Introduction to Real Time Embedded Systems Part I

Example, Definitions, Common Architecture

Instructional Objectives

After going through this lesson the student would be able to

- Know what an embedded system is
- distinguish a Real Time Embedded System from other systems
- tell the difference between real and non-real time
- Learn more about a mobile phone
- Know the architecture
- Tell the major components of an Embedded system

Pre-Requisite

Digital Electronics, Microprocessors

Introduction

In the day-to-day life we come across a wide variety of consumer electronic products. We are habituated to use them easily and flawlessly to our advantage. Common examples are TV Remote Controllers, Mobile Phones, FAX machines, Xerox machines etc.

However, we seldom ponder over the technology behind each of them. Each of these devices does have one or more programmable devices waiting to interact with the environment as effectively as possible. These are a class of "embedded systems" and they provide service in real time. i.e. we need not have to wait too long for the action.

Let us see how an embedded system is characterized and how complex it could be? *Take example of a mobile telephone:* (Fig. 1.1)

Fig. 1.1 Mobile Phones

When we want to purchase any of them what do we look for?

Let us see what are the choices available?

Phone Price	Weight / Size	Screen	Games	Camera	Radio	Ring tones	Memory
Phone 1 Rs 5000/-	88.1 x 47.6 x 23.6 mm 116 g	TFT[1] 65k Color 96x32 screen	Stauntman2 & Monopoly3 included more downloadable	Yes 4 x Zoom	No	Polyphonic	
Phone 2 Rs 6000/-	89 x 49 x 24.8 mm 123 g	TFT 65k Color 176x220 screen	J2ME Games: Stauntman and Monopoly More downloadable	Integrated Digital Camera 1 M Pixel	No	Polyphonic and MP3	
Phone 3 Rs 5000/-	133.7 x 69.7 x 20.2mm 137g	176 x 208 pixel backlit screen with 4096 colors	Symbian and Java download games or packaged on MMC cards	No	FM Stereo		3.4 MB user memory built in.

Besides the above tabulated facts about the mobile handset, being a student of technology you may also like to know the following
Network type GSM[2] or CDMA[3] (Bandwidth),

Battery: Type and ampere hour Talk-time per one charge, Standby time

[1] Short for ***t**hin **f**ilm **t**ransistor*, a type of LCD flat-panel display screen, in which each pixel is controlled by from one to four transistors. The TFT technology provides better resolution of all the flat-panel techniques, but it is also the most expensive. TFT screens are sometimes called *active-matrix LCDs*.

[2] short form of ***G**lobal **S**ystem for **M**obile Communications*, one of the leading digital cellular systems. GSM uses narrowband Time Division Multiple Access (TDMA), which allows eight simultaneous calls on the same radio frequency. GSM was first introduced in 1991. As of the end of 1997, GSM service was available in more than 100 countries and has become the *de facto* standard in Europe and Asia.

[3] Short form of ***Co**de-**D**ivision **M**ultiple **A**ccess*, a digital cellular technology that uses *spread-spectrum* techniques. Unlike competing systems, such as GSM, that use *TDMA*, CDMA does not assign a specific frequency to each user. Instead, every channel uses the full available spectrum. Individual conversations are encoded with a pseudo-random digital sequence. CDMA is a military technology first used during World War II by the English allies to foil German attempts at jamming transmissions. The allies decided to transmit over several frequencies, instead of one, making it difficult for the Germans to pick up the complete signal.

From the above specifications it is clear that a mobile phone is a very complex device which houses a number of miniature gadgets functioning coherently on a single device.

Moreover each of these embedded gadgets such as digital camera or an FM radio along with the telephone has a number of operating modes such as:

- you may like to adjust the zoom of the digital camera,
- you may like to reduce the screen brightness,
- you may like to change the ring tone,
- you may like to relay a specific song from your favorite FM station to your friend using your mobile
- You may like to use it as a calculator, address book, emailing device etc.

These variations in the functionality can only be achieved by a very flexible device.

This flexible device sitting at the heart of the circuits is none other than a Customized Microprocessor better known as an **Embedded Processor** and the mobile phone housing a number of functionalities is known as an **Embedded System.**

Since it satisfies the requirement of a number of users at the same time (you and your friend, you and the radio station, you and the telephone network etc) it is working within a time-constraint, i.e. it has to satisfy everyone with the minimum acceptable delay. We call this as to work in "**Real Time**". This is unlike your holidaying attitude when you take the clock on your stride.

We can also say that it does not make us wait long for taking our words and relaying them as well as receiving them, unlike an email server, which might take days to receive/deliver your message when the network is congested or slow.

Thus we can name the mobile telephone as a **"Real Time Embedded System" (RTES)**

Definitions

Now we are ready to take some definitions

Real Time

"'Real'-time usually means time as prescribed by external sources"
For example the time struck by clock (however fast or late it might be). The timings generated by your requirements. You may like to call someone at mid-night and send him a picture. This external timing requirements imposed by the user is the real-time for the embedded system.

Embedded (Embodiment)

"Embodied phenomena are those that by their very nature occur in real time and real space"
In other words, *A number of systems coexist to discharge a specific function in real time*

Thus "A Real Time Embedded System" (RTES) is precisely the union of subsystems to discharge a specific task coherently. Hence forth we call them as RTES. RTES as a generic term may mean a wide variety of systems in the real world. However we will be concerned about them which use programmable devices such as microprocessors or microcontrollers and have specific functions. We shall characterize them as follows.

Characteristics of an Rtes

Single-Functioned

Here "single-functioned" means specific functions. The RTES is usually meant for very specific functions. Generally a special purpose microprocessor executes a program over and over again for a specific purpose. If the user wants to change the functionality, e.g. changing the mobile phone from conversation to camera mode or calculator mode the program gets flushed out and a new program is loaded which carries out the requisite function. These operations are monitored and controlled by an operating system called as Real Time Operating System (RTOS) which has much simpler complexity but more rigid constraints as compared to the conventional operating systems such as Micro Soft Windows and Unix etc.

