Chapter 1

Introduction to Embedded Systems

CHAPTER OBJECTIVES

After reading this chapter, you will be able to:

- Understand what is an embedded system
- List the various application areas in which embedded systems are used
- Categorize embedded systems
- Gain knowledge of the architecture of an embedded system
- Understand the requirements of embedded systems
- Learn about the recent trends in embedded system development

Embedded systems are omnipresent. We find them everywhere—at our homes, in our offices, in shopping malls, in hospitals, in cars, in aircraft and so on. In this introductory chapter, we will define an embedded system and study the various market segments for the embedded systems. We will study how the embedded systems can be divided into different categories based on their functionality. We will discuss the architecture of an embedded system and the special requirements of these systems. We will also discuss the recent trends in embedded system development.

1.1 What is an Embedded System?

An embedded system can be defined as a computing device that does a specific focused job. Appliances such as the air-conditioner, VCD player, DVD player, printer, fax machine, mobile phone etc. are examples of embedded systems. Each of these appliances will have a processor and special hardware to meet the specific requirement of the application along with the embedded software that is executed by the processor for meeting that specific requirement. The embedded software is also called "firmware".

The desktop/laptop computer is a general purpose computer. You can use it for a variety of applications such as playing games, word processing, accounting, software development and so on. In contrast, the software in the embedded systems is always fixed.

Embedded systems are characterized by some special features listed below:

- Embedded systems do a very specific task, they cannot be programmed to do different things.
- Embedded systems have very limited resources, particularly the memory. Generally, they do not have secondary storage devices such as the CDROM or the floppy disk.
- Embedded systems have to work against some deadlines. A specific job has to be completed within a specific time. In some embedded systems, called real-time systems, the deadlines are stringent. Missing a deadline may cause a catastrophe—loss of life or damage to property.
- Embedded systems are constrained for power. As many embedded systems operate through a battery, the power consumption has to be very low.
- Embedded systems need to be highly reliable. Once in a while, pressing ALT-CTRL-DEL is OK on your desktop, but you cannot afford to reset your embedded system.
- Some embedded systems have to operate in extreme environmental conditions such as very high temperatures and humidity.
- Embedded systems that address the consumer market (for example, electronic toys) are very cost-sensitive. Even a reduction of \$0.1 is lot of cost saving, because thousands or millions systems may be sold.
- Unlike desktop computers in which the hardware platform is dominated by Intel and the operating system is dominated by Microsoft, there is a wide variety of processors and operating systems for the embedded systems. So, choosing the right platform is the most complex task.

In Brief...

An embedded system is a computing device that does a specific job. Both the hardware and software in an embedded system are optimized for that specific job.

1.3 Categories of Embedded Systems

Based on functionality and performance requirements, embedded systems can be categorized as:

- Stand-alone embedded systems
- Real-time systems
- Networked information appliances
- Mobile devices

In Brief...

Embedded systems can be categorized as: (a) stand-alone systems (b) real-time systems (c) networked information appliances, and (d) mobile devices.

1.3.1 Stand-alone Embedded Systems

As the name implies, stand-alone systems work in stand-alone mode. They take inputs, process them and produce the desired output. The input can be electrical signals from transducers or commands from a human being such as the pressing of a button. The output can be electrical signals to drive another system, an LED display or LCD display for displaying of information to the users. Embedded systems used in process control, automobiles, consumer electronic items etc. fall into this category. In a process control system, the inputs are from sensors that convert a physical entity such as temperature or pressure into its equivalent electrical signal. These electrical signals are processed by the system and the appropriate electrical signals are produced using which an action is taken such as opening a valve. A few embedded systems used at home are shown in Fig. 1.1

