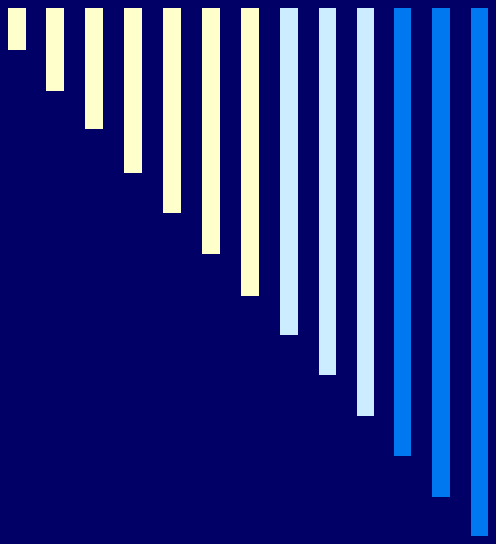


INTRODUCTION TO EMBEDDED SYSTEMS

What is an Embedded system?

- An embedded system is one that has computer hardware with software embedded in it as one of its components.

Or

- We can define an embedded system as **"A microprocessor based system that does not look like a computer".**

Or

- we can say that it is **"A combination of computer hardware and software, and perhaps additional mechanical or other parts, designed to perform a dedicated function. In some cases, embedded systems are part of a larger system or product, as is the case of an antilock braking system in a car ".**

- An **embedded system** is a special-purpose computer system designed to perform certain dedicated functions. It is usually *embedded* as part of a complete device including hardware and mechanical parts. (Wikipedia)

Significance

Due to their compact size, low cost and simple design aspects made embedded systems very popular and encroached into human lives and have become indispensable. They are found everywhere from kitchen ware to space craft. To emphasize this idea here are some illustrations.

Embedded systems everywhere?

Embedded systems span all aspects of modern life and there are many examples of their use.

a) Biomedical Instrumentation – ECG Recorder, Blood cell recorder, patient monitor system

b) Communication systems – pagers, cellular phones, cable TV terminals, fax and transreceivers, video games and so on.

c) Peripheral controllers of a computer – Keyboard controller, DRAM controller, DMA controller, Printer controller, LAN controller, disk drive controller.

d) Industrial Instrumentation – Process controller, DC motor controller, robotic systems, CNC machine controller, close loop engine controller, industrial moisture recorder cum controller.

e) Scientific – digital storage system, CRT display controller, spectrum analyser.

Were the embedded systems existing earlier ?

Yes, We have been enjoying the grace of embedded system quite a long time. But they were not so popular because in those days most of the embedded systems were designed around a microprocessor unlike today's systems which were built around a microcontroller.

As we know a microprocessor by itself do not possess any memory, ports etc. So everything must be connected externally by using peripherals like 8255, 8257, 8259 etc. So the embedded system designed using microprocessor was not only complicated in design but also large in size. At the same time the speed of microprocessor is also a limitation for high end applications.

Why a microcontroller ?

A microcontroller is a single silicon chip with memory and all Input/Output peripherals on it. Hence a microcontroller is also popularly known as a single chip computer. Normally, a single microcomputer has the following features :

- Arithmetic and logic unit
- Memory for storing program
- EEPROM for nonvolatile data storage
- RAM for storing variables and special function registers
- Input/output ports

- Timers and counters
- Analog to digital converter
- Circuits for reset, power up, serial programming, debugging
- Instruction decoder and a timing and control unit
- Serial communication port

So, its no wonder to say that the microcontroller is the most sought after device for designing an efficient embedded system.

What is inside an embedded system ?