Tightly Constrained

The constraints on the design and marketability of RTES are more rigid than their non-real-time non-embedded counter parts. Time-domain constraints are the first thing that is taken care while developing such a system. Size, weight, power consumption and cost[4] are the other major factors.

Reactive and Real Time

Many embedded systems must continually react to changes in the system's environment and must compute certain results in real time without delay. For example, a car's cruise controller continually monitors and reacts to speed and brake sensors. It must compute acceleration or deceleration amounts repeatedly within a limited time; a delayed computation could result in a failure to maintain control of the car. In contrast a desktop computer system typically focuses on computations, with relatively infrequent (from the computer's perspective) reactions to input devices. In addition, a delay in those computations, while perhaps inconvenient to the computer user, typically does not result in a system failure.

[4] Very few in India will be interested to buy a mobile phone if it costs Rs50,000/- even if it provides you a faster processor with 200MB of memory to store your address, your favorite mp3 music and plays them , acts as a small-screen TV whenever you desire, takes your call intelligently
However in USA majority can afford it !!!!!!

Common Architecture of Real Time Embedded Systems

Unlike general purpose computers a generic architecture can not be defined for a *Real Time Embedded Systems.* There are as many architecture as the number of manufacturers. Generalizing them would severely dilute the soul purpose of embodiment and specialization.

However for the sake of our understanding we can discuss some common form of systems at the block diagram level. Any system can hierarchically divided into subsystems. Each sub-system may be further segregated into smaller systems. And each of these smaller systems may consist of some discrete parts. This is called Hardware configuration.

Some of these parts may be programmable and therefore must have some place to keep these programs. In RTES the on-chip or on-board non-volatile memory does keep these programs. These programs are the part of the Real Time Operating System (RTOS) and continually run as long as the gadget is receiving power. A part of the RTOS also executes itself in the stand-by mode while taking a very little power from the battery. This is also called the sleep mode of the system.

Both the hardware and software coexist in a coherent manner. Tasks which can be both carried out by software and hardware affect the design process of the system. For example a multiplication action may be done by hardware or it can be done by software by repeated additions. Hardware based multiplication improves the speed at the cost of increased complexity of the arithmetic logic unit (ALU) of the embedded processor. On the other hand software based multiplication is slower but the ALU is simpler to design. These are some of the conflicting requirements which need to be resolved on the requirements as imposed by the overall system. This is known as *Hardware-Software Codesign* or simply *Codesign***.**

Let us treat both the hardware and the imbibed software in the same spirit and treat them as systems or subsystems. Later on we shall know where to put them together and how. Thus we can now draw a hierarchical block diagram representation of the whole system as follows:

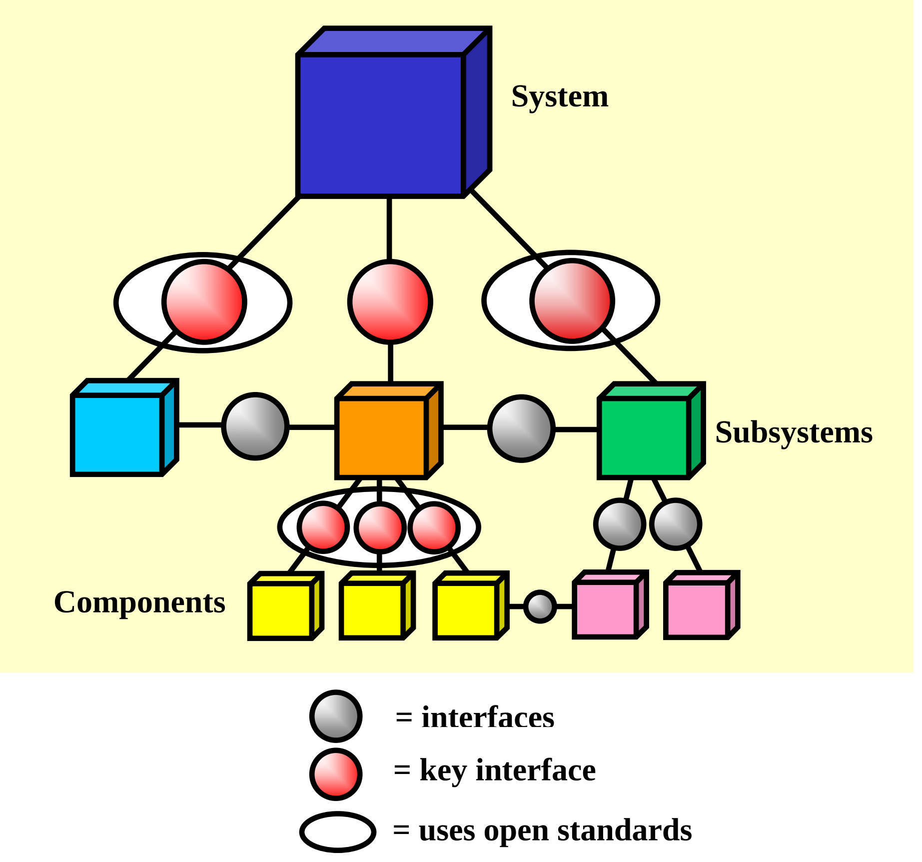


Fig. 1.2 The System Interface and Architecture

The red and grey spheres in Fig.1.2 represent interface standards. When a system is assembled it starts with some chassis or a single subsystem. Subsequently subsystems are added onto it to make it a complete system.

Let us take the example of a Desktop Computer. Though not an Embedded System it can give us a nice example of assembling a system from its subsystems.

You can start assembling a desktop computer (Fig.1.3) starting with the chassis and then take the SMPS (switched mode power supply), motherboard, followed by hard disk drive, CDROM drive, Graphic Cards, Ethernet Cards etc. Each of these subsystems consists of several components e.g. Application Specific Integrated Circuits (ASICs), microprocessors, Analog as well as Digital VLSI circuits, Miniature Motor and its control electronics, Multilevel Power supply units crystal clock generators, Surface mounted capacitors and resistors etc. In the end you close the chassis and connect Keyboard, Mouse, Speakers, Visual Display Units, Ethernet Cable, Microphone, Camera etc fitting them into certain well-defined sockets.