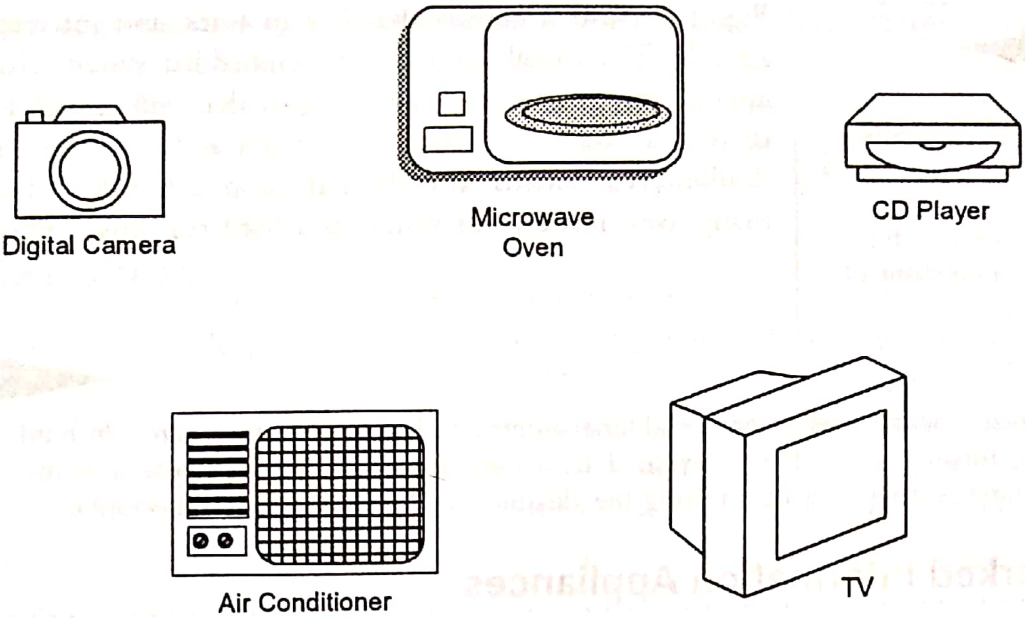

Fig. 1.1: Stand-alone Embedded Systems used at Home

1.3.2 Real-Time Systems

Embedded systems in which some specific work has to be done in a specific time period are called real-time systems. For example, consider a system that has to open a valve within 30 milliseconds when the humidity crosses a particular threshold. If the valve is not opened within 30 milliseconds, a catastrophe may occur. Such systems with strict deadlines are called hard real-time systems. In some embedded systems, deadlines are imposed, but not adhering to them once in a while may not lead to a catastrophe. For example, consider a DVD player. Suppose, you give a command to the DVD player from a remote control, and there is a delay of a few milliseconds in executing that command. But, this delay won't lead to a serious implication. Such systems are called soft real-time systems.

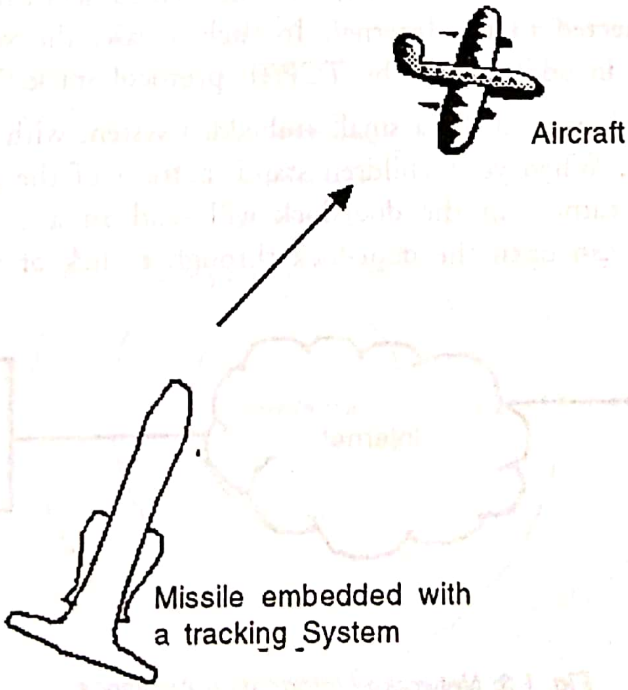

Fig. 1.2: Hard Real-Time Embedded System

In Brief...

Real-time embedded systems have to complete a specific task in a specified time period. Meeting the deadlines is the most important requirement of real-time systems.

Fig. 1.2 shows a missile that has to track and intercept an enemy aircraft. The missile contains an embedded system that tracks the aircraft and generates a control signal that will launch the missile. If there is a delay in tracking the aircraft and if the missile misses the deadline, the enemy aircraft may drop a bomb and cause loss of many lives. Hence, this system is a hard real-time embedded system.

