Al-Mustaqbal University
College of Sciences
Intelligent Medical System Department

# Embedded systems

# Lecture 3:Embedded System Basics and Application


Prof.Dr. Mehdi Ebady Manaa

جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

# Types of embedded systems

- When considering performance and functional requirements, embedded systems are categorized into real-time embedded systems, standalone embedded systems, networked embedded systems, and mobile embedded systems.
- **Real-time embedded systems** prioritize prompt output generation and can be classified as soft real-time (lenient deadlines) or hard real-time (strict deadlines).
- **Standalone embedded systems** can function independently without a host computer.
- **Networked embedded systems** rely on network connections and communication for output generation.
- **Mobile embedded systems** refer to small, portable devices such as smartphones and laptops.

# Examples of Embedded Systems

- Digital watches
- Washing Machine
- Toys
- Televisions
- Digital phones
- Laser Printer
- Cameras
- Industrial machines
- Electronic Calculators
- Automobiles
- Medical Equipment

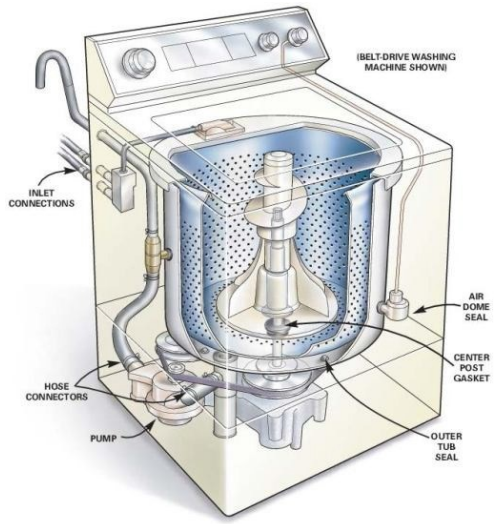
# WATCH

- **It is a time display SYSTEM**
- **Parts: Hardware, Needles, Battery, Dial, Chassis and Strap**


- **Rules**

1. All needles move clockwise only
2. A thin needle rotates every second
3. A long needle rotates every minute
4. A short needle rotates every hour
5. All needles return to the original position after 12 hours

- WASHING MACHINE
- **It is an automatic clothes washing SYSTEM**
- **Parts: Status display panel, Switches & Dials, Motor, Power supply & control unit, Inner water level sensor and**
- **solenoid valve.**




