A

Seminar report

on

"Embeded System"

Submitted in partial fulfillment of the requirement for the award of degree
Of Bachelor of Technology in Computer Science

SUBMITTED TO:
www.studymafia.org

SUBMITTED BY:
www.studymafia.org

ACKNOWLEDGEMENT

It is my pleasure to be indebted to various people, who directly or indirectly helped me in the seminar on “Embedded Systems ”.

I would like to express my gratefulness to………………**,** who has given me the opportunity to carry out this seminar.

Lastly, I would like to thank the almighty and my parents for their moral support.

CONTENT

EMBEDDED SYSTEMS

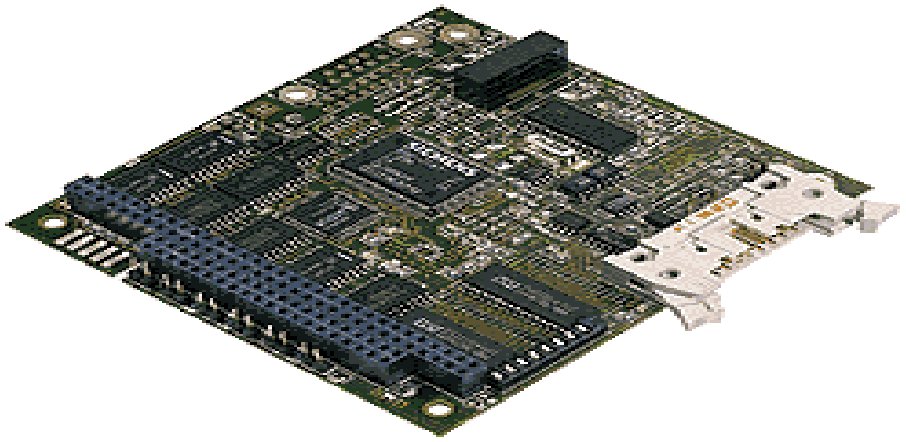
From handheld devices such as personal digit assistants and smart cell phones to automobiles and rocket propellers embedded systems are at the heart of it all. Embedded systems are basically a combination of hardware and software designed to perform a specific function.Software provides features to it and hardware is used for performance. These functions are performed with the help of a new type of operating systems known as Real Time operating system (RTO's).

The basic components of an embedded system are embedded hardware, embedded RTO's, device drivers, communication stacks and embedded applications. The challenge is to provide affordable, highly integrated devices meeting stringent requirements for safety, security, reliability, and availability and at the same time keeping the cost low.

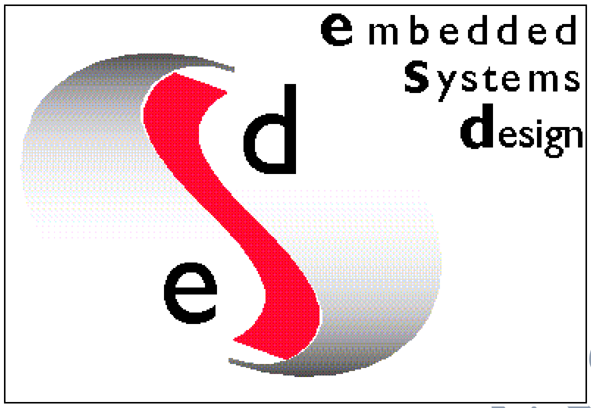
As microprocessors are becoming smaller and cheaper, more and more products are becoming 'smart' with microprocessor embedded in them. Few applications of embedded systems are ATM's, multifunction wristwatches, PDA's, PLC's, medical equipments, etc and many more. Embedded software requires software tools like cross compiler, cross assembler, linker, and locator along with the host and target systems. The application areas of this technology cover virtually everything from home appliances to communication equipments.

As the world is experiencing groundbreaking research in hardware technology (nano-technology, quantum mechanics, etc), packing more power into a single chip will soon be a reality. So it can be said that embedded systems is a dreamer's paradise with unlimited possibilities.