Notes...

Real-time systems are categorized as hard real-time systems and soft real-time systems. In hard real-time systems, missing a deadline may lead to a catastrophe. In soft real-time systems, meeting the deadline is important but missing the deadline will not lead to a catastrophe.

1.3.3 Networked Information Appliances

Embedded systems that are provided with network interfaces and accessed by networks such as Local Area Network or the Internet are called networked information appliances. Such embedded systems are connected to a network, typically a network running TCP/IP (Transmission Control Protocol/ Internet Protocol) protocol suite, such as the Internet or a company's Intranet. These systems have emerged in recent years. These systems run the protocol TCP/IP stack and get connected either through PPP or Ethernet to a network and communicate with other nodes in the network. Here are some examples of such systems:

- A networked process control system consists of a number of embedded systems connected as a Local Area Network. Each embedded system can send real-time data to a central location from where the entire process control system can be monitored. The monitoring can be done using a web browser such as the Internet Explorer.
- A web camera can be connected to the Internet. The web camera can send pictures in real-time to any computer connected to the Internet. In such a case, the web camera has to run the HTTP server software in addition to the TCP/IP protocol stack.
- The door-lock of your home can be a small embedded system with TCP/IP and HTTP server software running on it. When your children stand in front of the door-lock after they return from school, the web camera in the door-lock will send an alert to your desktop over the Internet and then you can open the door-lock through a click of the mouse.

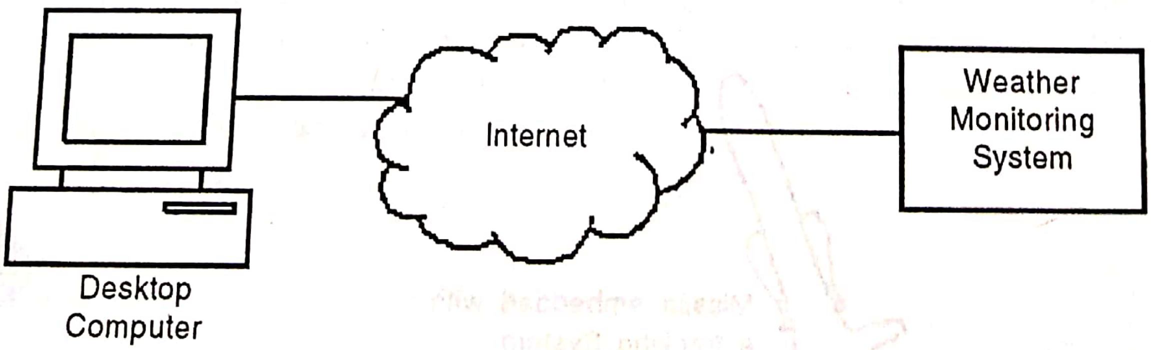

Fig. 1.3: Networked Information Appliance

Fig. 1.3 shows a weather monitoring system connected to the Internet. TCP/IP protocol suite and HTTP web server software will be running on this system. Any computer connected to the Internet can access this system to obtain real-time weather information. The networked information appliances need to run the complete TCP/IP protocol stack including the application layer protocols. If the appliance has to provide information over the Internet, HTTP web server software also needs to run on the system.

1.3.4 Mobile Devices

Mobile devices such as mobile phones, Personal Digital Assistants (PDAs), smart phones etc. are a special category of embedded systems. Though the PDAs do many general-purpose tasks, they need to be designed just like the 'conventional' embedded systems. The limitations of the mobile devices—memory constraints, small size, lack of good user interfaces such as full-fledged keyboard and display etc.—are same as those found in the embedded systems discussed above. Hence, mobile devices are considered as embedded systems. However, the PDAs are now capable of supporting general-purpose application software such as word processors, games, etc.