- **Rules**
- 1.Wash by

spinning 2.Rinse

- 3.Drying

4.Wash over by blinking

5.Each step display the process stage

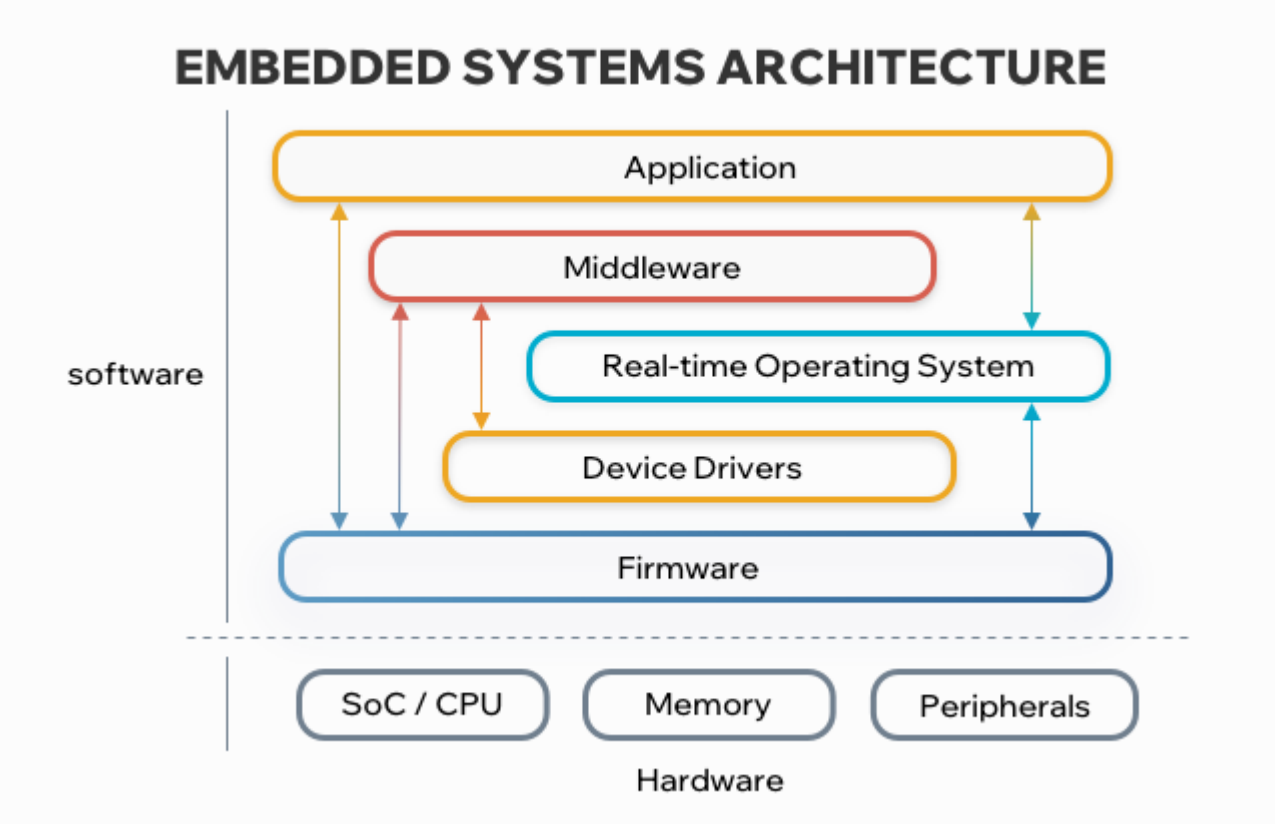
6.In case interruption, execute only the remaining

# Application of Embedded System

- Home appliances
- Transportation
- Health care
- Business sector & offices
- Defense sector
- Aerospace
- Agricultural Sector

# Components of Embedded Systems

- 1. Hardware
- 2. Software
- 3. Firmware



# Components of Embedded Systems

- The components of embedded systems consist of hardware and software elements that work together to enable the desired functionality of the system.
- **Hardware components of embedded systems**
- The hardware components of embedded systems encompass various physical elements that comprise the system infrastructure. These include power supply, microcontrollers and microprocessors, memory, timers and counters, communication interfaces, input/output, and electrical circuits, all of which work together to enable the desired functionality of the embedded system.

# COMPUTER HARDWARE

- A Microprocessor
- A Large Memory (Primary and Secondary) (RAM, ROM and caches)
- Input Units
  (Keyboard, Mouse, Scanner, etc.)
- Output Units
  (Monitor, printer, etc.)
- Networking Units
  (Ethernet Card, Drivers, etc.)
- I/O Units
  (Modem, Fax cum Modem, etc.)

# Microprocessors and Microcontrollers in Embedded Systems

- A **microcontroller** is a functional computer system-on-a-chip. It contains a processor, memory, and programmable input/output peripherals.
- Microcontrollers include an integrated CPU, memory (a small amount of RAM, program memory, or both) and peripherals capable of input and output.

Microprocessors and microcontrollers are the brains of embedded systems. They are responsible for executing instructions and controlling system operation. A microprocessor is a general-purpose central processing unit (CPU) that performs arithmetic, logic, control, and input/output operations. A microcontroller, on the other hand, is a specialized integrated circuit (IC) that combines a microprocessor core with memory and input/output peripherals.

# Microprocessors and Microcontrollers in Embedded Systems

- Microprocessors and microcontrollers are the brains of embedded systems. They are responsible for executing instructions and controlling system operation. A microprocessor is a general-purpose central processing unit (CPU) that performs arithmetic, logic, control, and input/output operations. A microcontroller, on the other hand, is a specialized integrated circuit (IC) that combines a microprocessor core with memory and input/output functionality. Microprocessors and microcontrollers have different architectures and different features. The choice of microprocessor or microcontroller depends on the specific requirements of the embedded system. Common microprocessor architectures include ARM, x86, and MIPS, and common microcontroller families include PIC, AVR, and Arduino.ipherals.