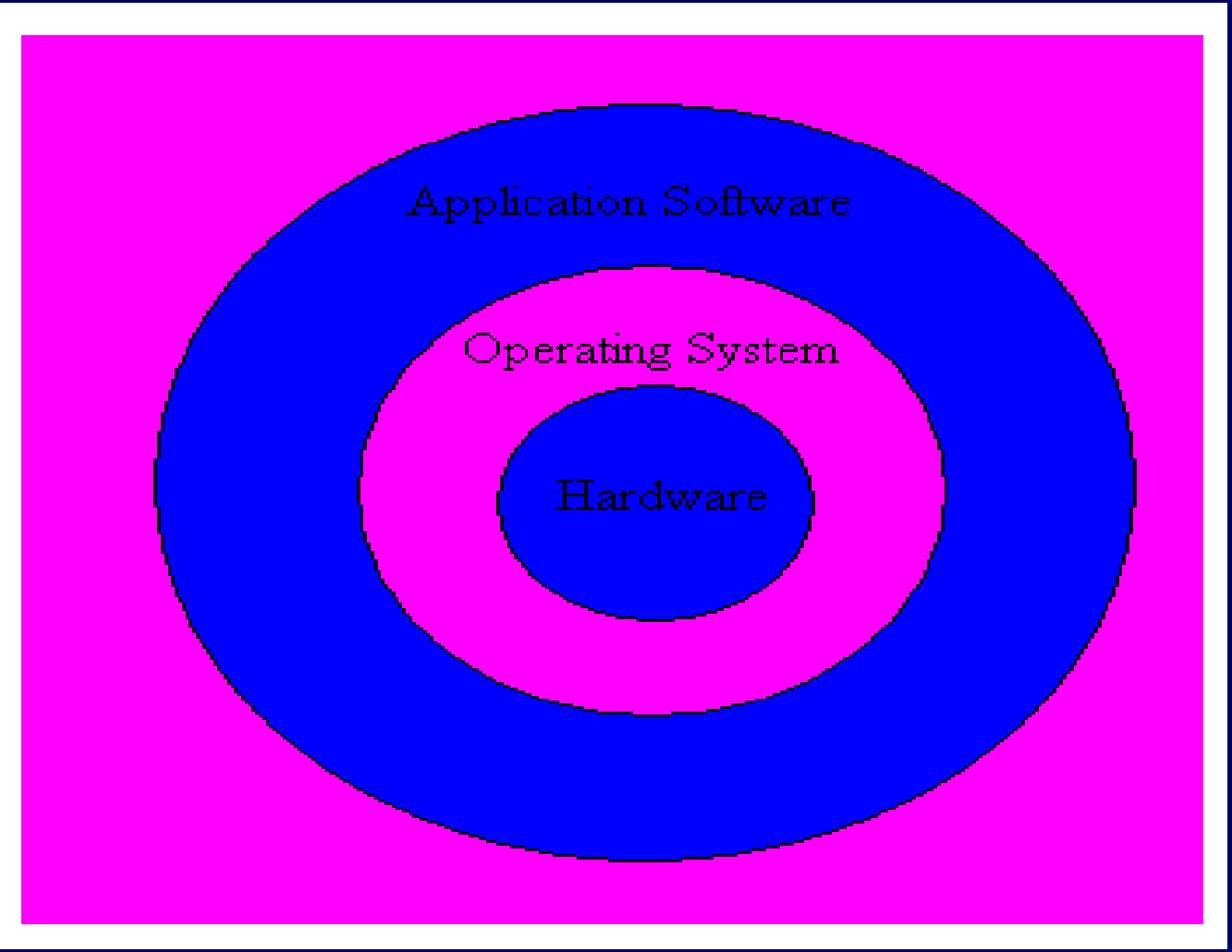
Every embedded system consists of custom-built hardware built around a Central Processing Unit (CPU). This hardware also contains memory chips onto which the software is loaded. The software residing on the memory chip is also called the ‘firmware’.

The 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 fir an operating system and we 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 .

The next slide shows the layered architecture of an embedded system.