As we can see that each of the subsystems inside or outside the Desktop has cables fitting well into the slots meant for them. These cables and slots are uniform for almost any Desktop you choose to assemble. The connection of one subsystem into the other and vice-versa is known as **Interfacing.** It is so easy to assemble because they are all **standardized**. Therefore, standardization of the interfaces is most essential for the universal applicability of the system and its compatibility with other systems. There can be open standards which makes it exchange information with products from other companies. It may have certain key standards, which is only meant for the specific company which manufactures them.

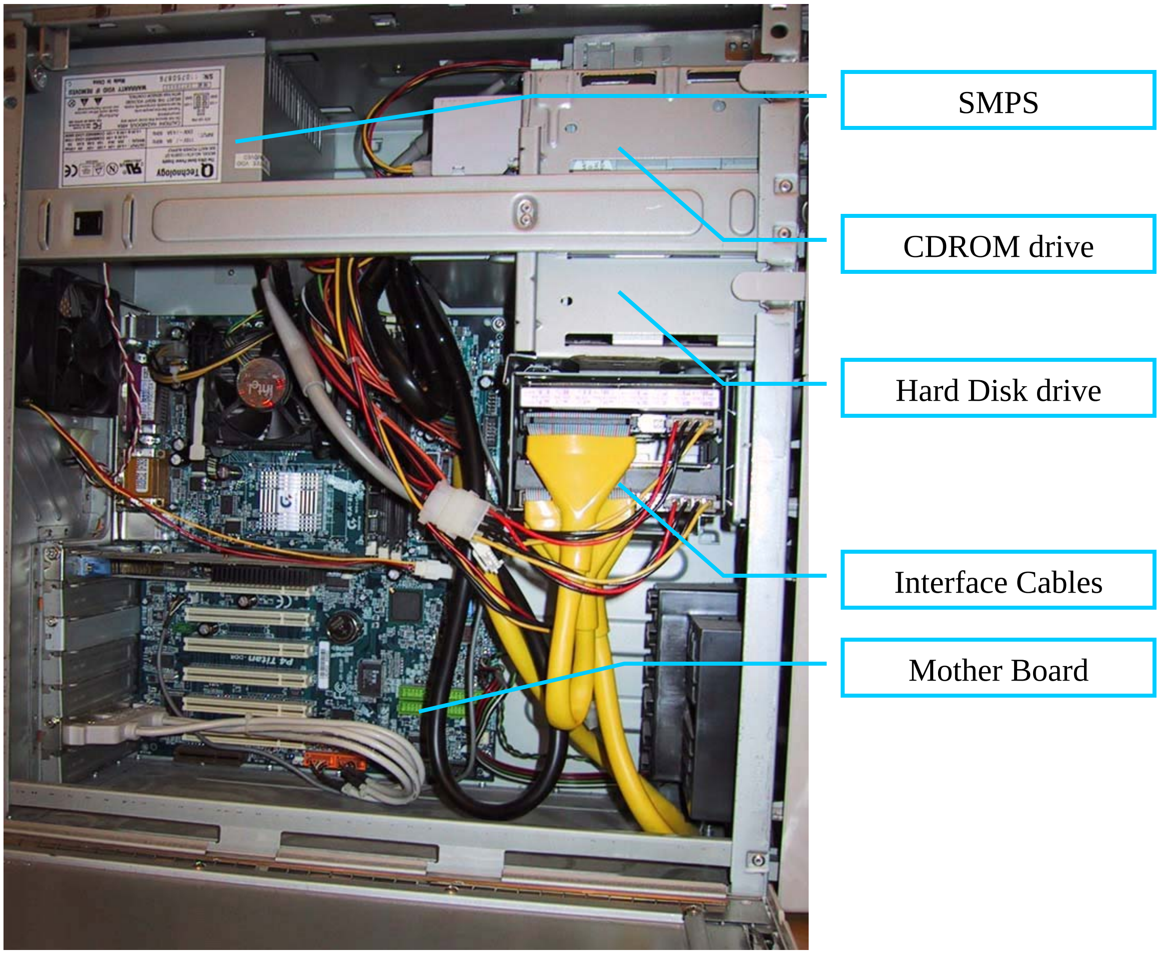


Fig. 1.3 Inside Desktop Computer

A Desktop Computer will have more open standards than an Embedded System. This is because of the level of integration in the later. Many of the components of the embedded systems are integrated on to a single chip. This concept is known as **System on Chip (SOC)** design**.** Thus there are only few subsystems left to be connected.

Analyzing the assembling process of a Desktop let us comparatively assess the possible subsystems of the typical RTES.

One such segregation is shown in Fig.1.4. The explanation of various parts as follows:

User Interface: for interacting with users. May consists of keyboard, touch pad etc

ASIC: Application Specific Integrated Circuit: for specific functions like motor control, data modulation etc.

Microcontroller(μC): A family of microprocessors

Real Time Operating System (RTOS): contains all the software for the system control and user interface

Controller Process: The overall control algorithm for the external process. It also provides timing and control for the various units inside the embedded system.

Digital Signal Processor (*DSP)* a typical family of microprocessors

DSP assembly code: code for DSP stored in program memory

Dual Ported Memory: Data Memory accessible by two processors at the same time

CODEC: Compressor/Decompressor of the data

User Interface Process: The part of the RTOS that runs the software for User Interface activities

Controller Process: The part of the RTOS that runs the software for Timing and Control amongst the various units of the embedded system