EMBEDDED SYSTEMS



EMBEDDED SYSTEMS



INTRODUCTION:

The term ‘embedded systems’ is quite a complex one. An embedded system is a special purpose computer that is used inside of a device. It is basically a combination of hardware and software designed to perform a specific function. These smart systems can take decisions in different conditions.

Built into the devices, these are invisible to our eyes but have all the processing power of computers. They form the components of a larger system. This, in turn, is preprogrammed to perform a range of functions, usually with minimal operator intervention**.**

High profile embedded chips is scaleable, generate small amounts of heat, and consume less power. These are generally preferred for their speed, accuracy and reliability. On larger scale, programmable devices or systems are generally used to monitor or control processes and equipment.

In embedded systems, the hardware is normally unique to a given application. Computer chips are embedded into the control electronics to manage the product functionality. Thanks to their compact size and ability to perform time critical and task specific operations, embedded devices find applications in all segments of the commercial and industrial marketplace. Home appliances, mobile phones, personal digital assistants (PDA’s), cars, tiny microchips and avionics are all using embedded technology.

A few years ago embedded technology existed in standalone devices such as vending machines and copiers that did their jobs with little regard to what went on around them. But as technology advanced to connect devices (mobile phones, PDA’s, and so on) to the Internet and to each other, the potential of embedded technology has grown. Suddenly, it is not so much about what devices do

on their own but about what they are connected to and what actions the connections let them perform.
For example, take the case of Internet-the most important of all applications. The Internet is itself a huge embedded system. Today, typical access to the Internet is via broadband technology, GSM/GPRS or CDMA modem, Ethernet LAN or Wi-Fi access points.

Characterstics of the Embedded Systems:

*They compute results in real time ,without any delay.
*They are low cost
*They use little power
*They are small in size

Why they are called so?

In Embedded Systems all the circuits are embedded on the chip but they are so small in size that they often go unnoticed

Examples of Embedded System

Suppose If an air conditioner breaks down and the general practice would be to phone the company that supplied it and ask them to send a service engineer at the earliest .

But if this air conditioner is fitted with embedded technology it could be programmed in such a way that the AC notifies the user before it breaks down.The next step could be that the AC connects to the supplies ‘s network and puts in a request for a sevice engineer

COMPONENTS OF AN EMBEDDED SYSTEM:

An embedded system comprises the following components:-

Embedded hardware: The embedded hardware mainly consists of a microcontroller with various peripheral IC's. A fixed size volatile memory such as DRAM or SRAM and non-volatile memory such as Flash or EPROM, connected to the microcontroller are an integral part of the device.

Depending on the target applications of the device, the peripherals can include communication devices such as serial controller, Ethernet controller, or a wireless communication controller and other application specific IC's. Many handheld devices, these devices, also have keypads and graphical LCD screens as user interfaces.