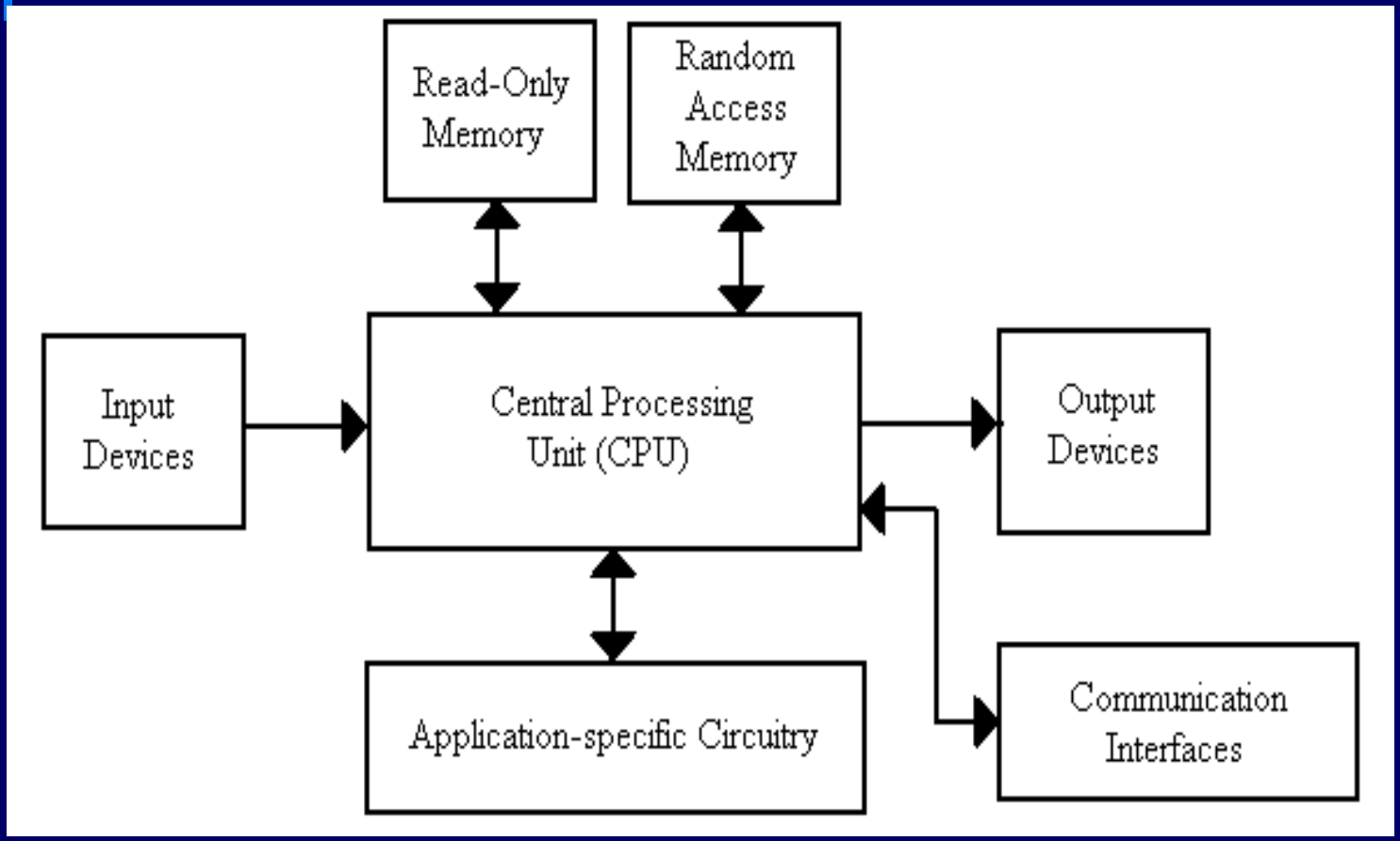


Layered architecture of an Embedded System

Now let us see the details of the various building blocks of the hardware of an embedded system.

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

This slide shows the Hardware architecture of an embedded system



Features of an embedded system

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 dead line 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-effective. Even a reduction of Rs.10 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 .

Classification of Embedded Systems

Based on functionality and performance requirements, embedded systems are classified as :

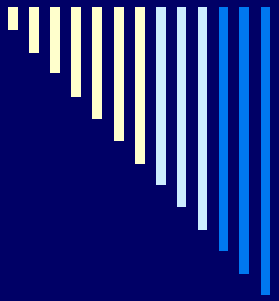
- Stand-alone Embedded Systems
- Real-time Embedded Systems
- Networked Information Appliances
- Mobile Devices

Stand-alone Embedded Systems

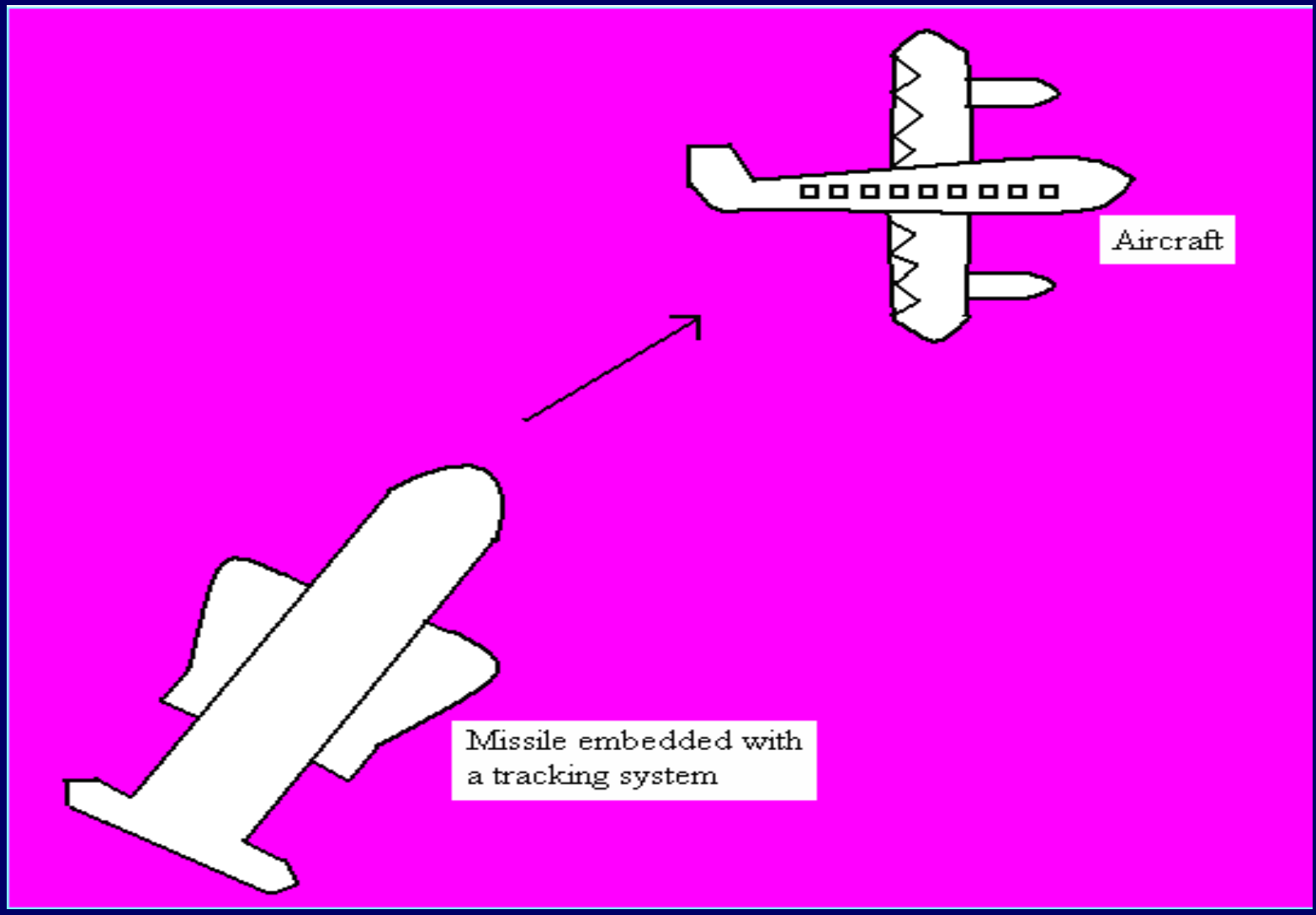
As the name implies, stand-alone systems work in stand-alone mode. They take inputs, process them and produce the desired output. The input can be electrical signals from transducers or commands from a human being such as the pressing of a button. The output can be electrical signals to drive another 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.