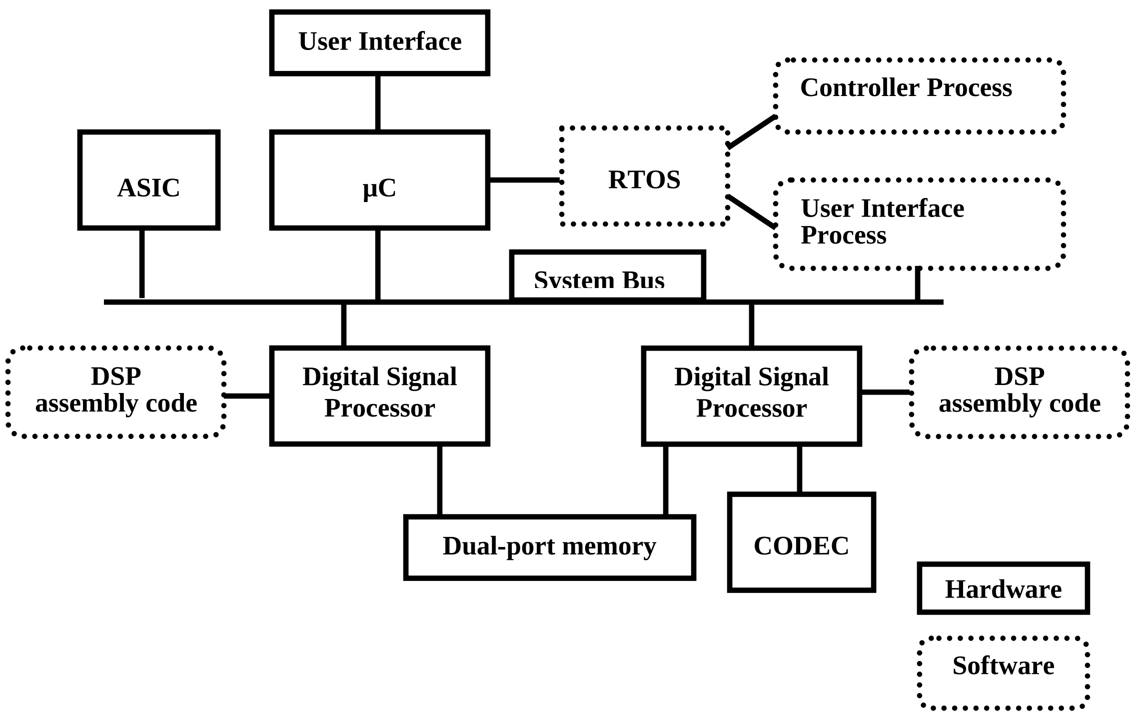


Fig. 1.4 Architecture of an Embedded System

The above architecture represents a hypothetical Embedded System (we will see more realistic ones in subsequent examples). More than one microprocessor (2 DSPs and 1 µC) are employed here to carry out different tasks. As we will learn later, the µC is generally meant for simpler and slower jobs such as carrying out a Proportional Integral (PI) control action or interpreting the user commands etc. The DSP is a more heavy duty processor capable of doing real time signal processing and control. Both the DSPs along with their operating systems and codes are independent of each other. They share the same memory without interfering with each other. This kind of memory is known as dual ported memory or two-way post-box memory. The Real Time Operating System (RTOS) controls the timing requirement of all the devices. It executes the over all control algorithm of the process while diverting more complex tasks to the DSPs. It also specifically controls the µC for the necessary user interactivity. The ASICs are specialized units capable of specialized functions such as motor control, voice encoding, modulation/demodulation (MODEM) action etc. They can be digital, analog or mixed signal VLSI circuits. CODECs are generally used for interfacing low power serial Analog-to-Digital Converters (ADCs). The analog signals from the controlled process can be monitored through an ADC interfaced through this CODEC.

Questions and Answers

Q1 Which of the following is a real time embedded system? Justify your answer

(a) Ceiling Fan

(b) Microwave Oven

(c) Television Set

(d) Desktop Key Board

(e) Digital Camera

Ans:

(b) and (e) are embedded systems

(a) Ceiling Fans: These are not programmable.

(b) & (e) obey all definitions of Embedded Systems such as

(i) Working in Real Time (ii) Programmable (iii) A number of systems coexist on a single platform to discharge one function(single functioned)

(c) Television Set: Only a small part of it is programmable. It can work without being programmable. It is not tightly constrained.

(d) Desktop Keyboard: Though it has a processor normally it is not programmable.

Definition of Real Time Systems

An operation within a larger dynamic system is called a real-time operation if the combined reaction- and operation-time of a task operating on current events or input, is no longer than the maximum delay allowed, in view of circumstances outside the operation. The task must also occur before the system to be controlled becomes unstable. A real-time operation is not necessarily fast, as slow systems can allow slow real-time operations. This applies for all types of dynamically changing systems. The polar opposite of a real-time operation is a batch job with interactive timesharing falling somewhere in between the two extremes.

Alternately, a system is said to be hard real-time if the correctness of an operation depends not only upon the logical correctness of the operation but also upon the time at which it is performed. An operation performed after the deadline is, by definition, incorrect, and usually has no value. In a soft real-time system the value of an operation declines steadily after the deadline expires.

Embedded System

An embedded system is a special-purpose system in which the computer is completely encapsulated by the device it controls. Unlike a general-purpose computer, such as a personal computer, an embedded system performs pre-defined tasks, usually with very specific requirements. Since the system is dedicated to a specific task, design engineers can optimize it, reducing the size and cost of the product. Embedded systems are often mass-produced, so the cost savings may be multiplied by millions of items.

Handheld computers or PDAs are generally considered embedded devices because of the nature of their hardware design, even though they are more expandable in software terms. This line of definition continues to blur as devices expand.

Q.2 Write five advantages and five disadvantages of embodiment.

Ans:

Five advantages:

1. Smaller Size
2. Smaller Weight
3. Lower Power Consumption
4. Lower Electromagnetic Interference
5. Lower Price

Five disadvantages

1. Lower Mean Time Between Failure
2. Repair and Maintenance is not possible
3. Faster Obsolesce
4. Unmanageable Heat Loss
5. Difficult to Design

Q3. What do you mean by *Reactive in Real Time*. Cite an example.

Ans:

Many embedded systems must continually react to changes in the system's environment and must compute certain results in real time without delay. For example, a car's cruise controller continually monitors and reacts to speed and brake sensors. It must compute acceleration or deceleration amounts repeatedly within a limited time; a delayed computation could result in a failure to maintain control of the car. In contrast a desktop computer system typically focuses on computations, with relatively infrequent (from the computer's perspective) reactions to input devices. In addition, a delay in those computations, while perhaps inconvenient to the computer user, typically does not result in a system failure.