1.4 Overview of Embedded System Architecture

Every embedded system consists of custom-built hardware built around a Central Processing Unit (CPU). This hardware also contains memory chips onto which the software is loaded. The software residing on the memory chip is also called the 'firmware'. The embedded system architecture can be represented as a layered architecture as shown in Fig. 1.4. The operating system runs above the hardware, and the application software runs above the operating system. The same architecture is applicable to any computer including a desktop computer. However, there are significant differences. It is not compulsory to have an operating system in every embedded system. For small appliances such as remote control units, air-conditioners, toys etc., there is no need for an operating system and you can write only the software specific to that application. For applications involving complex processing, it is advisable to have an operating system. In such a case, you need to integrate the application software with the operating system and then transfer the entire software on to the memory chip. Once the software is transferred to the memory chip, the software will continue to run for a long time and you don't need to reload new software.

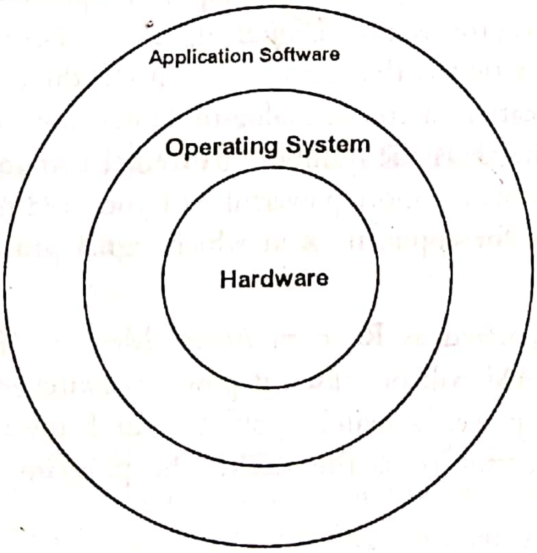

Fig. 1.4: Layered Architecture of an Embedded System

Now, let us see the details of the various building blocks of the hardware of an embedded system. As shown in Fig. 1.5, the building blocks are:

- Central Processing Unit (CPU)
- Memory (Read-only Memory and Random Access Memory)
- Input Devices
- Output devices
- Communication interfaces
- Application-specific circuitry