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 .



Hard Real-Time Embedded System

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.

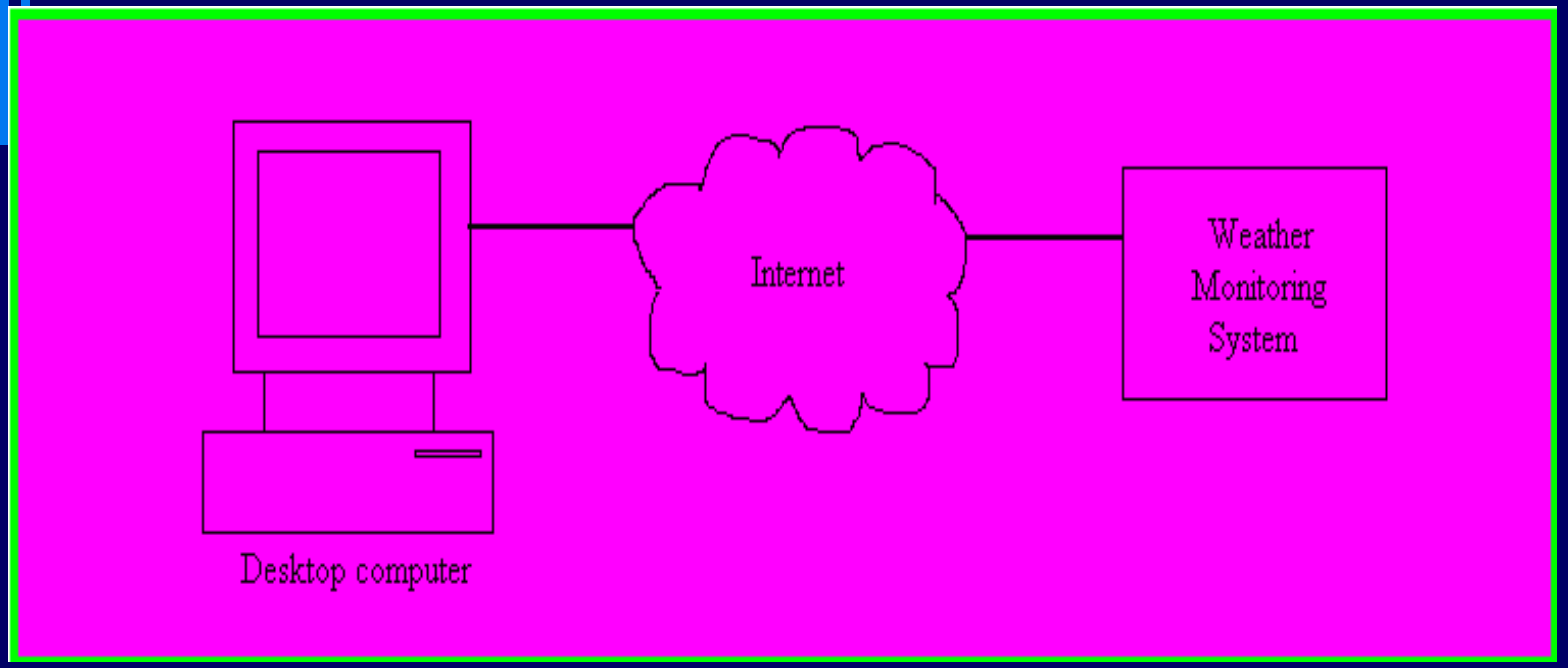
These systems have emerged in recent years. These systems run the protocol TCP/IP stack and get connected through PPP or Ethernet to an network and communicate with other nodes in the network.

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.



This slide 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.

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.

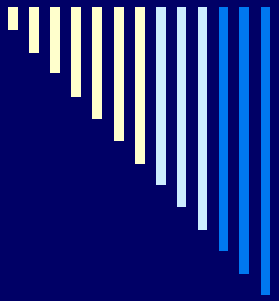
The limitations of the mobile devices – memory constraints, small size, lack of good user interfaces such as full fledged keyboard and display etc. are same as those found in the embedded systems discussed above. Hence, mobile devices are considered as embedded systems.