Q4. Give at least five examples of embedded systems you are using/watching in your day to day life.

(i) Mobile Telephone (ii)Digital Camera (iii) A programmable calculator (iv) An iPod (v) A digital blood pressure machine

iPod: The iPod is a brand of portable media players designed and marketed by Apple Computer. Devices in the iPod family are designed around a central scroll wheel (except for the iPod shuffle) and provide a simple user interface. The full-sized model stores media on a built-in hard drive, while the smaller iPod use flash memory. Like many digital audio players, iPods can serve as external data storage devices when connected to a computer.

Q5. Write the model number and detailed specification of your/friend's mobile telephone.

Manufacturer
Model:
Network Types: EGSM/ GSM /CDMA
Form Factor: *The industry standard that defines the physical, external dimensions of a particular device. The size, configuration, and other specifications used to describe hardware.*
Battery Life Talk (hrs):
Battery Life Standby (hrs):
Battery Type:

Measurements
Weight:
Dimensions:
Display Display Type: Colour or Black & White
Display Size (px):
Display Colours:

General Options
Camera:
Mega Pixel:
Email Client:
Games: Yes
High Speed Data:
MP3 Player:
PC Sync: Yes
Phonebook:
Platform Series
Polyphonic Ring tones:
Predictive Text:
Streaming Multimedia:
Text Messages:
Wireless Internet: Opera

Other Options
Alarm:
Bluetooth:
Calculator:
Calendar:
Data Capable:
EMS:
FM Radio:
Graphics (Custom):
Infrared:
Speaker Phone:
USB:
Vibrate:

Lesson 2

Introduction to Real Time Embedded Systems Part II

Structure and Design

Instructional Objectives

After going through this lesson the student will

- Learn more about the numerous day-to-day real time embedded systems
- Learn the internal hardware of a typical mobile phone
- Learn about the important components of an RTES
- Learn more about a mobile phone
- Learn about the various important design issues
- Also learn the design flow

Pre-Requisite

Digital Electronics, Microprocessors

Common Examples Of Embedded Systems

Some of the common examples of Embedded Systems are given below:

Consumer electronics cell phones, pagers, digital cameras, camcorders, DVD players, portable video games, calculators, and personal digital assistants etc.

Fig. 2.1(a) Digital Camera

Fig. 2.1(b) Camcorder

Fig. 2.1(c) Personal Digital Assistants

Home appliances microwave ovens, answering machines. thermostats, home security systems, washing machines. and lighting systems etc.

Fig. 2.1(d) Microwave Oven

Fig. 2.1(e) Washer and Dryers

office automation fax machines, copiers, printers, and scanners

Fig. 2.1(f) Fax cum printer cum copier

business equipment electronic cash registers, curbside check-in, alarm systems, card readers product scanners, and automated teller machines

Fig. 2.1(g) Electronic Cash Registers

Fig. 2.1(h)Electronic Card Readers

Fig. 2.1(i)Automated Teller Machines

automobiles Electronic Control Unit(ECU) which includes transmission control, cruise control, fuel injection, antilock brakes, and active suspension in the same or separate modules.

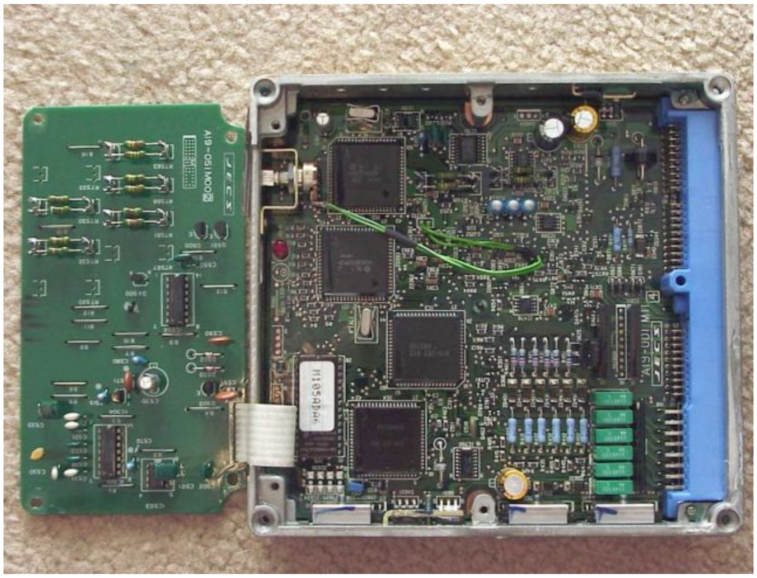


Fig. 2.1(j)ECU of a Vehicle

Mobile Phone

Let us take the same mobile phone as discussed in Lesson 1 as example for illustrating the typical architecture of RTES.

In general, a cell phone is composed of the following components:

- A Circuit board (Fig. 2.2)
- Antenna
- Microphone
- Speaker
- Liquid crystal display (LCD)
- Keyboard
- Battery