# VARIOUS MICROCONTROLLERS

- INTEL
  - 8031,8032,8051,8052,8751,8752
- PIC
  - 8-bit PIC16, PIC18,
  - 16-bit DSPIC33 / PIC24, PIC16C7x
- Motorola
  - MC68HC11

# MICROPROCESSOR Vs MICROCONTROLLER

| MICROPROCESSOR | MICROCONTROLLER |
|---|---|
| The functional blocks are ALU, registers, timing & control units | It includes functional blocks of microprocessors & in addition has timer, parallel i/o, RAM, EPROM, ADC & DAC |
| Bit handling instruction is less, One or two type only | Many type of bit handling instruction |
| Rapid movements of code and data between external memory & MP | Rapid movements of code and data within MC |
| It is used for designing general purpose digital computers system | They are used for designing application specific dedicated systems |

# Memory Devices in Embedded Systems

- Storage devices play an important role in embedded systems by storing program instructions and data. There are two main types of memory used in embedded systems: volatile memory and non-volatile memory. B. Random Access Memory (RAM), such as volatile memory. Temporarily stores data and program instructions during system operation. When the power is turned off, data stored in volatile memory is lost. Nonvolatile memory, on the other hand, stores data even when power is removed.
- Read-only memory (ROM) is commonly used in embedded systems to store system firmware or persistent software. This allows the software to remain intact even if power is removed. Additionally, flash memory is often used in embedded systems to store program code and data that can be modified at runtime.

# Input and Output Devices in Embedded Systems

- Input/output devices allow embedded systems to interact with the outside world. These devices allow users to enter input into the system and receive output or feedback. Common input devices for embedded systems include buttons, switches, sensors, and keyboards. These devices capture user input and convert it into a format that the system can understand.
- Output devices, on the other hand, send information from embedded systems to users or other devices. Examples of output devices include displays, LEDs, buzzers, and actuators. These devices allow the system to provide visual, audio, or physical feedback to the user or to control external devices.

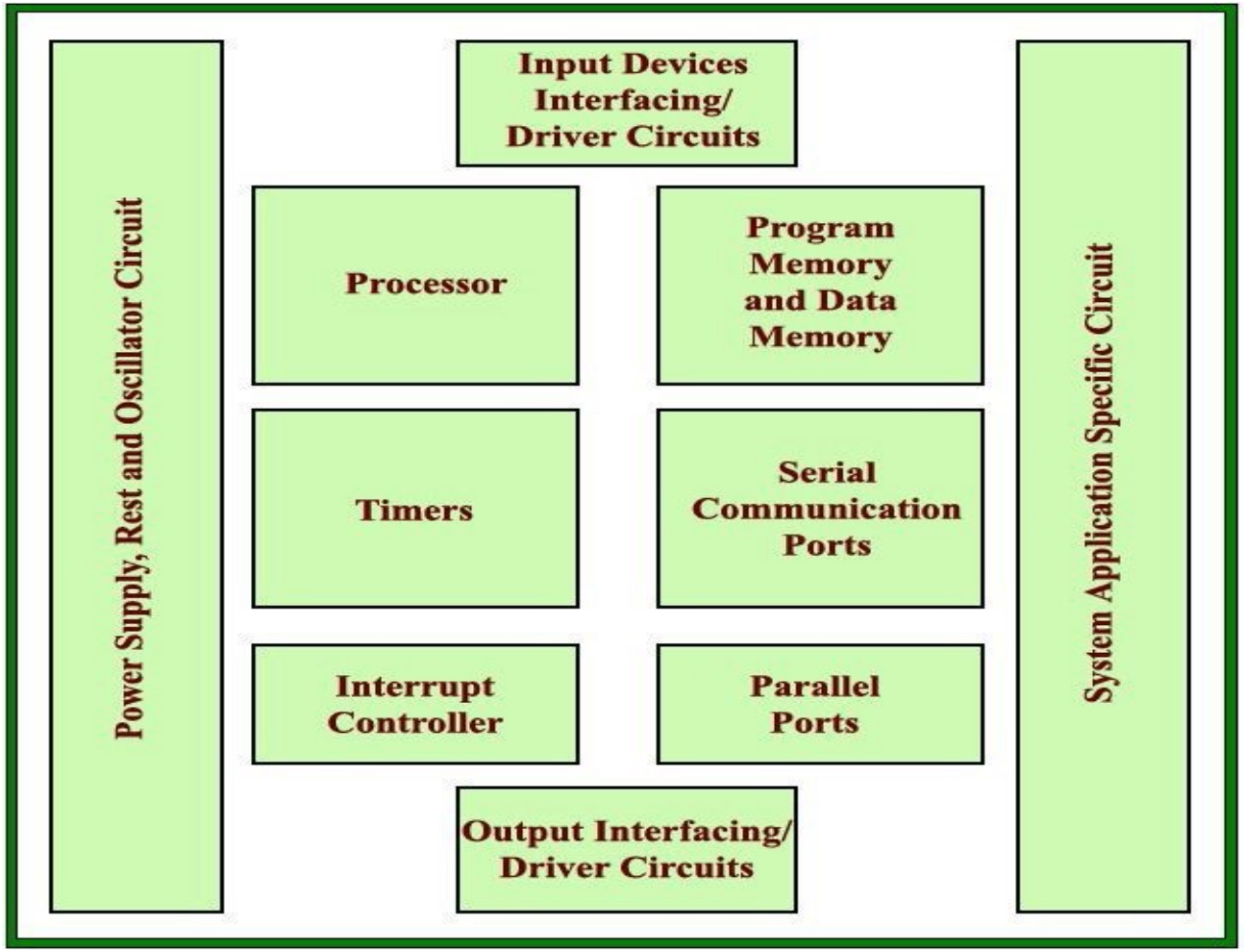
# Communication Interfaces in Embedded Systems