However, the PDAs are now capable of supporting general purpose application software such as word processors, games, etc.

Languages for Programming Embedded Systems

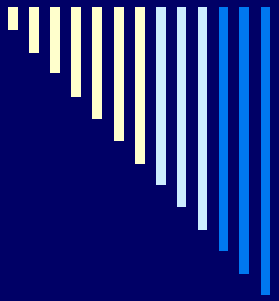
Assembly language was the pioneer for programming embedded systems till recently. Nowadays there are many more languages to program these systems. Some of the languages are C, C++, Ada, Forth, and Java together with its new enhancement J2ME.

The presence of tools to model the software in UML, SDL is sufficient to indicate the maturity of embedded software programming



The majority of software for embedded systems is still done in C language. Recent survey indicates that approximately 45% of the embedded software is still being done in C language.

C++ is also increasing its presence in embedded systems. As C++ is based on C language, thus providing programmer the object oriented methodologies to reap the benefits of such an approach.



C is very close to assembly programming and it allows very easy access to underlying hardware. A huge number of high quality compilers and debugging tools are available for the C language.

Though C++ is theoretically more efficient than C, but some of its compilers have bugs due to the huge size of the language. These compilers may cause a buggy execution.

C language can definitely claim to have more mature compilers C++. Now in order to avail the extra benefits of C++ and plus to avoid buggy execution, experts are doing efforts to identify a subset of C++ that can be used in embedded systems and this subset is called Embedded C++ .

Communication Interfaces

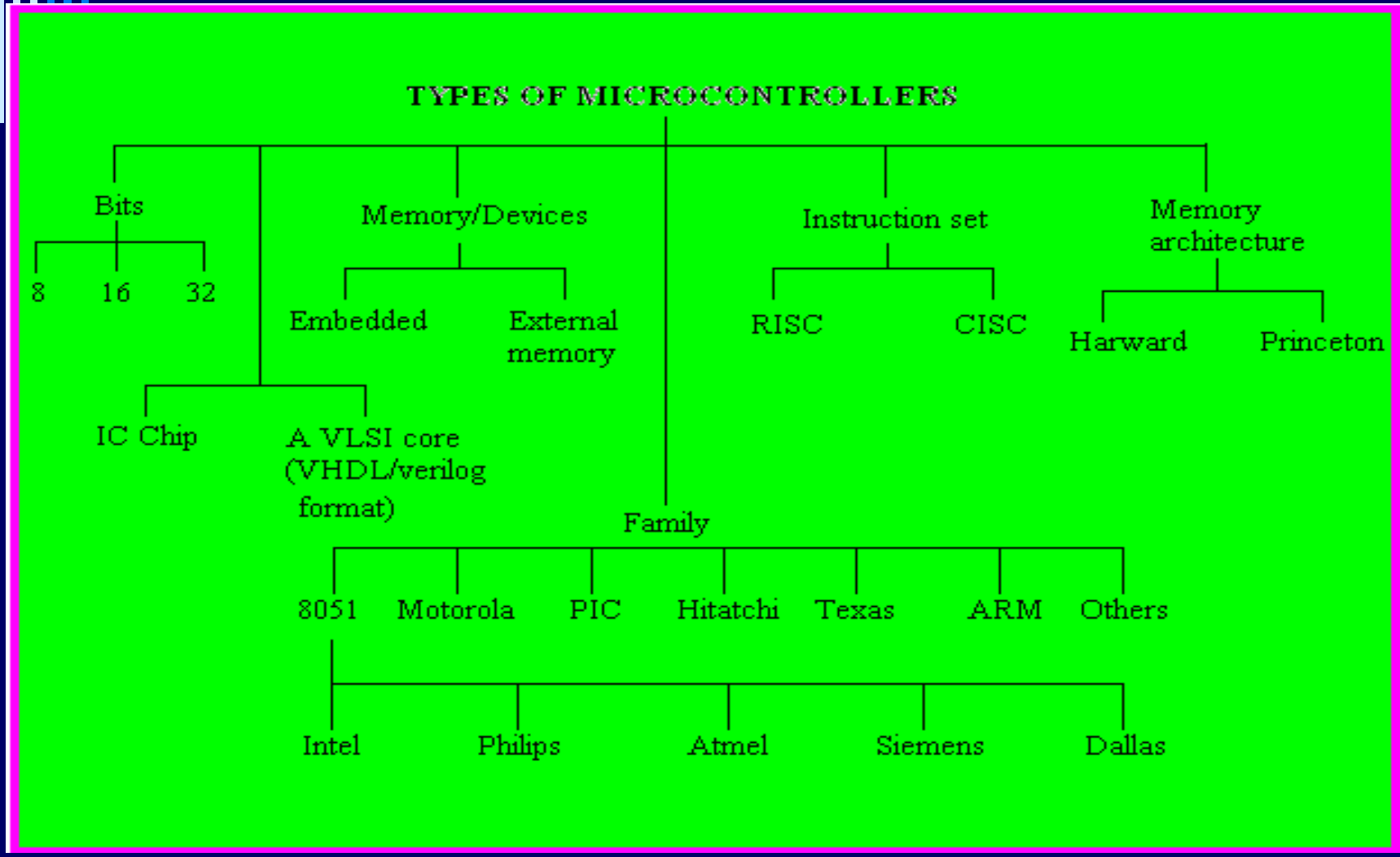
For embedded systems to interact with the external world, a number of communication interfaces are available. They are

