Embedded Systems

Parineeth M Reddy

Embedded systems differ from general purpose computers in many aspects. This article introduces the reader to embedded systems.

Parineeth M Reddy is a computer science BE graduate currently working as a software engineer at Philips Software Centre, Bangalore. His research interests include networking and distributed computing.

Introduction

An embedded system is a microprocessor-based system that is incorporated into a device to monitor and control the functions of the components of the device. They are used in many devices ranging from a microwave oven to a nuclear reactor. Unlike personal computers that run a variety of applications, embedded systems are designed for performing specific tasks. An embedded system used in a device (for instance the embedded system in washing machine that is used to cycle through the various states of the washing machine) is programmed by the designers of the system and generally cannot be programmed by the end user. Embedded systems possess the following distinguishing qualities.

Reliability: Embedded systems should be very reliable since they perform critical functions. For instance, consider the embedded system used for flight control. Failure of the embedded system could have disastrous consequences. Hence embedded system programmers should take into consideration all possibilities and write programs that do not fail.

Responsiveness: Embedded systems should respond to events as soon as possible. For example, a patient monitoring system should process the patient's heart signals quickly and immediately notify if any abnormality in the signals is detected.

Specialized Hardware: Since embedded systems are used for performing specific functions, specialized hardware is used. For example, embedded systems that monitor and analyze audio signals use signal processors.

Keywords
Embedded system, micro-controller, simulator, emulator.

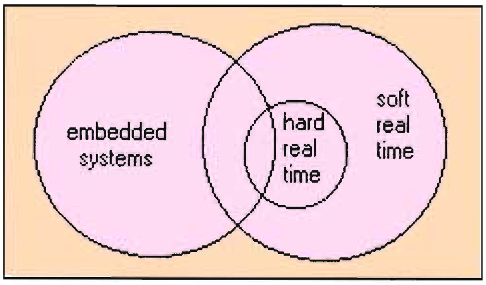


Figure 1. Embedded and real time systems.

Low cost: As embedded systems are extensively used in consumer electronic systems, they are cost sensitive. Thus their cost must be low.

Robustness: Embedded systems should be robust since they operate in a harsh environment. They should endure vibrations, power supply fluctuations and excessive heat. Due to limited power supply in an embedded system, the power consumed by the components of the embedded system should be kept to a minimum.

Embedded systems are often confused with real-time systems. A real-time system is one in which the correctness of the computations not only depends on the accuracy of the result, but also on the time when the result is produced. This implies that a late answer is a wrong answer. A hard real-time system should always respond to an event within the deadline or else the system fails. Soft real-time systems have less severe time constraints. All embedded systems are not real-time systems and vice-versa. *Figure* 1 shows the relation between embedded and real-time systems.

Components of an Embedded System

Embedded systems have the following components.

Embedded systems should be highly reliable, responsive, robust and inexpensive.

Processor: A processor fetches instructions from the memory unit and executes the instructions. An instruction consists of an instruction code and the operands on which the instruction should act upon. The format of instruction code and operands of a processor is defined by the processor's instruction set. Each

Embedded systems often employ specialized processors such as digital signal processors.

type of processor has its own instruction set.

Performance of the system can be improved by using specialized processors. These dedicated processors implement algorithms in hardware using building blocks such as hardware counters and multipliers. For example, digital signal processors are particularly designed for implementing digital signal processing algorithms such as the fast Fourier transform (FFT) algorithm and filtering algorithms. For the input $x(n)$, the output $y(n)$ of an FIR filter is given by the equation

$$y(n) = \sum_{k=1}^{m} x(n-k)*h(k) \quad \text{for } n=1,2,\ldots,$$

The equation clearly shows that filtering involves many multiply and accumulate operations. Hence digital signal processors use a dedicated circuit for performing multiply and accumulate operations.

Some embedded processors have special fuzzy logic instructions (such as the fuzzy AND instruction). This is because inputs to an embedded system are sometimes better represented as fuzzy variables. For instance, the mathematical model for a control system may not exist or may involve expensive computing

Fuzzy Logic

In fuzzy logic variables have value in the range [0.0, 1.0], with 0.0 representing absolute falseness and 1.0 representing absolute truth. For example, consider the statement:

"Ajay is tall."