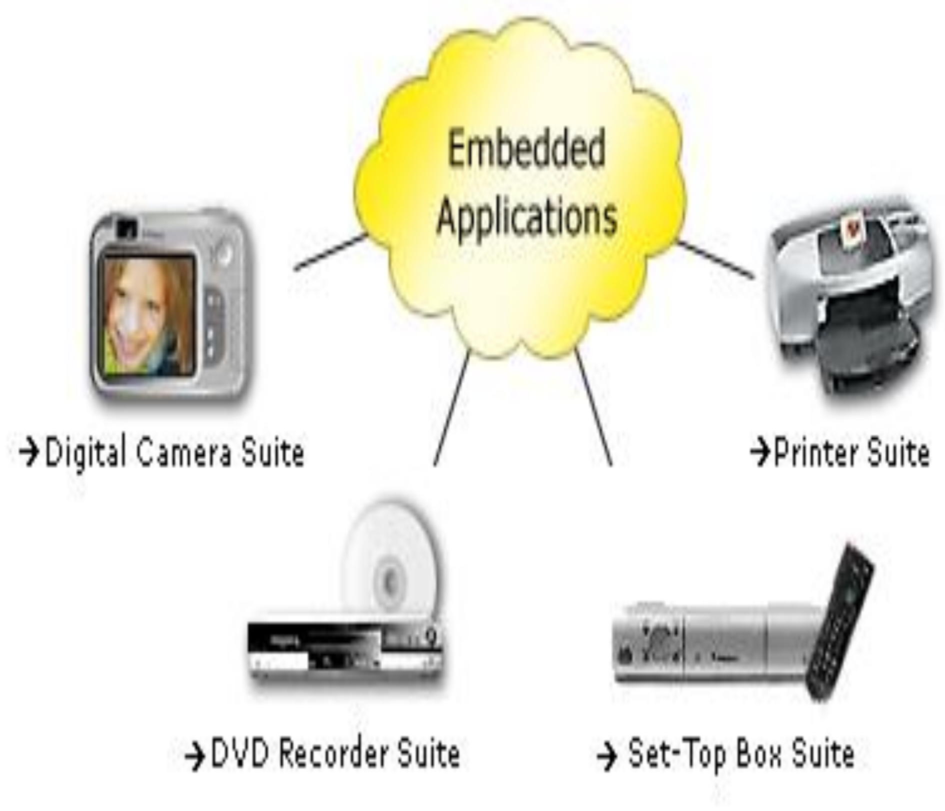
Embedded RTOS: All embedded devices that perform complex functions have an embedded operating system inside. This operating system is typically real time in nature, i.e. it is capable of responding deterministically to time-critical external events.

Device drivers: The lowest level software that acts as glue between the operating systems and the peripheral devices is called the device driver. The device driver software controls every peripheral device that is connected to the microcontroller.

Communication stacks: If the embedded device is capable of communicating to the external world, it has a communication software stack running on the top of the operating system. In order to connect to the internet, the embedded devices need a TCP/IP stack.

Embedded applications: Finally, embedded application software performs the predefined function of the embedded device. This software can support such application as the Internet, e-mail and MP3 decoders. Many of the embedded applications, these days, support sophisticated graphical user interface screens.

EMBEDDED APPLICATIONS



REAL TIME OPERATUING SYSTEMS (RTOS): The concept of real time operating systems is inseparable from when we talk about embedded systems. A real time operating system is built for specific applications and guarantees response to an external event within a specified time constraint. For example, when you suddenly apply brakes to your car to avoid an accident, the intelligent gadget responds immediately.

Real time operating systems are different from desktop operating systems such as windows or UNIX. On a desktop computer, the operating system takes control of the machine as soon as it is turned on and then lets you start your application. In embedded system, you usually link your application and the RTOS.

Your application gets the control first and then starts the RTOS. Many RTOS don't protect themselves as carefully from the applications as the desktop systems. To save memory, they typically include just the services that you need for your embedded systems. Most RTOS allow you to configure extensively before you can link them to the application, so you can leave out any function you don't plan to use.

The RTOS is also better than other software architectures like the Round Robin and Function Queue Scheduling. In RTOS, both the interrupt routine and the task code are in priority order. The worst response time for task code in RTOS is zero plus the execution time for routines. The stability of response when the code changes are very good. The only isadvantage with the RTOS is that the architecture of RTOS is most complex as compared to Round Robin and Function Queue Scheduling.

CHALLENGES FACED BY THE EMBEDDED TECHNOLOGY:

Creating computer hardware requires precise engineering and design. And when it is an embedded system, it has to be more than reliable besides being low in cost and sufficiently high on performance. Often an organization's embedded experts are directly involved in putting together embedded designs.

A thorough design analysis with other software counterparts produces the best possible solutions and then the product is delivered. In this varied discipline, it is the most important decision, which makes the product in tune with international standards.