- Embedded systems often need to communicate with other systems or devices, either locally or over a network. Communication interfaces facilitate this data exchange. Communication interfaces commonly used in embedded systems include Universal Serial Bus (USB), Integrated Circuit-to-Integrated Circuit (I2C), Serial Peripheral Interface (SPI), and Ethernet.
- These interfaces allow embedded systems to connect to external devices such as sensors, actuators, and displays. It also enables communication between multiple embedded systems, allowing them to collaborate and share data. The choice of communication interface depends on factors such as data transfer speed, distance, and compatibility with other devices.

# COMPONENTS OF EMBEDDED SYSTEM

- **It has Hardware**
  - Processor, Timers, Interrupt controller, I/O Devices, Memories, Ports, etc.
- **It has main Application Software**
  - Which may perform concurrently the series of tasks or multiple tasks.
- **It has Real Time Operating System (RTOS)**
  - RTOS defines the way the system work. Which supervise the application software. It sets the rules during the execution of the application program. A small scale embedded system may not need an RTOS.

# EMBEDDED SYSTEM HARDWARE



# EMBEDDED SYSTEM CONSTRAINTS

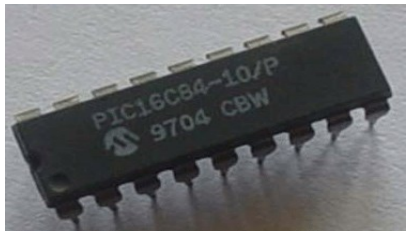
- An embedded system is software designed to keep in view three constraints:
  - Available system memory
  - Available processor speed
  - The need to limit the power dissipation
- When running the system continuously in cycles of wait for events, run, stop and wakeup.

## What makes embedded systems different?

- Real-time operation
- size
- cost
- time
- reliability
- safety
- energy
- security

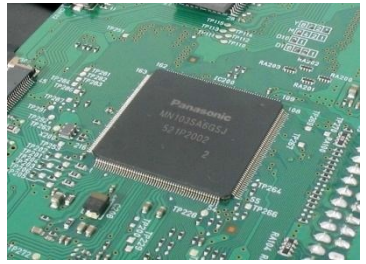
# CLASSIFICATIONS OF EMBEDDED SYSTEM

1. Small Scale Embedded System




2. Medium Scale Embedded System


3. Sophisticated Embedded System



# SMALL SCALE EMBEDDED SYSTEM

- Single 8 bit or 16bit Microcontroller.
- Little hardware and software complexity.


- They May even be battery operated.
- Usually “C” is used for developing these system.
- The need to limit power dissipation when system is running continuously.