If Ajay's height is 1.7 m, then we might assign the statement the truth value of 0.80. Fuzzy logic is used in many applications.

Consider a thermostat that regulates the temperature in a room. The variable temperature can be expressed using the states 'cold' and 'hot'. Defining the boundaries of these states is difficult. If an arbitrary value, say 23°C, is used to divide the 'cold' state (represented by 0) from the 'hot' state (represented by 1), there will be a discontinuous change of state at 23°C. This problem can be solved using fuzzy logic. Temperature can then be expressed as 0.4 'cold' and 0.6 'hot', or 0.3 'cold' and 0.7 'hot' thereby avoiding abrupt changes.

power. Fuzzy logic can be employed for such control systems to provide a cost-effective solution.

Memory: The memory unit in an embedded system should have low access time and high density (a memory chip has greater density if it can store more bits in the same amount of space). Memory in an embedded system consists of ROM (only read operations permitted) and RAM (read and write operations are permitted). The contents of ROM are non-volatile (power failure does not erase the contents) while RAM is volatile. The classification of the ROM is given in *Figure* 2. ROM stores the program code while RAM is used to store transient input or output data. Embedded systems generally do not possess secondary storage devices such as magnetic disks. As programs of embedded systems are small there is no need for virtual storage.

Embedded systems generally do not possess secondary storage devices such as magnetic disks.

Peripherals: Peripherals are the input and output devices connected to the serial and parallel ports of the embedded system. Serial ports transfer one bit at a time between the peripheral and the microprocessor. Parallel ports transfer an entire word consisting of many bits simultaneously between the peripheral and the microprocessor. The microprocessor generally communicates with the peripherals using a programmable interface device. Programmable interface devices provide flexibility since they can be programmed to perform I/O on different peripherals. The microprocessor monitors the inputs from peripherals and performs actions when certain events occur. For instance,

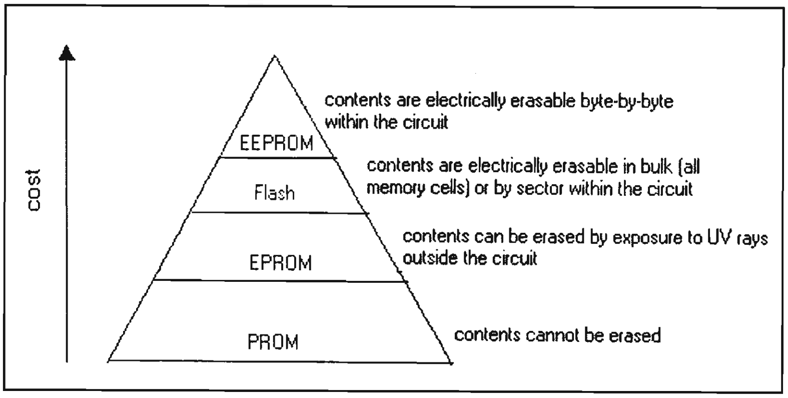

cost
EEPROM
contents are electrically erasable byte-by-byte within the circuit
Flash
contents are electrically erasable in bulk (all memory cells) or by sector within the circuit
EPROM
contents can be erased by exposure to UV rays outside the circuit
PROM
contents cannot be erased

Figure 2. Classification of ROMS.

when sensors indicate that the level of water in the wash tub of a washing machine is above the preset level, the microprocessor starts the wash cycle.

Hardware Timers: The clock pulses of the microprocessor periodically update hardware timers. The timers count the clock pulses and interrupt the processor at regular intervals of time to perform periodic tasks.

Due to the absence of secondary storage devices in an embedded system, program code and constant data reside in the ROM. During execution of the program, storage space for variables is allocated in the RAM.

Software: Due to the absence of secondary storage devices in an embedded system, program code and constant data reside in the ROM. During execution of the program, storage space for variables is allocated in the RAM. The programs should execute continuously and should be capable of handling all possible exceptional conditions. Hence the programs generally do not call the function exit.

Real-time embedded systems posses an RTOS (real-time operating system). The RTOS consists of a scheduler that manages the execution of multiple tasks in the embedded systems. Unlike operating systems for the desktop computers where scheduling deadlines are not critical, an RTOS should schedule tasks and interrupt service routines such that they are completed within their deadlines. The RTOS provides features that simplify the programmer's job. For example, an RTOS provides semaphores that can be used by the programmer to prevent multiple tasks from simultaneously writing into shared memory.