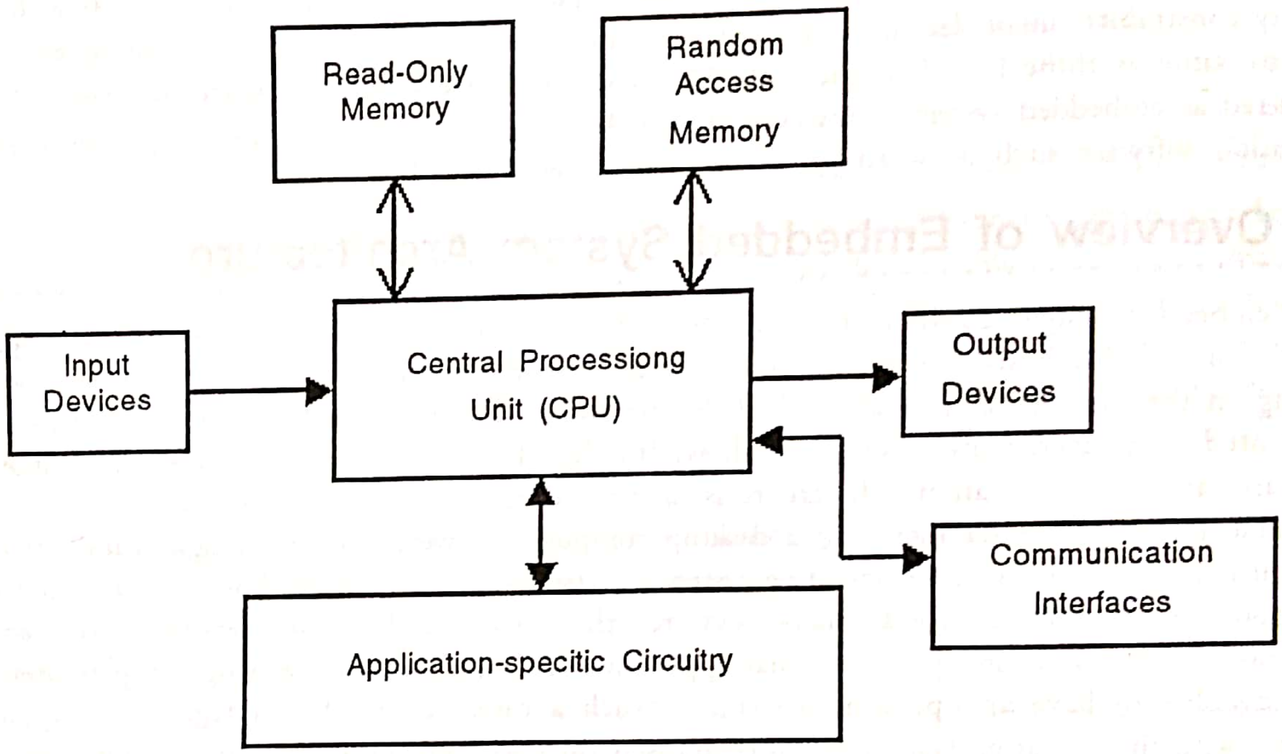

Fig. 1.5: Simplified Hardware Architecture of an Embedded System

Central Processing Unit (CPU): The Central Processing Unit (processor, in short) can be any of the following: micro-controller, microprocessor or Digital Signal Processor (DSP). A micro-controller is a low-cost processor. Its main attraction is that on the chip itself, there will be many other components such as memory, serial communication interface, analog-to-digital converter etc. So, for small applications, a micro-controller is the best choice as the number of external components required will be very less. On the other hand, microprocessors are more powerful, but you need to use many external components with them. DSP is used mainly for applications in which signal processing is involved such as audio and video processing.

Memory: The memory is categorized as Random Access Memory (RAM) and Read Only Memory (ROM). The contents of the RAM will be erased if power is switched off to the chip, whereas ROM retains the contents even if the power is switched off. So, the firmware is stored in the ROM. When power is switched on, the processor reads the ROM, the program is transferred to RAM and the program is executed.

Input devices: Unlike the desktops, the input devices to an embedded system have very limited capability. There will be no keyboard or a mouse, and hence interacting with the embedded system is no easy task. Many embedded systems will have a small keypad—you press one key to give a specific command. A keypad may be used to input only the digits. Many embedded systems used in process control do not have any input device for user interaction; they take inputs from sensors or transducers and produce electrical signals that are in turn fed to other systems.

Output devices: The output devices of the embedded systems also have very limited capability. Some embedded systems will have a few Light Emitting Diodes (LEDs) to indicate the health status of the system modules, or for visual indication of alarms. A small Liquid Crystal Display (LCD) may also be used to display some important parameters.

Communication interfaces: The embedded systems may need to interact with other embedded systems or they may have to transmit data to a desktop. To facilitate this, the embedded systems are provided with one or a few communication interfaces such as RS232, RS422, RS485, Universal Serial Bus (USB), IEEE 1394, Ethernet etc.

Application-specific circuitry: Sensors, transducers, special processing and control circuitry may be required for an embedded system, depending on its application. This circuitry interacts with the processor to carry out the necessary work.

The entire hardware has to be given power supply either through the 230 volts main supply or through a battery. The hardware has to be designed in such a way that the power consumption is minimized.

We will discuss each of these building blocks in greater detail in the next chapter.

In Brief...

The building blocks of an embedded system's hardware are: Central processing Unit, memory, input/output devices, communication interfaces and application-specific circuitry.

Notes...

The CPU of an embedded system can be: (a) micro-controller, (b) microprocessor, or (c) Digital Signal Processor.

A micro-controller is the best choice for small embedded systems because it has memory and peripherals in the same chip as the CPU and hence very few extra components are required. Digital Signal Processors are the best choice if the application demands signal processing such as audio/video processing.

1.5 Specialities of Embedded Systems

As compared to desktop computers, workstations or mainframes, embedded systems have many specialities. Developers need to keep these specialities in mind while designing embedded systems.

1.5.1 Reliability

When we use a desktop, sometimes the system 'hangs' and we need to reset the computer. Generally, this does not cause any problem. However, this is not the case with the embedded systems used in mission-critical applications. They must work with high reliability.

Reliability is of paramount importance in embedded systems. They should continue to work for thousands of hours without break. Many embedded systems used in industrial control are inaccessible. They are hidden in some other large-sized equipment; hence there will not be a reset button on such systems! So, the design of the embedded system should be such that in case the system has to be reset, the reset should be done automatically. Special hardware/software needs to be built into the system to take care of it. This special module is known as watchdog timer.