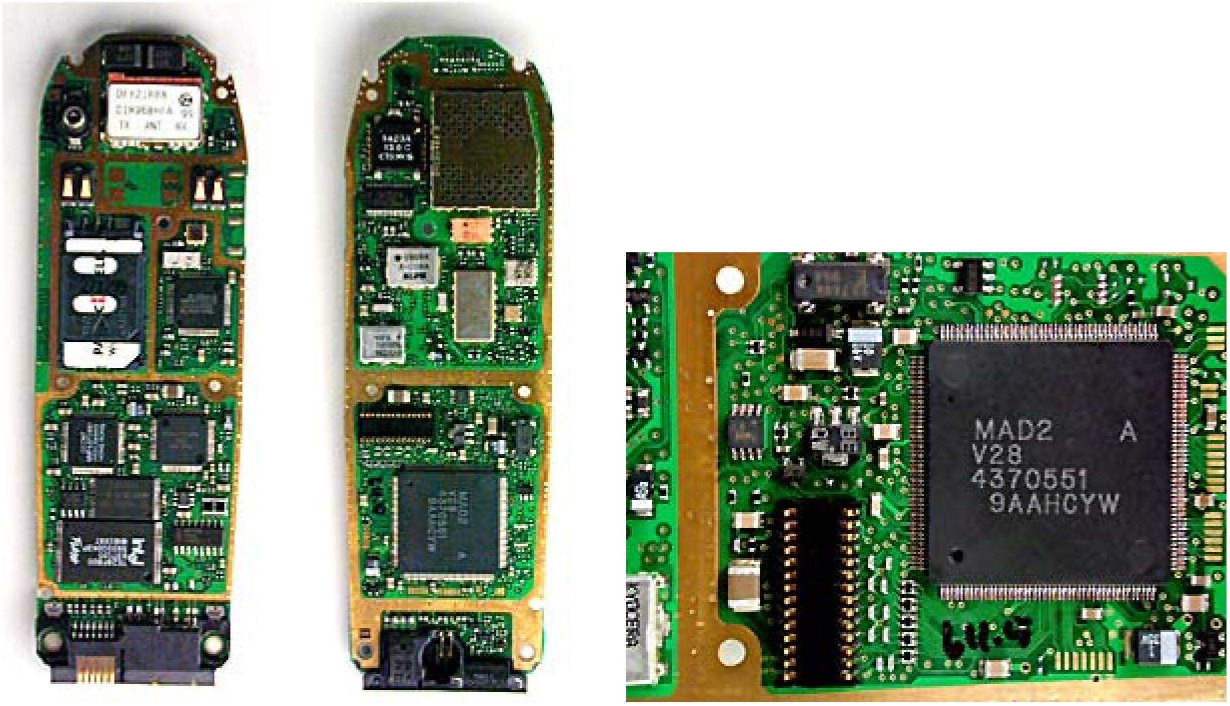


Fig. 2.2 The Cell Phone Circuitry

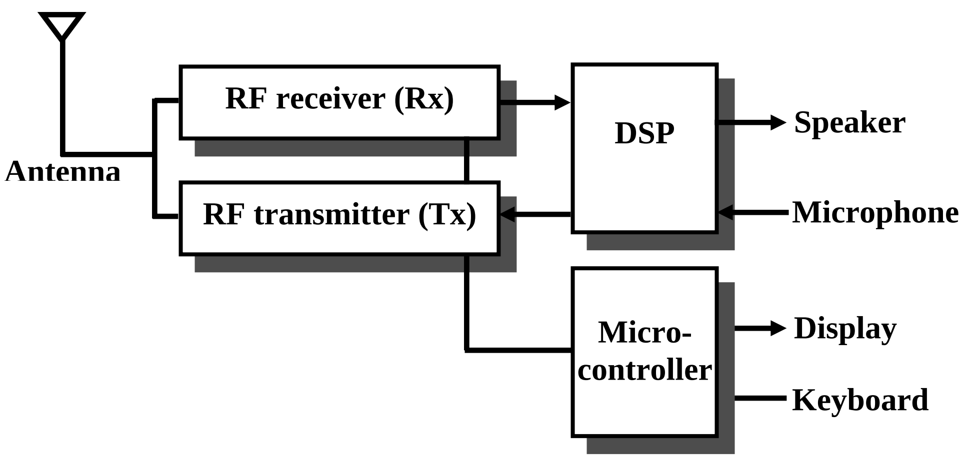


Fig. 2.3 The block diagram

A typical mobile phone handset (Fig. 2.3) should include standard I/O devices (keyboard, LCD), plus a microphone, speaker and antenna for wireless communication. The Digital Signal Processor (DSP) performs the signal processing, and the micro-controller controls the user interface, battery management, call setup etc. The performance specification of the DSP is very crucial since the conversion has to take place in real time. This is why almost all cell phones contain such a special processor dedicated for making digital-to-analog (DA) and analog-to-digital(AD) conversions and real time processing such as modulation and demodulation etc. The Read Only Memory (ROM) and flash memory (Electrically Erasable and Programmable Memory) chips provide storage for the phone's operating system(RTOS) and various data such as phone numbers, calendars information, games etc.

Components of an Embedded System

By this time we know where are our Embedded Systems and what makes them stand out from other systems like Calculators, Desktop Computers, and our Old Television Sets. We have also developed some 6th sense to guess the components of an RTES.

1. Microprocessor

This is the heart of any RTES. The microprocessors used here are different from the general purpose microprocessors like Pentium Sun SPARC etc. They are designed to meet some specific requirements. For example Intel 8048 is a special purpose microprocessor which you will find in the Keyboards of your Desktop computer. It is used to scan the keystrokes and send them in a synchronous manner to your PC. Similarly mobile phones Digital Cameras use special purpose processors for voice and image processing. A washer and dryer may use some other type of processor for *Real Time Control and Instrumentation.*

2. Memory

The microprocessor and memory must co-exit on the same Power Circuit Board(PCB) or same chip. *Compactness*, *speed* and *low power* consumption are the characteristics required for the memory to be used in an RTES. Therefore, very low power semiconductor memories are used in almost all such devices. For housing the operating system Read Only Memory(ROM) is used. The program or data loaded might exist for considerable duration. It is like changing the setup of your Desktop Computer. Similar user defined setups exist in RTES. For example you may like to change the ring tone of your mobile and keep it for some time. You may like to change the screen color etc. In these cases the memory should be capable of retaining the information even after the power is removed. In other words the memory should be non-volatile and should be easily programmable too. It is achieved by using Flash[1] memories.

3. Input Output Devices and Interfaces

Input/Output interfaces are necessary to make the RTES interact with the external world. They could be Visual Display Units such as TFT screens in a mobile phone, touch pad key board, antenna, microphones, speakers etc. These RTES should also have open interfaces to other devices such as Desktop Computers, Local Area Networks (LAN) and other RTES. For example you may like to download your address book into your personal digital assistant (PDA). Or you may like to download some mp3 songs from your favorite internet site into your mp3 player. These input/output devices along with standard software protocols in the RTOS provide the necessary interface to these standards.