With the recent developments in VLSI, the processor, memory, peripherals and the interfaces to the outside world are integrated into a single chip resulting in a microcontroller.

With the recent developments in VLSI, the processor, memory, peripherals and the interfaces to the outside world are integrated into a single chip resulting in a microcontroller. *Figure* 3 shows the details of 8051 microcontroller.

Example of an embedded system

Consider the anti-lock braking system in a car. When brakes are applied to a car traveling at high speed, the wheels stop much more quickly than the car and 'lock up'. An anti-lock brake system ensures that the wheels of the car gradually stop when

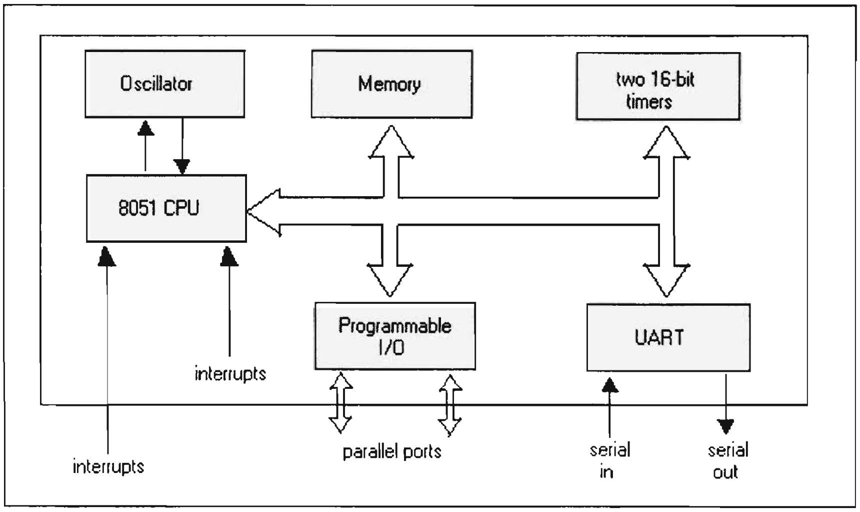


Figure 3. Block diagram of 8051 microcontroller.

brakes are applied . The anti-lock brake system has the following components.

Speed sensors: The speed sensors are located at each wheel and indicate the speed of the wheels.

Valves: Valves regulate the pressure applied on the brakes. An open valve allows pressure to be transmitted to the brake. A closed valve prevents the pressure on the brakes from rising even though the driver pushes the brake pedal harder

Microcontroller: The microcontroller monitors the speed sensors and checks for rapid decelerations in the wheels. On detecting a rapid deceleration, the microcontroller reduces the pressure on the brake by closing the valves until it sees an acceleration. The microcontroller then increases the pressure on the brakes by opening the valves until it sees a deceleration. This cycle continues and results in the wheel slowing down at the same rate as the car.

Polling and Interrupts

Devices connected to the embedded system are monitored using either polling or interrupts. In the polling scheme, the processor keeps checking the status of every device and service device as

In the interrupt scheme the processor has interrupt pins to which devices are connected. For each interrupt, there is a corresponding interrupt service routine that is executed when the interrupt is raised.

illustrated in the following pseudo-code:

```
while (TRUE)
{
        read status of device A
        if (device A needs service)
        {
                service device A
        }
        read status of device B
        if (device B needs service)
        {
                service device B
        }
        read status of device C
        if (device C needs service)
        {
                service device C
        }
}
```

Although polling is a simple technique, it is generally not used due to its drawbacks. Suppose the code that services device A and B takes 10 milliseconds each to execute and device C has to be serviced within 15 milliseconds. There is likelihood that device C is not serviced within the deadline. In the polling scheme, processor wastes its clock cycles by checking the status of idle devices. Thus an interrupt scheme is usually used.

In the interrupt scheme the processor has interrupt pins to which devices are connected. For each interrupt, there is a corresponding interrupt service routine that is executed when the interrupt is raised. The addresses of the interrupt service routines are stored in an interrupt vector table.