- Serial Communication Interfaces (SCI):
  RS-232, RS-422, RS-485 etc
- Synchronous Serial Communication Interface:
  I2C, JTAG, SPI, SSC and ESSI
- Universal Serial Bus (USB)

- Networks:
  Ethernet, Controller Area Network, LonWorks, etc
- Timers:
  PLL(s), Capture/Compare and Time Processing Units
- Discrete IO:
  General Purpose Input/Output (GPIO)
- Analog to Digital/Digital to Analog (ADC/DAC)

Which is the best suited microcontroller for design of embedded system?

There is always a trade off between efficiency and power dissipation. To know this, let us review the various types of microcontrollers and their specifications and the vendors.

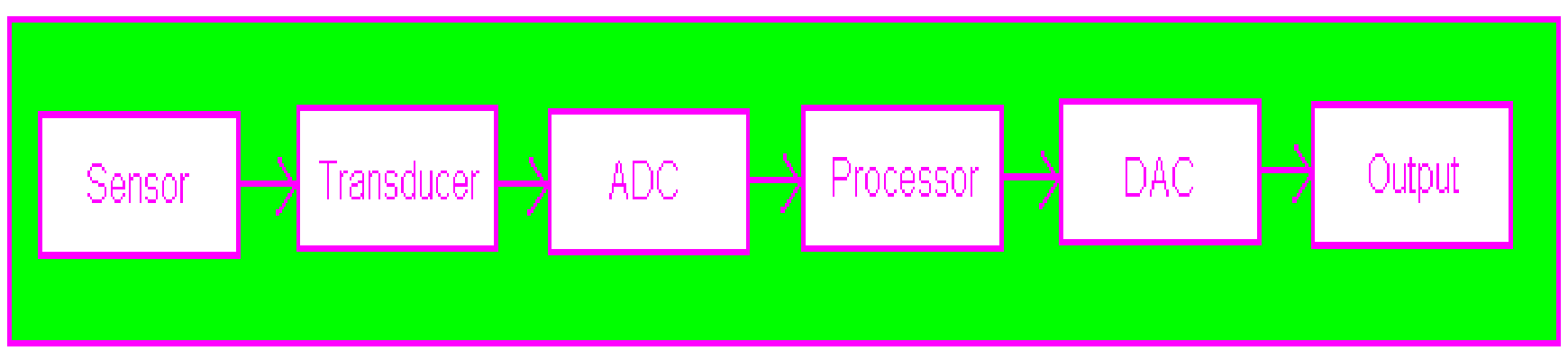

TYPES OF MICROCONTROLLERS
Bits
8
16
32
Memory/Devices
Embedded
External memory
Instruction set
RISC
CISC
Memory architecture
Harward
Princeton
IC Chip
A VLSI core (VHDL/verilog format)
Family
8051
Motorola
PIC
Hitatchi
Texas
ARM
Others
Intel
Philips
Atmel
Siemens
Dallas

From the previous slide we can find that the ARM processor is a strong option for better performance. But when we consider the power consumption, in the case of ARM it is around 400mW and the ATmega1031, AVR microcontroller consumes low power around 16.5mW, but provides low performance.

But the Texas instruments MSP430 with wide range of operation modes consumes only 1.2mW with reasonably good performance. So it is always left to the designer to choose a suitable device according to the requirement.

Design of an embedded system – a Case study

To understand the design of a simple embedded system let us first consider the idea of a data acquisition system. The data acquisition system is shown in the next slide.