It also involves careful selection of chips and then a thorough analysis to ensure that the selected chips will work and interact the way you intend for a particular application. The design has to be perfect and there is to be no room for failures.

In short, the challenge is to provide affordable, highly integrated solutions while meeting stringent requirements for safety, security, reliability and availability and real tile performance, and at the same time, keeping the cost low. This calls for a disciplined approach and of course the time factor to deliver the goods.
Embedded System need huge investment in R&D.There is need for technology branding ,which is lacking.

EMBEDDED SOFTWARE DEVELOPMENT TOOLS:

Host and target machines:. In the embedded world, there are many reason developers to do their actual programming work on a system other than the one on which the software will eventually run. The system that you ship may or may not have a keyboard, a screen, a disk drive and the other peripherals necessary for programming. It may not have enough memory to run a programming editor.

Therefore most programming work for embedded systems is done on host- a computer system on which all the programming tools run. It's only after the program has been written, compiled, assembled and linked that is moved on the target system, which is shipped to customers. Some people use the word 'work-station' instead of 'host'.

Cross-compilers: Most desktop system used as hosts come with compilers, assemblers, linkers and building programs that will run on host. These tools are called 'native' tools. The main function of the compiler is to run on your host computer but produces the binary instructions that will be understood by your target microprocessor. Such a program is called a cross-compiler. In theory, a program that compiles without error on your native should also compile without error on the cross-compiler. However, in practice, you should expect that another would not accept certain instructions accepted by one compiler. The fact that your program works on your host machine and compiles cleanly with your cross-compiler is no assurance that it will work on your target system. Because of this, you should accept a different set of warnings from your cross-compiler.

Cross-assemblers and tool chains: Another tool that you'll need if you must write your programming in Assembly language is a cross-compiler. The cross-assembler runs on your host but produces binary instructions appropriate for the target. There is no point in expecting that appropriate input for the cross-assembler has any relation to the input for the native assembler. The tools must be compatible with one another. A set of tools that is compatible with one another called as tool-chain.

Linker/Locator for Embedded Software: Although the job of the cross-compiler is same as that of the native computer-read in source file and produce a object file for the linker-a object file suitable for the linker for an embedded system must do a number of things differently from a native linker. In fact, two programs are different enough that linkers for embedded system are often called locators.

Address resolution: The first difference between a native linker and the compiler is the nature of file both can generate.

The native linker creates a file on the disk drive of the host computer that can be read by the part of the operating system called loader. Whenever the user requests to run the program, copies the program from disk to memory and then does various other processing jobs before starting the program

The locators, by contrast, create a file that will be used by some program that copies output to the target system. Later, the output from the locator will have to run on its own.

In most embedded systems, there is no loader. When the locator is done, its output is copied into the target system. Therefore the locator knows where in memory, the program will reside and fix up all the address. Locators have machines that allow you to tell them where the program will be on the target system.

Locators use any number of different output file formats and various tools. Clearly the tools you are using to load your program into the target system, must understand the file format your locator produces. The details of this format are unimportant. These are typically fairly simple. One common format is simply the binary image that is to be copied into the ROM.

Locating program components properly: **A**nother issue that the locators must resolve in the embedded environment is that some parts of the program need to enter the ROM and some parts of the program need to end up in the RAM.

Most tool chains deal with the problem by dividing programs into segments. Each segment is a piece of the program that the locator can place in the memory independently of the other segments. For example, the instructions that make up the program, which will go into the RAM, go into one collection of segments, and data, which will go into the RAM, goes into another collection of segments.

Segments solve another problem that embedded systems programmers must cope with. Whereas application programmers typically don't care wherein memory the instructions end up, it is an important consideration for at least some of the code in every embedded system. For example, when the microprocessor is turned on, it begins executing instructions at a particular address. To ensure that the first instruction of the program is in that particular address, the embedded system programmer puts the start-up code (usually some piece of assembly code) in its own segment and tells the locator to put that segment at the magic address.

