CHARACTERISTICS OF EMBEDDED SYSTEMS

Embedded systems exhibit a wide range of complexity levels, but they all share the common characteristic of being designed for specific tasks or functions.

- Time Specific
- Low Cost
- Easy Operation
- High efficiency
- Limited Memory
- Fault Tolerance
- High Stable
- Task Specific
- Minimal Interface
- Real Time
- Reliable
- Less Power

COMPONENTS OF EMBEDDED SYSTEMS

An embedded system is a combination of both hardware and software modules. Both computer software and embedded system software are different in terms of their purposes. The computer software can be installed on many devices to achieve the required goals while embedded system software, on the other hand, is specifically written for a certain device to meet a certain goal.

❖ HARDWARE (EMBEDDED SYSTEM HARDWARE)

The hardware provides the necessary computational power, connectivity, and interfacing capabilities for the system.

- Power Supply
  - A power supply is a crucial component of the embedded system design.
  - It is an electrical device mainly used to power up the electrical load.
  - To work the embedded system properly, a smooth and efficient power supply is needed. Both wall adopter and battery can be used as a power

supply. Some power supplies work as independent equipment while others are incorporated into the embedded technology they power.

- Micro Controller
  - An embedded system is either a microcontroller-based or microprocessor-based system. They give a system computing power and are called integrated circuits.
  - The embedded hardware performance is mainly dependent on the processor which is normally called the brain of the embedded system.
- Memory
  - Memory is essential to store important information in the embedded computer system.
  - Memory is integrated into a microcontroller or microprocessor.
- Timer/Counter
  - Sometimes you need to create a delay before a specific function. Timers are used in such cases. While at times you want to count the number of times a particular event occurs. Counters are used in such cases. If an up counter is used in the system, it will count up from the initial value to 0xFF and if it is down counter, it will count down to 0x00. The counters are integrated using register-type circuits like a flip-flop.
- Communication Port
  - Communication ports are used in embedded systems to establish communication with other embedded systems. There are several communication ports including USB, UART, USB, I2C, SPI, and RS-485. For simple applications, communications ports are utilized from the microcontroller, and for complex and advanced applications these ports are externally installed inside the embedded systems.

- Output/Input
  - Input is required to interact with the embedded system. A sensor can be used to provide input to the system. The microcontroller used in the system can be configured as an input or output port. In the microcontroller, there are a fixed number of input and output ports that you can utilize as per your requirement.

❖ SOFTWARE (EMBEDDED SYSTEM SOFTWARE)

Embedded systems require software that is specifically developed to meet the requirements of the intended application. This application-specific software controls the behavior and functionality of the embedded system, enabling it to perform its designated tasks. The software is typically designed to optimize resource usage and ensure efficient operation within the embedded system's constraints.

- EDITOR
  - The editor is the first tool you required for embedded system software.
  - The code you write in C and C++ programming languages will be saved in a text file in the editor.
  - Geany editor is a great example of a text editor.
  - This editor supports scores of languages including Java, C, HTML, Python, PHP, Pascal, and Pearl.
- COMPILER
  - A compiler is used to turn this written code into low-level machine language that the machine can comprehend.
- ASSEMBLER
  - The assembler tool converts the written code into a machine language. It is slightly different than a compiler.
  - The compiler directly converts the written code into machine language while the assembler, on the other hand, first converts source code to object code and then to the language that the machine can understand.
- EMULATOR
  - The main task of the emulator is to make the embedded system act like a real system in a simulation environment.
  - Using an emulator, you'll get an idea of how the code will function in real-time. It is used to simulate software performance, and it helps in achieving the ideal performance of the written code.
  - With an emulator, you can run one operating system into another device. For example, using an emulator you can run Mac operating system into your windows operating system.
- LINKER
  - Typically, software code is written in small modules and pieces.

- A linker, also called a link editor, is a tool that takes one or more object files and combines them to develop a single executable code.

- DEBUGGER
  - A debugger is a tool used for testing and debugging purposes. It scans the code thoroughly and removes the errors and bugs, and identifies the places where they occur.
  - Programmers can quickly address the errors and fix them.

❖ **Real-Time Operating System (RTOS)**

An RTOS is a specialized operating system designed for real-time applications. It provides services and functions that facilitate real-time task scheduling, prioritization, and resource management. The RTOS ensures that time-critical tasks are executed within specified time constraints, allowing the embedded system to respond to events or inputs in a timely and deterministic manner.

These three components—hardware, application-specific software, and an RTOS—work together to enable the operation and functionality of embedded systems. The hardware provides the physical platform, the application-specific software defines the behavior and functionality, and the RTOS provides the necessary real-time capabilities and task management for the system to operate effectively in time-sensitive environments.