- Programming tools:
  - Editor, Assembler and Cross Assembler

# MEDIUM SCALE EMBEDDED SYSTEM

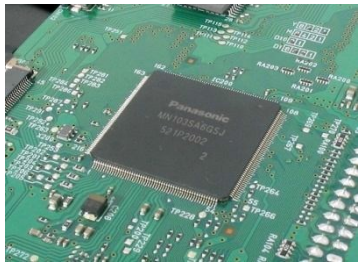
- Single or few 16 or 32 bit microcontrollers or Digital Signal Processors (DSP) or Reduced Instructions Set Computers (RISC).
- Both hardware and software complexity.


- Programming tools:
  - RTOS, Source code Engineering Tool, Simulator, Debugger and Integrated Development Environment (IDE).

# SOPHISTICATED EMBEDDED SYSTEM

- **Enormous hardware and software complexity**
- **Which may need scalable processor or configurable processor and programming logic arrays.**

- **Constrained by the processing speed available in their hardware units.**



- **Programming Tools:**
- **For these systems may not be readily available at a reasonable cost or may not be available at all. A compiler or retargetable compiler might have to be developed for this.**

# Software Components of Embedded Systems

- While hardware components form the foundation of embedded systems, software components bring them to life. Embedded software is responsible for controlling and coordinating hardware components to perform specific tasks. Let’s take a closer look at his two important software components: embedded operating systems and programming languages.
- **SOFTWARE**
- **SIMULATOR**
- **COMPILER**

# Software Components of Embedded Systems

- **Embedded Operating Systems :**An embedded operating system (OS) is specialized software that manages embedded system resources and provides an interface for running applications. Unlike general-purpose operating systems such as Windows or Linux, embedded operating systems are lightweight, efficient, and optimized for specific hardware.
- **Power Supply**: Reliable power supplies are essential for embedded systems to function properly. Depending on the application requirements, it can be powered via batteries, AC grid, or renewable energy sources. Power management circuits regulate voltage levels, manage power consumption, and ensure system stability, especially in battery-powered devices where energy efficiency is paramount.
- **Real-Time Operating System (RTOS) :**Many embedded systems use real-time operating systems to efficiently manage tasks, schedules, and resource allocation. RTOS provides deterministic behavior to ensure timely execution of critical tasks and response to external events. Features such as task prioritization, intertask communication, and synchronization make it ideal for applications that require precise timing and control.

# Software Components of Embedded Systems

- **Development Tools**

1. **Integrated Development Environments (IDEs)**: Provide a comprehensive environment for writing, compiling, debugging, and testing code.
2. **Compilers**: Convert high-level programming languages into machine code specific to the target hardware.
3. **Debuggers**: Help identify and fix errors in the code.
4. **Simulators**: Allow testing of embedded software in a virtual environment before deploying it on actual hardware.

- These tools are essential for ensuring the stability, performance, and maintainability of embedded software.
- **Embedded Operating Systems (OS)**

- **Embedded OS**: Manages system resources and provides an interface for running applications. Unlike general-purpose OS like Windows or Linux, embedded OS are lightweight and optimized for specific hardware.
- **Functions**: Include task scheduling, memory management, device drivers, and communication protocols.
- **Common Embedded OS**: FreeRTOS, VxWorks, Embedded Linux, and Android.

# Programming Languages for Embedded Systems

1. **C**:
    1. Most widely used in embedded systems development.
    2. Offers low-level hardware access, efficient code execution, and a wide range of libraries and tools.
    3. Known for its portability and suitability for resource-constrained systems.
2. **C++**:
    1. An extension of C with additional features like object-oriented programming.
    2. Provides higher-level abstractions for easier software development while retaining the benefits of C.
    3. Commonly used in systems where object-oriented design principles are desirable.
3. **Assembly Language**:
    1. A low-level language that directly corresponds to machine instructions.
    2. Provides the highest level of control over hardware but requires expertise and is less portable.
    3. Used in systems where performance optimization is critical.

# Functions of Embedded Systems

**1.Control and Automation:**

- •Automate processes in industrial machinery, consumer electronics, automotive systems, and smart infrastructure.
- •Monitor sensors, analyze data, and activate actuators to maintain desired conditions and respond to changes.

**2.Data Acquisition and Processing**:

- •Collect and process data from sensors in real-time.
- •Provide actionable insights for decision-making in various fields like environmental monitoring and healthcare.

**3.Communication and Networking**:

- •Facilitate device communication and networking.
- •Support data exchange, remote monitoring, firmware updates, and device management over wired and wireless networks.