When a device raises an interrupt the processor completes executing the current instruction and saves the program counter (program counter is a register in the processor that stores the address of next instruction to be executed) on the stack (in the RAM). The processor then sends an interrupt acknowledge signal. Depending on the interrupt pin that is active the processor fetches the address of the corresponding interrupt service routine from the interrupt vector table and begins to execute the routine. After completing the interrupt service routine the processor reloads the program counter with the value stored in the stack and resumes execution of the program that was interrupted. Since interrupts have different priorities, when two or more devices simultaneously raise an interrupt the highest priority device is serviced first.

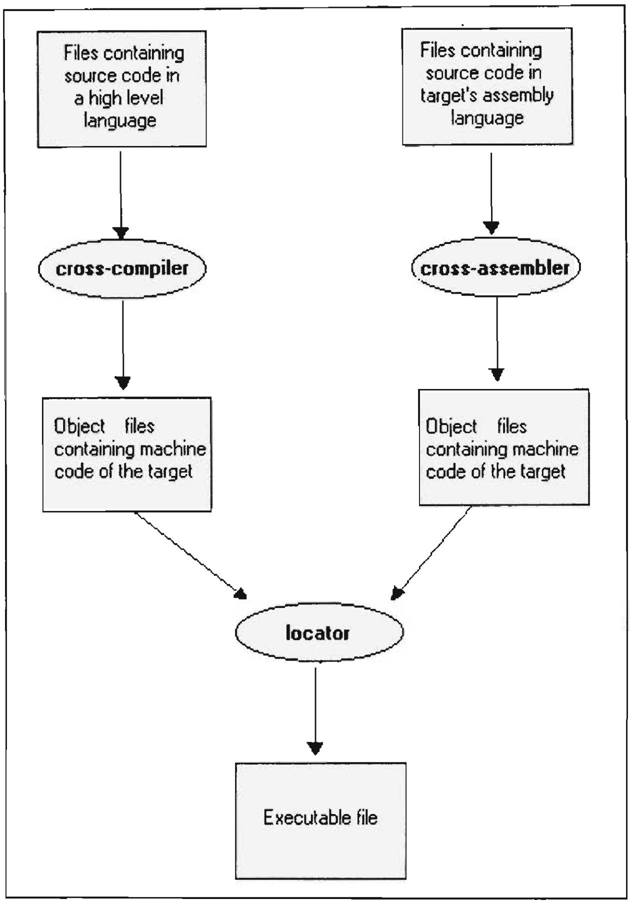
Most of the programming work for an embedded system, which includes writing, compiling, assembling and linking the program, is done on a general purpose computer called a host that has all the required programming tools.

Embedded Software Development

Programmers who write programs for desktop computers do their work on the same kind of computer on which their application will run. A programmer developing a program to run on a Linux machine edits the program, compiles it and debugs it on a Linux machine. This approach cannot be used for embedded systems due to the limited hardware and software in an embedded system. For example, the absence of a keyboard in the embedded system rules out editing a program in the embedded system. So, most of the programming work for an embedded system, which includes writing, compiling, assembling and linking the program, is done on a general purpose computer called a host that has all the required programming tools. The final executable consisting of machine code is then transferred to the embedded system (also referred to as target). *Figure* 4 shows the development of embedded software on the host.

Programs are written on the host in a high level language (such as C) or assembly language of the target system's processor. The program files written in the high level language are compiled on the host using a cross-compiler to obtain the corresponding object files. The assembly language files are assembled on the host using a cross-assembler to obtain the object files. The object files produced by cross-compilers and cross-assemblers contain instructions that are understood by the target's processor (na-

Figure 4. Software development on the host.


Files containing source code in a high level language
Files containing source code in target's assembly language
cross-compiler
cross-assembler
Object files containing machine code of the target
Object files containing machine code of the target
locator
Executable file

tive compilers and assemblers on the other hand produce object files containing instructions that are understood by the host's processor).

The object files are linked using a specialized linker called locator to obtain the executable code. This executable code is stored in the ROM. Since the program code already resides in memory, there is no need for a loader in an embedded system. In personal computers, on the other hand, the loader has to transfer the program code from the magnetic disk to memory to execute the program.