Many embedded systems used in industrial automation and defence equipment need to work in extreme environmental conditions such as very high/low temperatures, high humidity. Besides, they should be able to withstand bump and vibrations. Hence, very stringent environmental specifications have to be met by such systems. The ability to work reliably in extreme environmental conditions is known as ruggedness. Not only the military equipment, even the consumer appliances such as the mobile phone need to be very rugged. Many people keep dropping their mobile phone on the floor, but still it works because it is very rugged.

1.5.2 Performance

Many embedded systems have time constraints. For instance, in a process control system, a constraint can be: "if the temperature exceeds 40 degrees, open a valve within 10 milliseconds." The system must meet such deadlines. If the deadlines are missed, it may result in a catastrophe. You can imagine the damage that can be done if such deadlines are not met in a safety system of a nuclear plant.

1.5.3 Power Consumption

Most of the embedded systems operate through a battery. To reduce the battery drain and avoid frequent recharging of the battery, the power consumption of the embedded system has to be very low. To reduce power consumption such hardware components should be used that consume less power. Besides, emphasis should be on reducing the components count of the hardware. To reduce component count, the hardware designers have the option of using Programmable Logic Devices (PLDs) and Field Programmable Gate Arrays (FPGAs). Reducing the component count apart from reducing the power consumption also increases the reliability of the system.

1.5.4 Cost

For embedded systems used in safety applications of a nuclear plant or in a spacecraft, cost may not be a very important factor. However, for embedded systems used in consumer electronics or office automation, the cost is of utmost importance. Suppose you designed a toy in which the electronics will cost US$20. By a careful analysis of the design, if you can find a way to reduce the cost to US$19, it will be a great job. Don't underplay the importance of that one dollar because when you sell 10 million toys, the cost reduction is $10 million! Not surprisingly, the hardware engineers debate on component selection to reduce even $0.1.

1.5.5 Size

Size is certainly a factor for many embedded systems. We do not like a mobile phone that has to be carried on our backs. The size and the weight are important parameters in embedded systems used in aircraft, spacecraft, missiles etc. because in such cases, every inch and every gram matters. To reduce the size and the weight, again the hardware engineers have to design their boards by reducing the component count to the maximum possible extent. During the initial development of mobile phones based on GSM (Global System for Mobile Communication), GSM was expanded as "God, Send Mobiles" because of the complexity of the mobile phone. Today, the mobile phones go into our pockets, thanks to developments in microelectronics.

1.5.6 Limited User Interface

Unlike desktops, which have full-fledged input/output devices, embedded systems do not have sophisticated interfaces for input and output. Some embedded systems do not have any user interface at all. They take electrical signals as input and produce electrical signals as output. In many embedded systems, the input is through a small function keypad or a set of buttons. The output is displayed either on a set of LEDs or a small LCD. For example, the mobile phone has a small display of 4 lines x 16 characters. The input is through a keypad and composing a small text message is not and easy task. Developing a user friendly interface with limitation of the input/output devices is a challenging task for firmware developers.

1.5.7 Software Upgradation Capability

Embedded systems are meant for a very specific task. So, once the software is transferred to the embedded system, the same software will run throughout its life. However, in some cases, it may be necessary to upgrade the software. Consider the example of a Public Call Office (PCO). At the PCO, an embedded system is used which displays the amount to be paid by a telephone user. The amount is calculated by the firmware, based on the calling number and the duration of the call. From time to time, the telecom operator will change the algorithm for the calculation of the bill amount. So, every time there is a tariff change, the PCO operator has to replace the program stored in the memory of the embedded system with new program. This is very cumbersome, considering that a memory chip will have to be replaced in thousands of PCOs. Nowadays, software upgradation is done by downloading the software onto the embedded system through a network connection. In future, such an upgradation may become mandatory for many embedded systems.

In Brief...

While designing embedded systems, the following aspects need to be considered by the developers: reliability, performance, power consumption, cost, size, limited user interface and software upgradation capability.

Notes...

By using PLDs and FPGAs, the number of components in an embedded system can be reduced. Reducing the component count increases the system's reliability and also reduces the power consumption.