Most cross compilers divide each module they compile into two or more segments. Some cross compilers put each function into a separate segment, while some cross compilers build one segment for all of the codes in the module.

Cross assemblers also specify the segment into which the output from the assembler should be placed. Unlike cross compilers, cross assemblers don't have any default behavior. We have to specify the segments in which each part of our code is to reside.

Locator places segments in memory: The locator will place each segment one after the other in the memory, starting with the given address. To set up the instruction properly for the locator, you must know the names of segments into which the cross compiler divides the modules. These are typically in documentation for the cross compiler.

The locator offers some others feature also. For example, if we specify the address ranges of RAM and ROM, the locator will warn you if your program doesn't fit within those addresses. If we specify an address at which the segment is to end, the locator will place the segment below that address; this is useful in operating with stacks.

Each segment can be assigned to a group, and then the locator is told about where the groups will go instead of dealing with the individual segments. If the cross compiler puts the code from each module into a separate segment, or worse, puts the code from each function into a separate segment, it is convenient if all of these segments belong to one group, perhaps one called ‘code’, allowing to tell the locator where to put that group without worrying about individual segments.

Execution out of RAM: RAM is typically faster than various kinds of ROM’s and Flash memories. For many systems, this speed difference is irrelevant, because even the slower ROM’s are fast enough to keep up with the microprocessor. Systems that use the fastest microprocessors can execute more rapidly if the program is stored in RAM rather than in ROM. Such systems cannot rely on RAM to store their programs. Instead they store their programs in ROM and copy them to RAM when the system starts up.

The start up code runs directly from ROM and therefore executes slowly. It copies the rest of the code into the RAM, then calls or jumps at some entry point, after which the program can run at a higher speed. Sometimes, the program is compressed before it is placed in ROM and the start up code decompresses it as it copies it to RAM. A system that does this, places a new requirement upon its locator: the locator has to build a program that can be stored at one collection of addresses but then execute properly after being copied to another collection of addresses.

APPLICATIONS OF EMBEDDED SYSTEMS:

The various applications of embedded technology are:

Automatic teller machines (ATM): These are machines that will give you the money on click of some buttons. So they are popularly known as All Time Money Machine.

Cellular telephones and telephones switches: The telephones with legs that can walk with you and accept making calls can do various other things for you. Now it has mad possible to withdraw the money on telephones and to make business transactions.

Household appliances: They include washing machine, television sets, microwave ovens, etc and many more.

Inertial guidance systems in aircraft and missiles: These are GPS controlled pilot-less planes which are controlled by a from it's chamber on the ground.

Multifunction wristwatches: That days are gone when the wristwatches are used only to know the time but in fact now a days they are equipped with modern gadgets that besides time can find out direction, can guide you in dark, can tell the direction and at last when fitted with LCD's display can provide you the access to the internet.

Medical equipment: The fictions have become facts that you can be treated or operated for bypass, angiography or for the appendices without a cut on the body and without any admissions into the hospitals i.e. you are discharged on the same day the day you enter.

Factory automation: The factories got automated i.e. the tedious processes like controlling the temperature in the boiler, getting the exact ratios of the chemicals and many more related processes that involve human error are
eliminated and you get the exact figures that ultimately increase the productivity and minimizes accidents.

Personal digital assistants: These are new generations' mobile phones that enable you to access internet and to do all the computer related work on to your palm-top. These are pocket computers that are known as palm-top.

Programmable logic controllers: PLC's are the logical programmable devices that can be programmed to perform various activities and by changing the logic the entire function can be changed or inverted.

Measurement equipment likes oscilloscopes, logic analyzers and spectrum analyzer

Audio/visual equipment: The audio/ visual equipment have all been digitalized and they can be used for various purposes and are used in multiple applications.

<u>Personal computer peripheral office equipment</u>: It is an interface device that is used to interface your personal computer at home with your office PC.