[1] A memory technology similar in characteristics to EPROM(Erasable Programmable Read Only Memory) memory, with the exception that erasing is performed electrically instead of via ultraviolet light, and, depending upon the organization of the flash memory device, erasing may be accomplished in blocks (typically 64k bytes at a time) instead of the entire device.

4. Software

The RTES is the just the physical body as long as it is not programmed. It is like the human body without life. Whenever you switch on your mobile telephone you might have marked some activities on the screen. Whenever you move from one city to the other you might have noticed the changes on your screen. Or when you are gone for a picnic away from your city you might have marked the no-signal sign. These activities are taken care of by the Real Time Operating System sitting on the non-volatile memory of the RTES.

Besides the above an RTES may have various other components and Application Specific Integrated Circuits (ASIC) for specialized functions such as motor control, modulation, demodulation, CODEC.

The design of a Real Time Embedded System has a number of constraints. The following section discusses these issues.

Design Issues

The constraints in the embedded systems design are imposed by external as well as internal specifications. Design metrics are introduced to measure the cost function taking into account the technical as well as economic considerations.

Design Metrics

A Design Metric is a measurable feature of the system's performance, cost, time for implementation and safety etc. Most of these are conflicting requirements i.e. optimizing one shall not optimize the other: e.g. a cheaper processor may have a lousy performance as far as speed and throughput is concerned.

Following metrics are generally taken into account while designing embedded systems

NRE cost (nonrecurring engineering cost)

It is one-time cost of designing the system. Once the system is designed, any number of units can be manufactured without incurring any additional design cost; hence the term nonrecurring.

Suppose three technologies are available for use in a particular product. Assume that implementing the product using technology 'A' would result in an NRE cost of $2,000 and unit cost of $100, that technology B would have an NRE cost of $30,000 and unit cost of $30, and that technology **C** would have an NRE cost of $100,000 and unit cost of $2. Ignoring all other design metrics, like time-to-market, the best technology choice will depend on the number of units we plan to produce.

Unit cost

The monetary cost of manufacturing each copy of the system, excluding NRE cost.

Size

The physical space required by the system, often measured in bytes for software, and gates or transistors for hardware.

Performance

The execution time of the system

Power Consumption

It is the amount of power consumed by the system, which may determine the lifetime of a battery, or the cooling requirements of the IC, since more power means more heat.

Flexibility

The ability to change the functionality of the system without incurring heavy NRE cost. Software is typically considered very flexible.

Time-to-prototype

The time needed to build a working version of the system, which may be bigger or more expensive than the final system implementation, but it can be used to verify the system's usefulness and correctness and to refine the system's functionality.

Time-to-market

The time required to develop a system to the point that it can be released and sold to customers. The main contributors are design time, manufacturing time, and testing time. This metric has become especially demanding in recent years. Introducing an embedded system to the marketplace early can make a big difference in the system's profitability.

Maintainability

It is the ability to modify the system after its initial release, especially by designers who did not originally design the system.

Correctness

This is the measure of the confidence that we have implemented the system's functionality correctly. We can check the functionality throughout the process of designing the system, and we can insert test circuitry to check that manufacturing was correct.

The Performance Design Metric

Performance of a system is a measure of how long the system takes to execute our desired tasks.

The two main measures of performance are:

Latency or response time

This is the time between the start of the task's execution and the end. For example, processing an image may take 0.25 second.

Throughput

This is the number of tasks that can be processed per unit time. For example, a camera may be able to process 4 images per second

These are the some of the cost measures for developing an RTES. Optimization of the overall cost of design includes each of these factors taken with some multiplying factors depending on their importance. And the importance of each of these factors depends on the type of application. For instance in defense related applications while designing an anti-ballistic system the execution time is the deciding factor. On the other hand, for de-noising a photograph in an embedded camera in your mobile handset the execution time may be little relaxed if it can bring down the cost and complexity of the embedded Digital Signal Processor.

The design flow of an RTES involves several steps. The cost and performance is tuned and fine-tuned in a recursive manner. An overall design methodology is enumerated below.

Design Methodology (Fig. 2.4)

System Requirement and Specifications
Define the problem
What your embedded system is required to do?
Define the requirements (inputs, outputs, control)
What are the inputs and outputs of your system?
Write down the specifications for them
Specify if the signals are in digital or analogue form. Specify the voltage levels, frequency etc.
The design task can be further segregated into the following steps

System level Design

Find out the possible subsystems of the system and the interconnections between them.

Sub-system or Node Level design

Each of these subsystems can be termed as the nodes. Elaborate on each of these subsystems and further make the block diagram and component level interconnections.

Processor Level Design

Each subsystem may consist of processor, memory, I/O devices. Specification and design at this level is required now.

Task Level Design

Complete interconnection of these subsystems depending on the tasks they would perform.