Data acquisition system

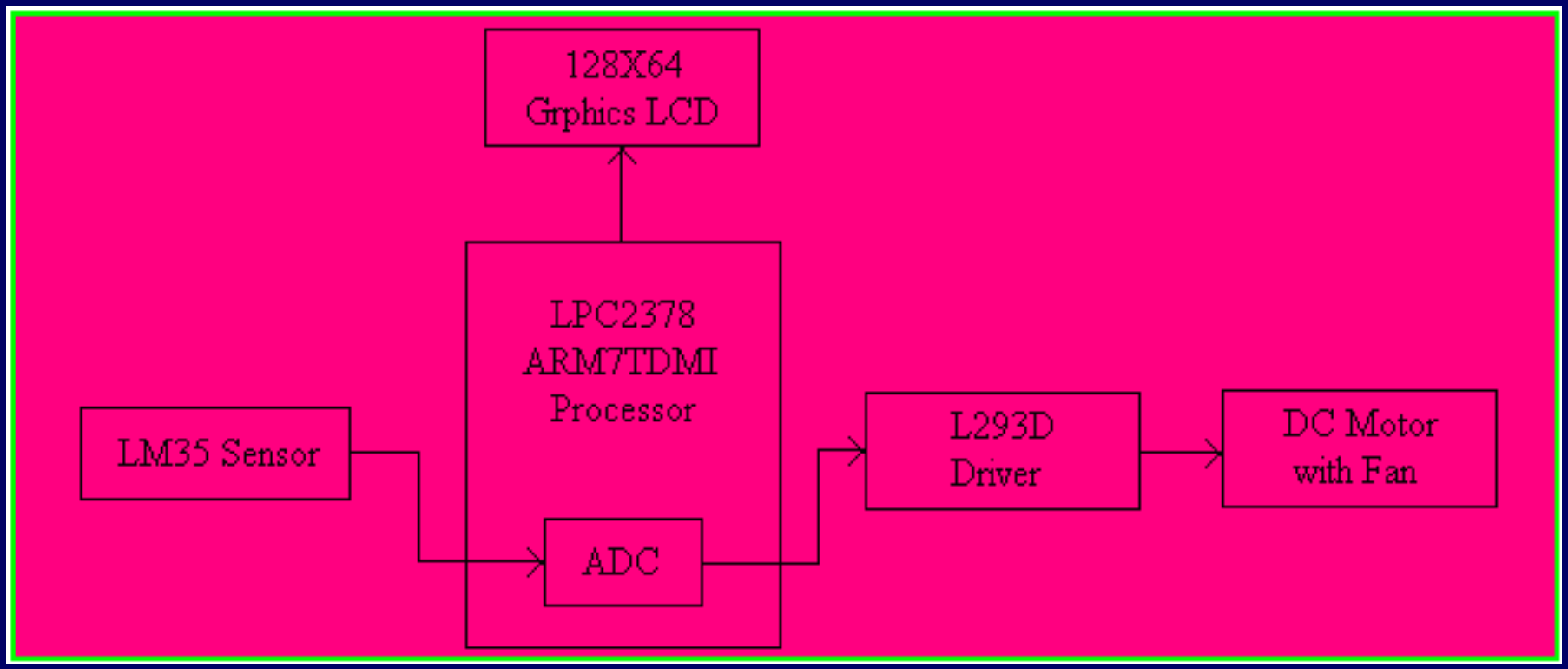
For example let me consider a simple case of temperature measurement embedded system.

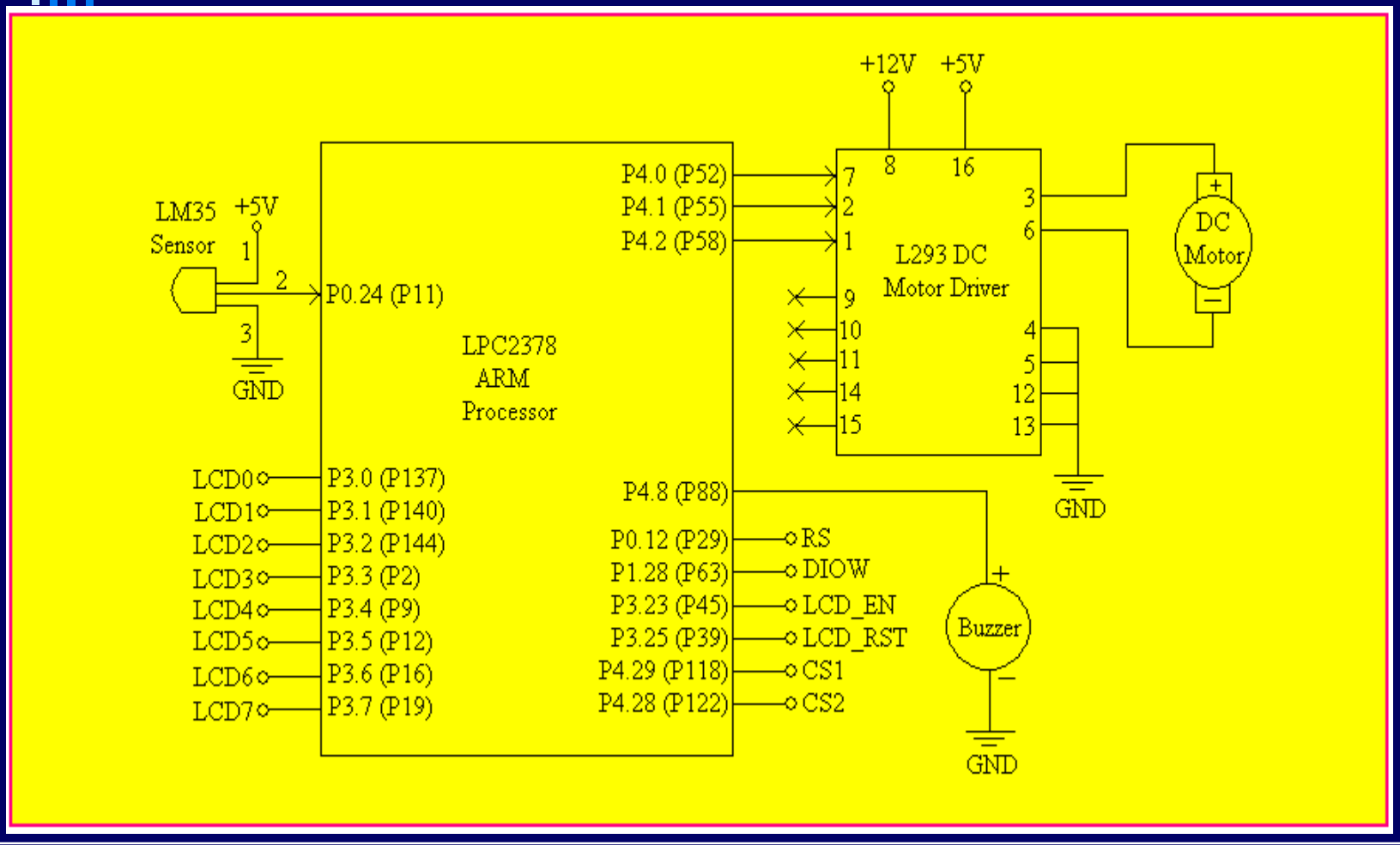
- First we must select a temperature sensor like thermistor or AD590 or LM35 or LM335 or LM75 etc.
- After this the analog data is converted into digital data and at the same time proper signal conditioning is done.