The binary code obtained by translating an assembly language program using an assembler is smaller and runs faster than the binary code obtained by translating a high level language using a compiler since the assembly language gives the programmer complete control over the functioning of a processor. The advantage of using a high level language is that a program written in a high level language is easier to understand and maintain than a program written in assembly language. Hence time-critical applications are written in assembly language while complex applications are written in a high level language.

Time-critical applications are written in assembly language while complex applications are written in a high level language.

The programs developed for an embedded system are tested using the following tools.

Simulator: A simulator is software tool that runs on the host and simulates the behavior of the target's processor and memory. The simulator knows the target processor's architecture and instruction set. The program to be tested is read by the simulator and as instructions are executed the simulator keeps track of the values of the target processor's registers and the target's memory. Simulators provide single step and breakpoint facilities to debug the program. Simulators cannot be used if the embedded system uses special hardware that cannot be simulated and the only way to test the program is to execute it on the target. Although simulators do not run at the same speed as the target microprocessor, they provide details from which the time taken to execute the code on the target microprocessor can be determined. For instance, the simulator can report the number of target microprocessor's bus cycles taken to execute the code. Multiplying this value with the time taken for one bus cycle gives the actual time taken by the target microprocessor to execute the code.

Emulator: An emulator is a hardware tool that helps in testing and debugging the program on the target. The target's processor is removed from the circuit and the emulator is connected in its place. The emulator drives the signals in the circuit in the same way as the target's processor and hence the emulator appears to be the processor to all other components of the embedded system. Emulators also provide features such as single step and breakpoints to debug the program.

Suggested Reading

[1] David E Simon, *An Embedded Software Primer*, Addison-Wesley Pub Co, 1999.

[2] Michael Barr, *Programming Embedded Systems in C and C ++*, O'Reilly & Associates, 1999.

[3] Myke Predko, *Programming and Customizing the 8051 Microcontroller*, McGraw-Hill, 1999.

Conclusion

Embedded systems have requirements that differ significantly from general purpose computers. The main goal of an embedded system developer is to design a lowest cost system, that performs the desired tasks without failing. Algorithms can be implemented in hardware or software. While the hardware approach improves performance the software approach provides flexibility. Recent developments in hardware-software co-design permit tradeoffs between hardware and software for cost-effective embedded systems.

Address for Correspondence
Parineeth M Reddy
3529, I Cross
13-H Main HAL 2nd Stage
Bangalore 560 008, India.
Email: parineeth@yahoo.com

Making of a Young Scientist

Several factors can influence the making of a scientist at various stages in his life. The formative young years are important, of course. Krishnan, in one of his articles in Tamil says: 'my first love for science came in my 4th form (class 9) in my high school in 1911. Even though my teacher was not a professional scientist, he was good at explaining science in a clear and captivating fashion. His lessons not only sunk deep into our mind but also made us crave for more science. Whether it is physics, geography or chemistry, his teaching style was unique. He did not simply reproduce the lessons from the book. He demonstrated many simple experiments for us and also encouraged us to do experiments ourselves. Very few teachers I know are of this type, and I feel fortunate to have had him as my first science teacher. This master teacher's name is Sri A Subramanya Iyer and he did not stay too long in my school.'

'My real involvement in science came after an year, when my physics teacher asked us to write an 'essay' about Archimides principle. At that time we had just learned the proof of this principle. But, in my article I wrote about an instrument that I constructed. on my own, for measuring the density of solids. A few days later I learned that my instrument is nothing new and it was invented by Nicholas many years ago – "the Nicholas hydrometer", by then text book material.'

'My whole hearted involvement in science came only after seven or eight years, when I got opportunities to read copies of research articles of Prof. C V Raman, then Palit Professor at Calcutta University, which appeared in *Nature*, *Philosophical Magazine* and other journals. This whole culture of eminent scientists publishing their work in 'Science Journals' and that some of our own scientists like Ramanujan, Raman are contributing first rate articles which are very much appreciated by the world came to me as an eye opener. This gave me a new feeling for science, scientists and the new world of science.'

KSK finished his article by saying: 'I relinquished the small job I had and decided to do research in physics and went to Prof. C V Raman at Calcutta. But, he did not agree for my starting research immediately. Only after learning various aspects of physics properly at Calcutta University for two years was I able to join his research group. I had the good fortune of having a five year 'Gurukula vasam' there. These five years turned out to be a festive season in my science life.'

G Baskaran