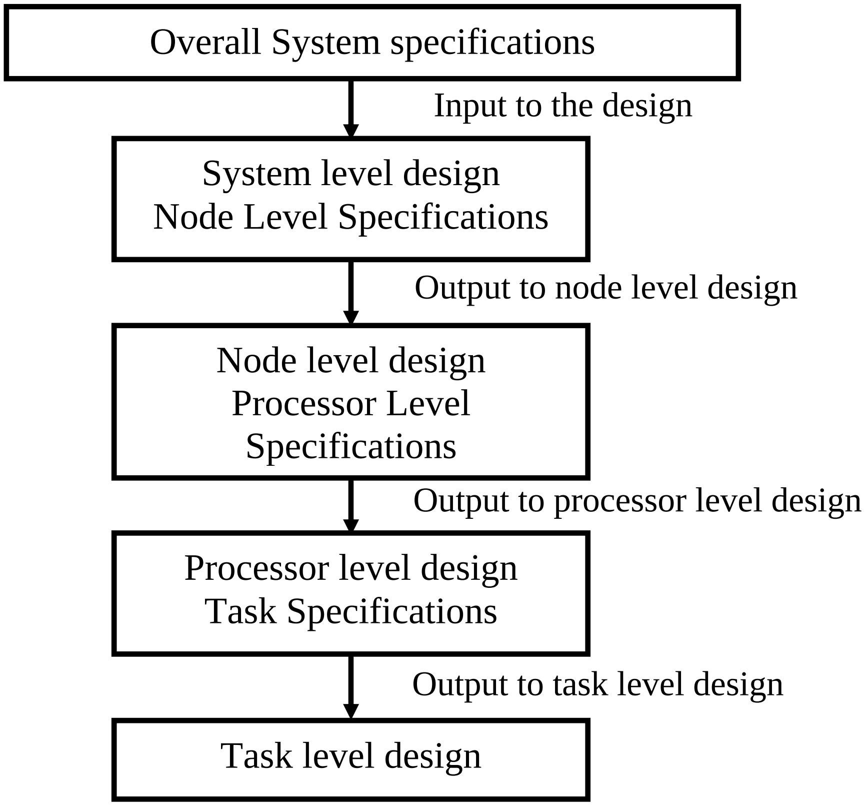


Fig. 2.4 The design approach

Conclusion

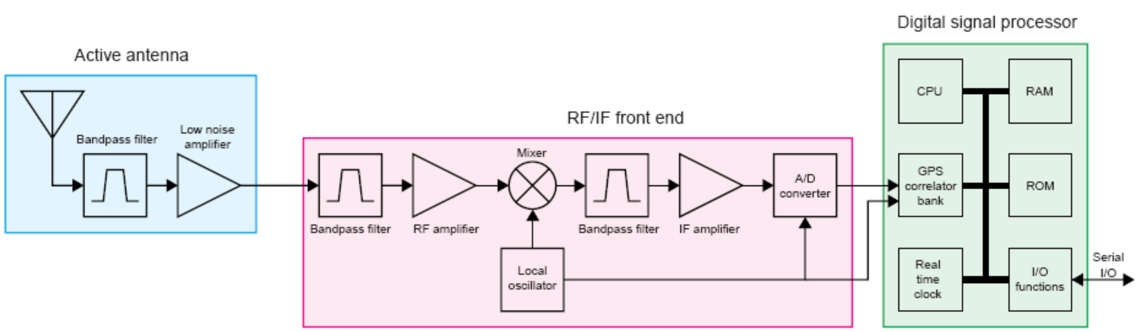
- The scope of embedded systems has been encompassing more and more diverse disciplines of technology day by day. Obsolescence of technology occurs at a much faster pace as compared to the same in other areas. The development of Ultra-Low-Power VLSI mixed signal technology is the prime factor in the miniaturization and enhancement of the performance of the existing systems. More and more systems are tending to be compact and portable with the RTES technology. The future course of embedded systems depends on the advancements of sensor technology, mechatronics and battery technology.
- The design of these RTES by and large is application specific. The time-gap between the conception of the design problem and marketing has been the key factor for the industry.
- Most of the cases for very specific applications the system needs to be developed using the available processors rather than going for a custom design.

Questions

Q1. Give one example of a typical embedded system other than listed in this lecture. Draw the block diagram and discuss the function of the various blocks. What type of embedded processor they use?

Ans:

Example 1: A handheld Global Positioning System Receiver



For details please http://www.gpsworld.com/

A GPS receiver receives signals from a constellation of at least four out of a total of 24 satellites. Based on the timing and other information signals sent by these satellites the digital signal processor calculates the position using triangulation.

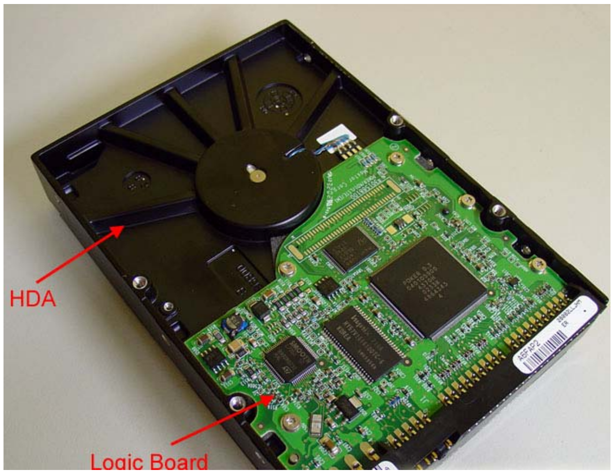
The major block diagram is divided into (1) Active Antenna System (2)RF/IF front end (3) The Digital Signal Processor(DSP)

The Active Antenna System houses the antenna a band pass filter and a low noise amplifier (LNA)

The RF/IF front end houses another band pass filter, the RF amplifier and the demodulator and A/D converter.

The DSP accepts the digital data and decodes the signal to retrieve the information sent by the GPS satellites.

Q2. Discuss about the Hard Disk Drive housed in your PC. Is it an RTES?



Ans:

Hard drives have two kinds of components: internal and external. External components are located on a printed circuit board called logic board while internal components are located in a sealed chamber called HDA or Hard Drive Assembly.

For details browse http://www.hardwaresecrets.com/article/177/3

The big circuit is the controller. It is in charge of everything: exchanging data between the hard drive and the computer, controlling the motors on the hard drive, commanding the heads to read or write data, etc.

All these tasks are carried out as demanded by the processor sitting on the motherboard. It can be verified to be single-functioned, tightly constrained,

Therefore one can say that a Hard Disk Drive is an RTES.

Q3. Elaborate on the time-to-market design metric.

Ans:

The time required to develop a system to the point that it can be released and sold to customers. The main contributors are design time, manufacturing time, and testing time. This metric has become especially demanding in recent years. Introducing an embedded system to the marketplace early can make a big difference in the system's profitability.

Q4. What is Moore's Law? How was it conceived?

Moore's law is the empirical observation that the complexity of integrated circuits, with respect to minimum component cost, doubles every 24 months. It is attributed to Gordon E. Moor, a co-founder of Intel.