded systems can be classified into the following types −

Small-Scale Embedded Systems

Small-scale embedded systems are designed using 8-bit or smaller microprocessor or microcontroller. They are limited in terms of memory and processing power. However, these systems are cost-effective and are used in traffic controllers, toys, smart TV remote controls, smart cards, etc.

Medium-Scale Embedded Systems

Medium-scale embedded systems use 16-bit or 32-bit microprocessors or controllers. These systems are relatively more complex and faster than small-scale systems. They are commonly used in smart home appliances, medical equipment, and automation systems.

Sophisticated-Scale Embedded Systems

Sophisticated-scale embedded systems are also referred to as **complex embedded systems**. They are designed using 64-bit or larger microprocessors or controllers. These systems are powerful in terms of memory and processing capabilities. However, these are highly complex and more expensive. Common applications of these embedded systems include advanced medical equipment, robotics, safety critical systems, etc.

Modern Trends in Embedded systems

Embedded systems have now become the integral parts of all digital smart devices, ranging from a simple digital watch to complex robotic systems.

The following points highlight the contribution of embedded systems in modern world −

- Embedded systems are making real-time data processing more advanced and faster. This is an essential need in complex systems like drones and other aviation systems.
- Embedded systems are the soul of IoT devices like wearables, smart appliances, etc.
- Embedded systems are being empowered with modern technologies like artificial intelligence (AI) and machine learning (ML). These technologies provide self-decision-making capabilities to the embedded systems.
- Embedded systems are also being equipped with edge computing that reduces the delay and bandwidth by processing data closer to its source. This technology is very important in real-time applications.
- Embedded systems are providing with advanced networking technologies like 4G, 5G, etc. for more efficient data communication.
- Embedded systems are being also integrating with quantum computing for complex problem solving, optimized data processing, enhanced security, etc.
- Embedded systems are enabling more precise 3D printing and improving printing processes through real-time monitoring.

Applications of Embedded Systems

Embedded systems are the key components of all smart devices or systems. Some of the very common applications of embedded systems across various fields are listed here −

- **Automobiles** − In modern cars and vehicles, the embedded systems are used to perform various functions such as safety, navigation, infotainment, cruise control, engine health monitoring, and much more.
- **Smartphones** − In a typical smartphone, tons of embedded systems are used. These systems are responsible for performing different functions, from touch screen sensing to signal transmission, camera control, voice recording, detecting peripherals, and debugging.
- **Industries** − Embedded systems are essential components of robotics and automation systems in industries. They are used for processing data from sensors connected across industrial machinery and produce action instructions for their smooth operations.
- **Medical Equipment** − The functioning of advanced medical equipment like heart monitors, ventilators, etc. is dependent on embedded systems. In these devices, embedded systems

automate their operation and collect data from sensors and convert them into meaningful results, helping medical staff to avoid errors and interpret the patient's conditions accurately.

- **Wearables** − Embedded systems are also used in wearables like smartwatches, fitness bands, etc. These systems are entirely responsible for connecting these devices with other IoT devices like internet and smartphones.

Embedded Systems Terminology

There are some very important terms related to embedded systems, which are briefly defined in this section. These definitions will be very helpful for readers throughout this tutorial.

1. Embedded Processor

A microprocessor which is specially designed to use in embedded systems is referred to as an **embedded processor**. These processors are designed to provide excellent performance in terms of processing power, efficiency, and real-time operations.

2. Microcontroller

A microcontroller, also referred to as microcontroller unit (MCU), is small-sized computer implemented on a single IC chip.

A typical microcontroller consists of all essential components of a microcomputer such as microprocessor, memory unit, IO peripherals, and software. 8085 and 8086 are common examples of simple microcontrollers.

3. Microprocessor

In an embedded system, a **microprocessor** is the main processing unit that executes instructions and processes. It is an integrated circuit chip having data processing circuitry.

4. 8051 Microcontroller

8051 Microcontroller is a single chip microcontroller developed by Intel Corporation in 1980 to use in embedded systems. It is also known as **Intel MCS-51**.

The 8051 is an 8-bit microcontroller, as it can process 8-bits of data at a time. It is usual employed in embedded system used in remote controls, robotics, and telecom applications.

5. System-on-Chip (SoC)

System-on-a-Chip is an integrated circuit (IC) design that combines all the major components like processor, memory, input-output peripherals, etc. of an electronic device or system onto a single chip. It does not have any separate components mounted on a motherboard.

6. Architecture

The fundamental structure and design of an embedded system that defines how it will handle the instructions and data and how its components will communicate is referred to as **architecture** of the embedded system. The commonly used architectures of embedded systems are Harvard architecture and Neumann architecture.

7. I/O Programming

In **embedded systems I/O programming** is referred to as the process of exchanging data and instructions between the embedded system and external devices like sensors, displays, motors, etc.

8. Assembly Language

Assembly Language is a low-level language used in computer programming. In this programming language, the instructions are written by using abbreviated names equivalent to machine language codes. Assembly language is primarily used to write programs for microprocessors and microcontrollers.

9. Registers

In embedded systems, a **register** is nothing but a small and high-speed temporary storage device used for holding instructions and data required for processing.

10. Register Bank / Stack

In embedded systems, the **register bank** is nothing but a part of the RAM (Random Access Memory) used for storing program instructions. While, a register stack is a part in the RAM that temporarily stores information and access this information using a stack pointer register.

11. Instructions

Instructions are the computer codes that the microprocessor of a computer or embedded system can understand and execute. These are generally written in binary language using 0s and 1s.

12. Addressing Modes

The methods of specifying operands of instructions or locations of data in memory are referred to as **addressing modes**. Therefore, addressing modes define the rules for interpreting and manipulating the address fields of the operands of instructions before their actual execution. In embedded systems, immediate, direct, indirect, and indexed are some commonly used addressing modes.

13. Special Function Registers (SFRs)

Special function registers (SFRs) are those registers in an embedded system that monitor and control various aspects of the system operations. These registers are closely associated with some special functions and provide communication interfacing between the microprocessor and the peripherals.

14. Timer and Counter

Timer and counter are two important features of embedded systems. Timer is used in embedded systems for measuring time and creating time delays. Whereas, counters are used for counting events occurring external to the system

15. Interrupts

In **embedded systems interrupts** are the signals produced by the microprocessor to stop the currently executing code or programs. These signals are important in embedded systems to enable real-time responses.

16. Real-Time Operating System (RTOS)

Real-time operating systems (RTOS) are specially design operating systems for processing data and events in real-time. These operating systems are programmed to complete tasks within a given time constraint or based on the demand of the event. Hence, these are also known as **event-driven operating systems**. RTOS are widely used in embedded systems to perform tasks based on real-time events.