**4.User Interface and Interaction**:

- •Provide intuitive interfaces such as displays, touch screens, buttons, and voice recognition.
- •Allow users to interact with and configure the system easily.

**5.Security and Encryption**:

- •Implement measures to protect data and prevent unauthorized access.
- •Use encryption, authentication, access control, and secure boot mechanisms to ensure software and firmware integrity.

# Significance of Embedded Systems

**1.Efficiency and Reliability**:
- •Optimized for specific tasks, offering higher efficiency and reliability.
- •Designed to operate under strict constraints like limited resources and harsh conditions.

**2.Miniaturization and Integration**:
- •Advances in technology have enabled the creation of compact, multifunctional devices.
- •Drives the development of wearables, IoT devices, medical implants, and more.

**3.Customization and Flexibility**:
- •Can be tailored to specific application requirements.
- •Provides flexibility and scalability, optimizing performance, cost, and time to market.

**4.Enabling Emerging Technologies**:
- •Support technologies like AI, machine learning, augmented reality, and IoT.
- •Provide the necessary computing infrastructure and connectivity.

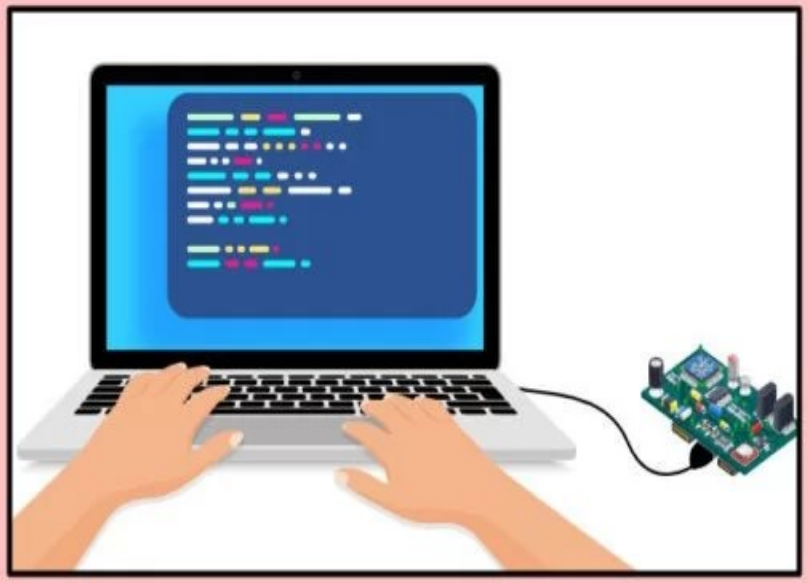
**5.Economic and Societal Impact**:
- •Enhance productivity, efficiency, and quality of life.
- •Enable automation, optimization, and innovation across industries, creating new opportunities.

Embedded systems are integral to modern technology, driving innovation and progress across various industries. Understanding their components and functions is crucial for designing and maintaining these intelligent systems.

# Firmware of Embedded System

- Embedded firmware is a specific type of computer software programmed on a hardware device that provides low-level control for a device's specific hardware. Firmware is implemented in non-volatile memory such as read-only memory (ROM) or electrically erasable programmable read-only memory (EEPROM) that cannot be easily modified or erased – hence the name "firm" – and is generally not intended to change once shipped. However, in more recent times, firmware is now commonly stored in Flash memory devices which is much easier to erase and reprogram and has greater storage capacity than its ROM/EEPROM predecessors.
- The primary responsibility of the firmware is to boot a device and to supply the instructions for the device to function and communicate with various hardware parts. Essentially, the firmware is the code that runs on a piece of hardware and controls a processor and peripherals embedded within a larger device.

# Embedded Firmware



- Firmware is a layer of software on top of which operating systems and other applications run.
- Firmware is usually found in general purpose computing devices like smartphones, PC's , laptops etc.
- The firmware does not include the end application.
- The firmware is not the only software that runs on the system.
- Firmware is stored in a flash.

# Embedded Software

- Embedded software is a software capable of running the entire system which might or might not include an Operating system.
- Embedded software is usually found on special purpose computing devices like embedded systems.
- Embedded software includes the end application.
- Embedded software will be the only software that runs on the system.
- Embedded software is also stored.

# Thank You