- This digital input is fed to the microcontroller through its ports.
- By developing a suitable program (Embedded C or Assembly) the data is processed and controlled.
- For this purpose keil or Ride or IAR ARM Embedded workbench C compilers can be used.

- Once the program is debugged, and found error free it can be dumped into the microcontroller flash memory using ISP (Philips - Flash magic or any ISP).
- Now, your microcontroller chip acts as an embedded chip.

For the sake of clarity I present the block diagram of a simple embedded system.




+12V
+5V
LM35
Sensor
+5V
1
2
3
GND
P0.24 (P11)
LPC2378
ARM
Processor
P4.0 (P52)
P4.1 (P55)
P4.2 (P58)
7
2
1
8
16
L293 DC
Motor Driver
9
10
11
14
15
3
6
4
5
12
13
DC
Motor
GND
LCD0
LCD1
LCD2
LCD3
LCD4
LCD5
LCD6
LCD7
P3.0 (P137)
P3.1 (P140)
P3.2 (P144)
P3.3 (P2)
P3.4 (P9)
P3.5 (P12)
P3.6 (P16)
P3.7 (P19)
P4.8 (P88)
P0.12 (P29)
P1.28 (P63)
P3.23 (P45)
P3.25 (P39)
P4.29 (P118)
P4.28 (P122)
RS
DIOW
LCD_EN
LCD_RST
CS1
CS2
Buzzer
GND

Embedded C softwares

Keil µvision evaluation version can be downloaded from www.keil.com

Embedded C compiler Ride can be downloaded freely from www.*raisonance*. *com*

Embedded IAR ARM Workbench can be downloaded from *www.**iar**.com*

The various vendors who can supply the microcontroller kits :

1.Power systems, Chennai (www.powersoftsystems .com)

2.Vi-microsystems - Chennai(*www.**vimicrosystems**.com)*

3.ESA systems- Bangalore(*www.**esa** india.com)*

4.SPJ Embedded Technologies .Ltd. (*www.**spjsystems**.com)*

5.Advanced Electronic systems-Bangalore (*www. **als**india.net)*

6. Front line electronics . *www. **frontline**-**electronics**. com*

Books that have helped me to understand the embedded systems :

1) Barr, Michael, Programming embedded sytems in C and C++ - OReilly Publ.
2) Raj Kamal, Embedded systems, TMG
3) Valvano, Introduction to Embedded microcomputer systems, Thomson Publ.

4) Mazidi and Mazidi, The 8051 microcontroller and embedded sytems - Pearson education.
5) Peatman,J.B. Design with microcontrollers and microcomputers, McGraw Hill
6) Sewart. J.W. The 8051 Microcontroller Hardware, Software and Interfacing – Prentice Hall

7) Ayala Kenneth, The 8051 Microntroller – Architecture, Programming and Applications – Delmar Publ.

8) Ajay Deshmukh, Microcontrollers – TATA McGraw Hill

9) Rajkamal, Microcontrollers - Architecture, Programming – Pearson Publ.

10) Myke Predko, Programming the 8051 Microcontroller – McGraw Hill

11) Michael J. Pont, Embedded C - Addison Wesely Publ.

12) Steve Heath, Embedded system design – Heinemann Publ.

13) Frank Vahid, Embedded systems – a unified hardware/software Introduction – John Wiley and sons Publ.

14) Barnett Cox & O'cull, "Embedded C Programming & the Microchip PIC", Thomson Delmar Learning.

Website References

1. http://www.eg3.com
2. http://www.ARM.MCU.com
3. http://www.mcjournal.com
4. http://www.iar.com
5. http://www.keil.com
6. http://www.semiconductors.philips.com/microcontrollers
7. http://www.embedded.com
8. http://www.powersoftsystems.com

- *www.macrovision.com/**newsletter**s*
- *www.planar**embedded**.com*
- ***www.8051.org***
- *www.**8051projects**.net*
- *www.programmersheaven.com*
- *microcontrollershop.com*

Concluding remarks

- There's lots more to learn, but with these basics under your belt it's just a small matter of gaining experience
- Good luck!