<u>Communication equipment</u>: Communication equipment like mobiles, satellites and wireless use embedded technology for their communication which made it possible to talk or to connect anywhere in the world within few minutes.

FUTURE TRENDS OF EMBEDDED TECHNOLOGY:

Some future trends of the embedded technology are given below:

Adaptive cruise control: The day is not far when all the automobiles would interact with computers on their dashboards. As a matter of fact, vehicles all over the world are now fitted with intelligent devices that make the vehicle respond to various factors-be it climate control, braking, sudden acceleration or even self repair of modules.

In the near future, you won't need to check oil levels or spark plugs or any other fitting. A drive along the highway would be a pleasure with the computer inside doing the varied calculations for you. The adaptive cruise control technology used in cars keeps advancing to new levels of safety. A microwave radar unit or a laser transceiver is fixed on the front of the car to determine the distance and relative speed of any vehicle on its path. The AC allows cars to keep safe distance from other vehicles on busy highways. The driver can set the speed of his car and the distance between his and other cars. When the traffic slows down, the vehicle speed is altered using moderate braking to maintain a constant between cars.

In advanced systems, just in case the driver overspends or suddenly falls asleep, the auto traction system takes over ant guides the car to safe halt. And if you have programmed it right, the GPS in the car would take you to your destination. So right from brakes to automatic traction control to air bags and fuel-air mixture control, the 'intelligence' takes over.

A few advanced car prototypes with embedded systems have been tried and tested where even if a smart thief has managed to break in through the car, the car doesn't start up. Even if it does, the computing in the car would lock the steering wheel and cut off the fuel injection supply. In the meantime, a signal is sent to the nearest police station and the owner informing them about the threat.

Telemedicine: The biomedical card developed by Mistral is a centralized patient monitoring system that allows remote monitoring of around 32 patients at a given time through a central computer. This DSP based system is capable of acquiring the ECG or EEG data from patients and analyzing it. It can process a maximum of 32 channels using the state of the art DSP in a PC environment.

Security: Regarding security in this application, it is said that it is a totally encrypted solution. It is not in a closed loop. The expert doctor at the remote location would guide properly and his assistants would act accordingly. So there is still the human aspect involved.

Tele-matics: Tele-matics is the vehicle's capability to communicate with the outside world and or the vehicle operator. It is a combination of telecommunication and computing. It wont be long when, if you are driving a car, you would be able to answer your phone call and at the same time guide yourself through the traffic.

Mistral software has developed a text-to-speech and speech recognition technology to give the car occupants the ultimate comfort. GPS navigation guides you safely through the traffic. The GPS interface in the car pinpoints your exact location on a map. In case, the GPS signal cant be received due to high density of tall buildings and other magnetic interferences, the ‘dead reckoning technique’ which works for short duration guides you effectively.

<u>Bluetooth application</u>: Bluetooth is the term used to describe the protocol of a short range (10 meters) frequency hopping radio link between devices. It is the global specification for wireless connectivity. Bluetooth supports only 780 kbps, which may be used for unidirectional data transfer at 721 kbps or for symmetric data transfer at up to 432.6kbps.

Targeted applications include PC and peripheral networking, hidden computing and data synchronization such as for address books and calendars. Other applications in the future would include networking of home appliances. Bluetooth is perfect for file transfer and printing applications.

CONCLUSION

The world of embedded systems is a dreamer's paradise with unlimited possibilities. Imagine you control all the systems around you just by a simple gesture and the things respond to you as if it was some magic. This could be possible with embedded systems.

As microprocessors are becoming smaller and cheaper, more and more products are becoming 'smart' with microprocessors embedded in them. From handheld devices such as personal digital assistants and smart cell phones to automobiles and rocket propellers, embedded systems are at the heart of it all.

The demand for embedded systems is, in fact, rising in integrated embedded solutions across various industry verticals. As the world is experiencing groundbreaking research in hardware technology (nano-technology, quantum mechanics, etc), packing more power into single chip will soon be a reality.